UNIVERSITÉ de GRENOBLE

No attribué par la bibliothèque
/_/_/_/_/_/_/_/_/_/

Habilitation à Diriger les Recherches

Mathématiques, Sciences et technologies de l'information, Informatique

Sélection de Publications Scientifiques

en vue de la soutenance de

Rémi Ronfard

le 2 décembre 2009

Titre :

ANALYSE AUTOMATIQUE DE FILMS : DES SEQUENCES D'IMAGES AUX SEQUENCES D'ACTIONS

JURY

Mme.	**Catherine BERRUT**	Président
M.	**Roger MOHR**	Rapporteur
M.	**Jean PONCE**	Rapporteur
M.	**Andrew ZISSERMAN**	Rapporteur
M.	**Patrick PEREZ**	Examinateur
Mme.	**Cordelia SCHMID**	Examinateur

Table des matières

CHAPITRE 1

Analyse Automatique de Films

Logiques de description pour l'Analyse Structurelle de Film, de Jean Carrive, François Pachet et Rémi Ronfard. Chapitre du livre Ingénierie des Connaissances : evolutions récentes et nouveaux défis, Eyrolles, Paris, 1999.

Audiovisual-based Hypermedia Authoring de Gwendal Auffret, Jean Carrive, Olivier Chevet, Thomas Dechilly, Remi Ronfard, Bruno Bachimont. ACM Conference on Hypertext and Hypermedia, 1999.

From Video Shot Clustering to Sequence Segmentation de Emmanuel Veneau, Rémi Ronfard et Patrick Bouthemy. International Conference on Pattern Recognition, 2000.

Clavis - a Temporal Reasoning System de Jean Carrive, François Pachet et Rémi Ronfard. Recherche d'Informations Assistee par Ordinateur (RIAO), 2000.

Logiques de descriptions pour l'analyse structurelle de film

Description Logics for Structural Analysis of Film

Jean Carrive[1], François Pachet[2], Rémi Ronfard[1]

[1]Institut National de l'Audiovisuel (INA)
[2]SONY CSL-Paris & Lip6 (Paris 6)

Institut National de l'Audiovisuel
Direction de la Recherche
4, avenue de l'Europe – 94 366 Bry sur Marne

{jean, remi}@ina.fr, pachet@csl.sony.fr

Résumé

Dans le contexte de l'annotation de documents filmiques assistée par ordinateur, nous posons le problème de l'analyse filmique automatique, et identifions 3 problèmes de base : classification de plans, regroupement temporel et pilotage d'algorithmes d'extraction. Ces trois processus mettent en œuvre des connaissances provenant de plusieurs domaines d'expertise : documentalistes, spécialistes d'analyse du signal et professionnels de l'audiovisuel.

Nous proposons d'utiliser le formalisme des logiques de descriptions comme paradigme principal de représentation pour représenter ces divers types de connaissances dans un environnement intégré. Nous proposons dans ce cadre un mécanisme de regroupement temporel fondé sur une restriction du formalisme de Allen qui permet de limiter les problèmes de complexité.

Mots Clef

Logiques de descriptions, indexation, raisonnement temporel, audiovisuel

Introduction

La numérisation systématique des documents filmiques ainsi que la production de documents accessibles au grand public ont connu ces dernières années une croissance exponentielle [1]. Le problème de l'indexation de grands volumes de documents filmiques devient ainsi une préoccupation importante à la fois des industriels de l'audiovisuel et des centres documentaires comme l'INA (Institut National de l'Audiovisuel).

Ce problème concerne des chercheurs de domaines très divers : bases de données, analyse d'image, traitement du signal sonore, recherche d'information, etc. Un schéma général d'indexation peut être vu comme étant composé de plusieurs étapes : analyse automatique (extraction de primitives et segmentation spatio-temporelle de bas niveau), annotation et découpage manuels, stockage en base de données. Les techniques de segmentation temporelle actuellement disponibles ne permettent pas d'extraire des unités temporelles suffisamment longues et pertinentes pour qu'il soit concevable de les annoter manuellement. Nous proposons une méthode de regroupement des unités temporelles de bas niveau, extraites automatiquement, en unités temporelles de plus haut niveau. Le but de cette méthode est d'offrir au documentaliste chargé de l'annotation un découpage du document en unités pertinentes – appelées *séquences* –, dont la cohésion permette qu'elles soient annotées comme un tout.

Cette extraction nécessite une ingénierie des connaissances adaptée qui permette à des experts des domaines concernés (documentalistes, professionnels de l'audiovisuel et experts en analyse du signal) de spécifier simplement leurs connaissances du domaine. Cette étude se place dans le contexte du projet européen DIVAN (Distributed Video Archives Network) auquel participent l'INA, l'IRISA, la Rai, etc.

Nous allons décrire précisément le problème de l'indexation dans ce contexte dans la première section, en identifiant trois problèmes de base. Nous identifions ensuite les logiques de descriptions comme un « bon » formalisme pour représenter le contenu des documents filmiques (radio, télévision) ainsi que les requêtes portant sur ces documents. Dans cet article, nous identifions le problème de l'analyse des documents à partir d'observations extraites par des automates. Nous proposons d'enrichir les logiques de descriptions avec un système de raisonnement temporel. Nous faisons le choix d'une représentation par intervalles permettant de modéliser des actions simultanées (par exemple événements visuels et sonores) et d'assurer le regroupement d'unités temporelles avec une complexité raisonnable. Nous considérons pour le moment que toutes les actions sont celles du monteur ou du réalisateur. Dans l'avenir, nous espérons modéliser également les personnages et leurs actions, ce qui pose des problèmes difficiles, à peine ébauchés dans le domaine de la vision par ordinateur (voir tout de même [2] et [3]).

Enfin, après une courte discussion, nous concluons sur l'état courant de notre implémentation et sur des perspectives d'extension.

1. Analyse filmique

L'analyse, dans le sens dans lequel nous l'entendons, consiste à reconstituer la structure temporelle d'un document à partir d'informations « primitives » obtenues le plus souvent par des algorithmes d'analyse du signal. Certaines de ces informations primitives peuvent être obtenues *a priori*, de manière systématique, pour tous les documents (découpage en plan, histogrammes de couleurs, etc.). D'autres informations nécessitent la mise en œuvre d'algorithmes plus complexes et ne peuvent être obtenues que sur demande explicite, assorties d'informations contextuelles nécessaires au bon fonctionnement de l'algorithme (reconnaissance de caractères ou de visages dans une image).

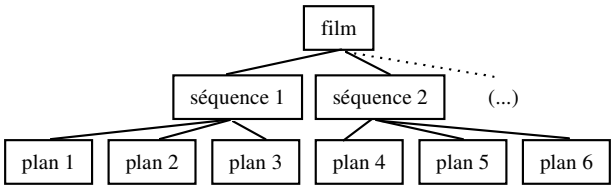


figure 1 : découpage d'un film

Dans la conception traditionnelle du film, représentée aujourd'hui par le cinéma hollywoodien, la structure temporelle du film se présente comme un arbre où les plans, unités élémentaires, sont regroupés en séquences [4] (voir figure 1). Cette analyse pose trois problèmes de fond : 1) la représentation des primitives d'extraction et leur organisation dans une taxonomie de plans, 2) le regroupement automatisé de ces plans en séquences, et 3) le pilotage explicite d'algorithmes complémentaires d'extraction permettant de raffiner l'analyse. Ce schéma général est représenté figure 2.

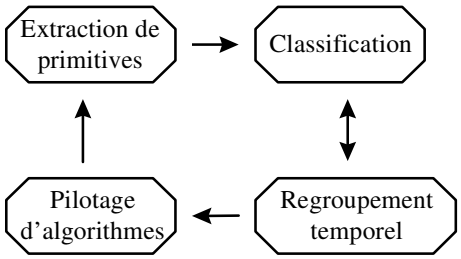


figure 2 : schéma général d'analyse

Après avoir présenté l'état actuel de la modélisation des documents filmiques à l'INA, nous allons présenter dans les sections suivantes chacun de ces problèmes.

1.1 Modèles de documents filmiques

Certains éléments de base du langage filmique sont représentés sous forme de taxonomie *explicite* ; d'autre part, il existe une taxonomie *implicite* des types de documents, qui sont représentés sous la forme d'un ensemble de « fiches collection ».

1.1.1 Taxonomie des événements filmiques

Les connaissances générales du domaine pouvant être formalisées recouvrent les éléments propres à la production : prise de vue et montage essentiellement. Les différents cadrages, par exemple – gros plan, plan moyen, plan serré, etc. – peuvent être décrits par une hiérarchie. De même, les mouvements de caméra traditionnels se prêtent au même type de description : caméra fixe (sans mouvement), les divers types de zooms (zoom avant, zoom arrière), de travellings (latéral, avant, arrière), etc. Une taxonomie des événements filmiques (TEF) existe ; il est possible d'en donner une représentation formelle partielle. La figure 3 donne une vision simplifiée des mouvements de caméra classiques.

Un des éléments importants du domaine est le *plan*. Le plan est généralement défini comme la plus petite unité syntaxique du film, et ne comporte ni coupure de caméra ni raccord [5], bien qu'on puisse le subdiviser en unités plus petites, surtout lorsque des mouvements de caméra complexes le composent. Les propriétés communément admises du plan sont la durée (plan long ou plan court), le type de transition avec le plan qui le précède et celui qui le suit : transition de type *cut* (coupe franche) ou de type graduel (fondus, volets, etc.), le nombre de personnages, le cadrage des personnages (gros plan, plan moyen, etc.). Une formalisation des propriétés générales d'un plan a été proposée dans [5].

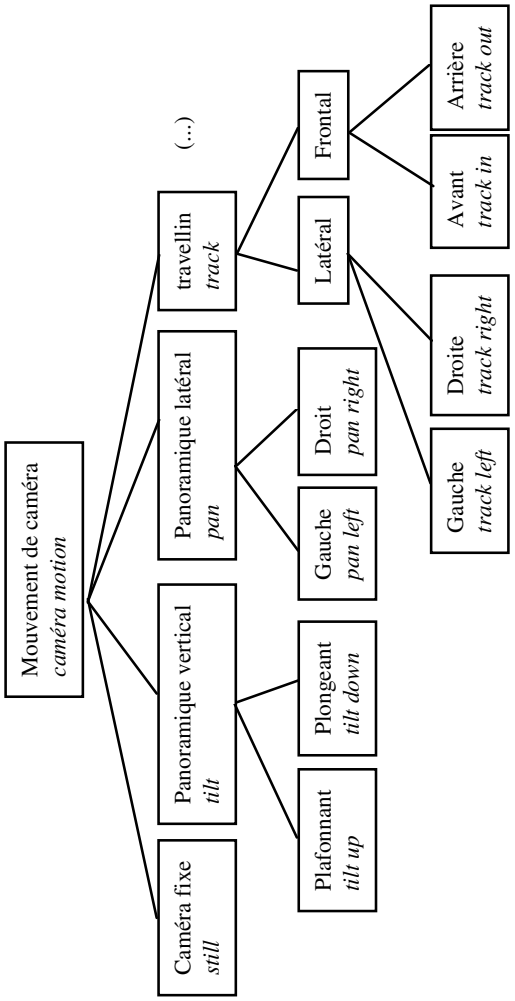


figure 3 : mouvements de caméra dans la TEF

1.1.2 Taxonomie des types de documents

Des plans consécutifs peuvent être regroupés en *séquences*, qui constituent des unités sémantiques. Les

émissions de télévision suivent très souvent un canevas relativement précis. C'est le cas pour les journaux télévisés, qui ont été jusqu'ici les plus étudiés, mais c'est également le cas de beaucoup d'autres émissions : magazines de reportages, de variétés, feuilletons ou sit-coms. Des modèles d'émission peuvent alors être établis. C'est le cas à l'INA, où de tels modèles, appelés « fiches collection », servent à indexer manuellement un grand nombre d'émissions. Une fiche collection est donnée à titre d'exemple par la figure 4.

Projection Privée
Produit par S. Bleckmans
Réalisé par E. David
Présenté par L. Weil

Magazine à rubriques hebdomadaire consacré à l'actualité cinématographique (...).

Chaîne de 1ère diffusion Métropole Télévision
Date de début 17/10/89
(...)

Genre : cinéma - **Forme** : magazine - **Descripteurs** : cinéma, film (...)

Résumé
Principe de l'émission : Ce magazine, présenté par L. Weil, est composé de présentation de sujets promotionnels sur les films qui vont sortir (...)
Déroulement chronologique : Durant le premier plateau, *xxx* présente l'émission (sommaire en images). Il lance ensuite le premier sujet (...)

Dispositifs
Générique : début sur fond noir, incrustation du titre
Dispositifs plateau : monocaméra, le siège du présentateur est (...)
Construction générale : plans généraux du plateau, plans taille du présentateur, passage d'un plan à l'autre par zoom avant
Construction des éléments : Sur les premières images, des sujets, le titre est incrusté sur un large bandeau noir (...)

etc.

figure 4 : exemple de fiche collection (source : Inathèque de France)

1.2 Agrégation temporelle de plans en séquences

Annoter un document au niveau du plan n'est pas réaliste : les plans sont en général courts, et un document peut comporter plusieurs centaines, voire plusieurs milliers de plans. Le regroupement automatique de plans en séquences constitue donc un axe de recherche important qui permettrait de faciliter l'indexation et l'annotation d'un plus grand nombre de documents.

Comme nous l'avons vu dans la section précédente, certaines caractéristiques d'un document filmique, film ou vidéo, peuvent être extraites du document de manière algorithmique. C'est le cas par exemple des mouvements de caméra (plans fixes, zooms, travellings), des transitions de plan (cuts, fondus) [6], de la présence de musique, etc. Cependant, ces caractéristiques se trouvent d'une part être en nombre limité et d'autre part présentent généralement un contenu sémantique faible. Il se pose alors le problème de les combiner entre elles afin de faire apparaître des entités de plus haut niveau, en rapport plus étroit avec l'idée que l'on peut se faire des éléments constitutifs du document. Le principe d'analyse dont nous avons besoin consiste à construire ces entités pertinentes en assemblant des caractéristiques primitives. L'analyse doit donc être générative.

1.2.1 Règles structurelles générales

Certaines règles de composition peuvent s'appliquer à un grand nombre de genres différents de films ou de documents vidéo. Il s'agit essentiellement de règles de montage qui, schématiquement, expriment la manière d'assembler les plans entre eux afin de ne pas rompre la continuité du discours : continuité des positions, des directions, des regards, des éclairages, etc. Le montage est parfois considéré comme l'essence même de l'art cinématographique [4].

L'étude menée dans [7] formalise quelques règles de montage ne nécessitant pas de connaissances préalables sur le document traité : les objets intervenants dans les prémisses des règles sont directement observables dans le flux vidéo et elles s'appliquent de manière générale pour tout type de document. Ces règles sont essentiellement fondées sur l'observation de l'alternance de différents types de transitions de plan (« cuts » et transitions graduelles), sur la similarité des plans entre eux, avec des mesures simples sur les histogrammes de couleurs, sur la présence ou l'absence de musique, sur la durée relative des plans, etc. La règle illustrée par la figure 5 signifie que lorsqu'on se trouve en présence d'au moins quatre plans séparés par des « cuts » suivis d'une transition graduelle, elle-même suivie à nouveau de quatre plans séparé par des « cuts », alors on peut estimer qu'il y a une rupture de séquence au moment de la transition graduelle.

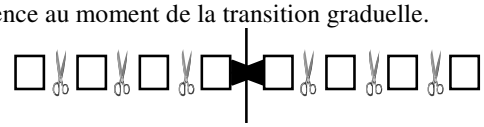


figure 5 : changement de séquence en fonction des transitions de plan, extrait de [7]

1.2.2 Connaissances spécifiques

Cependant, ces règles générales ne peuvent rendre compte de la diversité de tous les types de documents. Selon le genre d'émission (journaux télévisés, magazines de variété), la structure temporelle d'un document filmique peut être plus ou moins complexe et précise. Des modèles de documents spécifiques ont déjà été étudiés, concernant à notre connaissance principalement la structure du journal télévisé [8].

Dans les cas les plus simples, cette structure temporelle peut se formaliser comme une expression régulière. Ainsi, beaucoup de magazines alternent scènes de plateaux et scènes de reportage. La structure de ce type de documents peut alors se mettre sous la forme :

```
générique_début
(plateau reportage)*
générique_fin.
```

De la même manière, la structure de certains documents peut être décrite par une grammaire formelle, ou sous la forme d'une DTD SGML/HyTime, comme c'est le cas dans [9]. Il nous semble cependant que ces approches, si elles sont parfaitement justifiées pour un type particulier de documents, comme le journal télévisé, ne peuvent répondre au cas plus général de la description de différents types de documents.

Ces modèles de documents ne doivent pas seulement indiquer la structure temporelle respectée par les émissions appartenant à une collection donnée, ils doivent également indiquer les élément non temporels que l'on retrouve généralement d'une émission sur l'autre. Il peut s'agir du ou des présentateurs, d'éléments de décor, du logo de l'émission, des génériques, des jingles, etc. Dans la perspective d'un modèle automatisé, des descriptions numériques ou des exemples sont précieux, qui permettent une détection algorithmique des éléments récurrents : modèles phonétiques des voix des principaux intervenants, images numérisées des logos, etc.

Le modèle d'un document étant connu *a priori* à partir de simples données de catalogage, le problème se pose alors de contraindre les algorithmes d'analyse de bas niveau, en fournissant à ces algorithmes des données dépendantes du contexte.

1.3 Pilotage d'algorithmes

Nous décrirons par un exemple le type de situation dans lequel il est nécessaire de piloter des algorithmes d'extraction de primitives afin de raffiner l'analyse. Considérons un magazine de cinéma. Un tel magazine est composé principalement d'une alternance de scènes de studio et de séquences d'extraits d'un film à l'affiche. Les scènes de studio peuvent être isolées des séquences d'extraits en utilisant diverses méthodes : les scènes de reportages sont par exemple encadrées en haut et en bas par des bandes noires caractéristiques, ce qui est facile à détecter pour un algorithme d'analyse d'image. D'autre part, le modèle de document précise qu'au cours des cinq premières secondes des séquences d'extraits apparaît un panneau noir à gauche de l'écran où se trouve inscrit le titre du film dont il est question.

Il est dans ce cas possible dans un premier temps d'isoler les séquences d'extraits des scènes de studio par détection des bandes noires, et dans un second temps de n'exécuter un algorithme d'extraction de texte que sur une partie réduite du document.

2. Logiques de descriptions : un formalisme pour l'analyse

Le problème de l'analyse filmique étant posé dans son ensemble, se pose alors celui du cadre technologique dans lequel l'exprimer en vue d'une ingénierie des connaissances outillée et efficace. Le formalisme des logiques de descriptions est particulièrement bien adapté à la représentation et l'exploitation de taxonomies naturelles, et nous avons déjà proposé dans [10] de le prendre comme outil conceptuel de base pour nos travaux. Les logiques de descriptions sont des langages de représentation des connaissances bien étudiés et formalisés [11]. Cependant, s'il est naturel d'utiliser les logiques de descriptions comme langage de représentation pour les objets du domaine filmique, il paraît plus malaisé de s'en servir pour exprimer des connaissances à forte composante temporelle. Nous verrons plus loin que les possibilités de règles de production en chaînage avant qu'offre CLASSIC [12], système de logique de descriptions que nous avons utilisé pour l'implémentation, permet de résoudre un certain nombre de problèmes.

2.1.1 Classification et logiques de descriptions

Les logiques de descriptions forment un ensemble de langages de représentation de connaissances ; elles permettent de représenter des connaissances de manière structurée en séparant les définitions de concept (base terminologique ou *TBox*) des descriptions d'individus (base des assertions ou *ABox*). Un concept représente un un ensemble d'individus. Les *rôles* représentent des relations binaires. Descriptions de concepts et rôles sont organisés en hiérarchies par la relation de *subsomption* : le concept C subsume le concept D si les instances de D sont nécessairement instances de C. La *classification* est l'opération permettant d'attribuer une place à un concept (ou éventuellement à un rôle) dans la hiérarchie des concepts (ou des rôles). On trouvera dans [11] une introduction aux logiques de descriptions, un ouvrage de référence restant [13].

Nous proposons de poursuivre la piste indiquée par [7] en nous plaçant dans le cadre des logiques de descriptions. Nous posons ici le problème de l'agrégation temporelle décrit dans la section 1.2 dans le contexte des logiques de descriptions. Nous pensons d'autre part que dans le cadre de l'analyse de documents filmiques, seuls un nombre très restreint de règles peuvent s'appliquer en toute généralité, et qu'il convient lorsque cela est possible de formaliser des modèles de documents représentant un type particulier de documents (journal télévisé, magazine, par exemple).

2.2 Représentation de connaissances

La taxonomie TEF se représente aisément à l'aide du formalisme des logiques de descriptions. Comme il a été dit plus haut, certaines connaissances du domaine filmique peuvent être assez naturellement exprimées sous forme de hiérarchies : types de cadrage, mouvements de caméra, types de transition de plans, types d'éclairage (intérieur/extérieur, par exemple), nombre de personnages, etc. Tous ces éléments s'expriment de manière naturelle dans un langage de description comme CLASSIC.

Il est important de faire la distinction entre les aspects temporels d'un plan (ses dates de début et de fin) et sa description non temporelle, c'est-à-dire ses propriétés (ses *rôles*, pour reprendre la terminologie des logiques de descriptions).

Par exemple, conformément à la taxonomie TEF, on peut représenter le fait qu'un plan à 3 personnages dont le principal mouvement de caméra est un travelling avant (track-in), défini par l'expression CLASSIC suivante :

```
(define-concept 'SHOT-1 '(and
    (exactly 1 camera-motion)
    (fills camera-motion track-in)
    (exactly 1 character-num)
    (fills character-num 1)))
```

est plus spécifique que le concept de plan ayant entre 1 et 5 personnages et dont le principal mouvement de caméra est un travelling, défini par :

```
(define-concept 'SHOT-2 '(and
    (exactly 1 camera-motion)
    (fills camera-motion tranck)
    (exactly 1 character-num)
    (all character-num (min 1) (max 5)))
```

On obtient ainsi une opérationalisation de la taxonomie TEF, qui forme une ontologie explicite des types de plans.

2.3 Raisonnement temporel

Afin d'exprimer les règles reflétant la structure des documents, il convient de se doter d'un modèle temporel permettant d'exprimer des contraintes sur les occurrences d'événements. Il sera tout d'abord question du choix de ce modèle, puis nous nous intéresserons à un schéma de règles de regroupement temporel.

2.3.1 Un modèle temporel

Le choix du modèle de raisonnement temporel utilisé est important. Il s'agit en général d'un compromis entre expressivité et efficacité.

2.3.1.1 Logique de Allen restreinte

Pour exprimer la structure général d'un document filmique, il faut pouvoir exprimer des contraintes sur les occurrences d'événements temporels : un plan en intérieur suivi d'un plan en extérieur, une musique commençant pendant le dernier plan d'une séquence, etc.

Dans le domaine de l'analyse de la vidéo, [14] propose une théorie – PNF calculus[1] –, fondé sur la logique temporelle d'intervalles de Allen [15] et assortie d'un algorithme en temps polynomial pour la reconnaissance de structures temporelles. S'il est à noter que c'est l'une des rares tentatives d'exploiter le raisonnement temporel pour la représentation de la vidéo, il faut également remarquer que le pouvoir d'expression de cette théorie est trop limité : l'exigence d'un algorithme rapide (en $O(n^2)$) conduit les auteurs à regrouper les relations temporelles dans trois classes correspondant aux notions intuitives de *passé*, *présent* et *future*, ce qui constitue une perte d'information importante.

Nous proposons d'utiliser l'algèbre d'intervalles proposée par [16] – *Pointizable Interval Algebra* –, qui consiste à transformer les contraintes sur des disjonctions de relations de Allen (figure 6) entre intervalles temporels en conjonctions de contraintes sur les bornes de ces intervalles. Seul un sous-ensemble de l'algèbre proposé par Allen est ainsi représentable. Par exemple, la relation temporelle :

A {*before* ∨ *meets* ∨ *overlaps*} B

s'exprime par la conjonction de contraintes :

début(A) < début(B)
fin(A) < fin(B)[2]

mais la relation

A {*before* ∨ *after*} B

n'a pas d'équivalent.

Ce modèle est moins complet que celui de Allen, mais le test de cohérence peut être effectué en temps polynomial. D'autre part, la mise en œuvre de cette algèbre est relativement simple. Plus récemment, [17] a proposé un sous-ensemble maximal calculable de la logique de Allen, la sous-classe *ORD-Horn*, qui reprend les travaux de van Beek [16] en s'affranchissant du passage des intervalles aux bornes d'intervalles. Nous n'excluons pas d'utiliser ce formalisme dans un second temps.

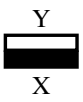
Relation	**Exemple**
X equals Y	
X meets Y Y met-by X	
X overlaps Y Y overlapped-by X	
X during Y Y includes X	
X starts Y Y started-by X	
X finishes Y Y finished-by X	
X before Y Y after X	

figure 6 : les 13 relations temporelles de Allen

2.3.1.2 Subsomption de relations temporelles

Nous avons choisi pour l'instant de réifier les relations temporelles. La principale raison est que nous bénéficions ainsi des mécanismes de classification de CLASSIC. Ainsi, la relation temporelle {*before* ∨ *meets* ∨ *overlaps*} subsume-t-elle la relation {*before* ∨ *meets*}, ce qui correspond bien à l'intuition : l'ensemble des segments temporels mis en relation par {*before* ∨ *meets*} l'est *a fortiori* par la relation {*before* ∨ *meets* ∨ *overlaps*}.

Du point de vue de l'implémentation, CLASSIC offre la possibilité sous certaines restrictions de définir un concept comme étant l'ensemble des instances qui satisfont un test (fonction Lisp renvoyant un booléen), et de définir explicitement des liens de subsomption entre ces tests afin que le système puisse classifier concepts et instances.

Il est donc possible de définir un concept temporel (TEMPORAL) comme ayant un début et une fin, le début précédent la fin[3] :

[1] PNF : past, now, future

[2] avec bien sûr début(A) < fin(A) et début(B)<fin(B)

[3] La syntaxe utilisée est proche de celle de CLASSIC, avec quelques facilités d'écriture pour les contraintes de

```
(define-concept 'TEMPORAL '(and
    (exactly 1 begin)
    (all begin integer)
    (exactly 1 end)
    (all end integer)
    (< begin end)))
```

De la même manière, on définit le concept de relation temporelle comme faisant intervenir deux instances de TEMPORAL.

```
(define-concept 'TEMPORAL-RELATION '(and
    (exactly 1 temporal-1)
    (all temporal-1 TEMPORAL)
    (exactly 1 temporal-2)
    (all temporal-2 TEMPORAL)))
```

La relation temporelle définie ci-dessus est la relation la plus générale. Quelles que soient deux intervalles temporels, ils sont toujours en relation !
Les 13 intervalles de base de Allen sont définis comme sous-concepts de TEMPORAL-RELATION. La relation *meets* est par exemple définie par :

```
(define-concept 'TEMPORAL-MEETS '(and
    TEMPORAL-RELATION
    (= (temporal-1 end) (temporal-2
    begin)))
```

CLASSIC ne permettant pas de définir des concepts comme disjonction de concepts, les concepts qui correspondent aux relations temporelles s'exprimant comme disjonction des relations de Allen doivent être décrites explicitement, ainsi que les relations de subsomption entre les tests correspondants.
Une remarque doit être faite quant à la manière dont est effectué le test *a* R *b* (*a* et *b* sont-ils mis en relation par R ?). Dans l'implémentation existante, une instance de TEMPORAL-RELATION est créée. Si cette instance se trouve être classifiée par CLASSIC comme instance de R, alors la relation est respectée. Il est évident que cette manière de procéder en réifiant les relations temporelles est très coûteuse, mais cela nous permet dans un premier temps d'une part d'exprimer clairement ces relations et d'autre part de profiter du mécanisme de subsomption offert par CLASSIC.

2.4 Règles de regroupement

Pour exprimer la structure temporelle des documents, nous proposons d'utiliser des règles de regroupement temporel qui agrègent des formes temporelles de bas niveau en des formes de plus haut niveau.

2.4.1 Principe général

Le principe de regroupement temporel présenté ici repose sur la possibilité offerte par CLASSIC d'associer des règles à un concept, lesquelles règles sont déclenchées à chaque instanciation de ce concept. Lorsqu'une instance *a* du concept A est créée, une règle et déclenchée. Dans notre cas, le traitement de cette règle consiste à chercher toutes les instances b_i du concept B telles que *a* R b_i, avec R instance d'un sous-concept de TEMPORAL-RELATION (R est une relation temporelle). *a* et b_i sont alors agrégées pour former une instance de plus haut niveau, de concept G. Ce type très général de règles fait intervenir quatre paramètres : les concepts A, B, R et G. Les nouvelles instances ainsi créées peuvent à leur tour être agrégés en groupes d'encore plus haut niveau.

2.4.2 Trois types de règles

Dans un premier temps, deux types de règles ont été identifiés qui permettent de regrouper des éléments temporels : les règles qui agrègent deux instances de concepts différents en une instance d'un concept de plus haut niveau, et les règles qui agrègent N instances du même concept en une instance d'un concept de plus haut niveau.
Dans le premier cas, on cherche à regrouper des formes qui se complètent. Ce sera le cas par exemple du champ-contre champ : admettons que l'on ait pu segmenter un document selon le schéma de la figure 7. Les cases représentent les plans. Les plans notés A et B sont des plans d'un type quelconque. Les plans notés I (resp. F) signalent la présence du personnage I (resp. F). Il paraît naturel ici de regrouper les plans sur le schéma :

A* (I F)* B*

La règle de regroupement qui s'applique ici regroupe un plan de type I suivi d'un plan (relation temporelle *meets*) de type F en un groupe de deux plans I-F. Le premier type de règles peut se décrire sous la forme :

$$C_1 \, R \, C_2 \rightarrow G \qquad (1)$$

avec :
C_1, C_2 : concepts héritant de TEMPORAL
R : concept héritant de TEMPORAL-RELATION
G : concept héritant de TWO-TEMP-GRP, groupe de deux instances de TEMPORAL. Ce concept est défini comme suit :

```
(define-concept 'TWO-TEMP-GRP '(and
    TEMPORAL
    (exactly 1 first-temporal)
    (all first-temporal TEMPORAL)
    (exactly 1 second-temporal)
    (all first-temporal TEMPORAL)))
```

Dans l'exemple cité, la règle de regroupement serait donc :

```
I meets F → I-F
```

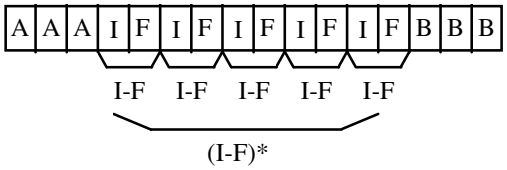


figure 7 : champ-contre champ

Une fois regroupés les plans I et les plans F en groupes I-F , on voudrait pouvoir regrouper ces derniers en une séquence d'instances de I-F. Un deuxième type de règles intervient ici qui agrège des instances de la même classe en une instance d'une classe de plus haut niveau :

$$C \, R \rightarrow G \qquad (2)$$

avec :
C : concept héritant de TEMPORAL

---

cardinalité et les tests Lisp. Ces derniers sont notés en italique.

R : concept héritant de TEMPORAL-RELATION
G : concept héritant de TEMPORAL-SEQ, séquence d'instances de TEMPORAL, défini par :

```
(define-concept 'TEMPORAL-SEQ '(and
    TEMPORAL
    (at-least 2 element)
    (all element TEMPORAL)))
```

Ce type de règles permet d'instancier une séquence temporelle dont les éléments appartiennent à la même classe et sont deux à deux mis en relation par une relation temporelle donnée.
Il est apparu qu'à ces deux types de règles d'agrégation devaient s'ajouter des règles permettant d'exprimer des ruptures temporelles ou des transitions. C'est le cas de la règle de transition entre séquences présentée plus haut (voir figure 5). Ces règles permettraient d'identifier des structures du type ABA' comme indiquant que B est un élément de transition. Dans l'exemple décrit, A et A' représentent une suite de plans séparés par des « cuts », et B représentent une transition graduelle. La règle tirée de [7] et illustrée par la figure 8 indique que lorsque deux plans consécutifs présentent des mesures colorimétriques voisines au sens d'une certaine distance, il y a de fortes chances qu'il y ait une rupture de séquences entre ces deux plans. Cette structure est de la forme ABB'C. Il est possible qu'un type spécifique de règles soit utile pour pouvoir effectuer des regroupements représentant ce genre de structure. Cependant, il faut noter que des structures de type AB A' ou ABB'C peuvent être exprimées avec une combinaison de règles de type (1). Dans le premier cas, on peut agréger A et B en A-B, puis A-B et A' en A-B-A', dans le deuxième on peut également décomposer la règle, ce qui nuit bien sûr à la clarté des règles, et ce qui conduit à créer des concepts intermédiaires n'ayant pas nécessairement de sens. Cependant, il doit être possible d'automatiser la décomposition de ces règles et ainsi de « cacher » ces concepts intermédiaires.

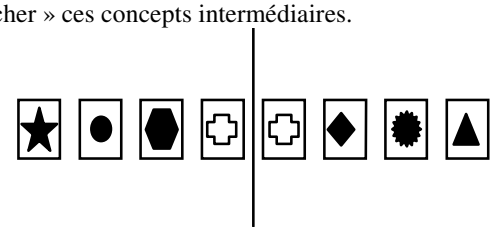


figure 8 : changement de séquence en fonction de similarités de plan

2.4.3 Principes d'exécution et monotonie

Les règles de regroupement étant exprimées de la manière exposée plus haut, il se pose alors un problème de monotonie lié à l'ordre dans lequel sont créées les instances. Une règle de type A R B → G est déclenchée à chaque création d'une instance *a* du concept A. L'action effectuée par la règle consiste à chercher toutes les instances b_i du concept B mis en relation par R avec *a*, puis à agréger *a* et b_i en une instance du concept G. Ce principe est illustré par l'algorithme suivant :

```
Création d'une instance a ⊂[4] A          (i)
Déclenchement de la règle                 (ii)
Recherche des b_i ⊂ B tq a R b_i          (iii)
Agrégation de a et b_i                    (iv)
```

Les règles étant déclenchées par l'instanciation de concept, on constate que l'ordre dans lequel les instances sont créées est important : deux instances ne peuvent être agrégées que si elles existent déjà au moment du déclenchement de la règle. Nous proposons ici une manière de résoudre ce problème pour chacun des deux types de règles exposés plus haut.
Pour les règles de type (1) il convient d'engendrer pour chaque règle la règle symétrique quant à la relation temporelle. Ainsi, pour chaque règle de type :

$$A\ R\ B \rightarrow G$$

doit-on également créer la règle :

$$B\ \overline{R}\ A \rightarrow G$$

avec $\overline{R}$ la relation temporelle inverse de R, définie par :

$$x\ R\ y \Leftrightarrow y\ \overline{R}\ x$$

Par exemple, la relation inverse de la relation temporelle {*before* ∨ *meets* ∨ *overlaps*} est {*after* ∨ *met-by* ∨ *overlapped-by*}.
Pour les règles de type (2), la procédure est un peu plus complexe. Le principe des règles inverses doit également être appliqué mais n'est cependant pas suffisant.
L'étape `(iv)` de l'algorithme consiste à agréger *a* et *a'*, instances du même concept A.
Pour la règle de type (2) A R → G regroupant N instances de A en une instance de G, une manière naïve de procéder consiste, lors du déclenchement de la règle par la création d'une instance *a* du concept A, à chercher une instance *g* du concept G telle que *a* soit en relation temporelle R avec l'une des valeurs du rôle `element` de *g*, puis à ajouter *a* comme valeur du rôle `element` à *g*. Si une telle instance *g* n'est pas trouvée, on cherche une instance *a'* de A telle que *a* et *a'* sont en relation temporelle R. Si un tel *a'* existe, *a* et *a'* sont alors agrégés en une nouvelle instance de G.

```
chercher g ⊂ G tq a' ∈[5] g
si g existe alors
    ajouter a comme valeur du rôle
    element à g
sinon
    chercher a' tq a R a'
    si a' existe
    créer g ⊂ G avec a et a' pour
    valeurs du rôle element
```

Sur l'exemple illustré par la figure 9, les plans grisés représentent des plans de la même classe L, par exemple une série de plans de plateau. Lorsque deux plans se chevauchent, la transition entre ces deux plans est graduelle. C'est le cas entre les plans 1 et 2, 2 et 3, 4 et 5, 5 et 6, alors qu'il y a un « cut » entre les plans 3 et 4.
Pour regrouper les plans de type L, on écrit la règle de type (2) :

$$L\ \{meets \vee overlaps\} \rightarrow GL$$

[4] ⊂ signifie « instance du concept »
[5] ∈ signifie ici « est valeur du rôle `temporal-element` de »

avec GL le concept définissant un groupe de plans L, concept héritant de TEMPORAL-SEQ.
On constate que si les plans sont créés dans l'ordre de leur numéro, l'algorithme indiqué agrège bien les plans en une seule instance de GL. Toutefois, si l'ordre de création est 1-6-2-3-5-4, les plans seront regroupés en deux instances de GL : 1-2-3-4 et 4-5-6. On constate ici que le plan n°4 peut être agrégé à deux groupes de plans distincts. Il convient dans ce cas d'agréger à nouveaux ces deux groupes.
L'étape (iv) de l'algorithme, pour les règles de type (2), doit donc prendre en compte le fait que lorsqu'une instance peut s'agréger à plusieurs groupes, ces groupes doivent être agrégés entre eux. Cela peut s'exprimer de la manière suivante :

```
extraire liste l des gᵢ ⊂ G tq a' ∈ gᵢ
si |l| = 0
    créer g ⊂ G et ajouter a et a' à g
si |l| = 1
    ajouter a à g₁
si |l|>1
    ajouter a à g₁
    agréger gᵢ, 2≤i≤|l|, à g₁
```

Agréger plusieurs instances de TEMPORAL-SEQ, comme il est indiqué à la dernière ligne de l'algorithme ci-dessus, peut ne pas être une opération triviale si d'autres rôles que `element` sont pourvus. Nous ne traiterons pas ce problème ici.

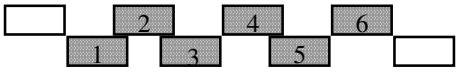


figure 9 : groupement de plans similaires

Enfin, un problème de monotonie se pose si les instances intervenant dans les règles de regroupement venaient à être modifiées. En effet, des propriétés peuvent être ajoutées aux instances ; ce serait le cas par exemple si plusieurs algorithmes de détection, ou bien une intervention manuelle, venaient successivement raffiner une description des plans.

2.5 Classification / Analyse

L'analyse telle qu'elle est présentée ici repose en dernier recours sur les algorithmes d'analyse. Il reste donc à préciser la manière dont vont interagir ces algorithmes et les processus de raisonnement présentés plus haut. La manière la plus simple consiste à exécuter tous les algorithmes d'analyse, puis à utiliser les résultats pour raisonner. Cependant, nous pouvons faire interagir plus étroitement analyse et raisonnement. Deux modes d'interaction entre analyse et raisonnement ont été identifiés. Le premier consiste pour un algorithme à créer des instances « primitives ». C'est le cas d'un algorithme de segmentation en plan, par exemple. Dans le second mode d'interaction, un algorithme spécialise une instance, par exemple en lui ajoutant des attributs. Cette instance peut alors à nouveau être classifiée. Ce second mode d'interaction est le plus complexe, car il fait intervenir des instances existantes. En utilisant ici encore le mécanisme de règles proposé par CLASSIC, il est possible de lier étroitement analyse de bas niveau et raisonnement.
Dans l'exemple du magazine de cinéma cité plus haut, une première étape consiste à exécuter un algorithme de segmentation en plans du document (premier mode d'interaction) des instances de la classe SHOT, concept primitif héritant de TEMPORAL. Dans une seconde étape, les plans d'une séquence d'extraits sont définis ainsi :

```
(define-concept 'TRAILER-SHOT '(and
    'SHOT
    (black-strip? begin-time end-time)))
```

black-strip? étant une fonction Lisp à deux paramètres calculant la présence de bandes noires dans le document entre deux dates. L'algorithme est donc déclenché automatiquement par le processus de classification (second mode d'interaction).
Les plans instances de TRAILER-SHOT sont alors regroupés par la règle de type (2) :
TRAILER-SHOT {*meets* ∨ *overlaps*} → TS-SEQ
avec TS-SEQ concept héritant de TEMPORAL-SEQ. Enfin, un algorithme d'extraction de texte peut être exécuté sur le début de chacune des séquences d'extraits instances de TS-SEQ.
Ainsi, un schéma général se dégage où alternent des phases de classification et de regroupement avec des phases d'analyse de bas niveau.

3. Discussion : reconnaissance de plan

Les règles de regroupement présentées ci-dessus constituent d'une certaine manière des *plans*, dans le sens de la reconnaissance de plan ou de scénario. Plusieurs travaux ont déjà été conduits qui utilisent le raisonnement terminologique dans le cadre de la reconnaissance de plan : par exemple, [18] propose d'étendre la notion de subsomption aux plans, et [19] définit un langage unifié de raisonnement terminologique et temporel. Dans le même ordre d'idées que [18], [20] propose d'organiser les plans en taxonomie en intégrant à un système de raisonnement terminologique (en l'occurrence CLASSIC) des techniques connues de planification utilisant des automates d'états finis. Une telle approche présente l'avantage d'offrir une formalisation claire du type de raisonnement mis en œuvre, ce qui permet d'avoir une bonne évaluation des types de problèmes pouvant être abordés ainsi que des limites du formalisme. En particulier, les plans qui peuvent être représentés dans [20] sont restreints à des expressions régulières, ce qui est trop peu expressif pour décrire le contenu de documents filmiques. Nous envisageons cependant de nous inspirer de cette démarche en formalisant d'avantage le raisonnement temporel à l'extérieur du système de représentation.

4. Expérimentations en cours

L'implémentation du système décrit dans ce papier est en cours. Nous utilisons la version 2.3 du système CLASSIC déjà mentionné dans l'environnement CLISP. Les principaux mécanismes de regroupement temporels ont été mis en œuvre ; des tests de plus grande envergure sur un

corpus documenté doivent être engagés. notamment dans le contexte du projet DIVAN.

Conclusion

L'indexation de documents filmiques est un sujet de recherche dont les enjeux sont considérables. La nature de la diffusion étant en train de changer considérablement, il sera de plus en plus important de pouvoir accéder à des bases de documents gigantesques, posant ainsi le problème de l'annotation dans un cadre nouveau. Nous pensons que la combinaison d'expertises de champs complémentaires est nécessaire pour obtenir des systèmes efficaces et opérationnels. Ceci nécessite d'une part la formalisation des divers types de documents existants, ce qui permet de restreindre le champ de l'analyse, et d'autre part la collaboration des techniques d'analyse « bas niveau » du traitement d'image et de l'analyse de signal d'une part et des techniques de représentation et de raisonnement de plus haut niveau d'autre part. Nous proposons dans ce papier un élément de réponse fondé sur l'exploitation du formalisme des logiques de descriptions qui nous paraît prometteuse.

Bibliographie

1. Poncin, P., *Audiovisuel : vers le tout numérique.* Les dossiers de l'audiovisuel, 1997. **74**.
2. Pinhanez, C.S., Bobick, A.F. *Approximate World Models: Incorporating Qualitative and Linguistic Information into Vision Systems*. in *AAAI 96*. 1996.
3. Herzog, G. *Utilizing Interval-Based Event Representations for Incremental High-Level Scene Analysis*. in *Proc. of the 4th International Workshop on Semantics of Time, Space, and Movement and Spatio-Temporal Reasoning*. 1992. Château de Bonas, France.
4. Katz, S.D., *Film Directing Shot by Shot*. 1991: Michael Wiese Production.
5. INA, *Glossaire de l'image et du son*, 1997, Collection techniques et production audiovisuelles, Institut National de l'Audiovisuel.
6. Yeo, B.-L., Liu, B., *Rapid Scene Analysis on Compressed Video.* IEEE Transactions on Circuits and Systems for Video Technology, 1995. **5**(6): p. 533-544.
7. Aigrain, P., Joly, P., Longueville, V., *Medium Knowledge-Based Macro-Segmentation of Video into Sequences*, in *Intelligent Multimedia Information Retrieval*, A.P.M. Press, Editor. 1997.
8. Merlino, A., Morey, D., Maybury, M. *Broadcast News Navigation using Story Segmentation*. in *Proceedings of the Fifth ACM International Conference*. 1997. Seattle, Washington, USA.
9. Özsu, M.T., Szafron, D., El-Medani, G., Vittal, C., *An Object-Oriented Multimedia Database for a News-on-Demand Application.* Multimedia Systems, 1995. **3**(5-6): p. 182-1995.
10. Ronfard, R. *Shot-level description and matching of video content*. in *SPIE 97*. 1997. San Diego.
11. Napoli, A., *Une introduction aux logiques de description*, 1997, Rapport de recherche n 3314, INRIA.
12. Borgida, A., Brachman, R.J., McGuiness, D.L., Resnick, L.A. *CLASSIC: A Structural Data Model for Objects*. in *ACM SIGMOD Int. Conf. on Management of Data*. 1989.
13. Nebel, B., *Reasoning and Revision in Hybrid Representation Systems.* LNAI, 1990. **422**.
14. Pinhanez, C., Bobick, A., *PNF Calculus : A Representation and a Fast Algorithm for Recognition of Temporal Structure*, 1996, Technical Report n 389, MIT Media Lab Perceptual Computing Section.
15. Allen, J.F., *Towards a general theory of action and time.* Artificial Intelligence, 1984. **23**(2): p. 123-154.
16. van Beek, P. *Reasoning about qualitative temporal information*. in *Proceedings of AAAI'90*. 1990.
17. Nebel, B., Bückert, H.J. *Reasoning about Temporal Relations: A Maximal Tractable Subclass of Allen's Interval Algebra*. in *Proceedings of AAAI'94*. 1994. Seattle, Washington.
18. Weida, R., Litman, D. *Terminological Reasoning with Constraint Networks and an Application to Plan Recognition*. in *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning (KR'92)*. 1992. Cambridge, Massachussetts.
19. Artale, A., Franconi, E. *A Computational Account for Description Logic of Time and Action*. in *Proc of the 4th International Conference on Principles in Knowledge Representation and Reasoning (KR94)*. 1994.
20. Devanbu, P.T., Litman, D., *Taxonomic Plan Reasoning.* Artificial Intelligence, 1996. **84**: p. 1-35.

Audiovisual-based Hypermedia Authoring: using structured representations for efficient access to AV documents

Gwendal Auffret, Jean Carrive, Olivier Chevet, Thomas Dechilly,

Rémi Ronfard, Bruno Bachimont

Institut National de l'Audiovisuel (INA)
4, avenue de l'Europe – 94366 Bry-sur-Marne Cedex, France
Tel: +33 1 49 83 20 00
E-mail: aedi@ina.fr

ABSTRACT

In this article we introduce the notion of audiovisual-based hypermedia authoring systems, *i.e.* systems mainly using documents from digital audiovisual (AV) archives as a source for hypermedia authoring. After showing that traditional hypermedia models are ill-designed for specific constraints implied by such systems, we propose a change of approach. We present a model based on formal structured representations of the content of documents as it is done in the field of structured documents. Since a specific model for the representation of AV content is needed, we introduce *Audiovisual Event Description Interface* (AEDI), which provides a model for the description of AV documents, and an XML-based syntax for the exchange of such descriptions. We describe AEDI's main concepts, how it can be related to a formal specification of the domain knowledge — called *ontology* — which allows efficient dynamic hyperlinking among elements. Finally, we describe the implementation of this model for the production of AV based hypermedia at INA's production department.

KEYWORDS: audiovisual, structured documents, hypermedia design, ontology, content indexing.

INTRODUCTION

Many hypertext and hypermedia applications and systems rely mainly on texts and still images to convey information. Audiovisual (AV) — i.e. cinema, TV, radio and video — source documents are quite often only used by hypermedia authors as small extracts linked to text for occasional illustrative purposes.

There are of course many economical and technical reasons for this lack of use. However, we claim that design issues remain some of the most fundamental obstacles. Indeed as [25] and [9] show, the creation of a design model adapted to specific AV constraints remains a challenge for the hypertext community.

In this article, we will describe our research on AV-based hypermedia authoring environments, which bring together concepts of three different domains: digital libraries (our source documents are provided by *INA*, French TV and Radio Archives), hypertext (links are used for browsing and relating AV documents) and multimedia (the intensive use of AV media implies temporal constraints among components). We will analyze current hypermedia models and explain why they remain ill-designed for hypermedia authoring tasks requiring intensive use of AV source documents. We will argue that the nature of AV media, as well as specific constraints applying to users' interaction with large digital AV repositories, calls for the creation of a hypermedia model based on structured representations of the content of source documents.

Our approach is inspired by structured document methodologies [4] and by their adaptation to traditional hypertext concepts [38] [36]. It provides hypermedia authors and/or readers with an efficient means of interpreting, manipulating, browsing, editing, searching and linking segments of documents. Moreover, relating structured representations of documents to a formal specification of the domain knowledge allows the application of inference mechanisms to the documents, as is done in [33], and dynamic linking to be performed among elements of the content as in [37].

We will argue that there is currently no perfectly suited model for the representation of AV documents. Therefore, we will introduce Audiovisual Event Description Interface (AEDI), which provides a description model and an XML-based exchange format for the description of AV documents. Moreover, as describing AV documents requires not only a model and a format, but also a description policy defining what should be described and how. We will show how to relate AEDI descriptors with concepts defined in an ontology [7] and how to use this ontology for dynamic linking of description elements.

Hypertext 99 Darmstadt Germany

Finally, before concluding, we will discuss the implementation of this model in INA's production department.

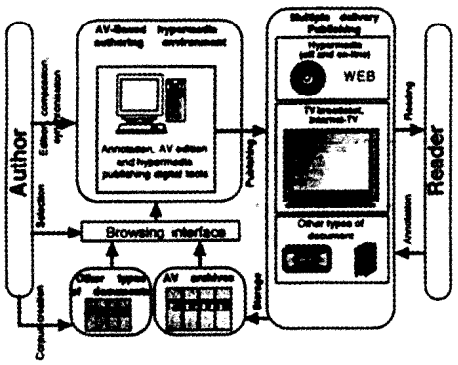


Figure 1: An AV-based hypermedia authoring environment

CONTEXT

As telecommunications, broadcasting and computer industries are converging into a single entity, new applications based on hypermedia concepts and including AV documents on a large scale are appearing: internet-enabled TV, hypermedia versions of TV programs published on CD-ROM, video on demand on cable TV, and remote access to digital AV documents in archives are typical examples.

For one year, INA's research department has been working on the design and development of an *AV-based hypermedia authoring environment*. Such an environment is illustrated in Figure 1. AV directors are provided with an indexed repository containing AV archival documents, AV rushes[1], still images and texts. The indexing is created during the production and archiving process following guidelines. It is stored in digital form and allows dynamic hyperlinking among documents. Using this relevant metadata, authors can select segments of AV source documents relevant to their needs, specify presentation guidelines and create new documents which can be published following diverse delivery specifications (as a broadcast program, a hypermedia application or even a book).

The main objective of this project — which brings together the concepts of the digital library, the hypertext and the multimedia domains — is to show that the digitization of AV archives' and broadcasters' documents modifies traditional AV production processes and leads to the creation of new tools and usages integrating AV documents as a primary resource.

DESIGN REQUIREMENTS

AV-based hypermedia authoring environments call for a design model. This model should conform to some specific requirements, which are listed below:

[1] A "rush" is a piece of AV content filmed by the author, which has not been edited yet. It is the produce of the original camera shooting.

Any AV-based hypermedia model should provide an adequate representation for what librarians and library users traditionally refer to as *document*, *i.e.* the elementary units manipulated by clients for browsing the library content, to interpret it and to create new documents from it. Furuta [18] describes library documents as self-contained units representing an identified intellectual contribution and exhibiting, to a certain extent, an intentional structure. We would add that any element of a document is interpreted by readers not only in terms of the information it contains, but also in terms of its place in the document structure. This is what we call its *source document context*, as opposed to the source context defined by [26], which is in fact the presentation context of a media element in a hypermedia document at runtime. Besides, most of the documents provide a canonical navigation order, or "guided tour" through their content structure (*e.g.* the continuous single timeline for traditional AV documents). All these elements should be represented formally in any model used for AV-based hypermedia model authoring.

Moreover, any AV-based hypermedia model should provide new ways to work efficiently with large collections of AV-documents. Indeed, currently, once a document is identified by using a library catalogue, the selection of relevant segments of its content remains a long, fastidious and little added value process which should be improved as much as possible.

Finally, a way to link efficiently not only *to* but also *from* segments of AV documents should be provided to allow content-based navigation of the AV material.

These are the requirements we have to fulfill. Unfortunately, as we will show in the next section, none of the hypermedia models currently referenced in the literature seems adequate.

Drawbacks of Current Hypermedia Models

Most hypertext and hypermedia models have been designed as static networks of nodes (or components) and links [12]. Nodes contain information (most of the time text and/or still images) and links are created by the author and used by readers to browse this information by "jumping" from one anchor to another.

As [25] or [9] show, this type of approach remains ill-designed for dynamic temporal media such as video or audio recordings. Even a widely accepted and used hypertext model such as the Dexter Model [24] has to be modified in order to add temporal constraints to its scope.

However, we claim that, even customized, the traditional Dexter-like approach is still inadequate for AV-based hypermedia authoring systems. Indeed, to our knowledge, there is nothing exactly equivalent to the notion of source document in current hypertext approaches. Most of the time, the basis of the hypermedia authors' work remains the node and link network, which itself might be partly

structured using, for instance, Dexter's composites[2] as in [19] or [21], or it might even be considered as a structured document as in CMIF [27]. Similarly, the notion of entity[3] in the Hypertext Design Model (HDM) [20] provides a way to structure objects used for the hypermedia authoring process.

However, none of the above models provides a notion corresponding to what librarians and library users consider as documents, i.e. the structured *sources* from which the hypermedia document is authored. AV sources are still considered only as a special type of computer resources (i.e. mostly AV files on disk, without any specific logical structure), used as bricks for building hypermedia documents, which imply fairly complex interaction and synchronization schemes such as the one developed in the Amsterdam Hypermedia Model [25].

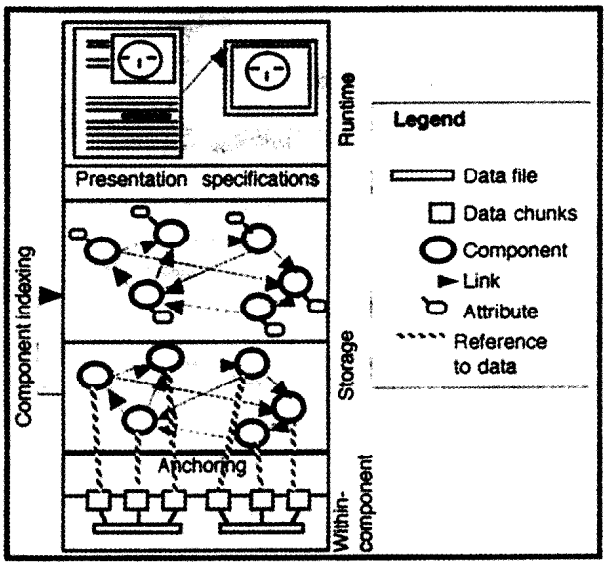


Figure 2: « Dexter-like » approach to hypermedia design

Moreover, classical hypermedia models are not better-suited than current analogue library systems for browsing and manipulating large collections of AV documents for hypermedia authoring purposes. Indeed, most of the time, components containing (or referencing) AV data are created the following way (cf. Figure 2): authors add a new component to the hypermedia network, then they fill in the component by selecting a temporal segment of AV data from an AV document[4], they index the component by typing some textual information in predefined fields (Dexter's "attributes") and, finally, they define presentation specifications controlling the execution of the application at the runtime layer.

This type of interaction is acceptable for authors when they are dealing with only a few seconds of AV data, or when all segments to be integrated in hypermedia components are available as well-identified files on disk, which just have to be imported without being watched. However, it is simply too time-consuming to proceed this way when one deals with hours of AV content! As a consequence, some mean of providing relevant random access to segments of AV documents is needed.

Finally, linking *from* AV content remains a thorny problem in traditional systems. Indeed, anchoring zones inside the AV document are evolving through time: if users play a segment of an AV document, different anchors to different nodes may be accessible at different moments in time. Allowing this type of anchoring in a static component and link network, as it has been done in [25], remains complicated.

As a conclusion, we claim that AV-based hypermedia environments call for a new approach allowing more efficient browsing and manipulation of the content of AV documents.

Representing the Content of AV Documents

One way to avoid most of the problems listed above would be to provide hypermedia authors with tools for automatic searching, retrieving, comparing and linking relevant segments of AV documents that they could then insert into their hypermedia components.

Unfortunately the very nature of the AV medium itself is a problem. Indeed, contrary to text, AV data is not made out of discrete symbolic elements, i.e. elements that can be computed by computers and be interpreted by humans, and therefore it cannot be easily processed. Of course, [3] shows that many research projects from the digital signal processing field (such as [11] or [32]) are working on automatic extraction algorithms. Some applications, such as QBIC [16] or Virage[5], have even become industrial products. However, such applications, though very helpful, remain only analytic tools and cannot replace human synthetic interpretation for the creation of the high level conceptual descriptions which are needed for an efficient manipulation of AV content. Therefore, other projects, such as [14], focus on computer-aided manual annotation of AV content.

In all these projects, the central element is an *intermediate representation* of the AV content which is used as a substitute for the AV document. Indeed, once related to the content by a temporal and spatial linking model, the intermediate representation acts as a *document index*. The changes, manipulations, filters and composition applied to the representation imply equivalent actions on the AV document itself. For example, it is possible to apply research, selection and linking functions to the representation of "Cyrano de Bergerac" in order to gather

[2] A composite component in the Dexter Model is a component containing other components

[3] An HDM entity is defined as an autonomous object composed of components related by structural links.

[4] *i.e.* run a digital AV file in a player, watch and/or listen to the AV document until the begining of the relevant segment and then select the begin time and the duration

[5] See http://www.virage.com

all the segments in which Gerard Depardieu is on screen, in close-up, etc.

This leads us to the conclusion that the crucial challenge of AV-based hypermedia environments is to provide a *representation framework for expressing the content of AV documents*. Many standardization bodies such as ISO MPEG-7[6], the joint EBU-SMPTE task force[7], W3C Metadata[8], the CEN/ISSS[9] or DAVIC[10] are currently addressing this issue by creating large scale initiatives aiming at defining a standard representation for metadata on AV and multimedia content. Prié & *al* [35] show that a generic model for representing (manual as well as automatic) annotations of AV documents is needed. Moreover, we claim that this representation has to be formally structured at the document level to allow easy computer-aided access to relevant parts of the content.

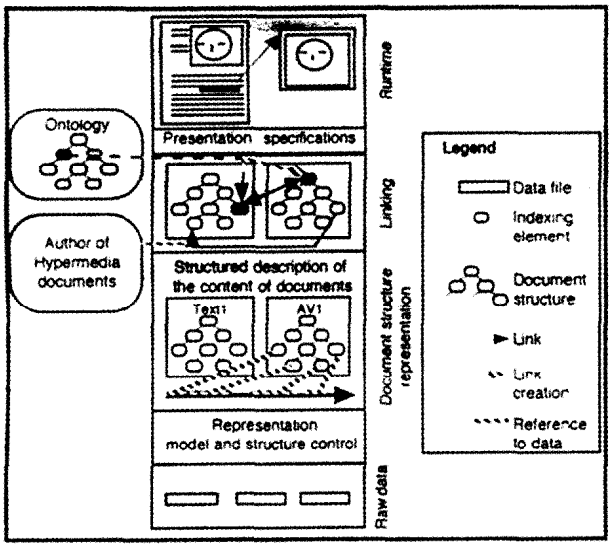


Figure 3: Our approach, based on structured representations of the content of documents

A « Structured Documents » Approach

Our model is based on the concept of structured document [4]. In such a model (cf. Figure 3), any source document available in the archives is referred to by using a *structured representation* of its content. These representations can then be used as a basis for linking. Links among structure elements can of course be created by the author, but they can also be automatically generated. Indeed, if the elements of the structure are related to some formal representation of the domain knowledge, such as an ontology [7] or a semantic network [33], inference mechanisms can be applied for linking the elements related to the same notion in the domain knowledge [10]. Once this interrelated network of structured documents is created, it can be browsed by users as a hypermedia application. This requires some presentation specifications, which build a "style sheet" for the hypermedia. This can be achieved by using a specific language such as SMIL[11].

This type of approach, though not much referenced in the hypertext literature [38], has proved efficient and relevant for electronic publishing tasks [37], for document manipulation and browsing in many industrial applications based on SGML [30]. It meets our requirements since it provides a model for the representation of documents (and thus for the source document context of an AV segment), it allows efficient manipulation of AV documents based on the manipulation of their representation structure and, finally, structure elements can be used as an anchoring system for linking among segments of the AV content.

It should be noted that our notion of structured representation is not restricted to what is traditionally called "logical" or "generic" structure of an SGML-encoded text (*i.e.* the inclusion of the different parts of a text such as subsections into sections, *etc.*). Indeed, the logical structure, though conveying relevant semantic information about the document, is often not informative enough to describe the content and, therefore, it has to be complemented by some other knowledge representations such as anchors related to a semantic network, as in [33]. Moreover, when dealing with AV documents, the notion of logical structure remains limited to the editing elements that can be adequately interpreted for an up-translation[12], *i.e.* shots[13], camera motion and, to a certain extent, scenes[14] [2]. However, trying to relate these AV editing elements with the semantics of the content is extremely difficult. Indeed, AV editing does not provide fairly stabilized traditions as text publishing does. As a consequence, we define a structured representation as *any formal structure representing the content of an AV document in terms of a manipulable semiotic form.* A semiotic form is a representation format which is directly human readable and does not require any machine-based computation and decoding (*e.g.*: characters, images, sketches...etc.). This type of representation is also called "content indexing".

---

[6] MPEG-7 ("Multimedia Content Description"). See http://drogo.cselt.it/mpeg/

[7] The European Broadcasters' Union/ Society of Motion Pictures & Television Engineers task force has created a metadata dictionary which has been published as an international standard in 1998 and should be adopted by the EBU and NATO from 1999 on. See http://www.ebu.ch/

[8] World Wide Web Consortium, see http://www.w3c.org/Metadata/

[9] CEN Information Society Standardization System is currently involved in the Metadata for Multimedia Information initiative (MMI). See. http://www2.echo.lu/oii/en/metadata.html

[10] Digital Audio Visual Council, see http://www.davic.org/

[11] See http://www.w3.org/TR/1998/REC-smil-1998-19980615

[12] An *up-translation* is an automatic translation from a physical appearance to a logical structure, as it is done when interpreting that all 12 points bold character strings in a text are "section title" elements, for instance.

[13] A *shot* is a stream of contiguous frames continuously recorded by a single camera, and a shot cut can be detected with rather good results.

[14] A *scene* is collection of one or more adjoining shots, that has the characteristic of perceptual continuity and semantic homogeneity.

A Model Specific to AV Documents

Having decided upon a structure-based approach, the nature of the formal model to be used for the representation of AV documents remains to be determined. Indeed, however simple they may seem at the surface, traditional (i.e. single-timeline) AV documents are fairly complex objects (elements are related by temporal and/or logical constraints, they overlap easily, they can be interpreted in many different and concurrent ways, etc.) and finding an adequate model for their representation is not easy.

For instance, SGML seems a very good candidate to encode document structure, but it cannot be used by itself and need some customization to be applied to AV documents. Indeed, describing AV data requires complex data types (such as those extracted by signal analysis algorithms) and a temporal and spatial model to relate the description to the content. Moreover, if SGML structures allow efficient representation of tree-structures such as segmentations in scenes and shots (cf. Fig. 4), they remain relatively ill-designed for describing overlapping stratification approaches (cf. Fig. 5) such as [1] or [15].

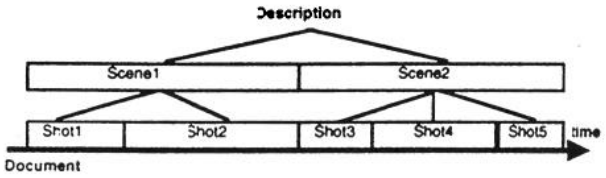


Figure 4: Segmentation

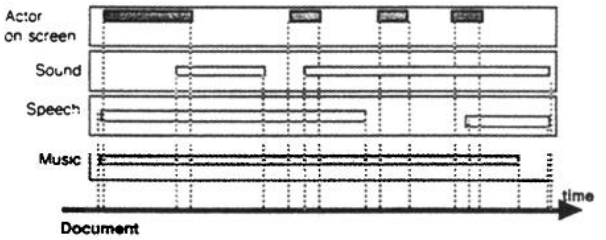


Figure 5: Stratification

On the other hand, HyTime [31] could seem a good candidate. However, it has been designed for the encoding of any linking between chunks of a hypermedia document. These chunks can be expressed in any user defined coordinate space. Therefore, HyTime is to open and not focused enough on AV document for our needs. Indeed, the efficient use of AV descriptions requires a specific model for representing logical, spatial and temporal constraints among segments of a document.

For the same reason, we cannot rely on very wide frameworks such as RDF[15], intended for the encoding of metadata on any type of computer resource.

Finally, languages or models such as SMIL, MHEG [34], PREMO [29] or HyTime's presentation module are intended for the synchronization of multimedia elements for *presentation* and *interaction*. They are concerned with the way the documents are presented on screen, which is useful at the presentation specification level (cf. Fig. 3). However, a model for the *representation* of AV-documents remains to be defined.

Since none of the above models and languages fulfills our needs, we believe that a specific representation model for continuous temporal media has to be defined[16] and, as a consequence, in the next section, we will introduce *Audiovisual Event Description Interface* (AEDI), the formal representation model we developed at *INA*.

OVERVIEW OF AEDI

Audiovisual Event Description Interface (AEDI) combines a representation model and an application and platform independent XML-based exchange format for the description of AV documents. It is being used in AV indexing research projects and will be submitted to MPEG-7 standardization in February 1999 [5]. It is currently being tested by scholars to produce, exchange and re-use annotations on AV documents using the French Legal Deposit's Computer-Aided AV Reading Station. This prototype, called *Médiascope*, proposes a graphical interface for the annotation of AV documents. some automatic segmentation facilities and an export function which outputs descriptions as AEDI text files.

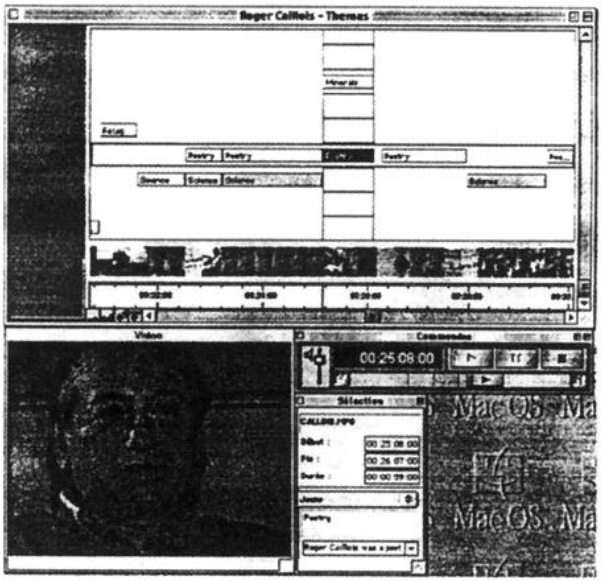


Figure 6: Representing thematic annotations with AEDI in Médiascope

Figure 6 shows how Médiascope can be used to annotate and browse thematic strata on a documentary: when clicking on a segment related to "poetry", the user visualizes the inclusion of this particular segment in a "theme = poetry" stratum (horizontal scale) and other segments available on the same temporal segment

[15] see http://www.w3.org/TR/WD-rdf-syntax

[16] One could argue that such formats already exist : Open Media Framework from AVID Technology Inc. (cf. http://www.avid.com/) for instance, is used in the professional world. But these formats are copyrighted, they depend on software from a specific company and therefore are never widely accepted.

(vertical scale): here, for instance, the theme of the selected temporal segment is related to "minerals" as well.

Temporal Model

An AEDI description is a graph containing description objects. These objects (called *segment*, *stratum* or *A-Temporal Entity (ATE)*) are organized in a tree-structure and may be related one to each other using reference links (cf. Fig. 7). They refer to a single AV document[17], which is defined in the following way: a document is *a continuous and contiguous timeline identified by a unique identifier.*

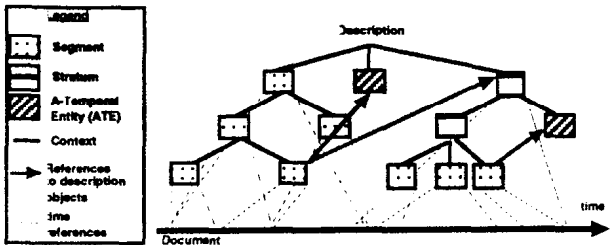


Figure 7: AEDI graph structure for the description of a single timeline AV document

However restrictive this definition may be (in many cases, such as hypermedia documents, a single temporal timeline is not adequate to describe a document), it is efficient. Indeed, it fits most of past and current AV productions, and it allows efficient and low-cost description generation.

A projection mechanism inserted into the description (called *timespace)* allows the user to project the description upon the document "virtual" timeline and, then, to project the document timeline itself upon media instances of the document (*cf.* Fig. 8).

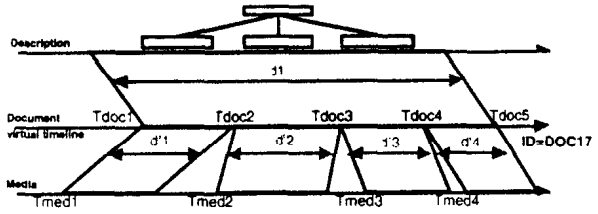


Figure 8: Using the document timeline abstraction in AEDI

AEDI-aware browsers determine the document identifier in the description and then look for a service able to relate this identifier to some chunks of media. This double projection mechanism is a way to allow more than one description of the same document, which is important, especially for user-centered annotation purposes. Moreover it enables the description to play documents that are specified on more than one chunk of media (*e.g.*: a single document can be divided among several files) without forcing users to get involved with files and media management systems. Finally it enables archives to change the media used for storage without any impact on user's interaction with the library system.

Description elements

AEDI defines various description elements, which are represented in Fig.9.

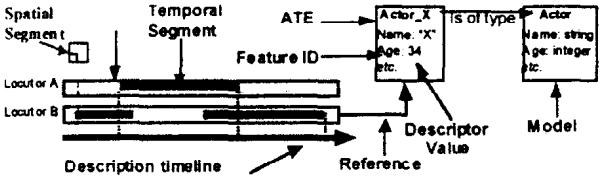


Figure 9:AEDI description elements

An *ATE (A Temporal Element)* is any non-temporal object used in a description. ATEs are defined independently of temporal objects but can be referenced for their description. A typical example of an ATE is the description of a given actor in a film in terms of name, date of birth, filmography, *etc.* which can be referenced as a characteristic of a shot.

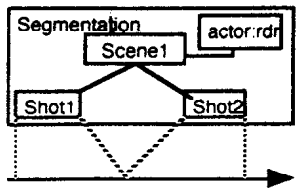
A *segment* is the definition of an actual part of an AV document, it may be temporal or spatial. A segment is associated with a locator which defines the type of object described: a temporal segment or a spatial region.

- A temporal segment has at least a begin time and a duration. The description of a shot or a scene, for instance, can be represented as a temporal segment. A tree-like structure of temporally located segments can be used to build a *segmentation* (cf. Figure 4) of a document.
- A spatial segment delimits a spatial region as a bounding box (rectangle) defined by positions relative to the side of the image (position of the top left corner and height and width).

A *stratum* is a logical object which holds a collection of description elements (segment, stratum or ATEs) for a certain purpose. A stratum constitutes the expression of a point of view on the AV document. It allows a *stratification* approach (cf. Figure 5) of the document. The description of the collection of all the segments related to a subject (such as, for instance, the set of all speeches of a character in a theater play, or the results of a specific automatic analysis algorithm) can be represented as a stratum.

Description elements are characterized by *properties.* Properties are associations of an attribute with a typed value. AEDI provides some basic types (such as integer, character string, float, boolean, time reference, date, etc.) and allows the definition of new types by users. It is possible to define properties such as "actor-on-screen" of type "string", or "color" of type "RGB", where "RGB" is defined as a structure of 3 integers, and so forth. Once a property is defined, it can be used to characterize any AEDI description element.

[17] We are working on multiple documents descriptions, but the current version is restricted to single document descriptions. However, descriptions can be linked and related into metadata repositories.



DTD

```
<!ENTITY % AEDIent SYSTEM "aedi.ent">
%AEDIent;

<!ELEMENT scene (segment, story,
actor_link*, shot+)>

<!ELEMENT story (#PCDATA)>
<!ATTLIST story
        aedi CDATA #FIXED "property"
        basetype CDATA FIXED "string">

<!ELEMENT actor_link EMPTY>
<!ATTLIST actor_link
        aedi CDATA #FIXED "intlink"
        reference IDREF #REQUIRED
        refersto CDATA #FIXED "actor">

<!ELEMENT actor (ate, name)>
<!ATTLIST actor
        id ID #REQUIRED>

<!ELEMENT name (#PCDATA)>
<!ATTLIST name
        aedi CDATA #FIXED "property"
        basetype CDATA FIXED "string">
<!ELEMENT shot (segment, begin_effect,
end_effect)>

...etc.
```

Instance

```
<?xml version="1.0" encoding="UTF-8"?>
<segmentation>
<timespace>
<documentID>H324</documentID>
<projection timebase="internal">
<projtuple>
<internal_timeref>00:00:00:00</internal_timeref>
<external_timeref>00:00:30:00</external_timeref>
<internal_duration>00:00:05:34</internal_duration>
    <external_duration>00:00:05:34</external_duration>
</projtuple>
</projection>
</timespace>

<actor id="RDN">
<ate/>
<name>Robert de Niro<name>
</actor>

<scene id="scene1">
<segment>
<begin>00:00:02:30c</begin>
<end>00:00:04:00</end>
</segment>
<story>The hero steps into the
room</story>
<actor_link reference="RDN">
<shot id="shot1">
<segment>
<begin>00:00:02:30</begin>
<end> 00:00:03:00</end>
</segment>
<begin_effect>cut</begin_effect>
<end_effect>cut</end_effect>
</shot>

<shot id="shot2">
<segment>
<begin>00:00:03:00</begin>
<end>00:00:03:20</end>
</segment>
<begin_effect>cut</begin_effect>
<end_effect>dissolve</end_effect>
</shot>
</scene>
...etc.
</segmentation>
```

Figure 10:Using XML to encode an AEDI description

Besides, a description element may be an instance of a *model*. A model defines, in a way, the "interface" implemented by a description element. More precisely, an AEDI model defines the content model of a description object[18], *i.e.* which properties can, must or may be instantiated in the entity as well as the potential links from this entity to others. For example, a model for entities called "shot" could specify that these entities *must* contain a property "cast" of type "list of string", which provides the list of actors on screen, and *might* contain a property "camera-angle".

Finally, a description element can reference (*i.e.* be linked to) another one. For instance, a stratum "locutor" can be linked to a ATE of model "actor". References can be named and typed: it is possible to specify to which model of an object a reference is pointing.

Structure encoding using XML

Being able to define a structured content model for a description is crucial for archives since descriptions are produced, following systematic guidelines: descriptions of AV document are structured documents. Therefore, we decided to use XML as an exchange format for AEDI descriptions. Indeed, XML provides efficient structure control through DTDs. Moreover, it provides a common format for descriptions of AV documents and textual documents such as TV newspapers, scripts or elements of the production file. Indeed, AV archives store not only AV data but also texts[19] which are extremely useful for the interpretation and the manipulation of AV documents and are potential hypermedia authoring sources. Using the same format for AV descriptions and text allows easy and generic linking and manipulation of data, which is generally considered as heterogeneous.

From an XML point of view, AEDI data types are expressed as attributes [8] and AEDI elements (segments, strata and ATEs) are stored in a parameter entity which is inserted in the DTD.

The insertion of XML elements is also used to control AEDI logical, spatial and temporal constraints. For example, if a segment S2 is included in segment S1 in the DTD, this means that the temporal and/or spatial borders of the instances of S2 should be included in the borders of the parent instance of S1. It is also possible to store the temporal and spatial references as external links. In such a case, our format can be considered as an XML flavor of HyTime using a single standard coordinate space for time and space.

From the XML description, the AEDI parser can build a tree-structure such as the one shown on Figure 7, which is then used for the manipulation of the AV document.

Relating descriptors to an ontology

AEDI properties and models define what we call *descriptors*, *i.e.* objects used to relate a concept and an

[18] AEDI content models are equivalent to SGML DTDs.

[19] French legal deposit receives 50.000 pages a year of such textual documents.

element of a document. We propose the use of an *ontology* to define the semantics of descriptors. An ontology is traditionally defined as the "specification of a conceptualization" [22], [23] and is used to represent the concepts associated with a domain. In this article, we will follow the definitions provided by [6], *i.e.* descriptors are linguistic terms[20]. The semantics of these descriptors is defined by the location of the terms in an "is-a" tree. Once the terms used in the domain are defined by experts using this tree structure, they can be used as primitives for any representation or encoding of the domain knowledge. This conforms to the ontologies as they are commonly defined in the literature [28]. In the end, each notion is represented by a term and defined in terms of its:

- "interpretative semantics": linguistic specification of the meaning and the knowledge content that corresponds to the symbol for a domain specialist. At this level, a symbol is understood as a term.
- "formal semantics": formal specification of the representation format and/or the possible values for the symbol. At this level, a symbol is understood as a logical predicate or function;
- "computational semantics": specification of the computational behavior of the symbol. At this level, a symbol is understood as an object to which messages can be sent so that it can exhibit some computational behavior.

An ontology of descriptors (or a semantic network) is a crucial tool for sharing descriptions of AV documents among members of a specific community. As a consequence, we are currently working on the creation of an ontology, which would allow the explicit representation of indexing methodologies used by the community of AV archivers. This ontology should also be able to represent the descriptors created during the production process (shooting, editing, broadcasting) as well as extracted by state of the art signal analysis algorithms.

However, this ontology will not be inserted into the AEDI model as a built-in feature. Indeed, efficient use of metadata requires agreement on a common ontology among users of specific community and different communities use different ontologies. Moreover, building a "one size fits all" ontology for describing AV content is probably impossible (cf. [13]) since as communities evolve, their ontologies also evolve. Communities adapt tools to their needs, so they would deviate from the use of a closed ontology and this would lead to ambiguity. Therefore, as we want AEDI to be as widely usable as possible, we provide specific links to relate descriptors to their ontological definition. Users are then free to organize their own ontological namespace and to declare their descriptors as they wish.

[20] In some contexts, especially signal processing analysis, descriptors might be numeric values, or even images or icons as in [15]

Dynamic hyperlinking

Our AV-based hypermedia authoring environment is based on a database containing our ontology as well as XML-encoded texts and AEDI descriptions of AV documents. These have been created following controlled vocabulary guidelines: the tags are terms related to concepts in the ontology.

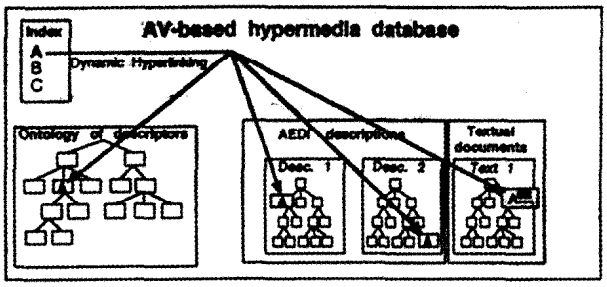


Figure 11: Dynamic linking in an indexed repository.

To help authors browse this primary resource and build up their hypermedia application, we provide dynamic linking among elements of the database. As they are all expressed using XML, documents of any kind are computable the same way. As shown on Figure 11, the application can thus dynamically link elements of any kind as long as they are indexed by the same concepts and constitute a link base similar to the ones developed in [17]. Authors can browse the base by taking as an entry point the ontology (*e.g.*: see all the elements indexed by a particular term) or any document, which becomes, in a way, a table of contents for the others. Moreover, as the linking is done dynamically, documents can be added and retrieved from the repository without any major consequences for the hyperlink navigation facilities.

Starting from this hyperlinked repository, authors can define presentation specifications by specifying a layout for the appearance of the information on the screen, which can then be applied as a style-sheet.

DISCUSSION AND FUTURE WORK

In this section we describe the current implementation of our model at INA's production department and we discuss the future of AEDI.

Implementation issues

INA's production department is dedicated to the production of experimental programs for broadcasters. Since 1998, it has set up a multiple delivery production studio which aims at creating new types of programs based on indexed repositories of AV documents and in which the model described in this article is currently being implemented.

Two initial prototypes have already proven the feasibility of the concept and contracts have been signed with French broadcasters for the production of experimental programs that could be "webcasted". For each of these prototypes, an indexing policy and presentation specifications have

been defined and hypermedia applications have been created by applying generic dynamic linking among elements based on the ontology. As a consequence, any properly indexed program can be processed the same way to create a new hyperdocument.

Users of these hyperdocuments can watch the program and the rushes used to create it. They can browse it and search it following transcriptions issued by the producer, descriptions and annotations issued by documentalists. Finally, they can put their own bookmarks on the document and come back to what they liked or even edit a new AV document based on their bookmarks. The whole process is based on manipulation of AV document descriptions and, in the end, is applied to the content.

Future Work

In the future, we plan to test the robustness and the genericity of our model by confronting it with real life multiple delivery productions. In particular, we shall develop interfaces such as the one described in [27] for the interaction with document descriptions. Indeed, AV directors are not computer specialists and their adaptation to hypermedia authoring implies user friendly tools.

Moreover, we intend to develop this model towards the specification of future hypermedia interfaces for accessing and browsing digital AV archives. The development of such systems should allow better access to archival documents and thus better re-use of available AV data in AV, hypermedia and multimedia productions.

CONCLUSION

In this article, we described our model for AV-based hypermedia authoring environments. We focused on the central role of document content representations for the efficient browsing and manipulation of large repositories of digital AV documents. Then we introduced *Audiovisual Event Description Interface* (AEDI), our model for the description of AV documents. We described AEDI temporal model, description elements and structure control facilities, then showed how relating descriptors to an ontology of the domain knowledge could help in creating links dynamically and provide better browsing facilities.

As a conclusion, we would like to stress the fact that the digitization of AV archives provides an important opportunity for the domains of digital libraries, hypertexts and multimedia. Indeed, until now, scientists from these communities have built very powerful models, but these have not yet been combined. Digital AV archives, and the opportunity that they represent for hypermedia authors, provide an adequate context for this convergence. Nevertheless, adequate models remain to be defined to allow future developments. We believe that our view of AV-based hypermedia authoring environments provides an efficient framework for such models.

REFERENCES

[1] T. G. Aguierre-Smith and G. Davenport, *The stratification system, a design environment for random access video, Proceedings Network and Operating System Support for Digital Video and Audio - 3d International Workshop*, La Jolla, 1992.

[2] P. Aigrain, P. Joly and V. Longueville, *Medium knowledge based macro-segmentation of video into sequences, International Joint Conference on Artificial Intelligence 95*, Maybury Ed., Montréal, 1995.

[3] P. Aigrain, D. Petkovic and H. J. Zhang, *Content-based representation and retrieval of visual media: a state-of-the-art review, Multimedia Tools and Applications special issue on representation and retrieval of visual media*, 1996.

[4] J. André, R. Furuta and V. Quint, *Structured Documents*, Cambridge University Press, 1989.

[5] G. Auffret, R. Ronfard and B. Bachimont, *Proposal for a minimal MPEG-7 DDL for temporal media*, ISO MPEG contribution 3782, Dublin Meeting, 1998.

[6] B. Bachimont, *MPEG-7 and Ontologies: an editorial perspective, Virtual Systems and Multimedia 98*, Gifu, Japan, 1998.

[7] J. Bouaud, B. Bachimont, J. Charlet and P. Zweigenbaum, *Methodological principles for structuring an ``ontology''. Proceedings of the IJCAI'95 Workshop on ``Basic Ontological Issues in Knowledge Sharing.''*, 1995.

[8] T. Bray, *Adding Strong Data Typing to SGML and XML*, W3C Report (1996).

[9] M. C. Buchanan and P. T. Zellweger, *Specifying Temporal Behavior in Hypermedia Documents, Proceedings of the ACM ECHT conference*, 1992.

[10] J. Carrive, F. Pachet and R. Ronfard, *Using Description Logics for Indexing Audiovisual Documents., International Workshop on Description Logics (DL'98)*, Trento, Italy, 1998, pp. 116-120.

[11] S.-F. Chang, J. R. Smith, M. Beigi and A. Benitez, *Visual Information Retrieval From Large Distributed Online Repositories*, Communications of the ACM, 40/12 (1997), pp. 63-71.

[12] J. Conklin, *Hypertext: an introduction and survey*, IEEE Computer, 20 (1987), pp. 17-41.

[13] D. Connolly, R. Khare and A. Rifkin, *XML principles, Tools and techniques, World wide web journal*, O'Reilly & Associates, 1997.

[14] J. M. Corridoni, A. d. Bimbo, D. Lucarella and H. Wenxue, *Multi-perspective Navigation of movies*, Journal of Vision Languages and Computing, 7 (1996), pp. 445-466.

[15] M. Davis, *Media Streams : an iconic visual language for video annotation*, *Proceedings of the 1993 IEEE Symposium on visual languages*, 1993, pp. 196-203.

[16] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele and P. Yanker, *Query by image and video content : The QBIC system*, IEEE Comput., 28/9 (1995), pp. 23-32.

[17] A. M. Fountain, W. Hall, I. Heath and H. C. Davis, *MICROCOSM: an Open Model for Hypermedia With Dynamic Linking*, in A. Ritz, N. Streitz and J. André, eds., *Hypertext: Concepts, Systems and Applications. Proceedings of the ECHT conference*, 1990.

[18] R. Furuta, *What can Digital Libraries teach us about hypertext?*, SIGLINK newsletter, 6/3 (1997), pp. 7-9.

[19] F. Garzotto, L. Mainetti and P. Paolini, *Adding Multimedia Collections to the Dexter Model*, *Proceedings of the ECHT conference*, 1994.

[20] F. Garzotto, P. Paolini and D. Schwabe, *HDM - A Model-Based Approach to Hypertext Application Design*, ACM Transactions on Information Systems, 11 (1993), pp. 1-26.

[21] K. Grønbæk, *Composites in a Dexter-Based Hypermedia Framework*, *Proceedings of the ECHT conference*, 1994.

[22] T. Gruber, *Toward principles for the design of ontologies used for knowledge sharing*, International Journal of Human-Computer Studies, 43 (1995), pp. 907-928.

[23] N. Guarino, *Formal ontology, conceptual analysis and knowledge representation*, International Journal of Human-Computer Studies, 43 (1995), pp. 625-640.

[24] F. Halasz and M. Schwartz, *The Dexter hypertext reference model*, Communication of the ACM, 37 (1994), pp. 30-39.

[25] L. Hardman, D. C. A. Bulterman and G. V. Rossum, *The Amsterdam Hypermedia Model: Adding Time and Context to the Dexter Model*, Communications of the ACM, 37/2 (1994), pp. 50-62.

[26] L. Hardman, D. C. A. Bulterman and G. v. Rossum, *Links in hypermedia: the Requirement for Context*, *Proceedings of ACM Hypertext 1993*, 1993.

[27] L. Hardman, G. V. Rossum and D. C. A. Bulterman, *Structured Multimedia Authoring*, *Proceedings of the ACM Multimedia conference 93*, 1993.

[28] G. v. Heijst, A. T. Schreiber and B. J. Wielinga, *Using Explicit Ontologies in KBS development*, International Journal of Human-Computer Studies, 46(2/3) (1997), pp. 183-292.

[29] I. Herman, G. S. Carson, J. Davy, P. t. Hagen, D. A. Duce, W. T. Hewitt, K. Kansy, B. J. Lurvey, R. Puk, G. J. Reynolds and H. Stenzel, *PREMO: An ISO standard for a presentation environment for multimedia objects*, *Proceedings of the ACM Multimedia'94 Conference*, ACM Press, 1994.

[30] ISO, *ISO 8879:1986 Information processing - Text and office systems - Standard Generalized Markup Language (SGML)*, ISO, Geneva, 1986.

[31] ISO, *ISO 10744:1992, Information technology - Hypermedia/Time based Structuring Language (HyTime)*, ISO, Geneva, 1992.

[32] R. Leinhart, S. Pfeiffer and W. Effelsberg, *Video Abstracting*, Communications of the ACM, 40/12 (1997), pp. 55-62.

[33] J. Nanard and M. Nanard, *Adding macroscopic semantics to anchors in knowledge-based hypertext*, Int. J. Human-Computer Studies, 43 (1995), pp. 363-382.

[34] R. Price, *MHEG: An Introduction to the future International Standard for Hypermedia Object Interchange*, *Proceedings of the conference on Multimedia '93*, 1993, pp. 121-128.

[35] Y. Prié, A. Mille and J. Pinon, *AI-STRATA: A User-centered Model for Content-based description and Retrieval of Audiovisual Sequences*, *First Int. Advanced Multimedia Content Processing Conf.*, Osaka, 1998.

[36] V. Quint and I. Vatton, *Combining Hypertext and Structured Documents in Grif*, *Proceedings of ECHT'92*, 1992.

[37] V. Quint and I. e. n. Vatton, *Making structured documents active*, Electronic Publishing, 7 (1194).

[38] A.-M. Vercoustre, *Structured Editing - Hypertext Approach: cooperation and complementarity*, *Proceedings of the International Conference on Electonic Publishing*, Cambridge University Press, 1990, pp. 65-78.

From Video Shot Clustering to Sequence Segmentation

Emmanuel Veneau, Rémi Ronfard
Institut National de l'Audiovisuel
4, avenue de l'Europe
94366 Bry-sur-Marne cedex, France
{eveneau,rronfard}@ina.fr

Patrick Bouthemy
IRISA/INRIA
Campus Universitaire de Beaulieu
35042 Rennes cedex, France
bouthemy@irisa.fr

Abstract

Segmenting video documents into sequences from elementary shots to supply an appropriate higher level description of the video is a challenging task. This paper presents a two-stage method. First, we build a binary agglomerative hierarchical time-constrained shot clustering. Second, based on the cophenetic criterion, *a* breaking distance *between shots is computed to detect sequence changes. Various options are implemented and compared. Real experiments have proved that the proposed criterion can be efficiently used to achieve appropriate segmentation into sequences.*

1 Introduction

Browsing and querying data in video documents requires to first extract and organize information from the audio and video tracks. The first step in building a structured description is to segment the video document into elementary shots which are usually defined as the smallest continuous units of a video document. Numerous methods for shot segmentation have been proposed (e.g., see [3]). Nevertheless, shots are often not the relevant level to describe pertinent events, and are too numerous to enable efficient indexing or browsing.

The grouping of shots into higher-level segments has been investigated through various methods which can be gathered into three main families. The first one is based on the principle of the Scene Transition Graph (STG) [9], which can be formulated in a continuous way [7], or according to alternate versions [4]. Methods of the second family [1, 2] use explicit models of video documents or rules related to editing techniques and film theory. In the third family [5, 8], emphasis is put on the joint use of features extracted from audio, video and textual information. These methods achieve shot grouping more or less through a combination of the segmentations performed for each track.

We present a method based on a so-called *cophenetic criterion* which belongs to the first family. The sequel is organized as follows. Section 2 describes our method involving an agglomerative binary hierarchy and the use of the cophenetic matrix. Section 3 specifies the various options we have implemented with respect to extracted features, distance between features, hierarchy updating, and temporal constraints. Experimental results are reported in Section 4, and Section 5 contains concluding remarks.

2 Binary hierarchy for describing shot similarity

We assume that a segmentation of the video into shots is available, where each shot is represented by one or more extracted keyframes. The information representing a shot (except its duration) is given by the (average) signature computed from the corresponding keyframes. We build a spatio-temporal evaluation of shot similarity through a binary agglomerative hierarchical time-constrained clustering.

2.1 Binary agglomerative hierarchical time-constrained clustering

To build a hierarchy following usual methods [10], we need to define a similarity measure s between shots, and a distance between shot clusters, called index of dissimilarity δ. The temporal constraint, as defined in [9], involves a temporal distance d_t. We introduce a temporal weighting function W accounting for a general model for the temporal constraint. The formal definitions of all these functions will be given in Section 3. The time-constrained distance $\tilde{d}$ between shots is defined (assuming that similarity is normalized between 0 and 100) by :

$$\tilde{d}(i,j) = \begin{cases} 100 - s(i,j) \times W(i,j) & \text{if } d_t(i,j) \leq \Delta T \\ \infty & \text{otherwise} \end{cases} \quad (1)$$

0-7695-0750-6/00 $10.00 © 2000 IEEE

where i and j designate two shots and ΔT is the maximal temporal interval for considering any interaction between shots.

At the beginning of the process, each shot forms a cluster, and the time-constrained dissimilarity index $\tilde{\delta}$ between clusters is then the time-constrained distance $\tilde{d}$ between shots. A symmetric time-constrained $N \times N$ proximity matrix $\tilde{\mathcal{D}} = [\tilde{d}(i,j)]$ is considered [6], using $\tilde{\delta}$ to evaluate the dissimilarity between clusters. The hierarchy is built by merging the two closest clusters at each step. The matrix $\tilde{\mathcal{D}}$ is updated according to the index of dissimilarity $\tilde{\delta}$ to take into account each newly created cluster. This is iterated until the proximity matrix contains only infinite values. The resulting binary time-constrained hierarchy supplies a description of the spatio-temporal proximity of the extracted shots.

2.2 Cophenetic dissimilarity criterion

In [6], another proximity matrix $\mathcal{D}_c$, called *cophenetic* matrix, is proposed to capture the structure of the hierarchy. We will use the time-constrained version $\tilde{\mathcal{D}}_c$ of this matrix to define a criterion for the segmentation of the video into sequences. The *cophenetic* matrix is expressed as $\tilde{\mathcal{D}}_c = [\tilde{d}_c(i,j)]$, where $\tilde{d}_c$ is the so-called *clustering distance* defined by :

$$\tilde{d}_c(i,j) = \max_{p \neq q/(i,j) \in C_p \times C_q} \{\tilde{\delta}(C_p, C_q)\}$$

where $\tilde{\delta}$ is the index of dissimilarity constructed from $\tilde{d}$, and C_p and C_q are two clusters. Assuming that the shot indices follow a temporal order, the *cophenetic* matrix leads to the definition of our criterion for sequence segmentation, called *breaking distance*, calculated between two consecutive shots as : $\tilde{d}_b(i, i+1) = \min_{k \leq i < l} \{\tilde{\mathcal{D}}_c(k,l)\}$.

2.3 Segmentation using the breaking distance

If the breaking distance $\tilde{d}_b$ between consecutive shots exceeds a given threshold τ_c, then a sequence boundary is inserted between these two shots. An example is presented on Fig 1 where two different thresholds to perform segmentation into sequences $\tau_1 = 20$ and $\tau_2 = 45$ are considered. Fig. 2 displays results corresponding to thersholds τ_1 and τ_2.

2.4 Comparison with the STG method

We have formally proved that our method delivers the same segmentation into sequences as the STG method described in [9]. Considering that STG method considers in a binary way inter-shot spacing and implies non-obvious setting of parameters [7], the advantage of our formulation is to smooth the effects of time, in the time-constrained distance, using continuous temporal weighting functions, and to consider a threshold parameter related to sequence segmentation and not to shot clustering. As a consequence, our approach allows one to visualize what the segmentation results are according to the selected threshold value which can then be appropriately tuned by the user. There is no need to rebuild the STG whenever the threshold is changed.

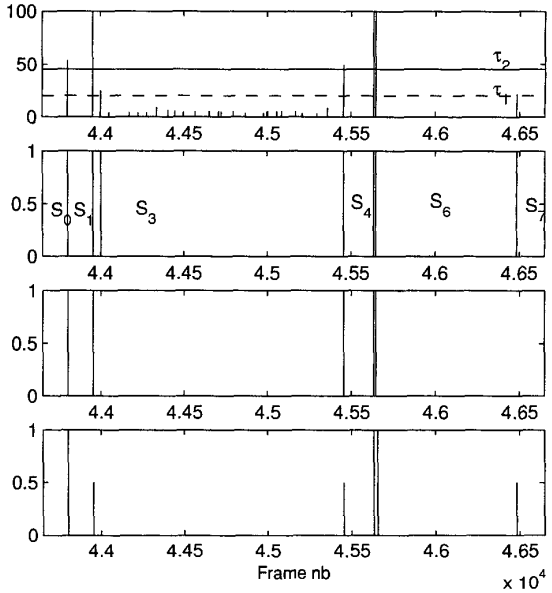


Figure 1. Thresholding the breaking distance values on excerpt 1 of *Avengers* movie (upper row), detected sequence boundaries for τ_1 (upper middle row) and τ_2 (lower middle row), and manual segmentation (lower row)

3 Description of implemented options

3.1 Signatures for shots

We have considered in pratrice three kinds of signatures : shot duration, color and region-based color histograms. Color and region-based color histograms are defined in the (Y, C_b, C_r) space with respectively 16, 4, and 4 levels, and 12 image blocks are considered for region-based histograms. The shot duration gives a relevant information on the rhythm of the action and on the editing work.

3.2 Distances between signatures

Various distances between signatures have been tested. Comparison between histograms can be achieved using his-

togram intersection, euclidian distance, χ_2-distance. The distance chosen between shot durations is the Manhattan distance.

3.3 Updating of the agglomerative binary hierarchy

In order to update the classification hierarchy, two algorithms are available [10] :

- the *Complete Link* method. The index of dissimilarity between clusters is defined by :

$$\tilde{\delta}(C_p, C_q) = \max_{(i,j)\in C_p \times C_q} \{\tilde{d}(i,j)\}$$

- the *Ward's* method. The index of dissimilarity between clusters is given by :

$$\tilde{\delta}(C_p, C_q) = \frac{n_{C_p}.n_{C_q}}{n_{C_p}+n_{C_q}} \tilde{d}(G_{C_p}, G_{C_q})$$

where G_{C_i} is the gravity centre of cluster C_i, n_{C_i} represents either $Cardinal(C_i)$ or $Duration(C_i)$.

In both cases, the Lance and William formula, given by $\tilde{\delta}(A \cup B, C) = a_1\tilde{\delta}(A,C) + a_2\tilde{\delta}(B,C) + a_3\tilde{\delta}(A,B) + a_4|\tilde{\delta}(A,C) - \tilde{\delta}(B,C)|$, is used to update the proximity matrix. We have $a_1 = a_2 = a_4 = \frac{1}{2}$, $a_3 = 0$ for the *Complete Link* method, and $a_1 = \frac{n_A+n_C}{n_{A\cup B}+n_C}$, $a_2 = \frac{n_B+n_C}{n_{A\cup B}+n_C}$, $a_3 = 0$, $a_4 = \frac{n_C}{n_{A\cup B}+n_C}$ for the *Ward's* method.

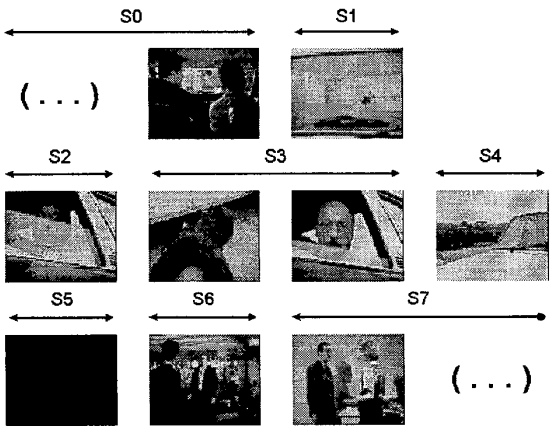


Figure 2. Obtained sequence segmentation on excerpt 1 of *Avengers* movie for threshold τ_1. S_3 is an angle / reverse angle sequence. S_5 is a fade out / fade in effect.

3.4 Temporal weighting function

The temporal weighting function is used to constrain the distance and the index of dissimilarity as introduced in equation 1. In [9], only one type of temporal weighting function was proposed, i.e. rectangular function which is not smooth. We have tested three smooth functions : linear, parabolic, and sinusoidal.

4 Experimental results

We have evaluated our method on a three hour video corpus. We report here results on four excerpts of two minutes. Three excerpts are taken from *Avengers* movies to evaluate the segmentation into sequences in different contexts. The first one comprises an angle / reverse angle editing effect and a transition with a dissolve effect. The second one includes a set change, and the third one involves color and rhythm changes. Obtained segmentations can be compared with a hand segmentation acting as ground truth. In plots displayed in Figures 1, 3 and 4, main sequence changes are represented by a value of 1 and secondary changes by a value of 0.5. The last excerpt is extracted from a news program to test the relevance of the built hierarchy.

Among the implemented options, three sets of descriptors and functions are selected : (O_1) color histograms intersection, rectangular temporal weighting function, and Complete Link method, (O_2) color histograms intersection, parabolic temporal weighting function, and Ward's method based on cluster duration, (O_3) Manhattan distance on shots duration, parabolic weighting function, and Ward's method based on cluster duration.

Results obtained on the news program excerpt show that the clustering distance $\tilde{d}_c$ provides a correct description of the similarity between shots at different levels, even if the information distribution is not homogeneous in the various levels of the hierarchy. An adaptive thresholding applied to breaking distance values would be nevertheless necessary to avoid heterogeneous results. Tests have shown that the best video segmentation into sequences is found using option set O_2.

In the processed excerpts, most of the sequence changes were correctly detected, when the proper options were selected. On Fig.1, we can point out that, using τ_1 and option O_1, all changes are detected with only one false alarm, the angle / reverse angle effect is recognized. Selecting the threshold value is nevertheless a rather critical issue. On excerpt 2, with a relevant threshold, we extract all the correct boundaries with option O_1, with only one false alarm (Fig. 3). Using option O_2 false alarms and missed detections increase on excerpt 2. The color and rhythm changes in excerpt 3 (Fig. 4) have been better detected using option

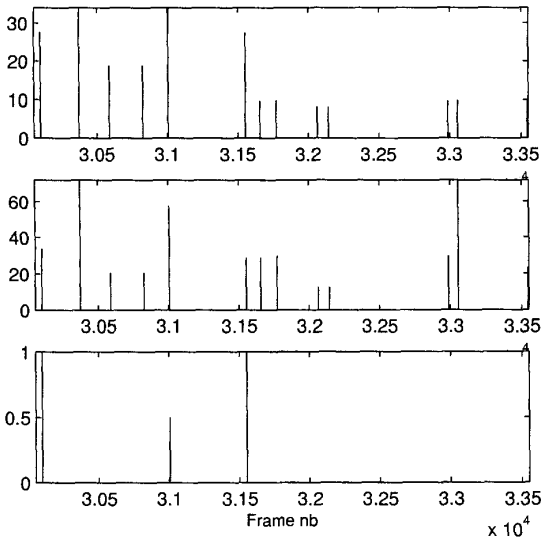


Figure 3. Breaking distance values on excerpt 2 of *Avengers* movie using option O_1 (upper row), option O_3 (middle row), and manual segmentation (lower row)

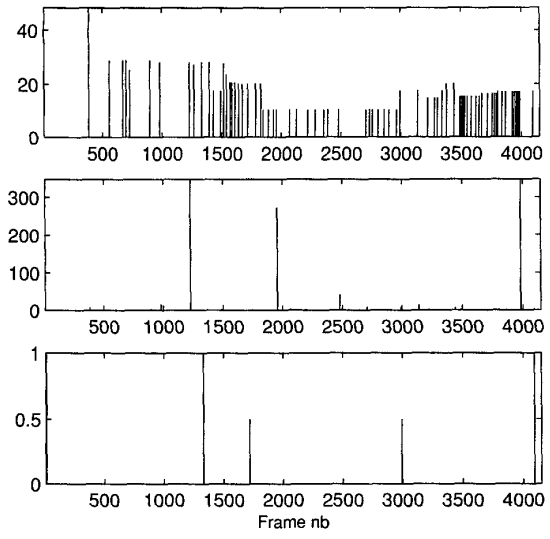


Figure 4. Breaking distance values on excerpt 3 of *Avengers* movie using option O_1 (upper row), option O_3 (middle row), and manual segmentation (lower row)

O_3, rather than O_1. Consequently, how to automatically select the proper option remains an open issue.

5 Conclusion

The method described in this paper, based on the cophenetic matrix, enables to accurately and efficiently segment video documents into sequences by building a binary agglomerative time-constrained hierarchy. We have implemented several versions. Selecting the most appropriate one improved results and gave a better description of the similarity of the shots through the hierarchy. Experiments on a larger base will be conducted in future work for selecting the best parameter set and evaluating alternative thresholding stategies.

References

[1] P. Aigrain, P. Joly, and V. Longueville. Medium knowledge-based macro-segmentation of video into sequences. In M. T. Maybury, editor, *Intelligent Multimedia Information Retrieval*, pages 159–173. AAAI/MIT Press, 1997.

[2] J. Carrive, F. Pachet, and R. Ronfard. Using description logics for indexing audiovisual documents. In ITC-IRST, editor, *Int. Workshop on Description Logics (DL'98)*, pages 116–120, Trento, 1998.

[3] A. Dailianas, R. B. Allen, and P. England. Comparison of automatic video segmentation algorithms. In *SPIE Photonics West*, volume 2615, pages 2–16, Philadelphia, 1995.

[4] A. Hanjalic, R. L. Lagendijk, and J. Biemond. Automatically segmenting movies into logical story units. In *Third Int. Conf. on Visual Information Systems (VISUAL'99)*, volume LNCS 1614, pages 229–236, Amsterdam, 1999.

[5] A. G. Hauptmann and M. A. Smith. Text, speech, and vision for video segmentation : The informedia project. In *AAAI Fall Symposium, Computational Models for Integrating Language and Vision*, Boston, 1995.

[6] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1988.

[7] J. R. Kender and B.-L. Yeo. Video scene segmentation via continuous video coherence. Technical report, IBM Research Division, 1997.

[8] R. Lienhart, S. Pfeiffer, and W. Effelsberg. Scene determination based on video and audio features. Technical report, University of Mannheim, November 1998.

[9] M. Yeung, B.-L. Yeo, and B. Liu. Extracting story units from long programs for video browsing and navigation. In *Proc. of IEEE Int. Conf. on Multimedia Computing and Systems*, Tokyo, 1996.

[10] J. Zupan. *Clustering of Large Data Sets*. Chemometrics Research Studies Series. John Wiley & Sons Ltd., 1982.

Acknowledgements Images from the *Avenger* movie, part of the AIM corpus, were reproduced thanks to INA, Department Innovation.

Clavis: a temporal reasoning system for classification of audiovisual sequences

Jean Carrive & François Pachet & Rémi Ronfard

Institut National de l'Audiovisuel (I.N.A)
4, av. de l'Europe
94 366 Bry sur Marne Cedex, France
{jcarrive, rronfard}@ina.fr

SONY CSL–Paris
6, rue Amyot
75005 Paris, France
pachet@csl.sony.fr

Abstract

In the context of video indexing, we present the Clavis system in which typical video sequences of television programs are represented by templates. Templates are terminological constraint networks in which video segments coming from automatic analysis tools are represented in a description logic formalism. Templates allow to express complex classes of video sequences with temporal constraints associated to a regular expression operator. Recognizing occurrences of a template in a video program is a plan recognition problem for which efficient methods have been implemented in a constraint satisfaction problem framework. The paper describes the system and illustrates its use with several experiments that were done in the context of the DiVAN european project.

Introduction

An important step towards content-based indexing television programs is the segmentation of the video into independent meaningful units, intermediate between the shot and the complete program. This is an inherently ill-posed problem in general, since even experts fail to agree on a common vocabulary of those units, and how to compare different segmentations. The problem becomes more tractable in more constrained situations, such as a collection of videos built on a common pattern or model. In that case, we can view segmentation as a plan recognition problem, where the plans to be recognized are characteristic of a collection of videos – such as a particular broadcast news or variety show. In this communication, we introduce a representation language and a computational framework for building such abstract models of television program collections, and recognizing the models from observations.

In (Ronfard, 1997) it was first proposed to use a description logic to describe and index video shots with a rich set of film concepts. (Carrive et al., 1998) further elaborated on this idea, and extended the proposal to a general taxonomy of film events, useful in the description and the analysis of video documents in the large. In this paper, we present Clavis (Classification of Video Sequences), a system which classifies typical video sequences found in collections of television programs, using temporal compositions of film events. The classes of sequences are represented as *templates* which are terminological constraints networks. Recognizing occurrences of templates within the video is presented as a plan recognition problem and the solution proposed is an extension of the T-Rex system originally proposed by (Weida, 1995), which combines symbolic temporal relations with a regular expression operator.

The paper is structured as follows. In section 0, we describe how classes of video segments coming from automatic analysis tools are represented in a description logic formalism, and used as building blocks of the Clavis system. In Section 3, we further explain how templates are defined as terminological constraints network and how the recognition process is designed. Finally in Section 4,

we present some experiments that were done in the context of the DiVAN (Distributed Audiovisual Archives Network) project[1].

Templates: descriptions of video sequences

A lot of periodic television programs, such as newscasts, variety shows or magazines, follow a predefined scenario which presents very little variation from one edition to another. These programs are part of what is called a collection. We propose to describe typical sequences of such programs using templates and we use a plan recognition algorithm based on Weida's work on terminological constraints network (Weida, 1995).

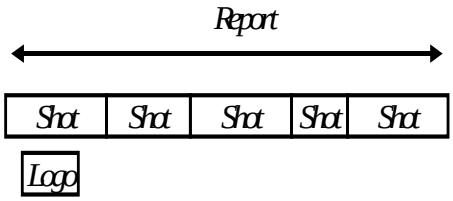
A plan recognition problem

We claim that recognising the occurrences of a template in a television program is similar to recognising which *plan* some active agent is following (Kautz, 1991) or recognising which predefined *scenario* best describes the evolution of a dynamic system (Ramaux et al., 1996), and we present this problem as a plan recognition problem. In (Fontaine, 1996), the author distinguishes four steps in a plan recognition process: opportunity, filtering, activation and discrimination. We will concentrate in this paper on the last step, which can be formulated in our case as the problem of finding in a video program made of automatically labelled segments the occurrences of a typical sequence represented by a template[2]. This recognition process often results in attributing a modality to the plan being processed, depending on whether the observations satisfy, don't satisfy or are compatible with the plan (Weida, 1995). (Ramaux and Fontaine, 1996) present a method that computes a proximity measure between the plan and the observations. We will focus in this paper on determining whether a set of observations satisfies or don't satisfies a template. It has turned out that determining whether the observations are compatible with a template is difficult and we temporarily put this task to one's side.

Following (Weida, 1995), we define a template as a terminological constraint network. The vertices of the network are associated with *concepts* which describe classes of video segments coming from audio or video analysis tools. The edges of the network are temporal constraints which have to be respected by the *observations*, *i.e.* the video segments coming from the analysis tools. In addition to Weida's formalism, we define an iteration operator "*" which expresses a contiguous sequence of video segments. For instance, a simple template may describe a report as a sequence of consecutive shots with a logo appearing during the first shot of the sequence. This template is defined by the following expression:

Report

nodes:	s1 [*Shot*]
	s2 [*Shot**]
	s3 [*Logo*]
constraints :	s1 {*meets*} s2
	s3 {*starts* $\vee$ *during* $\vee$ *finishes*} s1
	s1 {*starts*} this
	s3 {*finishes*} this



Classification of basic film events

Everything that appears on the screen, or is heard in the soundtrack constitutes an *event*. In this paper, we will consider that the temporal part of an event is only a time interval – for example a start point and a duration – identifying the temporal occurrence of the event within the video. Much like in grammars, we assume that a video is composed of several terminals, which we call segments. These

---

[1] DiVAN is the Esprit project N° 24956.

[2] We choose the term "template" because *scenario* or *plan* ("plan" means "shot" in French) are ambiguous in the television domain.

events are terminal because they are not themselves composed of other segments. An event can be a particular shot, which is what is filmed in one shot of the camera, a segment of music, a gradual transition between two shots as a dissolve, etc.

When applied in the context of DiVAN, the complete indexing process consists of the following steps: 1) initial segmentation of the audio and video track, producing at least two separate segmentation layers (usually more); 2) feature extraction and classification of segments, based on learned statistical models; 3) symbolic classification, using DL descriptions; and 4) recognition of composite events. In this paper, we assume the results from 1) and 2) provide a set of terminal events and focus on step 3) and 4).

Description Logics Description logics (Nebel, 1990) form a family of knowledge representation languages which derive from works on semantic networks and frame languages. In a description logic, *concepts* represent sets of *individuals*, and *roles* represent binary relations between individuals. Concepts can be seen as unary logical predicates, as roles are similar to binary logical predicates. Concepts can be described by syntactic operators as intersection (AND), union (OR), restrictions on the domain of a role or on the cardinality of a role. Concepts (and sometimes roles) are organized into a taxonomy according to a generality link – a *subsumption* link. Computing the subsumption relation between two concepts is one of the principal task of a description logic system. *Instantiation* is one other important operation, which compute the set of concept an individual belongs to. *Primitive* concepts are defined with necessary – and not sufficient – conditions, as *defined* concepts are defined with necessary and sufficient conditions.

Taxonomy of film events Using the CLASSIC system (Borgida, 1989), we define classes of events as concepts, focusing of classes of events which can be automatically recognized by audio and/or video analysis tools, such as those which are integrated in the DiVAN prototype: segmentation into shots, face regions and text regions detection, music/speech discrimination, jingle detection in the audio track, for example.

Figure 1 shows five concepts corresponding to cases where a face region can be detected clearly. These concepts are derived from terms of a cinematographic vocabulary, called “shot values”, and range from close-up (CU), where the face occupies approximately half of the screen, to the long shot (LS), where the human figure is seen entirely, and the face occupies around ten percent of the screen. Intermediate shot values are the medium shot (MS), the medium-close-up (MCU) and the medium-long-shot (MLS) (Thompson, 1998). Shot values are usually defined in relative and imprecise terms, based on the distance of the subject to the camera. We use the fact that the apparent size of faces on screen vary inversely with their distance to the camera to provide a computable definition of shot values. The ratio of the width of the face to the width of the frame is used to classify a shot among the five concepts. For example, the CLASSIC definition of a MCU shot is the following:

```
(cl-define-concept 'MCU-Face
    `(and
        face
        (all face-ratio (and (min ,(/ 1 6)) (max ,(/ 1 2))))))
```

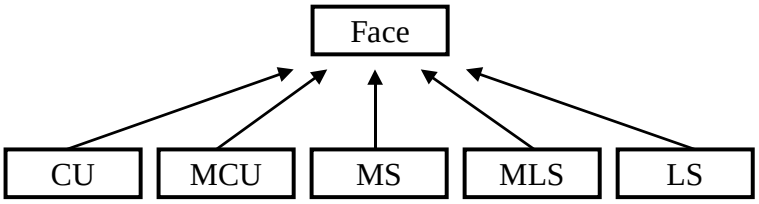


Figure 1: shot values

The algorithm used in DiVAN to segment a video into shots also detects two different classes of progressive transitions between shots, namely dissolves and wipes. In a dissolve, the existing image is

progressively replaced by superimposing a new image. In a wipe, a geometric pattern – often a simple line – erases the existing image and reveals the new one. This editing effects are illustrated by Figure 2.

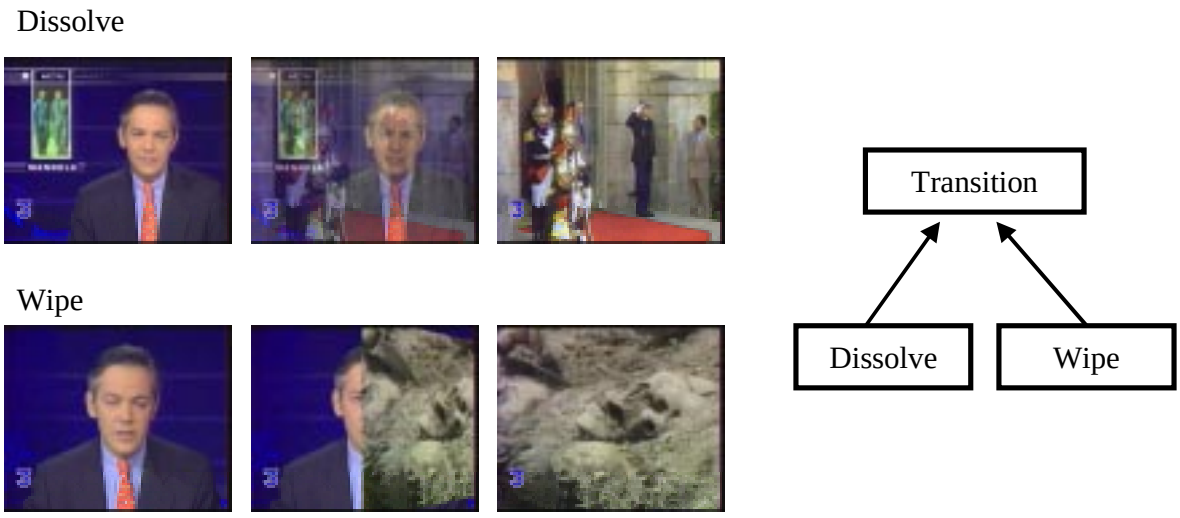


Figure 2: progressive transitions between shots

In may happen in some collections of documents that low-level features can directly provide high-level information. For example, in the five editions of the "France 3" evening newscast "Soir 3" from the DiVAN corpus, when some text is displayed at the bottom left of the screen during a medium close-up shot of a character, the text always refers to the character on-screen, mentioning its name and sometimes its function or role (see Figure 3). Thus, a "named person shot" can be defined by the very simple CLASSIC expression:

```
(cl-define-concept 'NamedPersonShot
    '(and
        MCU
        BottomLeftTextShot)))
```

Figure 3: example of a *NamedPersonShot*

When a shot is classified both as a MCUShot and as a BottomLeftTextShot, it is automatically classified as a NamedPersonShot. The definition of this concept is specific to the "Soir 3" newscast. Specific concepts are defined for each collection, starting from generally defined concepts as MCU. In other words, a general taxonomy of concepts is specialized for each collection of programs.

Constraint networks

A template is a temporal constraint network whose vertices are associated with concepts defined in a description logic formalism, or with other templates. A vertex associated with a concept is called an "elementary" vertex and a vertex associated with a template is told a "composed" vertex. An iteration operator "*" is defined which can be applied on the vertices of the network which are associated with concepts or with certain types on templates. This types of templates will be discussed later. The "*" operator can be compared to the "+" operator in regular expressions, as it indicates a contiguous sequence of at least one element. Non iterated vertices are told "simple" vertices. This iteration operator indicates a sequence of contiguous events. This leads to four different types of vertices:

- simple elementary vertices (v_C)
- simple composed vertices (v_T)
- iterated elementary vertices (v_{C*})
- iterated composed vertices (v_{T*})

The edges of the network are temporal constraints. In the current implementation of the system, these temporal constraints are temporal relations in the full interval algebra. A template is recognized – or satisfied – if and only if :

- each simple elementary vertex v_C is matched with an observation which is an instance *C*;
- each simple composed vertex v_T is matched with a subset of the observations which satisfies *T*;
- each iterated elementary vertex v_{C*} is matched with a subset of the observations forming a contiguous sequence of instances of *C*;
- each iterated composed vertex v_{T*} is matched with a subset of the observations forming a contiguous sequence of satisfied templates *T*;
- the matching respect the temporal constraints defined in the template.

When a set of observations satisfies a template *T*, it is said to be an instance of *T*. Without iterated vertices, the problem of template recognition comes down to find a matching between the constraint network and the observation network. This network matching problem has been proved to be NP-hard (Weida, 1995). Recognizing templates with iterated vertices lead to matching vertices of the template with sub-networks of the observation network. In order to avoid a combinatorial explosion, we designed methods which are very efficient for the type of cases we have to deal with. An overview of these methods is presented in section 0.

Temporal constraints

Several representations of time may underlie a temporal constraint network. The two main categories are the time point algebra (Vilain et al., 1989) and the time interval algebra (Allen, 1983). Complete constraint propagation in the latter representation is NP-hard and tractable subclasses of this algebra have been proposed (Nebel et al., 1994; Drakengren et al., 1997). Following (Weida, 1995) we have chosen the full interval algebra with a 3-path consistency constraint propagation algorithm which is potentially non complete. The reason why we choose this formalism is that when we started this work we didn't have a precise idea on what would be the most appropriate representation of time. We thus adopted this very general and expressive formalism. In a second step, we should work on determining what are the temporal constraints we really need. For example, it appeared on the one hand that numerical constraints such as "a segment of music which starts less than 30 seconds before a shot", as in (Aigrain, 1997), would enhance the powerfulness of our system. On the other hand, the full expressiveness of Allen's algebra didn't appear yet to be necessary.

The temporal extension of an instance *t* of a template *T* can be specified in the template definition by setting temporal constraints between *t* – or more precisely what will be *t* when *T* will be recognized – and its components. The instance of a template is designed by "this" in the template definition. For example, the template illustrated by Figure 4 defines a "musical shot" as a shot which appears during a musical segment. The temporal extension of a musical shot is naturally set as being equal to the observed shot. Note that any disjunction of Allen's relations may be set between the instance of a template and its components.

MusicalShot

nodes: s1[*Shot*]
s2[*Music*]

constraints: s2 *during* s1
s1 *equals* this

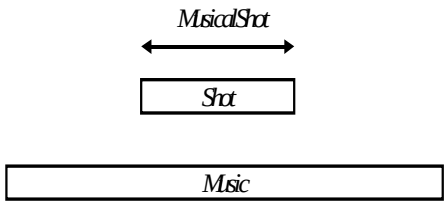


Figure 4: temporal extension of an instance of template

Iterated sequences

An iterated vertex v_{C*} or v_{T*} in a template represent a contiguous sequence of observations which are instances of *C* or instances of *T*. An iterated sequence contains at least one element, and the elements of an iterated sequence are contiguous, which means that two successive elements must be related by the Allen's *meets* relation. The temporal extension of an iterated sequence is defined as the temporal union of the temporal extensions of its elements. This implies that the temporal extensions of those elements are known, which lead to limit the types of templates which can be associated to an iterated vertex (see section 0). These types of templates are what we call "bounded" templates, *i.e.* templates whose temporal extension of instances may be computed. Roughly, a template is bounded if some of its vertices are in such a temporal relation with "this" that the temporal extension of its instances may be computed from the temporal extension of its components. This vertices have to be associated with event concepts or with other bounded templates. The temporal relations that allow to compute the starting point are *starts*, *is-started*, *meets*, *equals*, and the temporal relations that allow to compute the ending point are *finishes*, *is-finished*, *is-met*, *equals*.

The Figure 5 illustrates an iterated sequence of the template *MusicalShot* illustrated by Figure 4. Note is this example that the same segment of music was used to recognize each of the shots as instances of the *MusicalShot* template.

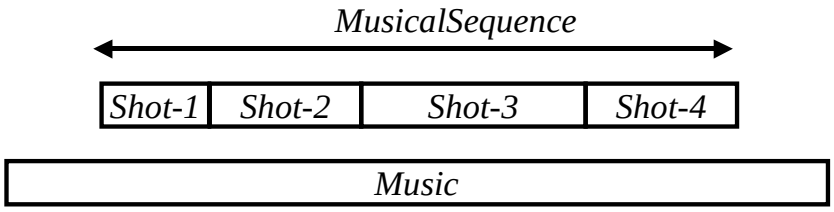


Figure 5: example of an iterated sequence

Other constraints

Once the general architecture of a template is designed as a temporal constraint network whose vertices are associated to concepts in a taxonomy of audiovisual events or with other templates, and whose vertices may be iterated in some cases, it is possible to define other constraints on its vertices. For example, we define a "no … between" constraint telling that for a template to be recognized, there should be no instance of some concept *C* or some template *T* between the observations matched with two of the vertices of the template. For example, the template illustrated by Figure 6 defines a report as the sequence of shots that is shown between two consecutive "jingle" shots, a jingle shot being a shot during which some instance of *Jingle* is heard.

Report

nodes: s1[*Shot*]
s2[Shot*]
s3[Shot]
s4[Jingle]
s5[Jingle]

constraints s1 *meets* s2
s2 *meets* s3
s4 *during* s1
s5 *during* s3
s2 *equals* this
no *Jingle* between s4 s5

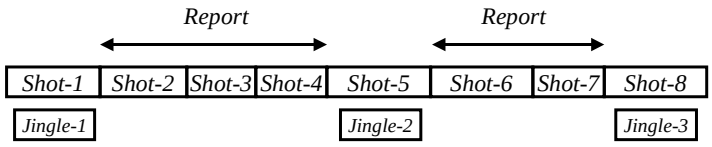


Figure 6: example of template with a "no … between" constraint

Other constraints could be easily defined, which would be taken into account by the constraint solver during the recognition process. It might be for cardinality constraints on the number of elements of an iterated sequence, or even numerical constraints on the temporal dimension of vertices, as "two instances of *Jingle* separated by at least 30 seconds". Note however that these constraints would not be taken into account during the construction of the temporal constraint network, and thus that inconsistencies would not be detected. In this case, the recognition process would scan the whole search space before answering that there are no solutions.

CSP techniques for template recognition

The problem of recognizing instances of a template *T* among the observations is expressed as a constraint satisfaction problem (CSP). Roughly, each vertex of *T* gives a *variable* of the CSP. Each variable takes its values from its *domain*, a finite set of possible values. The solutions of the problem are expressed as a set of *constraints* which are boolean functions whose arguments are variables.

In this section, we give an overview of the implementation of the Clavis system, focusing on the recognition of iterated sequences. The implementation deeply relies on BackJava (Roy et al., 1999), a (CSP) framework which allows to implement specific classes or variables, constraints and even domains or heuristics as subclasses of general purpose predefined classes. Thus, general mechanisms as arc-consistency can be used, as specialized filtering methods can be defined. In the system presented here, we use general unary constraints for checking that a simple elementary vertices are matched with observations that are instances of the concept associated with the vertex, general binary constraints for temporal constraints between simple vertices and specialized filtering method for temporal constraints which implies iterated vertices. These methods are sketched below. Finally, we let the default resolution mechanism of BackJava realize the recognition process, *i.e.* we let it choose when it should instantiate a new variable, which variable to choose, when it should backtrack, etc. The time responses we get during the experimentations we did – a few seconds for templates with iterated composed vertices with about 200 observed events – were quite encouraging and we didn't try to optimize the resolution phase.

The most complicated part of the implementation concerns iterated vertices of templates. We describe what is done with iterated elementary vertices. Iterated composed vertices associated with a template *T* are managed in a similar way after all instances of *T* have been recognized. Each iterated elementary vertex v_{C*} gives an "iterated variable" of the CSP. The first problem is to represent the domain of the variable, *i.e.* the set of iterated sequences of instances of *C*. The number of such sequences can be very important. For example, for an observation network made of *N* consecutive shots, there are $\frac{N(N+1)}{2}$ distinct iterated sequences of shots. In this case, illustrated by Figure 7, all sequences of shots may be represented as sub-sequences of the biggest sequence of shots, which is called a "maximal" sequence. A maximal sequence of instances of *C* is an iterated sequence of instances of *C* where no observation is both an instance of *C* and is in the *meets* relation (respectively the *is-met*) relation with its first (respectively its last) element. Each sub-sequence is uniquely determined by its size and the index of

its first element in the maximal sequence. An indexing function is used, which associates a unique integer between 0 and $\frac{N(N+1)}{2}$ -1 to all sub-sequences of a maximal sequence of size *N* according to their size and the index of their first element in the maximal sequence. The domain of an iterated variable is thus represented as a list of integers which is internally implemented as a list of intervals. This kind of integer variables are already implemented in BackJava which takes in charge basic operations on intervals, like union or intersection.

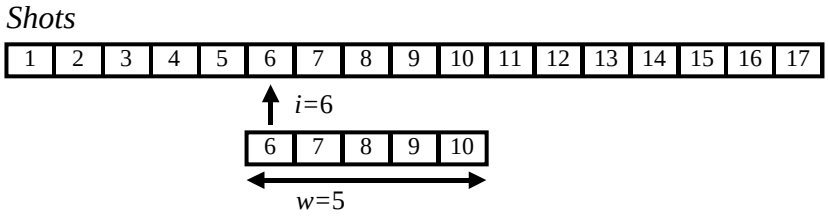


Figure 7: representation of sub-sequences

In order to compute the domain of an iterated variable representing an iterated vertex v_{C*}, one's must first compute the set of all maximal sequences of instances of *C*. This computation could be very expensive in the general case, but is quite acceptable for the kind of cases we have to process.

Temporal constraints which imply an iterated variable are implemented using a set of several filtering methods. Each of this method is intended to reduce the domain of the iterated variable in a given context. The general principal is the following. Let $C_R(v, v^*)$ be a temporal constraint which imposes hat the values of the two variables v and v* are in the *R* temporal relation, v* being an iterated variable, v being either iterated or not. When v is instantiated, *i.e.* when the solver chooses a value for v, some filtering methods are called which suppress from the domain of v* the iterated sequences which don't respect the *R* relation.

Let us take as example a template which specifies that a jingle must precisely *meets* an iterated sequence of shots. The *meets* relation is expected to be frequently encountered in templates, and thus a specialized filtering method has been designed for it. The v_J variable represents the jingle and the v_{S*} variable represents the shot sequence. The observations are illustrated by Figure 8. During the recognition process, the solver may choose to affect the *jingle-1* to v_J. In this case, the *doMeets* filtering method is called, which suppresses from the domain of v_{s*}. all the sequences which don't start with the sixth shot, as illustrated in the figure. Similar methods are designed for other cases which are expected to frequently happen, as the *starts*, *equals* or *finishes* temporal relations.

Another set of 13 methods are designed to cope with any temporal constraint, one for each of the Allen's basic temporal relation. For the *r* Allen's basic relation, the filtering method *doNot-r* is implemented, which takes as argument an iterated variables v* and an observed event *e*, and which suppresses from the domain of v* all the sequences which are in the *r* relation with *e*. In order to process a $C_R(v, v^*)$ constraint in the case where *R* is any disjunction of Allen's basic relations, the *doNot-r* method is called for all the Allen's basic relation *r* which are not part of *R*.

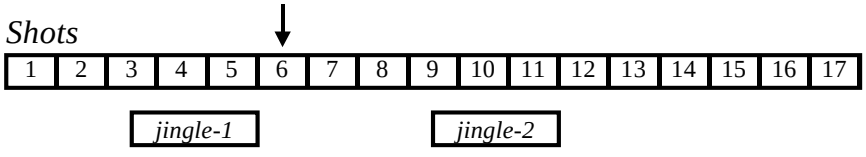


Figure 8: filtering iterated variables

Template subsumption

It would be appreciable if the template library was hierarchically organized according to subsumption links, as are concepts in a taxonomy. In (Weida, 1995), recognizing instances of templates among the

observations amounts to testing subsumption between templates, as plans and observations are roughly the same kind of temporal constraint networks. Unfortunately, the iteration operator we introduce in the template definition language results in an important complexity in the computation of subsumption between templates, as the '*' may appear in both the subsuming and the subsumee template. This implies that computing subsumption would necessitate to find a matching from sub-networks of the subsuming constraint network to sub-networks of the subsumee constraint network. Consider for example the templates T_1 and T_2 illustrated by Figure 9. T_1 subsumes T_2, as any set of observations recognized as an instance of T_2 would also be recognized as an instance of T_1.

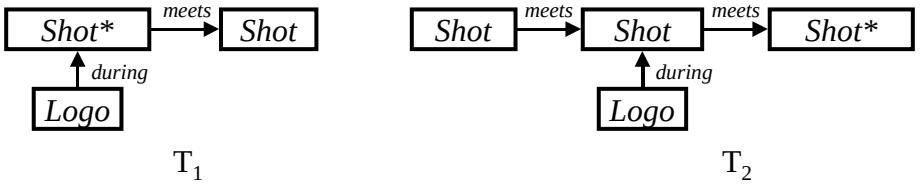


Figure 9: templates subsumption

Experimentation

We present in this section some experimentation we made on different broadcast news. Some of the documents come from an annotated corpus which was provided by INA and the AIM[3] group, the other are part of the corpus of the DiVAN project. The results presented here rely on reference segmentations established for evaluation of analysis and classification tools of the DiVAN project (Bouthemy et al., 1999). The newscasts presented here fall into two main categories:

- Traditional newscasts, alternating between the anchor person in the studio, and pre-recorded stories;
- Short newscasts composed of a small set of pre-recorded stories separated by jingles and/or graphics.

In order to recognize reports from this two types of newscast, two different methods are used which use different templates. It should be noted that similar classes of events – similar concepts – in two different newscast may have different definitions at the signal level. For example, the name of a character being filmed appears differently in a "Soir 3" and in a "France 2" evening newscasts, as well as the place of the logo is different (see Figure 3 above and Figure 10 below).

Reports of "France 2" newscast

In an edition of the "France 2" evening newscast, report sequences alternate with shots showing the anchorman in a studio. The anchorman can be filmed from different cameras. The logo of the channel always appear during report sequences on the bottom right of the screen, and never appears on studio shots, except once[4]. During report sequences, a text inscription on the bottom left of the screen always indicates the name of the person being filmed as a text inscription on the bottom right of a medium close-up shot always indicates the name of the location where the action takes place. Detecting these types of shots is interesting for at least two reasons. First, it may help to temporally structure the report sequence itself. Second, these shots may be used for summarize the report sequence, by for instance preferably select keyframes coming from these shots.

By exploiting the results of three analysis tools, namely a logo, a text region and a face region detection tool, four classes of shots can be defined for this newscast. These four classes are illustrated by Figure 10 and are organized according to subsumption links as shown in the figure.

---

[3] Action Indexation Multimedia

[4] We do not take this shot into account here

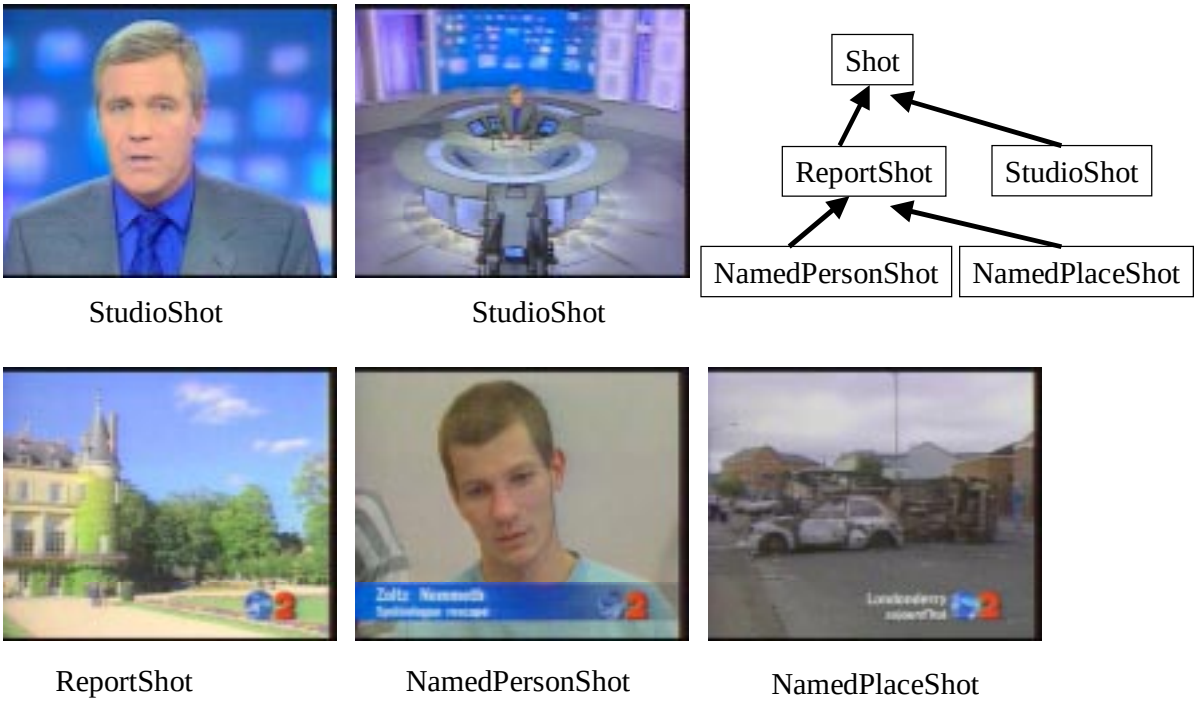


Figure 10: the four classes of shots from "France 2" evening newscast

We present here an edition of a "France 2" newscast coming from the AIM corpus. This document contains 157 shots, including 15 studio shots, 7 shots of named persons and 11 shots of named places. The temporal order of the shots is shown Figure 11. In the figure, the missing shots are report shots which are not shots of named place or named person.

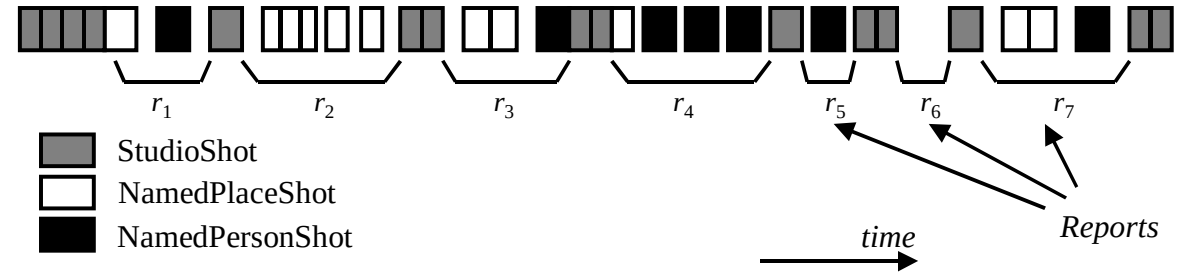


Figure 11: temporal order of shots in "France 2" example

In this example, we define a report as being what is shown between two shots of the anchorman. We thus extract the reports from the list of shots by defining the following template:

Report

nodes:	s1[StudioShot]
	s2[*ReportShot**]
	s3[*StudioShot*]
constraints:	s1 *meets* s2
	s2 *meets* s3
	s2 *equals* this

This template specifies that a report is recognized when it occurs that a studio shot is followed by a sequence of report shots which is followed by another studio shot, and that the temporal extension of

the report is equals to the temporal extension of the report shot sequence, *i.e.* it excludes the studio shots. This template is illustrated by Figure 12.

Figure 12: example of a "France 2" report

The recognition of this template gives 7 reports, as expected. The recognized reports are shown Figure 11. We then try to establish which reports contain at least one shot of a named person. The following template is used for this purpose:

ReportWithNamedPerson

nodes:	s1[*Report*]
	s2[*NamedPersonShot*]
constraints:	s1 {*starts finishes during equals*} s2
	s1 *equals* this

This template specifies that the reports we are looking for are reports "during" which a shot of a named person occurs, the meaning of "during" being given by the disjunction of Allen relations: {*starts* ∨ *finishes* ∨ *during* ∨ *equals*}. This relation states that the shot of the named person can be at any place in the report and can even temporally equals the report, which would be the case if the report is composed of a single shot of a named person – intervention of a foreign correspondent, for example. The temporal extension of such a report is obviously the same as the original report, as stated by the last constraint of the template definition.

The reports r_1, r_3, r_4, r_5 and r_7 of Figure 11 are recognized as reports with a named person. Note that the r_4 report contains three distinct shots of a named person. Thus, there are three different ways to recognize r_4 as a report with a named person. The recognition process gives three responses for r_4 t o such a report, and a post-treatment is needed in order to keep only one answer.

Reports recognized from audio jingle occurrences

Several short newscasts share a common very simple temporal structure: reports are only separated by jingles. This is the case for example for the French channels M6 (the "6 minutes" newscast), Canal+ and Arte (the "8 ½" newscast). Most of time, the jingles of such newscasts are sequences of very similar images accompanied with very similar sound samples. We are interested here in the case where two analysis algorithms are applied, one providing a segmentation of the visual part into shots and the other detecting occurrences of jingles within the audio track. Jingles and shots are temporally independents, as shown in Figure 13.

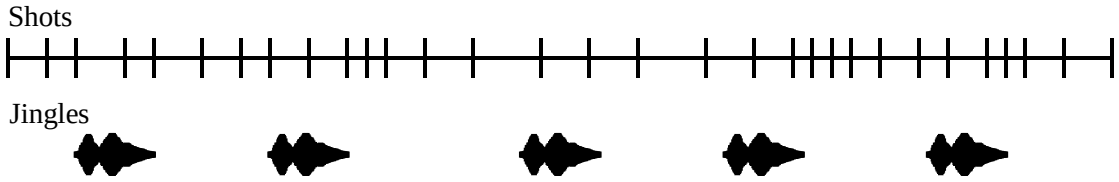


Figure 13: shot segmentation and jingle detection

We define in this case a report as a sequence of shots which is delimitated – in a way that will be précised later – by two jingles. We consider the first jingle as being part of the report in order to be

able to build sequences of reports, *i.e.* to recognize the *Report** part of the whole newscast template. We choose to include the first jingle as a jingle is often announcing the report and may contain visual or audio information on the report. A report is illustrated Figure 14. The first and the last reports constitute special cases, which are dealt with separately.

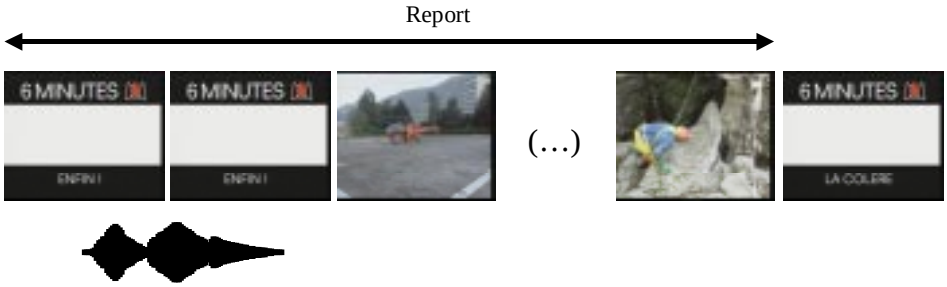


Figure 14: example of a "M6" report

We first define the whole temporal structure of the newscast, which is simpler that the definition of the report, and which is given by the following template:

Newscast

nodes:	s1[*FirstReport*]
	s2[*Report**]
	s3[*LastReport*]
constraints:	s1 *meets* s2
	s2 *meets* s3
	s1 *starts* this
	s3 *finishes* this

This template indicates that the newscast is composed of its first report followed by a sequence of reports followed by its last report, and that its temporal extension is the temporal union of all the reports. The template which defines a report is a bit more complicated, in order to handle all the possible relative temporal positions of the shots which are part of the jingles with respect to the audio part of those jingles. The definition of this template is the following:

Report

nodes:	s1[*Shot*]
	s2[*Shot**]
	s3[*Shot*]
	s4[*Jingle*]
	s5[*Jingle*]
constraints:	s1 *meets* s2
	s2 *meets* s3
	s1 {*overlaps starts is-started is-during*} s4
	s3 {*overlaps starts is-started is-during*} s5
	s1 *starts* this
	s2 *finishes* this
	no *Jingle* between s4 s5

At least three shots are necessary to recognize a report. Shots labelled s_1 and s_3 in the template delimitate the report. The temporal relation {*overlaps* $\vee$ *starts* $\vee$ *is-started* $\vee$ *is-during*} which constraint s_1 and s_3 stands for "the earliest shot which temporally intersect the jingle". Roughly, a report is said to last from the beginning of one jingle to the beginning of the next jingle.

The way the temporal constraints are set in the template ensures that recognized reports form a contiguous sequence. The last constraint imposes that there is no jingle between the jingles matched with s_3 and s_4, *i.e.* s_3 and s_4 are matched with consecutive jingles. Figure 15 shows an example of such a report. The matching of the template nodes are indicated on the figure, as are indicated the observed constraints between the shots which bound the report and the jingles.

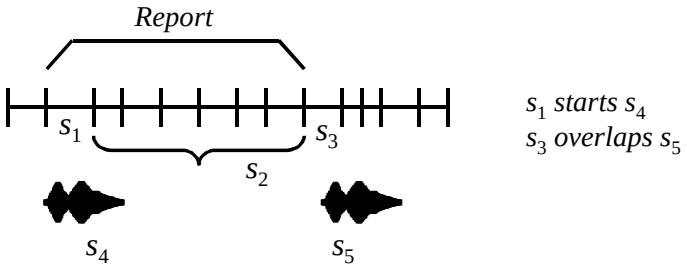


Figure 15: example of a report recognized from jingles

First and last reports are defined in a similar way, assuming that the first and the last shots have been labelled as *FirstShot* and *LastShot*. The definition of the template of the first report is given bellow:

FirstReport

nodes:	s1[*FirstShot*]
	s2[*Shot**]
	s3[*Shot*]
	s4[*Jingle*]
constraints:	s1 *meets* s2
	s2 *meets* s3
	s3 {*overlaps starts is-started is-during*} s4
	s1 *starts* this
	s2 *finishes* this
	no *Jingle* between s1 s3

At a first glance, it may not appear necessary to explicitly refer to the shots which bound the report – shots indicated by s_1 and s_3 in the report template – and a report may seem to be more simply defined as a sequence of shots "between" two consecutive jingles, which would naturally lead to define the template:

NaiveReport

nodes:	s1[Shot*]
	s2[Jingle]
	s3[Jingle]
constraints:	s1 {*is-started is-overlapped*} s2
	s1 {*meets overlaps*} s3
	s1 *equals* this
	no *Jingle* between s2 s3

Now consider the shots and the jingles illustrated by Figure 16. In the figure, shots which can be the first shot of a naïve report are indicated by a star and shots which can be the last shot of a naïve report are indicated by a diamond. Thus, there are 9 possible naïve reports, as there is only one report when applying the previous template. Moreover, this template doesn't take into account some special cases, as the case where the audio jingle is totally temporally contained in one single shot.

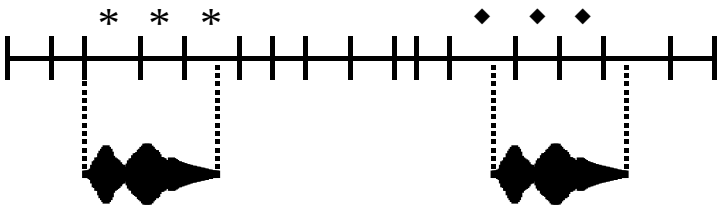


Figure 16: multi occurrences of a naïve report

We applied the newscast template to a "M6" newscast[5] containing 174 shots and 10 jingles. The recognition process gives exactly 1 matching for the newscast, which is made of 1 first report, a sequence of 9 reports, and 1 last report, which means that all reports have been recognized and that each report was recognized is only one way. On the other side, the recognition of the "naïve" report template gives 485 reports.

Other types of programs follow similar structures. For instance, programs of the "Top A" variety show from the DiVAN's corpus present successive songs which can be segmented by only using the applause occurring at the end of each song. Delimitating the temporal boundaries of a song is however more difficult than those of a "M6" report, because applause can temporally overlap the music.

Reports recognized from progressive transitions

We compare in this section two different ways of recognizing reports from five editions of "Soir 3", the evening newscast of the french France 3 channel. Four of these documents are part of the DiVAN corpus and the last one comes from the AIM corpus. A report of this newscast is defined as the sequence of shots that is showed between two consecutive shots of the anchorman. This definition is taken as the reference. The first method for recognising reports uses only one concept, *AnchorManShot,* and one template:

Report	
nodes:	s1[*AnchorManShot*]
	s2[*AnchorManShot*]
constraints:	s1 *before* s2
	s1 *meets* this
	s2 *is-met* this
	no *AnchorManShot* between s1 s2

Most of time, reports in a "Soir 3" newscast begins or ends with a progressive transition, as a dissolve, as illustrated by Figure 17.

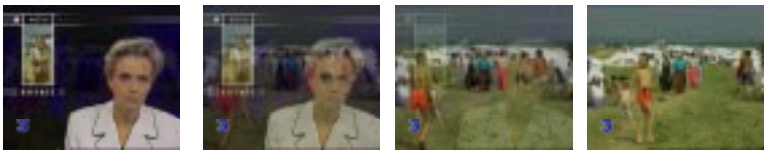


Figure 17: progressive transition at the beginning of a "Soir 3" report

Shots of the anchorman and progressive transitions of a typical "Soir 3" edition is illustrated by Figure 18. As progressive transitions may appear during a report, and as reports don't always start or end with a progressive transition, recognising reports with using only progressive transitions will produce some errors. These errors may come from a shot of the anchorman or a sequence of shots of the anchorman

[5] Still from AIM corpus

being classified as a report if it is surrounded by progressive transitions, or from "missed" reports – or more precisely missed transitions between two consecutive reports – caused by an anchorman shot or a sequence of anchorman shots neither starting nor ending by a progressive transitions.

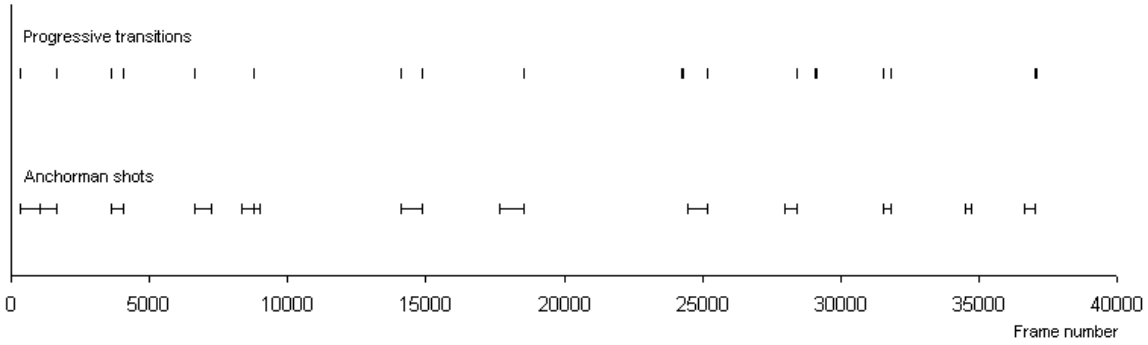


Figure 18: anchorman shots and progressive transitions in a "Soir 3" edition

The template used to recognize reports using the *ProgressiveTransition* event concept is the same as the template used with the *AnchorManShot* concept, by replacing *AnchorManShot* with *ProgressiveTransition*. Table 1 summarizes the number of reports recognized with the two templates, as the number of anchormans shots or sequences classified as reports and the number of missed reports.

Reports from anchorman shots	**Reports from progressive transitions**	**Anchorman shots or sequences classified as reports**	**Missed reports**
12	15	3	1
11	24	11	0
10	17	6	1
13	26	4	1
10	31	6	0

Table 1: reports from "Soir 3" editions

This experience shows that detecting reports in this case by using only the progressive transitions entails some over-segmentation due to progressive transitions appearing during the reports and anchorman shots or sequences of shots surrounded by progressive transitions. The quasi systematic use of progressive transitions at the beginning or at the end of reports leads to a very small number of forgotten reports.

Conclusion

We have shown in this paper that a plan recognition approach, making use of complex temporal relations between video segments can be both useful and efficient for solving a variety of video segmentation and indexing tasks. This temporal framework is obviously not sufficient for solving all problems, and should be extended to deal with other relations, such as image or sound similarity

between segment classes, and numerical temporal constraints. While this paper focused on the macro-segmentation task, such extensions could be even more useful in other applications, such as the automatic generation of video abstracts. More work is also needed to determine whether this temporal framework is necessary (compared to other simpler approaches using regular expressions and finite automata) for solving the task at hand.

This work should be extended in two main directions. First, experimentations showed that designing a template for a collection is not a trivial task, even for television experts, and that the quality of the results depends critically on such difficult design choices as temporal constraints. Therefore, we are now turning to machine learning techniques for creating templates from annotated examples. Second, we currently assume that the initial segmentation and classifying results observed by Clavis are perfect. In the future, we would like to relax this assumption, for instance by defining preferences in the space of solution of the CSPs.

References

Aigrain, P., Joly, P., Longueville, V. (1997). Medium Knowledge-Based Macro-Segmentation of Video into Sequences. Intelligent Multimedia Information Retrieval. A. P. M. Press.

Allen, J. F. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM* 26, 832-843.

Borgida, A., Brachman, R.J., McGuiness, D.L., Resnick, L.A. (1989). CLASSIC: A Structural Data Model for Objects. *ACM SIGMOD Int. Conf. on Management of Data* (pp. 59-67)

Bouthemy, P.& Garcia, C.& Ronfard, R.& Tziritas, G.& Veneau, E. & Zugaj, D. (1999). Scene segmentation and image feature extraction for video indexing and retrieval. *Third International Conference on Visual Information Systems (VISUAL'99)*, Amsterdam, june 1999

Carrive, J.& Pachet, F. & Ronfard, R. (1998). Using Description Logics for Indexing Audiovisual Documents. *Proceedings of the International Workshop on Description Logics*, Trento, Italy

Drakengren, T. & Jonsson, P. (1997). Twenty-one Large Tractable Subclasses of Allen's Algebra. *Artificial Intelligence* 93, 297-319.

Fontaine, D. (1996). Une approche par graphes pour la reconnaissance de scénarios temporels. *Revue d'Intelligence Artificielle* 10(4), 439-468.

Kautz, H. A. (1991). A Formal Theory of Plan Recognition and its Implementation. Reasoning about Plans. J. F. Allen& H. A. Kautz& R. N. Pelavin & J. D. Tenenberg, Morgan Kaufman Plubishers, Inc.

Nebel, B. (1990). Reasoning and Revision in Hybrid Representation Systems. *LNAI* 422.

Nebel, B. & Bückert, H. J. (1994). Reasoning about Temporal Relations: A Maximal Tractable Subclass of Allen's Interval Algebra. *Proceedings of AAAI'94*, Seattle, Washington (pp. 356-361)

Ramaux, N. & Fontaine, D. (1996). Recognising a Scenario by Calculating a Temporal Proximity Index between Constraint Graphs. *8th International Conference on Tools with Artificial Intelligence (ICTAI 96)*

Ronfard, R. (1997). Shot-level description and matching of video content. *SPIE 97*, San Diego

Roy, P.& Liret, A. & Pachet, F. (1999). The Framework Approach for Constraint Satisfaction. *Object Oriented Application Frameworks, Wiley Eds.* 2.

Thompson, R. (1998). *Grammar of the shot*, Focal Press.

Vilain, M.& Kautz, H. & Van Beek, P. (1989). Constraint propagation algorithms for temporal reasoning: A revised report. *Readings in Qualitative Reasoning about Physical Systems*, 373-381.

Weida, R. (1995). Knowledge Representation for Plan Recognition. *IJCAI 95 Workshop on the Next Generation of Plan Recognition Systems*, Montréal, Québec, Canada

A FRAMEWORK FOR ALIGNING AND INDEXING MOVIES WITH THEIR SCRIPT

Remi Ronfard and Tien Tran Thuong

INRIA Rhone Alpes
Montbonnot, France

ABSTRACT

A continuity script describes very carefully the content of a movie shot by shot. This paper introduces a framework for extracting structural units such as shots, scenes, actions and dialogs from the script, and aligning them to the movie based on the *longest matching subsequence* between them. We present experimental results and applications of the framework with a full-length movie and discuss its applicability to large-scale film repositories.

1. INTRODUCTION

Choosing terms for describing and indexing video content is a difficult and important problem. We believe not enough attention has been given to a very important source of video descriptions - the *continuity script* which describes very carefully the content of a movie shot by shot. In this paper, we discuss some of the issues related with synchronizing and aligning a movie with its script using a combination of cues from the dialogs and the image track. We describe grammars and automata for formatting the script into structural units such as shots, scenes, actions and dialogs. We then introduce a dynamic programming algorithm for finding the longest matching subsequence between the formatted script and the video content. This procedure aligns the script to the temporal axis of the movie at the shot and dialog levels, and therefore allows dialogs and action descriptions in the script to be used as indices to the video content. We illustrate the framework with 'The wizard of Oz ', a well-known masterpiece released in 1939, whose continuity script was carefully edited and published on the Internet [1].

Alignment of script to video was mentioned by other researchers [2, 3, 4] as a means to provide training data for learning models of objects, scenes and actors. But contrary to the similar problem of aligning bilingual translations of the same text [5, 6], it was never formalized properly. With this work, we would like to contribute to such a formalization.

2. SCRIPT FORMATTING

In this section, we introduce our model of the continuity script for 'The wizard of Oz', and algorithms for automatically formatting the script from plain text to XML. Typically, a continuity script is updated throughout shooting of the movie and includes the breakdown of scenes into shots. In contrast, a production script only breaks down the movie into its master scenes. In that case, the alignment and indexing can only be performed at a much grosser scale. In this work, we are particularly interested in continuity scripts, such as 'The wizard of Oz'.

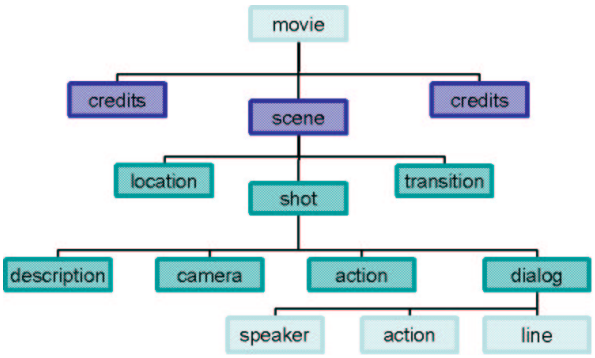


Fig. 1. *High-level grammar of film structure. A movie is composed of scenes, which are composed of shots. Transitions can occur between shots or scenes. Shots are composed of actions, camera movements and dialogs.*

From our own analysis of many film scripts and related books in film studies, we found that the structural components of a film script were - the scene, the shot, the transition, the action, the camera action and the dialog, as represented in Fig. 1. *Scene* is a segment of the movie taking place in a given location. It is described as 'interior' or 'exterior' and with the name of a place or location. It contains a sequence of contiguous shots. *Shot* has type close-shot (CS), medium-close-shot (MCS), medium shot (MS), medium-long-shot (MLS), long-shot (LS) or extreme-long-shot (ELS). It starts with a description, usually naming the actors, the settings and the camera viewpoint, followed by a

sequence of actions, camera actions and dialogs. *Transition* has type dissolve or fade and separates two shots or scenes. *Camera* is an informal textual description of the camera motion. *Action* is an informal textual description of an action taking place within a shot. It usually names the action with a verb as well the actors performing the action, and includes references to places in the scene and on the screen. *Dialog* starts with the name of a speaker. It contains a sequence of utterances (broken into lines) and actions.

The organization of those components in a particular script is embodied by a set of typographic and stylistic rules. In order to format the script into a strict, structured representation, we need to further describe those rules as a grammar, down to terminal symbols such as letters, tabulations and line breaks. It turns out that in many classical Hollywood-style scripts, the grammar is regular. In other words, film scripts can be modelled with regular expressions and recognized with finite-state automata. As an example, the formatting rules for the continuity script of the 'Wizard of Oz' follows the grammar of Fig 2.

SCENARIO	→	CREDITS? SCENE + CREDITS?
SCENE	→	LOCATION (SHOT[TRANSITION]) +
LOCATION	→	("INT." | "EXT.") – TEXT
TRANSITION	→	TAB? ("FADE IN" | "FADE OUT"
TRANSITION	→	TAB? "LAP DISSOLVE TO")
SHOT	→	SIZE DESCRIPTION DIALOG +
SIZE	→	"CS" | "MCS" | "MS" | "MLS" | "LS"
DESCRIPTION	→	ACTION [–ACTION | CAMERA]+
DIALOG	→	TAB SPEAKER "(O.S.)"? [LINE | ACTION]+
CAMERA	→	"CAMERA" TEXT
ACTION	→	TEXT

Fig. 2. *A grammar for the continuity script for 'The Wizard of Oz'. Higher-level symbols from Fig. 1 are explicitly decomposed into lower-level entities and terminals (formatting and tabulations). This grammar is easily found to be regular since all productions are either of type $A \rightarrow a$ or $A \rightarrow aB$, where a is a terminal and A, B are non-terminals.*

As a result, the script can be analyzed as a regular expression with a finite state automaton. Once this grammar has been fully worked out, it is easy to write down an automaton for transcribing the entire script into an XML tree in the form of Fig. 1. Fig. 3 shows the three shots of Fig. 4 translated into XML using specialized tags for shots, actions, cameras and dialogs.

3. SCRIPT ALIGNMENT

Given the formatted script, we now have to align its elements with the temporal axis of the movie, so that the descriptions from the script can be used as indices to the video content. This is not a trivial task because the video comes as large chunk of data, which must be parsed into elements corresponding to the scenes, shots, actions and dialogs in

```
<shot size="CS">
<action>Toto by wheel of rake</action>
<action>listening to song</action>
</shot>
<shot size="MCS">
<action>Dorothy singing</action>
<action>swings on wheel of rake</action>
<action>then walks forward around wheel</action>
<action>Toto jumps up onto seat of rake</action>
<action>Dorothy pets him</action>
<camera>CAMERA PULLS back</camera>
<dialog speaker="DOROTHY">
<line>Someday I'll wish upon a star</line>
</dialog>
</shot>
<shot size="LS">
<action>Miss Gulch rides forward</action>
<action>stops and gets off her bicycle</action>
</shot>
```

Fig. 3. *Example of xml-formatted script. For lack of space, we did not reproduce the scene level, where in fact the first two shots are part of the same scene and the third shot introduces a new scene.*

the script. Since there may be errors in both the formatting of the script and the parsing of the video, the alignment should be flexible enough. While video parsing has a long and active history, we do not believe that the results of video analysis can be trusted to generate a full tree structure allowing to formulate the alignment as a tree-matching problem. Instead, we temporally sort all the script elements and extracted video segments (shot transitions and subtitles) and apply string matching techniques to align them.

In this section, we reformulate the alignment problem as one of finding the longest matching subsequences (LMSS) between the movie and the script, and we describe an efficient dynamic programming algorithm which solves this problem. Dynamic programming has been used with much success for aligning bilingual corpora [5] and for matching video sequences [7].

For the purpose of aligning a movie and a script, we extract subtitles and candidate shot cuts. We implemented and used an algorithm described by Salesin et al. for shot change detection using Haar wavelet coefficients [8]. The algorithm computes a distance between successive frames, and uses thresholds to detect candidate shot cuts. We carefully tuned the thresholds over multiple temporal resolutions to obtain quasi perfect precision (no false detection). This results in a fast and reasonably robust detection, except for the case of dissolves and fades, which result in a large number of missed shot transitions. Gradual transitions are still an open issue for many other algorithms, because they typically introduce large numbers of false detections. In the context of

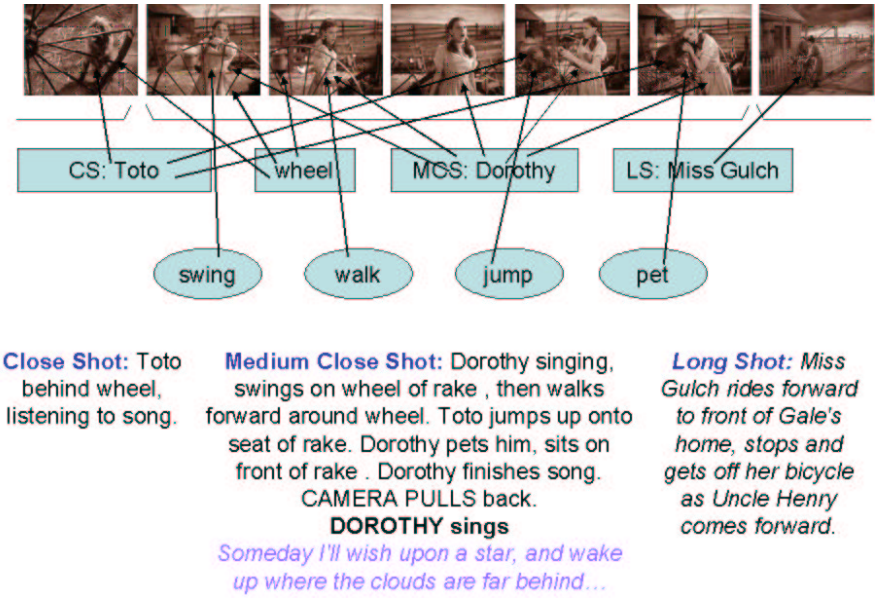


Fig. 4. *Example of three shots aligned to their script descriptions in* The wizard of Oz. *Shots in the script are aligned to automatically detected shot changes in the video. Aligned shots are described by the locations, actors and actions mentioned in the script.*

this work, we were interested to verify the viability of our alignment framework with imperfect shot detection, under the assumption that the effect of missed transitions would remain local (as was effectively verified).

Separately, we extracted the English subtitles from the same video stream and performed optical character recognition on them to produce a stream of time-stamped short texts. The detected shots and subtitles were translated into an MPEG7-like XML format for matching.

The alignment between the script and the detected video segments was performed by matching a temporally sorted string of shots and dialog lines from the script with the shots and subtitles from the video. More specifically, we searched for the longest increasing subsequence of matched shots and dialog/subtitles, a problem which can be solved efficiently with a classic dynamic programming algorithm [9]. In its simplest form, this approach accounts for the following three cases when comparing segments from the script and the movie - either they match, or the script segment was deleted, or the video segment was inserted. Note that this approach can be generalized in many ways to include more sophisticated editing models.

4. INDEXING AND SYNCHRONIZATION

Formatting and synchronizing the movie script for 'The wizard of Oz' opened up two useful and interesting applications, which proved surprisingly easy to implement using XSL transformations on the matched subsequences. In the first application, we created a database of all the scenes, shots, actions and dialogs of the movie and indexed them with the corresponding text from the script. In addition, the formatting of the script allowed us to extract and categorize place/location names (from scene descriptions), speaker/actor names (from dialogs) and action verbs (from shot descriptions).

In the second application, we generated MPEG-7 like elements and their temporal relations for use in an enhanced multimedia player for film studies which we implemented using our MDEFI framework. MDEFI [1] is an advanced environment for playing and editing multimedia documents [10]. MDEFI is based on Madeus, an extension of SMIL with the additional features of (1) enhanced separation of media content location, temporal information and spatial information, (2) hierarchical, operator-based temporal model complemented with relations, (3) rich spatial specification model with relative placements and (4) media fragment integration. MDEFI allows to reformat media content descriptions based on the MPEG-7 standard. This description is then used for specifying fine-grained composition between media objects. In this work, the fine-grained composition features of MDEFI were used to synchronize the video shots and the film script. When playing the movie, for example, the corresponding parts of the film script can be highlighted in synchronization. In addition, the user can jump to video

[1] Multimedia DEscription and Fine-grained Integration

segments by clicking anywhere on the script - as long as a matched segment can be found.

5. EXPERIMENTAL RESULTS

We performed the alignment of 'The wizard of Oz', starting with 791 shots in the script and 683 detected shots. Dissolves and special effects such as explosions and transitions through a crystal ball could not be detected at this stage. The alignment was performed using 2649 subtitles extracted from the video and 3041 dialog lines in the script. We compared dialogs and subtitles using approximate string matching [2], successfully matching a total of 1866 dialog lines. As a result, we were able to automatically align 604 shots, leaving 187 scenario shots unmatched and 79 video shots unmatched. A rapid manual inspection revealed that most of the matched shots were matched correctly, except for a few, highly localized segments of the movies with either (1) a fast succession of *missed* dissolves and special effects or (2) a missing scene, which was edited out from the script in the final movie. The latter case accounts for 80 unmatched shots. Of the remaining 107 unmatched shots in the script, half were due to undetected transitions and half to smaller variations between the final movie and the script. Our alignment algorithm therefore correctly matched 82% of the script shots and 88% of the detected video shots, reducing the number of outstanding shots from 791 to 107.

Of course, future work will be devoted to the remaining fraction of shots and dialogs which could not be matched with our current method. We are following two main directions of research in this respect. On the one hand, we can improve the alignment of shots (especially those without dialogs) by matching visual descriptors in addition to subtitles and compensating for inaccuracies in the shot detection algorithm by matching all frames, using models of the expected shot durations. This would match at the frame, rather than shot level, and use shot transition probabilities, rather than hard decisions in order to handle the more difficult cases of dissolves and special effects better[3]. On the other hand, we are extending our alignment algorithm following previous work in machine translation [5] to account for more elaborate models of insertions, deletions and replacements between the movie and script shots, based on the experimentation reported here. We are also interested in generalizing to other scripts and script formats, which entails discovering the formatting rules for the new scripts, writing down their grammars and checking that they remain consistent. Finally, we believe this work opens the way for even more ambitious developments such as tracking and hyper-linking of video objects and spatio-temporal synchronization, which are already part of the MDEFI framework.

[2] Actually, another instance of the longest common subsequence search!

[3] This will effectively turn our longest matching subsequence algorithm into a Hidden Markov Model decoding algorithm

6. CONCLUSION

By examining the script of 'The wizard of Oz', we have found that *at the structural level* at least, a movie and its script can be analyzed and synchronized with simple tools (regular expressions and dynamic programming). This has allowed us to format the script into high-level components and to align some of them to the movie itself. As a result of this work, we are currently building a large database of movie shots, indexed by dialogs, actors, settings and action descriptions. We believe such a database can be useful for film studies as well as for learning statistical models of video content.

7. REFERENCES

[1] Noel Langley, Florence Ryerson, and Edgar Allen Woolf, "The wizard of oz– movie script," 1939, Cutting Continuity Script, Taken From Printer's Dupe, Last revised March 15, 1939. This script was transcribed by Paul Rudoff.

[2] Joshua S. Wachman and Rosalind W. Picard, "Tools for browsing a TV situation comedy based on content specific attributes," *Multimedia Tools and Applications*, vol. 13, no. 3, pp. 255–284, 2001.

[3] Salway and Tomadaki, "Temporal information in collateral texts for indexing moving images," in *Proceedings of LREC 2002 Workshop on Annotation Standards for Temporal Information in Natural Language*, 2002.

[4] C.G.M. Snoek and M. Worring, "Multimodal video indexing: A review of the state-of-the-art," *Multimedia Tools and Applications*, 2003, Accepted for publication.

[5] W. A. Gale and K. W. Church, "A program for aligning sentences in bilingual corpora," in *In Proceedings of ACL-91, Berkeley CA.*, 1991.

[6] P. F. Brown, S. A. D. Pietra, V. J. D. Pietra, and R. L. Mercer, "The mathematics of machine translation: Parameter estimation," *Computational Linguistics*, vol. 19, no. 2, 1993.

[7] Milind R. Naphade, Roy Wang, and Thomas S. Huang, "Supporting audiovisual query using dynamic programming," in *ACM Multimedia*, 2001, pp. 411–420.

[8] Xiaodong Wen, Theodore D. Huffmire, Helen H. Hu, and Adam Finkelstein, "Wavelet-based video indexing and querying," vol. 7, no. 5, pp. 350–358, 1999.

[9] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein, *Introduction to Algorithms*, MIT Press, second edition edition, 2001.

[10] T. Tran Thuong and C. Roisin, *Media content modelling for Authoring and Presenting Multimedia Document*, World Scientific - Series in Machine Perception and Artificial Intelligence, 2002.

CHAPITRE 2

Analyse du mouvement humain

Learning to Parse Pictures of People de Rémi Ronfard, Cordélia Schmid et Bill Triggs, European Conference on Computer Vision, Copenhague, 2002.

Human Motion Tracking with a Kinematic Parameterization of Extremal Contours de David Knossow, Rémi Ronfard et Radu Horaud, International Journal of Computer Vision, vol. 2, no. 79, Septembre 2008.

Learning to Parse Pictures of People

Remi Ronfard, Cordelia Schmid and Bill Triggs*

INRIA, 655 avenue de l'Europe, 38330, Montbonnot, France

Abstract Detecting people in images is a key problem for video indexing, browsing and retrieval. The main difficulties are the large appearance variations caused by action, clothing, illumination, viewpoint and scale. Our goal is to find people in static video frames using learned models of both the appearance of body parts (head, limbs, hands), and of the geometry of their assemblies. We build on Forsyth & Fleck's general 'body plan' methodology and Felzenszwalb & Huttenlocher's dynamic programming approach for efficiently assembling candidate parts into 'pictorial structures'. However we replace the rather simple part detectors used in these works with dedicated detectors learned for each body part using Support Vector Machines (SVMs) or Relevance Vector Machines (RVMs). We are not aware of any previous work using SVMs to learn articulated body plans, however they have been used to detect both whole pedestrians and combinations of rigidly positioned subimages (typically, upper body, arms, and legs) in street scenes, under a wide range of illumination, pose and clothing variations. RVMs are SVM-like classifiers that offer a well-founded probabilistic interpretation and improved sparsity for reduced computation. We demonstrate their benefits experimentally in a series of results showing great promise for learning detectors in more general situations.

Keywords: object recognition, image and video indexing, grouping and segmentation, statistical pattern recognition, kernel methods.

1 Introduction

Detecting people in images is an important practical challenge for content-based image and video processing. It is difficult owing to the wide range of appearances that people can have. There is a need for methods that can detect people in general everyday situations. For instance, actors in typical feature films are shown in a great variety of activities, scales, viewpoints and lightings. We can not rely on frequently-made simplifying assumptions such as non-occlusion, perfect background subtraction, *etc*.

To address this issue, Forsyth & Fleck introduced the general methodology of *body plans* [8] for finding people in images. However, they relied on simplistic body part detectors based on generalized cylinders. This is problematic, especially in the case of loose clothing. Similarly, Felzenszwalb & Huttenlocher [6] showed how dynamic programming could be used to efficiently group body plans cast as 'pictorial structures' [7], but they relied on simplistic colour-based part detectors. Both of these works make strong photometric assumptions about the body parts. We retain their ideas for composing parts into assemblies by building tree-structured models of people, but propose

* This work was supported by the European Union FET-Open research project VIBES

a more general approach to learning the body part detectors and the underlying geometric model, based on Support Vector Machines (SVM) [24,4] or Relevance Vector Machines (RVM) [22,23]. In the past, SVM classifiers have been learned for entire humans [18] and also for rigidly connected assemblies of subimages (typically, upper body, arms, and legs) [16], but not for flexibly articulated body models.

We present a series of experiments showing the promise of learning the articulated structure of people from training examples with hand-labelled body parts, using SVMs or RVMs. Our contribution is three-fold. Firstly, our feature set and training method builds reasonably reliable part detectors from as few as 100 hand-labelled training images, and the final RVM detectors are very efficient, often involving comparison with only 2–3 positive and 2–3 negative exemplars. Secondly, we sketch a method for learning a body joint model using the recently proposed Adaptive Combination of Classifiers (ACC) framework [16]. Thirdly, we describe an efficient decoder for the learned models, that combines kernel based detection with dynamic programming. Our initial experiments demonstrate that body part detectors learned with only 100 images from the MIT pedestrian database can give reliable detection with as few as 4 false alarms per image on this data set. This is remarkable as even humans often find it difficult to classify the isolated part subimages correctly. The detected parts can be efficiently assembled into correct body plans in 70% of cases.

The paper is structured as follows. We introduce our body plan model in §2, then discuss body part detectors learned by two competing algorithms, SVM and RVM, in §3. §4 presents our approach for learning and decoding body plans. Finally, §5 presents some results and discusses future work.

2 The Pictorial Structure of People

In the work of Marr & Nishihara [15] and others [10,19], people are described as hierarchical 3D assemblies of generalized cylinders and components. The position of a part C relative to its parent P is parametrized by C's position (p, r, θ) and angular orientation (ψ, ϕ, χ) in P's cylindrical coordinate system. Each joint is thus represented as a 6-vector, with discrete toleranced values for each parameter. They note that perspective projection makes many parameters unobservable and that the image signature of a joint is a pair of axes, but still emphasize, and attempt to recover, 3D structure.

Recovering articulated 3D models from single images is difficult. Felzenszwalb & Huttenlocher recently reconsidered Fischler & Elschlager's notion of *pictorial structure* [7] and demonstrated its usefulness for detecting people in indoor scenes [6]. Pictorial structures are collections of image parts arranged in deformable configurations. They are directly adapted to monocular observations. Similarly, Morris & Rehg argued that 3D tracking singularities can be removed using image based 'scaled prismatic models' [17] — essentially, pictorial structure models. Other 2D part-based models use image edges [25] or motion models derived from dense optical flow [14] as features for detection and/or tracking.

Following this line of research, we represent people using a 2D articulated appearance model composed of 15 part-aligned image rectangles surrounding the projections of body parts: the complete body, the head, the torso, and the left and right upper arms,

forearms, hands, thighs, calves and feet, numbered from 1 to 15 as in Figure 1. Each body part P_i is a rectangle parametrized in image coordinates by its centre $[x_i, y_i]$, its length or size s_i and its orientation θ_i. A coarse resolution whole-body image is also included in case 'the whole is greater than the sum of the parts'. During training and detection, we discretize the admissible range of sizes and orientations. As discussed later, we use a range of 8 scales, and 36 orientations equally spaced every 10 degrees. 14 body joints connect the parts: the plexus between body and torso, the neck between head and torso, the hips between torso and thighs, the knees between thighs and calves, the ankles between calves and feet, the shoulders between torso and upper arms, the elbows between upper arms and forearms and the wrists between forearms and hands. Figure 1 shows the body model in average position, using a single aspect ratio of 16:9 for all body parts.

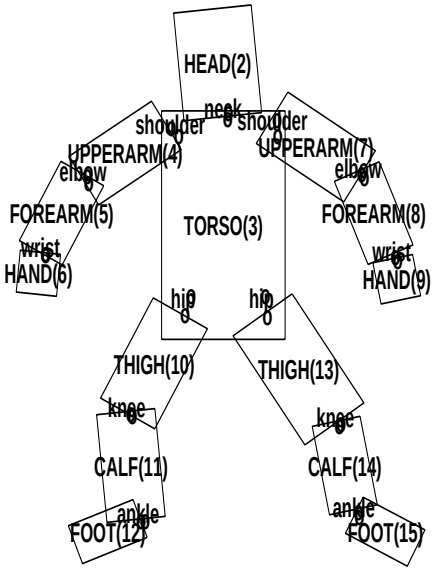


Figure 1. Our articulated body model with its 14 joints and 15 body parts.

Expressed in terms of the probabilistic formulation of pictorial structure, the posterior likelihood of there being a body with parts P_i at image locations l_i ($i \in \{1...15\}$) is the product of the *data likelihoods for the 15 parts* (*i.e.* the classification probabilities for the observed subimages at the given part locations to be images of the required parts) and the *prior likelihoods for the 14 joints* (*i.e.* the probabilities for a coherent body to generate an image with the given relative geometric positionings between each part and its parent in the body tree). The negative log likelihood for the whole body assembly $A = \{l_1, \ldots, l_{15}\}$ can thus be written as follows, where E is the list of body

joints ('edges' of the body tree):

$$L(A) \;=\; -\sum_i \log p_i(l_i) - \sum_{(ij)\in E} d_{ij}(l_i, l_j)$$

Felzenszwalb & Huttenlocher model body parts as constant color regions of known shapes and body joints as rotational joints. In this paper, we machine-learn the 29 distributions $p_i(l_i)$ and $d_{ij}(l_i, l_j)$ from sets of positive and negative examples. We model the part and articulation likelihoods using linear Support Vector or Relevance Vector Machines. Our work can be viewed as an extension of Mohan's recent work on *combined classifiers* [16], where 'component' classifiers are trained separately for the limbs, torso and head based on image pixel values, and 'combination' classifiers are trained for the assemblies based on the scores of the component classifiers in fixed image regions. However, we learn part-aligned, rather than image-aligned, classifiers for each body part, and we extend the 'combination' classifier to include deformable, articulated structures rather than rigid assemblies.

3 Detecting Body Parts

In our model, learning each body part amounts to estimating its probability given the observed image distribution at its location. Detecting and labelling body parts is a central problem in all component-based approaches. Clearly the image must be scanned at all relevant locations and scales, but there is also a question of how to handle different part orientations, especially for small, mobile, highly articulated parts such as arms and hands. One can work either in the image frame, trying to build a general detector that is capable of finding the part whatever its orientation, or in a part-aligned frame, building a detector that works for just one orientation and scanning this over all relevant orientations. The part-aligned approach has the potential to produce simpler detectors from less (but better labelled) training data, and the advantage that it also recovers the part orientation. Which approach is faster or better must depend on the relative complexity and reliability of all-orientation and one-orientation detectors, but in general it is difficult to build good transformation invariance into general-purpose detectors. The image-frame approach is well adapted to pedestrian detection applications such as Mohan's [16], where one wants a relatively coarse whole person detector for distant people with similar poses (mainly standing or walking). But our ultimate goal is to detect people and label them with detailed part locations, in applications where the person may be in any pose and partly occluded. For this we believe that the part-based body plan approach is preferable.

Our detector works with a generalized feature pyramid spanning 8 scales and 36 orientations $0° \ldots 350°$. During training, the articular structure of each training image is clicked, and for each designated part a 14×24 subimage aligned with its axes and scaled to its size is extracted as shown in Figure 2. We learn 15 Support Vector or Relevance Vector Machines for the individual parts and the whole body, and during detection run each of them over the scale-orientation-position feature pyramid, then assemble the results as discussed in the next section.

Figure 2. A hand-labelled training image from the MIT database and its extracted body part subimages. Reading vertically from left to right: left upper arm, forearm, hand; left thigh, calf and foot; head, torso and whole body; right thigh, calf, foot; right upper arm, forearm and hand.

3.1 Feature Sets

The problem of choosing features for object recognition has received a lot of interest in recent years and numerous feature sets have been suggested, including image pixel values, wavelet coefficients and Gaussian derivatives. Wavelets are currently popular, but as a general representation for human body parts it is unclear whether standard (rectangular) or non-standard (square) wavelet constructions are most suitable [9,16]. Heisele *et al* obtained better results for their SVM face detector using gray levels rather than Haar wavelets [9]. Some authors also feel that wavelets are unsuitable as a general image representation because they represent point events rather than line or curve ones, and instead propose ridgelets and curvelets [2,5]. These might prove useful for detecting human limbs.

Here we leave such issues for future work and use a feature set consisting of the Gaussian filtered image and its first and second derivatives. Although simple, these features seem to represent the variations of body part detail effectively over a range of scales and orientations. The feature vector for an image rectangle at location-scale-orientation $[x_i, y_i, s_i, \theta_i]$ contains the absolute values of the responses of the six Gaussian $\sigma = 1$ filters $\{G, \nabla_x G, \nabla_y G, \nabla_{xx} G, \nabla_{xy} G, \nabla_{yy} G\}$ in the rectangle's (rescaled and reoriented) 14×24 window. There are thus $14 \times 24 \times 6 = 2016$ features per window. For color images we use only the luminance values Y. The absolute values of the filter responses are normalized across each image. The extracted features are not required to be scale- or orientation-invariant. On the contrary, we seek features that are tuned to

the characteristic scales and orientations of the detail in the aligned body-part images. Some examples of the feature vectors are shown in Figure 3.

To implement this, the Gaussian filters are computed using 9 rotated images from 0 to 80 degrees and 8 scales. We resample according to scale in each window, so the standard deviation of the filters in their resampled 14×24 windows is always 1. For any given size and orientation, we select the feature vector that best approximates the part-aligned region as an axis-aligned rectangle of height 24. This choice of primitives makes reasonably few assumptions about the nature of the features to be learned, which can be arbitrary combinations of shape, luminance and texture.

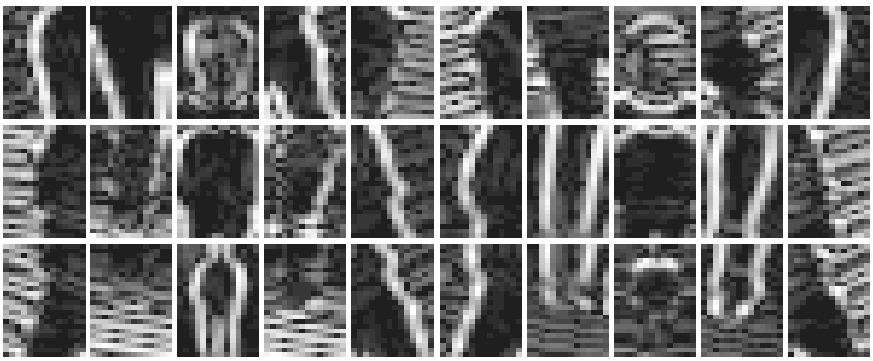


Figure 3. The $\nabla_x G$ and $\nabla_y G$ feature images for the example in Figure 2.

3.2 Training

Using the 2016-dimensional feature vectors for all body parts in the training set, we trained two linear classifiers for each part, one using a Support Vector Machine and the other using a Relevance Vector Machine. SVMs and RVMs are grounded on statistical learning results that suggest that they should give good classification performance even when there are relatively few training examples. Here we decided to put this claim to a severe test by training on the smallest sets of examples that give reasonable results — in our case, about 100.

We trained the 15 part classifiers individually against a common 'background' data set consisting of random pieces of the training images that do not contain people. Note that we are not attempting to learn isolated part detectors or multi-class part-type classifiers, but *reliable filters* for rejecting non-parts within an articulated body plan framework. We expect the overlap in appearance between different parts to be significant, but we do not want this to cause missed detections in ambiguous cases.

Support Vector Machines: SVMs are discriminant classifiers that give a yes/no decision, not a probability. However in our experiments we treat the SVM scores (scalar

products in feature space) as if they were log likelihoods for the body parts given the image values[1].

Relevance Vector Machines: RVMs [22,23] are Bayesian kernel methods that choose sparse basis sets using an 'Automatic Relevance Determination' [1] style prior that pushes non-essential weights to zero. They do not usually give significantly better error rates than the corresponding SVMs, but they do often give similar results with many fewer kernels. The functional form of the final classifier is the same as that of an SVM — only the fitted weights are different. Here we use logistic linear discriminant RVMs, whose output directly models the log-odds for a part versus a non-part at the given point. In this paper, we use RVMs mainly to reduce the number of kernels ('relevance vectors') and hence the computational complexity. The trained RVM classifiers typically use only 2–3 positive and 2–3 negative relevance vectors each, as compared to 100–200 support vectors for a comparable SVM classifier.

Currently we train the linear RVMs to make sparse use of *examples*, but they could also be trained to make sparse use of *features*. This would potentially mean that fewer image features would have to be extracted, and hence that the method would run faster. We plan to investigate this in future work.

3.3 Detection

We detect all of the body parts at once, in a single scan over the orientation-scale pyramid. The detection score for each part reduces to a simple convolution product against a mask containing the discriminant sum of weighted support or relevance vectors. Conceptually, this amounts to generalized template matching over images of local feature vectors, with weighted sums of training examples as templates. The nonlinearity of the process is hidden in the rectified, normalized local feature vectors. For efficiency in the assembly stage, we currently retain only the 50 best candidates for each part. The observed detection rates suggest that this strategy suffices for simple images, but it is not ideal for robustness against occlusions and we ultimately plan to use a more sophisticated strategy based on adaptive thresholds.

4 Parsing the body tree

In a non-articulated, image-aligned method such as that of Mohan [16], assembling the part detections is relatively straightforward: decompose the search window into subwindows, keep the highest score for the appropriate part in each subwindow, and compose the scores into a single, low-dimensional feature vector. Given these second-stage feature vectors, a single linear SVM can be learned for the overall body detection.

In our articulated, part-aligned method, the composition of part-models is only slightly more difficult, and can be cast as a combinatorial search: from all detected

[1] A more principled approach to converting the scores of a discriminant classifier to probabilities is as follows: run the detector over a validation set and fit density models to its positive-example and negative-example output scores. At any given score, the ratio of the positive-example density to the negative-example one is an estimate of the positive-to-negative odds ratio for detections at that score.

parts, search for the assemblies looking most like people. Since assemblies are naturally described as trees, efficient dynamic programming algorithms can be used to build the second-stage classifier, as we now describe.

4.1 Parsing/decoding algorithm

Given N candidate body part locations l_{kn} detected by each body part classifier C_k, we are looking for a 'parse' of the scene into one or more 'body trees'. One important subproblem is to assign a 'valid detection' or 'false alarm' label to each candidate, based not only on the candidate's scores, but on the local configuration between the candidates and its neighbours. Our approach relies on an extension of the Viterbi decoding algorithm, as described by Ioffe & Forsyth [13] and Felzenszwalb & Huttenlocher [6], which we sketch only briefly here. Given the detection scores $D_k(l_{kn})$ for all candidates $n = 1...N$, we search for the best candidate as a function of their direct parents $pa(n)$ in the body tree. For the leaves (i.e. hands, feet and head), this is computed by algorithm 1:

Algorithm 1 leaf location

$$B_k(l_{jm}) = \min_{\{n=1...N\}} -D_k(l_{kn}) + d_{kj}(l_{kn}, l_{jm})$$
$$l_k^*(l_{jm}) = \arg\min_{\{n=1...N\}} -D_k(l_{kn}) + d_{kj}(l_{kn}, l_{jm})$$

Based on this computation, we can score candidates from the bottom up, using the recursion algorithm 2:

Algorithm 2 bottom up

$$B_k(l_{jm}) = \min_{\{n=1...N\}} -D_k(l_{kn}) + d_{kj}(l_{kn}, l_{jm}) + \Sigma_{\{c|k=pa(c)\}} B_c(l_{kn})$$
$$l_k^*(l_{jm}) = \arg\min_{\{n=1...N\}} -D_k(l_{kn}) + d_{kj}(l_{kn}, l_{jm}) + \Sigma_{\{c|k=pa(c)\}} B_c(l_{kn})$$

At the root node we obtain the simple formula 3 for scoring the high level hypotheses.

Algorithm 3 root location

$$B_r = \min_{\{n=1...N\}} -D_r(l_{rn}) + \Sigma_{\{c|r=pa(c)\}} B_c(l_{rn})$$
$$L_r^* = \arg\min_{\{n=1...N\}} -D_r(l_{rn}) + \Sigma_{\{c|r=pa(c)\}} B_c(l_{rn})$$

Choosing the most probable root node, we can then assign the other nodes in a top down fashion by choosing $L_k^* = l_k^*(L_{pa(k)})$ for each node given its parent. Note that this algorithm has a complexity $O(MN^2)$ with M the number body parts and N the number of candidates per body part. As an example of the detection results obtained with this method, Figure 6 shows the three most probable parses for four test images, ranked in order of decreasing likelihood.

4.2 Learning the body tree

The cost functions used in our body tree model are based on geometric constraints on the relative positions of parts at a body articulation, as in Felzenszwalb & Huttenlocher [6]. Essentially, the articulation model is a linear combination of the differences between two joint locations, as predicted separately by the two body parts meeting at the articulation.

Algorithm 4 joint distance(l_i, l_j)

Compute joint location x_{ij}, y_{ij} given first body part location l_i
Compute joint location x_{ji}, y_{ji} given second body part location l_j
Return $d_{ij} = w^x_{ij}|x_{ij} - x_{ji}| + w^y_{ij}|y_{ij} - y_{ji}| + w^\theta_{ij}|\theta_i - \theta_j - \theta_{ij}| + w^s_{ij}|\log \frac{s_i}{s_j} - \log s_{ij}|$

Each body joint is parametrized by the relative sizes s_{ij} and angles θ_{ij} between its parts, and the four rigidity parameters $w^x_{ij}, w^y_{ij}, w^\theta_{ij}, w^s_{ij}$ governing the admissible range of apparent deformations of the articulation in position, size and orientation. We learned the relative sizes s_{ij} and angles θ_{ij} of each articulation by simply taking the average relative positions of all pairs of body parts over the training set.

To learn the rigidity parameters, we again used a Support Vector Machine. For each articulation A_{ij} between parts P_i and P_j, we learned a 'combination classifier' based on a five-dimensional feature vector $F^0_i = D_i + D_j$, $F^x_i = |x_{ij} - x_{ji}|$, $F^y_i = |y_{ij} - y_{ji}|$, $F^\theta_i = |\theta_i - \theta_j - \theta_{ij}|$, $F^s_i = |\log \frac{s_i}{s_j} - \log s_{ij}|$.

Using positive and negative examples from our training set, we used a linear SVM classifier to learn a set of weights $w^0_{ij}, w^x_{ij}, w^y_{ij}, w^\theta_{ij}, w^s_{ij}$ such that the score is positive for all positive example, and negative for all negative examples. We experimentally verified that the learned weights have the expected signs, $w^0_{ij} > 0$ and $w^x_{ij} < 0, w^y_{ij} < 0, w^\theta_{ij} < 0, w^s_{ij} < 0$, so that the learned model can indeed be related to the log-likelihood of the articulation

$$L(A_{ij}) = F^0_i - \frac{|w^x_{ij}|}{w^0_{ij}} F^x_i - \frac{|w^y_{ij}|}{w^0_{ij}} F^y_i - \frac{|w^\theta_{ij}|}{w^0_{ij}} F^\theta_i - \frac{|w^s_{ij}|}{w^0_{ij}} F^s_i$$

In our experiments with the MIT pedestrian database, the learned models performed slightly better than the naive approach of assigning equal weights to all parameters and all articulations, and we expect the method to be of even greater benefit for dealing with the more complicated cases of people in action such as running or jumping.

5 Implementation and results

We implemented and tested our method in Matlab. The system consists of several components. There is an interactive program for hand-labelling examples and storing the locations of the body joints and parts. Another function computes image pyramids and extracts image signatures at all locations x, y, s, θ . These are used both to generate feature vectors for SVM/RVM training, and to perform detection against the learned models. Finally, a parser based on the above dynamic programming approach reads

candidate locations from the 15 body part detectors and produces a ranked list of candidate assemblies.

We used MIT's public domain program SvmFu-3.0 to train the SVM classifiers. We trained the RVM classifiers in Matlab using a new algorithm that will be described in detail elsewhere.

5.1 Experimental setup

We selected 100 frontal images from the MIT pedestrian database and labelled their 15 parts, as shown in Figure 2. Each example is labelled by clicking 14 body joints. Occluded parts are clicked at their most likely (hidden) location, but flagged as occluded. Only visible parts are used to train the image part models, but the hidden parts can be included when training the geometric models. We also picked 5 background regions in each image, for use as negative examples. As a result, each body part classifier was trained with slightly less than 100 positive examples, and 500 negative examples.

Separate examples are needed for training and testing, so we selected and labelled another 100 images from the MIT pedestrian database to serve as a test set. This was used to evaluate the body part and assembly detectors.

5.2 Detection of body parts.

Detectors are traditionally compared by tracing ROC curves, i.e. true detection rates (recall) as a function of false alarm rates ($1-$precision). In our case the detectors must be tuned to function as filters, so most important parameter is the false alarm rate needed to achieve 'total recall'. Hence, we compared the two detectors by measuring the false detection rates required to detect all visible body parts in our test set. The resulting true positive rates for each part detector are shown in Figure 4.

As can be seen, individual part images are not very discriminative, so the absolute false alarm rates remain quite high. In fact, they become still higher (up to 15:1) once confusions between parts are included. Even so, the linking stage manages to resolve most of the ambiguity, and the number of candidates that have to be examined remains quite tractable, at most about 75 candidates per part for these images. Ignoring spatial contiguity, the worst-case number of body joint hypotheses is therefore $14 \times 75^2 = 78750$. In practice, we observed an average number closer to $14 \times 20^2 = 5600$ and used 50 candidates as a safe bet in all of our experiments. The RVM classifiers perform only slightly worse than their SVM counterparts, with mean false detection rates of 80.1% and 78.5% respectively. This is remarkable given the very small number of relevance vectors used by the RVM detectors. For the purpose of rapid filtering, the advantages of the RVM clearly outweigh their inconvenience.

Also note that the worst results are obtained for the torso (3) and head (2) models. The torso is probably the hardest body part to detect as it is almost entirely shapeless. It is probably best detected indirectly from geometric clues. In contrast, the head is known to contain highly discriminant features, but the training images contain a wide range of poses and significantly more training data (and perhaps some bootstrapping on false alarms) is probably needed to build a good detector.

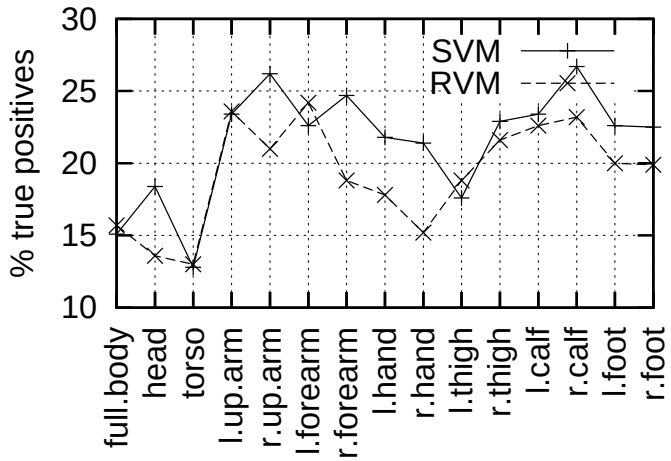


Figure 4. True positive rates for SVM and RVM body part detectors.

5.3 Detection of body trees

We evaluated the final body detector by visually comparing the best (highest probability) three configurations returned with the correct interpretation in each of the 100 test set images. Thus, the task was purely that of detecting humans using the 50 best candidates for each body part and the body tree model. Our first experiment used 100 training exemples. We obtained correct detections rates of 72 % using RVM scores and 83 % using SVM scores, while using a naive geometric model with uniform rigidity parameters for all of the body joints. We then learned a geometric model using labelled body joints from the 100 training images. We used the correct assemblies as positive examples, and circular permutations of the body parts as negative ones. Using the learned model, the correct detection rates improved to 74 % and 85 %. We should note that detection is a relatively easy task with this data set, and our method should be evaluated also with regards to the pose estimates. We plan to investigate this area quantitatively in later work. Qualitatively, we noted that a majority of the body parts were correctly positioned in only 36 % of the test images for RVM and 55 % for SVM.

In a second experiment, we increased the size of the training set to 200 examples. This resulted in a slight increase of the detection rates, to 76 % for SVM and 88 % for RVM, and a much vaster improvement of the pose estimates, resulting in qualitatively correct poses in 54 % of the test examples for RVM and 75 % for SVM.

6 Discussion and Future Work

The good detection rates achieved by the method make a convincing case that the body-plan strategy is applicable to real problems in image and video indexing. We plan to extend this work to video, where we hope to improve the detection rates even further

Figure 5. Part detection results from test collection.

by making use of temporal and kinematic constraints. But the construction of the image pyramid is computationally expensive, and we plan to move to a more efficient implementation, which could rely on a more thorough selection of the feature vectors. One way to do this will be to use RVM classifiers that learn relevant features rather than relevant examples. As a complement, Sidenbladh & Black's [20,21] approach for learning the image statistics of people vs. background could prove useful for learning better models by selecting better features. In the assembly phase, the complexity of the dynamic programming algorithm is quadratic n the number of candidate parts which need to be stored, which in turn depends on the precision of the individual body part detectors. By fine-tuning the body part detectors, we expect to achieve significant improvements also in the overall performance of the global detector.

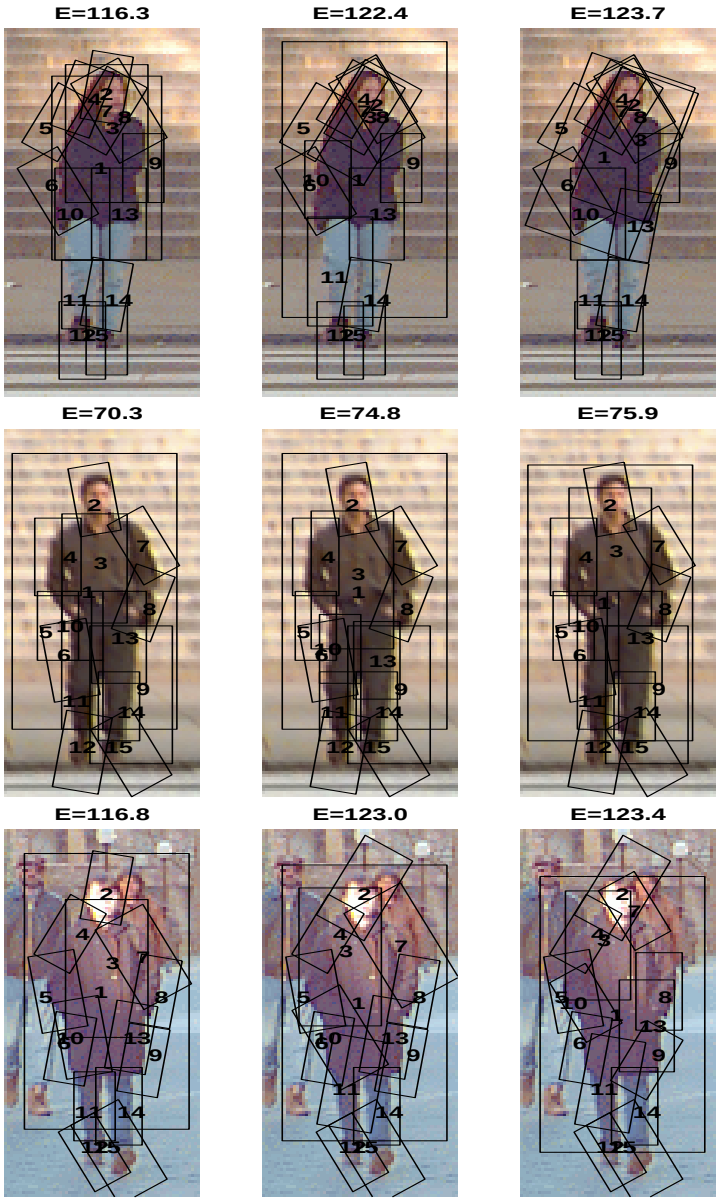


Figure 6. Ranked detections and their energies, using the learned body model and SVM scores.

Further work will be needed for assessing the correctness of the detection and pose estimation results in a more systematic way and for 'bootstrapping' the learned models (adding examples on which our current model fails, and retraining). Even without boot-

strapping, we have verified experimentally that the quality of the body part classifiers is improved significantly by increasing the size of the training data. We will need to quantify this observation in future work.

We also plan to extend the method to handle multiple persons in a greater variety of backgrounds and poses, by explicitly representing occlusions in the decoding process as in the work of Coughlan et al. [3] or by introducing mixtures of partial body trees, as in the recent proposal made by Ioffe and Forsyth [11,12]. The cost functions used to evaluate the assembly of the body plans could also benefit from a richer geometric model and additional photometric constraints (e.g. similarity of color and texture between the body parts for the same person). There are cases where we would like to move even further away from the human anatomic model, and replace it with a small set of 'clothing models', which could be learned in much the same way and provide additional flexibility. Those are avenues for further experimental work.

7 Conclusion

Detecting humans is a challenging problem in computer vision, with considerable practical implications for content-based indexing. We believe we have reached three useful concusions with the work reported in this paper. Firstly, it is possible to learn appearance models for human body parts from examples and to use them as input to a body plan parser, at least for a modest-size problem such as pedestrian detection. Secondly, we have been able to learn geometric models for the combination of the detected parts, allowing us to robustly estimate the likelihood of a body part assembly, without recourse to sampling or HMM distributions, which require thousands of examples to be learned efficiently. Thirdly, the learned models lead to an efficient decoding algorithm that combines kernel based learning and dynamic programming techniques, and is simple enough to be extended to video sequences.

References

1. C. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.
2. E. Candes and D. Donoho. Curvelets – a surprisingly effective nonadaptive representation for objects with edges. In L. L. Schumaker et al., editor, *Curves and Surfaces*. Vanderbilt University Press, 1999.
3. J. Coughlan, D. Snow, C. English, and A. Yuille. Efficient optimization of a deformable template using dynamic programming. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2000.
4. N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and other Kernel Based Learning Methods*. Cambridge University Press, 2000.
5. M. Do and M. Vetterli. Orthonormal finite ridgelet transform for image compression. In *Int. Conference on Image Processing*, volume 2, pages 367–370, 2000.
6. Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient matching of pictorial structures. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2000.
7. M. Fischler and R. Elschlager. The representation and matching of pictorial structures. *IEEE Trans. Computer*, C-22:67–92, 1973.

8. D. Forsyth and M. Fleck. Body plans. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1997.
9. B. Heisele, T. Poggio, and M. Pontil. Face detection in still gray images. Technical report, AI Memo 1687, Massachusetts Institute of Technology, 2000.
10. D. Hogg. Model-based vision: A program to see a walking person. *Image and Vision Computing*, 1(1):5–20, 1983.
11. Sergey Ioffe and David Forsyth. Human tracking with mixtures of trees. In *Proc. Int. Conf. Computer Vision*, 2001.
12. Sergey Ioffe and David Forsyth. Mixtures of trees for object recognition. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2001.
13. Sergey Ioffe and David Forsyth. Probabilistic methods for finding people. *Int. J. of Computer Vision*, 43(1), 2001.
14. S. Ju, M. Black, and Y. Yacoob. Cardboard people: a parameterized model of articulated image motion. In *Int. Conference on Automatic Face and Gesture Recognition*, 1996.
15. D. Marr and H.K. Nishihara. Representation and recognition of the spatial organization of three dimensional structure. *Proceedings of the Royal Society of London B*, 200:269–294, 1978.
16. Anuj Mohan, Constantine Papageorgiou, and Tomaso Poggio. Example-based object detection in images by components. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 23(4), 2001.
17. D. Morris and J. Rehg. Singularity analysis for articulated object tracking. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1998.
18. C. Papageorgiou. Object and pattern detection in video sequences. Technical report, Master's thesis, Massachusetts Institute of Technology, 1997.
19. K. Rohr. Incremental recognition of pedestrians from image sequences. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 8–13, 1993.
20. H. Sidenbladh and M. Black. Learning the statistics of people in images and video. *Int. Journal of Computer Vision*, 2001.
21. H. Sidenbladh, F. Torre, and M. Black. A framework for modeling the appearance of 3d articulated figures. In *Int. Conference on Automatic Face and Gesture Recognition*, 2000.
22. M. Tipping. The relevance vector machine. In *Advances in Neural Information Processing Systems*. Morgan Kaufmann, 2000.
23. M. E. Tipping. Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, 2001.
24. V. Vapnik. *Statistical Learning Theory*. Wiley, 1998.
25. Liang Zhao and Chuck Thorpe. Recursive context reasoning for human detection and parts identification. In *IEEE Workshop on Human Modeling, Analysis and Synthesis*, 2000.

Int J Comput Vis (2008) 79: 247–269
DOI 10.1007/s11263-007-0116-2

Human Motion Tracking with a Kinematic Parameterization of Extremal Contours

David Knossow · Rémi Ronfard · Radu Horaud

Received: 26 July 2006 / Accepted: 19 November 2007 / Published online: 8 December 2007

Abstract This paper addresses the problem of human motion tracking from multiple image sequences. The human body is described by five articulated mechanical chains and human body-parts are described by volumetric primitives with curved surfaces. If such a surface is observed with a camera, an extremal contour appears in the image whenever the surface turns smoothly away from the viewer. We describe a method that recovers human motion through a kinematic parameterization of these extremal contours. The method exploits the fact that the observed image motion of these contours is a function of both the rigid displacement of the surface and of the relative position and orientation between the viewer and the curved surface. First, we describe a parameterization of an extremal-contour point velocity for the case of developable surfaces. Second, we use the zero-reference kinematic representation and we derive an explicit formula that links extremal contour velocities to the angular velocities associated with the kinematic model. Third, we show how the chamfer-distance may be used to measure the discrepancy between predicted extremal contours and observed image contours; moreover we show how the chamfer distance can be used as a differentiable multi-valued function and how the tracker based on this distance can be cast into a continuous non-linear optimization framework. Fourth, we describe implementation issues associated with a practical human-body tracker that may use an arbitrary number of cameras. One great methodological and practical advantage of our method is that it relies neither on model-to-image, nor on image-to-image point matches. In practice we model people with 5 kinematic chains, 19 volumetric primitives, and 54 degrees of freedom; We observe silhouettes in images gathered with several synchronized and calibrated cameras. The tracker has been successfully applied to several complex motions gathered at 30 frames/second.

Keywords Articulated motion representation · Human-body tracking · Zero-reference kinematics · Developable surfaces · Extremal contours · Chamfer distance · Chamfer matching · Multiple-camera motion capture

D. Knossow · R. Ronfard · R. Horaud (✉)
INRIA Rhône-Alpes, 655, avenue de l'Europe,
38330 Montbonnot Saint-Martin, France
e-mail: radu.horaud@inrialpes.fr

1 Introduction and Background

In this paper we address the problem of tracking complex articulated motions from multiple image sequences. The problem of articulated motion (such as human-body motion) representation and tracking from 2-D and 3-D visual data has been thoroughly addressed in the recent past. The problem is difficult because it needs to solve an inverse kinematic problem, namely the problem of finding the parameters characterizing the control space (the space spanned by the articulated parameters) from a set of measurements performed in the observation space. In general this problem cannot be solved explicitly because the dimensionality of the observation space is much smaller than the dimensionality of the control space. More formally, the problem can be stated as the following minimization problem:

$$\min_{\mathbf{\Phi}} E(\mathcal{Y}, \mathcal{X}(\mathbf{\Phi})) \tag{1}$$

where $\mathcal{Y}$ denotes a set of observations, $\mathcal{X}$ denotes a set of predictions using the direct kinematic model, and $\mathbf{\Phi}$ is the vector of motion parameters to be estimated.

In this paper we will embed the human-motion tracking into the minimization problem defined by (1). We will emphasize a human-body model composed of articulated mechanical chains and of rigid body parts. Each such part is defined by a developable surface. An intrinsic property of such a surface is that it projects onto an image as a pair of straight extremal contours (an extremal contour appears in an image whenever a curved surface turns smoothly away from the viewer). We develop a direct kinematic representation of extremal contours based on the differential properties of developable surfaces and on the zero-reference kinematic representation of articulated chains with rotational joints. This kinematic description encapsulates the constrained articulated motions as well as a free rigid motion and allows us to predict both the position and the velocity of extremal contours.

Therefore, human motion tracking may be formulated as the problem of minimizing equation (1) using the chamfer distance between predicted extremal contour points, $\mathcal{X}(\mathbf{\Phi})$ and contour points detected in images, $\mathcal{Y}$. We show how the chamfer distance can be used as a differentiable multi-valued function and how the tracker based on this distance can be cast into a non-linear optimization framework. Even if, in theory, one camera may be sufficient for recovering the motion parameters, we show that a multiple-camera setup brings in the necessary robustness for implementing the tracker.

There is a substantial body of computer vision literature on articulated motion tracking and excellent reviews can be found in (Gavrila 1999), (Moeslund et al. 2006), and (Forsyth et al. 2006).

Monocular approaches generally require a probabilistic framework such as in (Deutscher et al. 2000; Toyama and Blake 2002; Song et al. 2003; Agarwal and Triggs 2006) to cite just a few. The probabilistic formulation has the attraction that both prior knowledge and uncertainty in the data are handled in a systematic way. The first difficulty with these methods is that the image data must be mapped onto a vector space with fixed dimension such that statistical methods can be easily applied. The second difficulty is to establish a relationship (between the space of articulated poses and the space spanned by the vectors mentioned above) that should be learnt prior to tracking. This is not an obvious task because it is virtually impossible to scan in advance the space of all possible poses of an articulated object with many degrees of freedom. Other methods attempted to recover articulated motion from image cues such as optical flow through sophisticated non-linear minimization methods (Bregler et al. 2004; Sminchisescu and Triggs 2003; Sminchisescu and Triggs 2005).

A second class of approaches relies on multiple-video sequences gathered with multiple cameras. One pre-requisite of such a camera setup is that the frames are finely synchronized—a not so obvious task. One can either perform some kind of 3-D surface or volumetric reconstruction prior to tracking (Cheung et al. 2005a; Mikic et al. 2003; Plaenkers and Fua 2003), or use 2-D features such as silhouettes, color, or texture (Delamarre and Faugeras 2001; Drummond and Cipolla 2001; Gavrila and Davis 1996; Kakadiaris and Metaxas 2000). Others used a combination of both 2-D and 3-D features (Plaenkers and Fua 2003; Kehl and Van Gool 2006).

In (Cheung et al. 2005a) and (Cheung et al. 2005b) the authors develop a shape-from-silhouette paradigm that is applied to human motion tracking. They describe a volumetric-based method that assigns a voxel to a body part and a method based on colored surface points (CSP) that combines silhouette-based reconstruction with color information. Both these methods require 3-D reconstruction from perfect silhouettes. A similar voxel-based method is described in (Mikic et al. 2003). The tracker minimizes a cost function that measures the consistency between the 3-D data (a set of voxels) and the model (ellipsoids linked within an articulated chain).

In (Kehl and Van Gool 2006) image edges, color, and a volumetric reconstruction are combined to take advantage of these various 2-D and 3-D cues. The authors notice that while volumetric data are strong features, image edges are needed for fine localization and hence accurate pose computation. The use of edges implies that one is able to predict model edges. Since the authors use superquadrics, it is necessary to compute their contour generator (referred in Kehl and Van Gool 2006 as the occluding contour) and project it in the images using perspective projection. This is done through a series of approximations since a closed-form solution is difficult to compute. Finally the authors cast the tracking problem into a stochastic optimization framework that uses a three-term cost function for surface, edge, and color alignment. The experimental setup uses 16 cameras. A similar approach based on both 3-D data (depth from a stereo image pair) and 2-D silhouettes is proposed in (Plaenkers and Fua 2003).

In (Kakadiaris and Metaxas 2000) and (Delamarre and Faugeras 2001) two similar methods are presented. Multiple-camera tracking is performed by projecting the 3-D model onto the images and building a cost function that measure the distance between the projected model and the 2-D silhouettes. This distance sums up the squares of the projected-model-point-to-silhouette-point assignments to estimate the 2-D force field and to infer the "physical forces" that allow the alignment.

In this paper we use neither color nor photometric information because it is not robust to illumination changes. We do not use texture because it is not a shape-invariant feature. We decided to concentrate on contours because they have been recognized as strong cues for representing shape (Koenderink 1990; Forsyth and Ponce 2003) and therefore

the tracker that we implemented projects predicted model contours onto the images and compares them with observed contours (edges, silhouettes, etc.). Nevertheless, the tasks of computing contours from 3-D models, of projecting these contours onto images, and of comparing them with observed ones are not straightforward. Previous methods have not made explicit the analytic representation allowing the mapping of articulated objects (and their surfaces) onto 2-D edges or silhouettes. Formally, a silhouette is the *occluding contour* (Barrow and Tenenbaum 1981) that separates an object from the background. Occluding contours are built up of *discontinuity* and *extremal* contours. The former correspond to sharp edges arising from surface discontinuities. The latter occur where a curved surface turns smoothly away from the viewer.

In the case of sharp edges there are well documented methods allowing for an explicit (analytic) representation of the mapping between the object's constrained (articulated) motion parameters and the observed image contours both under orthography (Bregler et al. 2004) and under perspective projection (Drummond and Cipolla 2001; Martin and Horaud 2002). In the presence of smooth surfaces, an extremal contour is the projection of a *contour generator*—a virtual contour that lies onto the surface where the lines of sight are tangent to the surface. Therefore, the apparent image motion of an extremal contour is a function of both the motion of the object itself and the motion of the contour generator, the latter being a function of the relative position of the object's surface with respect to the viewer. It turns out that the link between the differential properties of certain classes of surfaces and the rigid motion of these surfaces has barely been addressed.

In more detail, we use *elliptical cones* to model body parts. These shapes belong to a more general class of developable surfaces that have interesting differential properties that were not fully exploited in the past. Elliptical cones in particular and developable surfaces in general project onto images as a set of straight lines. By deliberately considering only these contours we simplify both the tasks of interpreting the image contours and of comparing them to the predicted object contours. Moreover, the body parts are joined together to form an articulated structure composed of five *open kinematic chains*. Therefore, each body-part motion is composed of two motions: a motion constrained by a number of rotational joints (the motion of its associated kinematic chain) and a free motion, i.e., the motion of the root body-part with respect to a world reference frame. We derive an analytic expression for the motion of a predicted extremal-contour point as a function of both the body-part motion as well as the motion of its contour generator lying onto the curved surface of that body part. Figure 1 briefly illustrates how the method operates in practice.

Therefore, the problem of articulated motion tracking may be formulated as the problem of minimizing a metric between image contours (gathered simultaneously with several cameras) and extremal contours (predicted from the model). There are several ways of defining a distance between two contours, including the sum of squares of the point-to-point distances, the Hausdorff distance, the chamfer distance, and so forth. We decided to capitalize onto the chamfer distance, and unlike previous approaches, we developed an analytic expression allowing us to compare the real-valued contour points predicted from the model with the chamfer-distance image computed from binary-valued image contours. This image-to-model metric thus defined does not require point-to-point matches, its computation is very efficient, and it can be analytically differentiated. We analyse in detail the numerical conditioning of the tracker, which amounts to the rank analysis of the Jacobian associated with the direct kinematic model. Although, in principle, one camera may be sufficient for gathering enough data, we claim that a multiple-camera setup provides the redundancy that is absolutely necessary for robust tracking.

Paper Organization The remainder of this paper is organized as follows. In Sect. 2 we consider the case of developable surfaces and we show that their contour generators are rulings of the surface. We derive a closed-form solution for the velocity of the contour generators (and of the corresponding extremal contours) as a function of the kinematic screw associated with the motion of the surface. In Sect. 3 we develop an explicit solution for the human-body kinematics using the zero-reference kinematic model and in Sect. 4 we derive the Jacobian that maps joint and free-motion velocities onto the 2-D velocity of an extremal-contour point. Section 5 describes in detail how to fit predicted extremal contours to detected image contours and how to carry out the minimization process using the chamfer distance. Section 6 describes experiments performed with simulated and real data. Finally, Sect. 7 draws some conclusions and give directions for future work.

2 The Kinematics of Extremal Contours

2.1 Definitions and Notations

We use shapes with smooth surfaces in order to represent rigid body parts. Each such body-part is linked to a *root* body-part through a kinematic chain of body parts. Each joint in the kinematic chain—the link between two adjacent body parts—is modeled by a rotational joint. Each such joint may have one, two, or three degrees of freedom. Moreover, the root body-part is allowed to freely move in the 3-D space with six degrees of freedom (three rotations and three translations).

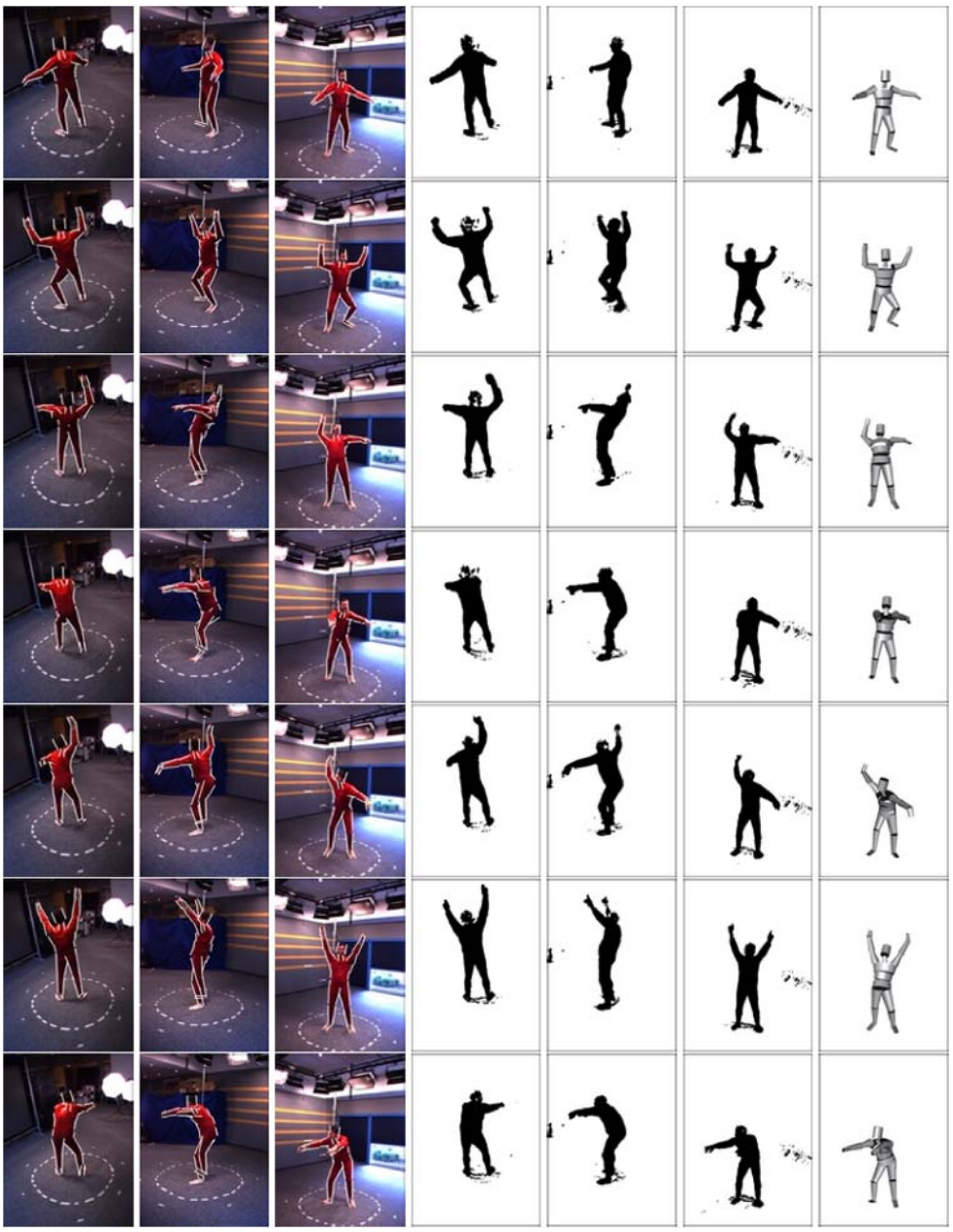


Fig. 1 An example of human-motion tracking based on extremal contours and using six cameras. The extremal contours fitted to the image data are shown superimposed onto the raw images. The tracker uses image silhouettes to fit the parameterized extremal contours to the data. The recovered pose of the human-body model is shown from the viewpoint of the third camera. There are 250 frames in these six image sequence. Notice that this apparent simple gesture (raising the arms and then leaning forward) involves almost all the degrees of freedom of the model as well as a motion of the root body-part

Therefore, the motion of any part of the kinematic chain is obtained by a combination of a *constrained motion* and of a *free motion*. We denote by $\mathbf{\Phi} = (\phi_1, \ldots, \phi_n)$ all these motion parameters. The first q parameters correspond to the motion of the root with $q \leq 6$ and the remaining p parameters correspond to the joint angles: $n = q + p$. The kinematic parameterization will be made explicit in the next section. In this section we will describe the motion of a body-part by a 3×3 rotation matrix $\mathbf{R}$ and by a 3-D translation vector $\boldsymbol{t}$. Both these rotation and translation are in turn parameterized by $\mathbf{\Phi}$, i.e., we will have $\mathbf{R}(\mathbf{\Phi})$ and $\boldsymbol{t}(\mathbf{\Phi})$.

It will also be convenient to consider a body part as a rigid object in its own right. The *pose of a rigid object* is described by six parameters and let $\boldsymbol{r}$ be the pose vector. If a body-part is treated as a free-moving rigid body, then the 6 components of $\boldsymbol{r}$ are the free parameters. If a body-part is treated as a

component of a kinematic chain, $\boldsymbol{r}$ is parameterized by $\boldsymbol{\Phi}$, i.e., $\boldsymbol{r}(\boldsymbol{\Phi})$. Finally we denote by $\dot{\boldsymbol{x}}$ the time derivative of $\boldsymbol{x}$.

We consider now the smooth surface of a body-part. This surface projects onto the image as an extremal contour. The apparent image motion of such an extremal contour depends on the motion of the body-part and on the local shape of the surface. Indeed, let's consider the *contour generator* that lies onto the smooth surface—the locus of points where the surface is tangent to the lines of sight originating from the camera's center of projection. When the surface moves, the contour generator moves as well and it's motion is constrained both by the rigid motion of the surface and by the relative position of the surface with respect to the camera. Therefore, the contour generator has two motion components and we must explicitly estimate these two components.

First, we will determine the constraints that formally define the contour generator. The extremal contour is simply determined by projecting the contour generator onto the image plane. Second, we will derive a closed-form solution for the *extremal-contour Jacobian*, i.e., the Jacobian matrix that maps 3-D joint velocities onto 2-D contour-point velocities.

2.2 The Contour-Generator Constraint and Extremal Contours

Let X be a 3-D point that lies onto the smooth surface of a body part, and let $\boldsymbol{X} = (X_1, X_2, X_3)$ be the coordinates of this point in the body-part frame, Fig. 2. Without loss of generality, the camera frame will be chosen to be identical to the world frame. Hence, the world coordinates of X are:

$$\boldsymbol{X}^w = \mathbf{R}(\boldsymbol{\Phi})\boldsymbol{X} + \boldsymbol{t}(\boldsymbol{\Phi}). \tag{2}$$

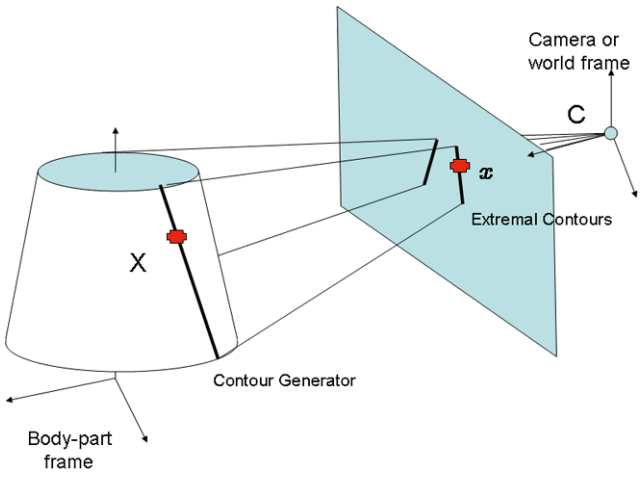


Fig. 2 A truncated elliptical cone is an example of a developable surface used to model a body part. Such a surface projects onto an image as a pair of *extremal contours*. The 2-D motion of these extremal contours is a function of both the motion of the body-part itself as well as the sliding of the *contour generator* along the smooth surface of the part

The contour generator is the locus of points lying onto the surface where the lines of sight (originating at the optical center of the camera and passing through image points) are tangent to that surface. Obviously, the contour generator is defined by:

$$(\mathbf{R}\boldsymbol{n})^\top (\mathbf{R}\boldsymbol{X} + \boldsymbol{t} - \boldsymbol{C}) = 0 \tag{3}$$

where the surface normal $\boldsymbol{n}$ is defined by the following cross-product:

$$\boldsymbol{n} = \frac{\partial \boldsymbol{X}}{\partial z} \times \frac{\partial \boldsymbol{X}}{\partial \theta} = \boldsymbol{X}_z \times \boldsymbol{X}_\theta. \tag{4}$$

Here the couple (z, θ) is a parameterization of the body-part's surface and $\boldsymbol{C}$ denotes the camera's optical center. The equation above becomes:

$$\boldsymbol{X}^\top \boldsymbol{n} + (\boldsymbol{t} - \boldsymbol{C})^\top \mathbf{R}\boldsymbol{n} = 0 \tag{5}$$

or:

$$(\boldsymbol{X} + \boldsymbol{m})^\top \boldsymbol{n} = 0 \tag{6}$$

with $\boldsymbol{m} = \mathbf{R}^\top(\boldsymbol{t} - \boldsymbol{C})$. Equation (6) is the contour-generator constraint that must be satisfied at each time instant. Once the contour generator is determined, the 2-D extremal contour (the projection of the contour generator) can be found in the camera frame from:

$$\begin{pmatrix} s\boldsymbol{x} \\ s \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} \boldsymbol{X}^w \\ 1 \end{pmatrix}. \tag{7}$$

2.3 The Contour Generator of a Developable Surface

It would be difficult to treat the general case of curved surfaces. An interesting case is the class of developable surfaces which are a special case of ruled surfaces (Do Carmo 1976). We prove the following result:

Proposition *Under perspective projection, the contour generators of a developable surface are rulings of the surface, i.e., they are line segments.*

This also means that the extremal contours of a developable surface are straight lines. In practice we need to consider surfaces that are well suited to model body parts. We will use elliptical cones but the result of this section allows one to use any kind of developable surfaces.

Consider a differentiable one-parameter family of straight lines $(\boldsymbol{\alpha}(\theta), \boldsymbol{\beta}(\theta))$ where to each θ are assigned a 3-D point $\boldsymbol{\alpha}(\theta)$ and a 3-D vector $\boldsymbol{\beta}(\theta)$, so that both $\boldsymbol{\alpha}(\theta)$ and $\boldsymbol{\beta}(\theta)$ depend differentiably on θ. The parametrized surface:

$$\boldsymbol{X}(\theta, z) = \boldsymbol{\alpha}(\theta) + z\boldsymbol{\beta}(\theta) \tag{8}$$

is called a ruled surface, and the normal to this surface is given by (4):

$$\boldsymbol{n} = \boldsymbol{X}_\theta \times \boldsymbol{X}_z = (\boldsymbol{\alpha}' + z\boldsymbol{\beta}') \times \boldsymbol{\beta}. \tag{9}$$

Since a developable surface is a ruled surface whose Gaussian curvature is null everywhere on the surface, one can show ((Do Carmo 1976), (Kreyzig 1991)) that the normal to this surface can be written as:

$$\boldsymbol{n} = (1 + bz)\boldsymbol{\beta}' \times \boldsymbol{\beta}. \tag{10}$$

Notice that the direction of the normal is given by the cross-product of $\boldsymbol{\beta}'$ and $\boldsymbol{\beta}$ and it depends only on the parameter θ. Using this parameterization of the normal, we can rewrite the contour generator constraint, (6), for developable surfaces as follows:

$$(\boldsymbol{\alpha}(\theta) + \boldsymbol{m})^\top (\boldsymbol{\beta}'(\theta) \times \boldsymbol{\beta}(\theta)) = 0. \tag{11}$$

One should notice that this contour-generator constraint involves only the surface parameter θ and not the z parameter. Therefore, any solution of (11), say $\hat{\theta}$, will yield the entire ruling line $\boldsymbol{X}(\hat{\theta}, z)$. This proves that under perspective projection, the contour generators of a developable surface are rulings of the surface, i.e. line segments. As a result, *the kinematics of the contour generators are fully determined by the evolution of the solutions $\hat{\theta}(t)$ of the contour generator equation over time*.

2.4 Truncated Elliptical Cones

In practice will model body parts with truncated elliptical cones. Such a shape is bounded by two planar faces which produce discontinuity contours, as well as a curved surface which produces a pair of extremal contours. The latter can be easily parameterized in cylindrical coordinates by an angle θ and a height z as a ruled surface:

$$\boldsymbol{X}(\theta, z) = \begin{pmatrix} a\cos\theta \\ b\sin\theta \\ 0 \end{pmatrix} + z \begin{pmatrix} ak\cos\theta \\ bk\sin\theta \\ 1 \end{pmatrix} \tag{12}$$

where a and b are minor and major half-axes of the elliptical cross-section, k is the tapering parameter of the cone and $z \in [z_1, z_2]$. It is straightforward to verify that an elliptical cone is a developable surface. Below we provide an analytical expression of its associated contour generators.

With this parametrization, (11) can be easily expanded to yield a trigonometric constraint of the form $F\cos\theta + G\sin\theta + H = 0$ where F, G and H depend on $\mathbf{R}(\boldsymbol{\Phi})$, $\boldsymbol{t}(\boldsymbol{\Phi})$ and C while they are independent of the parameter z. In order to solve this equation and find its roots we use the standard trigonometric substitution, i.e., $\tan\frac{\theta}{2}$ and obtain a second-degree polynomial:

$$(H - F)\tan^2\frac{\theta}{2} + 2G\tan\frac{\theta}{2} + (F + H) = 0. \tag{13}$$

This equation has two real solutions, θ_1 and θ_2, whenever the camera's optical center lies outside the cone that defines the body part (a constrained that is rarely violated). Therefore, in the case of elliptical cones the contour generator is composed of two straight lines parameterized by z, i.e., $X(\boldsymbol{\Phi}, \theta_1, z)$ and $X(\boldsymbol{\Phi}, \theta_2, z)$.

2.5 The Motion of Extremal Contours

We turn our attention back to extremal contours—the projection onto the image plane of the contour generator. We denote by $\boldsymbol{x} = (x_1, x_2)$ the real-valued image coordinates of an extremal-contour point, i.e., (7). The motion of this point depends on both:

- The *rigid motion* of the body-part with respect to the world reference frame, and
- the *sliding motion* of the contour generator onto the part's curved surface, as the relative position and orientation of this part varies with respect to the camera.

We formally derive the motion of an extremal contour point in terms of these two components. The 2-D velocity of an extremal-contour point is:

$$\frac{d\boldsymbol{x}}{dt} = \frac{d\boldsymbol{x}}{d\boldsymbol{X}^w}\frac{d\boldsymbol{X}^w}{d\boldsymbol{r}}\frac{d\boldsymbol{r}}{d\boldsymbol{\Phi}}\frac{d\boldsymbol{\Phi}}{dt}. \tag{14}$$

Vector $\boldsymbol{X}^w$, already defined by (2), denotes the contour-generator point in world coordinates. Its projection is obtained from (7):

$$x_1 = \frac{X_1^w}{X_3^w}, \qquad x_2 = \frac{X_2^w}{X_3^w}. \tag{15}$$

We recall that $\boldsymbol{r}$ was already defined in Sect. 2.1 and it denotes the pose parameters associated with the body-part. Since the latter is linked to the root part by a kinematic chain, $\boldsymbol{r}$ is in it's turn parameterized by $\boldsymbol{\Phi}$. We have:

- The first term of the right-hand side of (14) is the image Jacobian denoted by $\mathbf{J}_I$:

$$\frac{d\boldsymbol{x}}{d\boldsymbol{X}^w} = \mathbf{J}_I = \begin{bmatrix} 1/X_3^w & 0 & -X_1^w/(X_3^w)^2 \\ 0 & 1/X_3^w & -X_2^w/(X_3^w)^2 \end{bmatrix}. \tag{16}$$

- The second term is a transformation that allows to determine the velocity of a point from the motion of the part on which this point lies. When the point is rigidly attached to the part, this transformation is given by matrix $\mathbf{A}$ (see below). When the point slides onto the smooth surface there

is a second transformation— matrix $\mathbf{B}$—that remains to be determined:

$$\frac{d\boldsymbol{X}^w}{d\boldsymbol{r}} = \mathbf{A} + \mathbf{B}. \tag{17}$$

- The third term is the Jacobian of the kinematic chain that links the body part to a root body part and to a world reference frame. This Jacobian matrix will be denoted by $\mathbf{J}_H$.
- The fourth term is the vector composed of both the joint velocities and the velocity of the root body part.

With these notations, (14) becomes:

$$\dot{\boldsymbol{x}} = \frac{d\boldsymbol{x}}{d\boldsymbol{\Phi}}\dot{\boldsymbol{\Phi}} \tag{18}$$

where:

$$\frac{d\boldsymbol{x}}{d\boldsymbol{\Phi}} = \mathbf{J}_I(\mathbf{A} + \mathbf{B})\mathbf{J}_H \tag{19}$$

is the extremal-contour Jacobian that will be used by the tracker. It is useful to introduce the kinematic-screw notation, i.e., a six dimensional vector concatenating the rotational velocity, $\boldsymbol{\Omega}$, and the translational velocity, $\boldsymbol{V}$ (see below):

$$\frac{d\boldsymbol{r}}{dt} = \begin{pmatrix} \boldsymbol{\Omega} \\ \boldsymbol{V} \end{pmatrix}. \tag{20}$$

The velocity of an extremal-contour point can therefore be written as:

$$\dot{\boldsymbol{x}} = \mathbf{J}_I(\mathbf{A} + \mathbf{B}) \begin{pmatrix} \boldsymbol{\Omega} \\ \boldsymbol{V} \end{pmatrix}. \tag{21}$$

Let us now make explicit the 3×6 matrices $\mathbf{A}$ and $\mathbf{B}$. By differentiation of (2), we obtain:

$$\dot{\boldsymbol{X}}^w = \dot{\mathbf{R}}\boldsymbol{X} + \dot{\boldsymbol{t}} + \mathbf{R}\dot{\boldsymbol{X}}. \tag{22}$$

Equation (22) reveals that unlike the motion of a point that is rigidly attached to a surface, the motion of a contour-generator point has two components:

- A component due to the rigid motion of the smooth surface, $\dot{\mathbf{R}}\boldsymbol{X} + \dot{\boldsymbol{t}}$, and
- a component due to the sliding of the contour generator onto this smooth surface, $\mathbf{R}\dot{\boldsymbol{X}}$.

2.5.1 The Rigid-Motion Component

The first component in (22) can be parameterized by the kinematic screw and it becomes:

$$\dot{\mathbf{R}}\boldsymbol{X} + \dot{\boldsymbol{t}} = \dot{\mathbf{R}}\mathbf{R}^\top(\boldsymbol{X}^w - \boldsymbol{t}) + \dot{\boldsymbol{t}} = \mathbf{A}\begin{pmatrix} \boldsymbol{\Omega} \\ \boldsymbol{V} \end{pmatrix} \tag{23}$$

with $[\boldsymbol{\Omega}]_\times = \dot{\mathbf{R}}\mathbf{R}^\top$, $\dot{\boldsymbol{t}} = \boldsymbol{V}$, and where $\mathbf{A}$ is the 3×6 matrix that allows to compute the velocity of a point from the kinematic screw of the rigid-body motion:

$$\mathbf{A} = [[\boldsymbol{t} - \boldsymbol{X}^w]_\times \ \mathbf{I}_{3\times 3}]. \tag{24}$$

The notation $[\boldsymbol{m}]_\times$ stands for the 3×3 skew-symmetric matrix associated with the 3-vector $\boldsymbol{m}$. Vectors $\boldsymbol{\Omega}$ and $\boldsymbol{V}$ can be concatenated to form a 6-vector $(\boldsymbol{\Omega}\ \boldsymbol{V})^\top$ which is known as the kinematic screw—the rotational and translational velocities of the body part in world coordinates. This factorization is strictly equivalent with $\boldsymbol{V} = \dot{\boldsymbol{t}} - \dot{\mathbf{R}}\mathbf{R}^\top\boldsymbol{t}$ and $\mathbf{A} = [[\boldsymbol{X}^w]_\times \ \mathbf{I}]$.

2.5.2 The Sliding-Motion Component

It is interesting to notice that, although the link between image contours and smooth surfaces has been thoroughly studied in the past, the problem of inferring the velocity of these contours when the smooth surface undergoes a general 3-D motion has not yet been addressed. In the general case, the sliding-motion component is a complex function of both the local surface shape and of the relative motion between the surface and the observer. The problem is strongly linked to the problem of computing the aspects of a smooth surface (Koenderink 1990) and (Forsyth and Ponce 2003) (Chaps. 19 and 20). Both these textbooks treat the case of a static object viewed under orthographic projection.

We establish a mathematical formalism for developable surfaces, i.e., (Do Carmo 1976) when they are viewed under perspective projection. As it has been shown above, the contour generators are rulings of the surface and their motion are fully determined by computing the time derivatives of their θ parameters. If a surface point $\boldsymbol{X}$ lies onto the contour generator, then its observed sliding velocity is:

$$\dot{\boldsymbol{X}} = \frac{\partial \boldsymbol{X}}{\partial \theta}\dot{\theta} + \frac{\partial \boldsymbol{X}}{\partial z}\dot{z} = \boldsymbol{X}_\theta\,\dot{\theta} + \boldsymbol{X}_z\,\dot{z}. \tag{25}$$

The sliding velocity along the contour generator itself, $\dot{z}$, is not observable because the contour generator is the ruling of the surface—a straight line. Therefore one may assume that:

$$\dot{z} = 0. \tag{26}$$

Therefore, the sliding-motion component in (22) can be written as:

$$\mathbf{R}\dot{\boldsymbol{X}} = \mathbf{R}\boldsymbol{X}_\theta\,\dot{\theta}. \tag{27}$$

Since $\boldsymbol{X}$ lies onto the contour generator, it verifies the contour generator constraint, i.e., (5). By differentiation of this equation we obtain a constraint for the surface parameter velocity, $\dot{\theta}$, as follows. We differentiate equation (5), we perform the substitutions $\dot{\mathbf{R}}^\top = -\mathbf{R}^\top[\boldsymbol{\Omega}]_\times$ and $\dot{\boldsymbol{t}} = \boldsymbol{V}$, and we

notice that the velocity of a surface point is tangent to the surface, i.e., $\dot{\boldsymbol{X}}^\top \boldsymbol{n} = 0$. We obtain the following expression for the derivative of (5):

$$(\boldsymbol{X} + \mathbf{R}^\top(\boldsymbol{t} - \boldsymbol{C}))^\top \dot{\boldsymbol{n}} = ([\boldsymbol{\Omega}]_\times(\boldsymbol{t} - \boldsymbol{C}) - \boldsymbol{V})^\top \mathbf{R}\boldsymbol{n}. \tag{28}$$

With $\dot{\boldsymbol{n}} = \boldsymbol{n}_\theta \dot{\theta}$ and with $[\boldsymbol{a}]_\times \boldsymbol{b} = -[\boldsymbol{b}]_\times \boldsymbol{a}$, we obtain from (28):

$$\dot{\theta} = \frac{(\mathbf{R}\boldsymbol{n})^\top [[\boldsymbol{C} - \boldsymbol{t}]_\times - \mathbf{I}_{3\times 3}]}{(\boldsymbol{X} + \mathbf{R}^\top(\boldsymbol{t} - \boldsymbol{C}))^\top \boldsymbol{n}_\theta} \begin{pmatrix} \boldsymbol{\Omega} \\ \boldsymbol{V} \end{pmatrix}. \tag{29}$$

Therefore, the sliding velocity $\dot{\theta}$ can be expressed as a function of (i) the surface parameterization, (ii) the relative position and orientation of the camera with respect to the surface, and (iii) the rigid motion of the surface (the kinematic screw). To summarize, (27) becomes:

$$\mathbf{R}\dot{\boldsymbol{X}} = \mathbf{B} \begin{pmatrix} \boldsymbol{\Omega} \\ \boldsymbol{V} \end{pmatrix} \tag{30}$$

where $\mathbf{B}$ is the 3×6 matrix:

$$\mathbf{B} = \frac{1}{b} \mathbf{R} \boldsymbol{X}_\theta (\mathbf{R}\boldsymbol{n})^\top [[\boldsymbol{C} - \boldsymbol{t}]_\times \quad -\mathbf{I}_{3\times 3}] \tag{31}$$

and the scalar b is defined by:

$$b = (\boldsymbol{X} + \mathbf{R}^\top(\boldsymbol{t} - \boldsymbol{C}))^\top \boldsymbol{n}_\theta.$$

2.5.3 The Velocity of Extremal Contours

To conclude this section, the velocity of an extremal contour point has a rigid-motion component and a surface-sliding component:

$$\dot{\boldsymbol{x}} = \dot{\boldsymbol{x}}^r + \dot{\boldsymbol{x}}^s. \tag{32}$$

The explicit parameterization of the sliding component, as shown above, allows its incorporation into the explicit representation of the observed image velocities as a function of the kinematic-chain parameters, as described in detail below.

The sliding velocity depends both on the curvature of the surface and on the velocity of the surface. In practice it will speed up the convergence of the tracker by a factor of two, as described in Sect. 6.

3 The Human-Body Kinematic Chain

In the case of a kinematic chain, the rigid motion of a body-part can be parameterized by the joint parameters. Kinematic chains are widely used by human-body trackers and motion capture systems. In this section we introduce the use

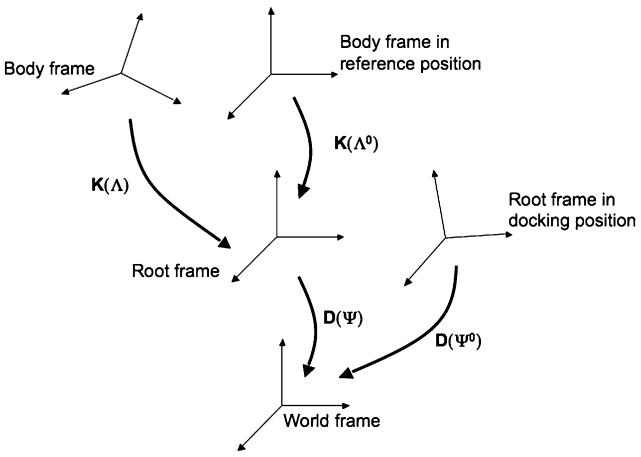


Fig. 3 Each body part has a frame associated with it, therefore motions are represented by changes in coordinate frames. There is a reference position for each body part in the chain defined by the joint angles $\boldsymbol{\Lambda}^0$. Similarly, there is a docking position for the root body-part defined by the six-dimensional vector $\boldsymbol{\Psi}^0$

of the *zero-reference kinematic representation* for modeling the human-body articulated chains. The Zero-reference kinematic representation was studied for robot manipulators (Mooring et al. 1991) and (McCarthy 1990). The parameterization introduced in this section combines the zero-reference representation with the free motion of the root body-part, i.e., Fig. 3.

Without loss of generality we consider any one among the several kinematic chains needed to describe the human body. A body part P is linked to the root body-part R by a kinematic chain with p rotational degrees of freedom. The root body part itself moves freely with six degrees of freedom (three rotations and three translations) and with respect to the world coordinate frame. Let $\boldsymbol{\Lambda} = (\lambda_1, \ldots, \lambda_p)$ denote the joint angles associated with the kinematic chain, and let $\boldsymbol{\Psi} = (\psi_1, \ldots, \psi_q)$ denote the rotational and translational degrees of freedom of the free motion. In the most general case we have $q = 6$. Therefore, there are $p + q$ motion parameters embedded in the vector $\boldsymbol{\Phi} = (\boldsymbol{\Psi}, \boldsymbol{\Lambda})$.

With the same notations as in the previous section, we consider a point X that belongs to the contour generator associated with a developable surface and body part. The point's *homogeneous coordinates* in the local frame are denoted by $\widetilde{\boldsymbol{X}} = (X_1\ X_2\ X_3\ 1)^\top$. We also denote by $\boldsymbol{X}^r$ the coordinates of the same point in the root body-part frame, and by $\boldsymbol{X}^w$ its coordinates in the world frame.

Moreover, we denote with $\mathbf{D}(\boldsymbol{\Psi})$ the 4×4 homogeneous matrix associated with the free motion of the root body part with respect to a fixed world frame, and with $\mathbf{K}(\boldsymbol{\Lambda})$ the 4×4 homogeneous matrix associated with the constrained motion of a body part with respect to the root part. Let $\boldsymbol{\Lambda}^0$ be the joint angles for a particular *reference position* of the kinematic chain. Obviously we have

$\widetilde{\boldsymbol{X}}^r(\boldsymbol{\Lambda}) = \mathbf{K}(\boldsymbol{\Lambda})\widetilde{\boldsymbol{X}}$ and $\widetilde{\boldsymbol{X}}^r(\boldsymbol{\Lambda}^0) = \mathbf{K}(\boldsymbol{\Lambda}^0)\widetilde{\boldsymbol{X}}$. We obtain $\widetilde{\boldsymbol{X}}^r(\boldsymbol{\Lambda}) = \mathbf{K}(\boldsymbol{\Lambda})\mathbf{K}^{-1}(\boldsymbol{\Lambda}^0)\widetilde{\boldsymbol{X}}^r(\boldsymbol{\Lambda}^0)$. With this formula and from $\widetilde{\boldsymbol{X}}^w(\boldsymbol{\Psi}, \boldsymbol{\Lambda}) = \mathbf{D}(\boldsymbol{\Psi})\widetilde{\boldsymbol{X}}^r(\boldsymbol{\Lambda})$ we obtain:

$$\widetilde{\boldsymbol{X}}^w(\boldsymbol{\Psi}, \boldsymbol{\Lambda}) = \mathbf{D}(\boldsymbol{\Psi})\mathbf{K}(\boldsymbol{\Lambda})\mathbf{K}^{-1}(\boldsymbol{\Lambda}^0)\widetilde{\boldsymbol{X}}^r(\boldsymbol{\Lambda}^0).$$

We also consider a reference or a *docking* position for the root body-part, defined by the free-motion parameters $\boldsymbol{\Psi}^0$, i.e., $\widetilde{\boldsymbol{X}}^w(\boldsymbol{\Psi}^0, \boldsymbol{\Lambda}^0) = \mathbf{D}(\boldsymbol{\Psi}^0)\widetilde{\boldsymbol{X}}^r(\boldsymbol{\Lambda}^0)$. Finally we obtain:

$$\begin{aligned}\widetilde{\boldsymbol{X}}^w(\boldsymbol{\Psi}, \boldsymbol{\Lambda}) &= \mathbf{D}(\boldsymbol{\Psi})\mathbf{K}(\boldsymbol{\Lambda})\mathbf{K}^{-1}(\boldsymbol{\Lambda}^0)\mathbf{D}^{-1}(\boldsymbol{\Psi}^0)\widetilde{\boldsymbol{X}}^w(\boldsymbol{\Psi}^0, \boldsymbol{\Lambda}^0) \quad (33)\\ &= \mathbf{H}(\boldsymbol{\Psi}, \boldsymbol{\Psi}^0, \boldsymbol{\Lambda}, \boldsymbol{\Lambda}^0)\widetilde{\boldsymbol{X}}^w(\boldsymbol{\Psi}^0, \boldsymbol{\Lambda}^0). \quad (34)\end{aligned}$$

It will be convenient to write the above transformation as:

$$\mathbf{H}(\boldsymbol{\Psi}, \boldsymbol{\Psi}^0, \boldsymbol{\Lambda}, \boldsymbol{\Lambda}^0) = \mathbf{F}(\boldsymbol{\Psi}, \boldsymbol{\Psi}^0)\,\mathbf{Q}(\boldsymbol{\Lambda}, \boldsymbol{\Lambda}^0, \boldsymbol{\Psi}^0) \quad (35)$$

with:

$$\mathbf{F}(\boldsymbol{\Psi}, \boldsymbol{\Psi}^0) = \mathbf{D}(\boldsymbol{\Psi})\mathbf{D}^{-1}(\boldsymbol{\Psi}^0) \quad (36)$$

and:

$$\mathbf{Q}(\boldsymbol{\Lambda}, \boldsymbol{\Lambda}^0, \boldsymbol{\Psi}^0) = \mathbf{D}(\boldsymbol{\Psi}^0)\mathbf{K}(\boldsymbol{\Lambda})\mathbf{K}^{-1}(\boldsymbol{\Lambda}^0)\mathbf{D}^{-1}(\boldsymbol{\Psi}^0). \quad (37)$$

3.1 The Kinematic-Chain Model

The transformation $\mathbf{K}$ describes an open kinematic chain and the transformation $\mathbf{Q}$ describes exactly the same chain but relatively to a *reference position* of the chain. $\mathbf{K}$ may be written as a composition of fixed transformations $\mathbf{L}_1, \ldots, \mathbf{L}_p$, and of one-degree-of-freedom rotations $\mathbf{J}(\lambda_1), \ldots, \mathbf{J}(\lambda_p)$:

$$\mathbf{K}(\boldsymbol{\Lambda}) = \mathbf{L}_1\mathbf{J}(\lambda_1)\cdots\mathbf{L}_p\mathbf{J}(\lambda_p) \quad (38)$$

where the matrices $\mathbf{L}_1 \ldots \mathbf{L}_p$ are fixed transformations between adjacent rotational joints, and matrices of the form of $\mathbf{J}$ are the canonical representations of a rotation.

Matrix $\mathbf{Q}$ in (37) can now be written as a product of one-degree-of-freedom transformations $\mathbf{Q}_i$:

$$\begin{aligned}\mathbf{Q}(\boldsymbol{\Lambda}, \boldsymbol{\Lambda}^0, \boldsymbol{\Psi}^0) = \mathbf{Q}_1(\lambda_1 - \lambda_1^0)\cdots\mathbf{Q}_i(\lambda_i - \lambda_i^0)\cdots\\ \times\, \mathbf{Q}_p(\lambda_p - \lambda_p^0) \quad (39)\end{aligned}$$

where each term $\mathbf{Q}_i$ is of the form $\mathbf{U}_i\mathbf{J}(\lambda_i - \lambda_i^0)\mathbf{U}_i^{-1}$, i.e., (McCarthy 1990):

$$\begin{aligned}\mathbf{Q}_i(\lambda_i - \lambda_i^0) = \underbrace{\mathbf{D}(\boldsymbol{\Psi}^0)\mathbf{L}_1\mathbf{J}(\lambda_1^0)\cdots\mathbf{L}_i}_{\mathbf{U}_i}\,\mathbf{J}(\lambda_i - \lambda_i^0)\\ \times\, \underbrace{\mathbf{L}_i^{-1}\cdots\mathbf{J}(-\lambda_1^0)\mathbf{L}_1^{-1}\mathbf{D}^{-1}(\boldsymbol{\Psi}^0)}_{\mathbf{U}_i^{-1}}. \quad (40)\end{aligned}$$

Notice that matrices $\mathbf{U}_i, \{i = 1 \ldots p\}$ remain fixed when the joint parameters vary **and** when the root body-part undergoes a free motion. Others used the exponential representation for this one-dimensional transformations (Murray et al. 1994; Bregler et al. 2004).

3.2 The Zero-Reference Kinematic Model

Without loss of generality, one may set the initial joint-angle values to zero, i.e., $\lambda_1^0 = \cdots = \lambda_p^0 = 0$. In this case, the kinematic chain does not depend any more on its reference pose, since $\mathbf{J}(\lambda_i^0) = \mathbf{I}$ for all i. The kinematic chain writes in this case:

$$\mathbf{Q}(\boldsymbol{\Lambda}, \boldsymbol{\Psi}^0) = \mathbf{Q}_1(\lambda_1, \boldsymbol{\Psi}^0)\cdots\mathbf{Q}_i(\lambda_i, \boldsymbol{\Psi}^0)\cdots\mathbf{Q}_p(\lambda_p, \boldsymbol{\Psi}^0). \quad (41)$$

The human-body zero-reference kinematic chain From the equations above, one may write a compact and convenient factorization of matrix $\mathbf{H}$, i.e., (35):

$$\begin{aligned}\mathbf{H}(\boldsymbol{\Psi}, \boldsymbol{\Psi}^0, \boldsymbol{\Lambda}) = \mathbf{F}(\boldsymbol{\Psi}, \boldsymbol{\Psi}^0)\mathbf{Q}_1(\lambda_1, \boldsymbol{\Psi}^0)\cdots\\ \times\, \mathbf{Q}_i(\lambda_i, \boldsymbol{\Psi}^0)\cdots\mathbf{Q}_p(\lambda_p, \boldsymbol{\Psi}^0). \quad (42)\end{aligned}$$

4 The Jacobian of the Human-Body Kinematic Chain

In this section we make explicit the Jacobian matrix associated with the kinematic chain of the human-body, $\mathbf{J}_H$. This matrix appears in (14); From this equation and from (21) we obtain:

$$\begin{pmatrix}\boldsymbol{\Omega}\\ \boldsymbol{V}\end{pmatrix} = \mathbf{J}_H\dot{\boldsymbol{\Phi}}. \quad (43)$$

The Jacobian of a kinematic chain such as the one described above is intrinsic to the mechanical and geometric structure of the kinematic chain and it does not depend on a particular choice of a point X, is it sliding onto the surface or rigidly attached to it. The Jacobian $\mathbf{J}_H$ maps joint velocities onto the kinematic screw of a body part whose kinematic chain is denoted by $\mathbf{H}$. In order to establish an expression for the Jacobian, we will first need to determine the tangent operator $\widehat{\mathbf{H}}$ of $\mathbf{H}$:

$$\widehat{\mathbf{H}} = \dot{\mathbf{H}}\mathbf{H}^{-1}. \quad (44)$$

Second, we parameterize $\widehat{\mathbf{H}}$ such that it depends only on the kinematic parameters, i.e., we must take the derivative of a body-part point with respect to the motion variables, i.e., $d\boldsymbol{X}^w/d\boldsymbol{\Phi}$. The case of human-body motion is different than the classical case studied in the robotics literature, (McCarthy 1990; Mooring et al. 1991; Murray et al. 1994) because one must take into account the fact that the root-part of the chain undergoes a free rigid motion.

4.1 A Rotational Joint

First, we consider the case of a single rotational joint. It's tangent operator is defined by $\widehat{\mathbf{Q}}_i = \dot{\mathbf{Q}}_i \mathbf{Q}_i^{-1}$, and we obviously have $\widehat{\mathbf{Q}}_i = \mathbf{U}_i \widehat{\mathbf{J}}_i \mathbf{U}_i^{-1}$. From $\widehat{\mathbf{J}} = \dot{\mathbf{J}}\mathbf{J}^{-1}$, we have:

$$\widehat{\mathbf{J}}(\lambda_i) = \dot{\lambda}_i \widetilde{\mathbf{J}},$$

with

$$\widetilde{\mathbf{J}} = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

Therefore, we obtain a simple expression for the tangent operator associated with one joint:

$$\widehat{\mathbf{Q}}_i(\lambda_i - \lambda_i^0) = \dot{\lambda}_i \mathbf{U}_i \widetilde{\mathbf{J}} \mathbf{U}_i^{-1} = \dot{\lambda}_i \widetilde{\mathbf{Q}}_i. \tag{45}$$

Matrix $\widetilde{\mathbf{J}}$ is called the *Lie-algebra of the Lie-group* defined by the matrices of the form of $\mathbf{J}$. If one prefers the exponential representation, $\widetilde{\mathbf{J}}$ is called a *twist*.

4.2 The Tangent Operator

Second, we determine the tangent operator of the human-body kinematic chain. The zero-reference kinematic chain, $\mathbf{H}$, may well be viewed as an Euclidean transformation and is composed of a rotation matrix and a translation vector: $\mathbf{R}_H, \boldsymbol{t}_H$. Hence, its tangent operator has a rigid-motion component, i.e., (23) and (24), as well as a sliding-motion component, i.e., (30). When applied to (34) we obtain *the action of the tangent operator* onto a surface point:

$$\begin{aligned} \dot{X}^w(\boldsymbol{\Psi}, \boldsymbol{\Lambda}) &= \dot{\mathbf{R}}_H X^w(\boldsymbol{\Psi}_0, \boldsymbol{\Lambda}_0) + \dot{\boldsymbol{t}}_H + \mathbf{R}_H \dot{X}^w(\boldsymbol{\Psi}_0, \boldsymbol{\Lambda}_0) \\ &= \dot{\mathbf{R}}_H \mathbf{R}_H^\top (X^w(\boldsymbol{\Psi}, \boldsymbol{\Lambda}) - \boldsymbol{t}_H) + \dot{\boldsymbol{t}}_H \\ &\quad + \mathbf{R}_H \dot{X}^w(\boldsymbol{\Psi}_0, \boldsymbol{\Lambda}_0) \\ &= (\mathbf{A}_H + \mathbf{B}_H) \begin{pmatrix} \boldsymbol{\Omega} \\ \boldsymbol{V} \end{pmatrix} \end{aligned} \tag{46}$$

where we have $\boldsymbol{\Omega} = \dot{\mathbf{R}}_H \mathbf{R}_H^\top$, $\boldsymbol{V} = \dot{\boldsymbol{t}}_H$ and with 3×6 matrices $\mathbf{A}_H$ and $\mathbf{B}_H$ as defined by (24) and (31). The 3-D vectors $\boldsymbol{\Omega}$ and $\boldsymbol{V}$ form the kinematic screw which we seek to estimate:

$$\widehat{\mathbf{H}}(\boldsymbol{\Psi}, \boldsymbol{\Lambda}) = \begin{bmatrix} [\boldsymbol{\Omega}]_\times & \boldsymbol{V} \\ \mathbf{0}^\top & 0 \end{bmatrix}. \tag{47}$$

Since $\mathbf{H} = \mathbf{F}\mathbf{Q}$, we have $\dot{\mathbf{H}} = \dot{\mathbf{F}}\mathbf{Q} + \mathbf{F}\dot{\mathbf{Q}}$ and:

$$\widehat{\mathbf{H}}(\boldsymbol{\Psi}, \boldsymbol{\Lambda}) = \widehat{\mathbf{F}}(\boldsymbol{\Psi}) + \mathbf{F}\widehat{\mathbf{Q}}(\boldsymbol{\Lambda})\mathbf{F}^{-1}. \tag{48}$$

As detailed below, the tangent operator can be written as the sum:

$$\widehat{\mathbf{H}}(\boldsymbol{\Psi}, \boldsymbol{\Lambda}) = \widehat{\mathbf{H}}_r(\boldsymbol{\Psi}) + \sum_{i=1}^{p} \widehat{\mathbf{H}}_i(\lambda_i). \tag{49}$$

- *The tangent operator associated with the free motion of the root body-part*, $\widehat{\mathbf{H}}_r(\boldsymbol{\Psi}) = \widehat{\mathbf{F}}(\boldsymbol{\Psi})$; from (36) we obtain: $\widehat{\mathbf{F}}(\boldsymbol{\Psi}) = \widehat{\mathbf{D}}(\boldsymbol{\Psi})$. $\widehat{\mathbf{H}}_r$ is the 4×4 matrix parameterized by the rotational velocity $\boldsymbol{\omega}_r$ and the translational velocity $\boldsymbol{v}_r$ of this free motion:

$$\widehat{\mathbf{H}}_r(\boldsymbol{\Psi}) = \begin{bmatrix} [\boldsymbol{\omega}_r]_\times & \boldsymbol{v}_r \\ \mathbf{0}^\top & 0 \end{bmatrix}. \tag{50}$$

  This motion has six degrees of freedom and can be parameterized by three rotations and three translations:

$$\begin{aligned} \mathbf{R} &= \mathbf{R}_z(\psi_3)\mathbf{R}_y(\psi_2)\mathbf{R}_x(\psi_1), \\ \boldsymbol{t} &= \psi_4 \boldsymbol{e}_x + \psi_5 \boldsymbol{e}_y + \psi_6 \boldsymbol{e}_z, \end{aligned}$$

  where $\boldsymbol{e}_x = (1\,0\,0)^\top$ and so forth. The kinematic screw of this motion can therefore be written as:

$$\begin{pmatrix} \boldsymbol{\omega}_r \\ \boldsymbol{v}_r \end{pmatrix} = \begin{bmatrix} \boldsymbol{\omega}_x & \boldsymbol{\omega}_y & \boldsymbol{\omega}_z & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{e}_x & \boldsymbol{e}_y & \boldsymbol{e}_z \end{bmatrix}_{6\times 6} \dot{\boldsymbol{\Psi}} \tag{51}$$

  with $[\boldsymbol{\omega}_z]_\times = [\boldsymbol{e}_z]_\times$, $[\boldsymbol{\omega}_y]_\times = \mathbf{R}_z[\boldsymbol{e}_y]_\times \mathbf{R}_z^\top$, and $[\boldsymbol{\omega}_x]_\times = \mathbf{R}_y\mathbf{R}_z[\boldsymbol{e}_x]_\times \mathbf{R}_z^\top \mathbf{R}_y^\top$.
- *The tangent operator associated with the constrained motion of the kinematic chain*, $\mathbf{F}\widehat{\mathbf{Q}}(\boldsymbol{\Lambda})\mathbf{F}^{-1}$; it is expressed in world coordinates and with respect to a reference position defined by both $\boldsymbol{\Psi}^0$ and $\boldsymbol{\Lambda}^0$. This tangent operator can be expanded as (McCarthy 1990):

$$\mathbf{F}\widehat{\mathbf{Q}}(\boldsymbol{\Lambda})\mathbf{F}^{-1} = \begin{bmatrix} \boldsymbol{\omega}_1 & \dots & \boldsymbol{\omega}_p \\ \boldsymbol{v}_1 & \dots & \boldsymbol{v}_p \end{bmatrix} \dot{\boldsymbol{\Lambda}}, \tag{52}$$

  with $\dot{\boldsymbol{\Lambda}} = (\dot{\lambda}_1 \dots \dot{\lambda}_p)^\top$.

Therefore, by combining (49), (51), and (52) we obtain the following expression for the kinematic screw:

$$\begin{aligned} \begin{pmatrix} \boldsymbol{\Omega} \\ \boldsymbol{V} \end{pmatrix} &= \begin{bmatrix} \boldsymbol{\omega}_x & \boldsymbol{\omega}_y & \boldsymbol{\omega}_z & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{e}_x & \boldsymbol{e}_y & \boldsymbol{e}_z \end{bmatrix} \dot{\boldsymbol{\Psi}} \\ &\quad + \begin{bmatrix} \boldsymbol{\omega}_1 & \dots & \boldsymbol{\omega}_p \\ \boldsymbol{v}_1 & \dots & \boldsymbol{v}_p \end{bmatrix} \dot{\boldsymbol{\Lambda}} \end{aligned} \tag{53}$$

with $\dot{\boldsymbol{\Psi}} = (\dot{\psi}_1 \dots \dot{\psi}_6)^\top$ and $\dot{\boldsymbol{\Lambda}} = (\dot{\lambda}_1 \dots \dot{\lambda}_p)^\top$. Finally, the Jacobian of the human-body kinematic chain writes as

a $6 \times (6+p)$ matrix:

$$\mathbf{J}_H = \begin{bmatrix} \boldsymbol{\omega}_x & \boldsymbol{\omega}_y & \boldsymbol{\omega}_z & \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\omega}_1 & \dots & \boldsymbol{\omega}_p \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{e}_x & \boldsymbol{e}_y & \boldsymbol{e}_z & \boldsymbol{v}_1 & \dots & \boldsymbol{v}_p \end{bmatrix}. \quad (54)$$

To conclude this section we remind that the relationship between the kinematic velocities $\dot{\boldsymbol{\Phi}} = (\dot{\boldsymbol{\Psi}}\ \dot{\boldsymbol{\Lambda}})$ and the image velocity of an extremal contour point x writes:

$$\dot{\boldsymbol{x}} = \mathbf{J}_I(\mathbf{A}_H + \mathbf{B}_H)\mathbf{J}_H\dot{\boldsymbol{\Phi}}. \quad (55)$$

This corresponds to (21) where the kinematic screw is given by (53).

5 Fitting Extremal Contours to Image Contours

In this section we consider the problem of fitting extremal contours—model contours predicted in the image plane, with image contours—contours extracted from the data. Therefore, we have to measure the discrepancy between a set of predictions (extremal contours) and a set of observations (image contours): we want to find the model's parameters that minimize this discrepancy.

Although the human body comprises several (five) kinematic chains, for the sake of clarity of exposition we consider only one such kinematic chain. We collect extremal-contour points from all the body-parts. Let $\mathcal{X} = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_j, \dots, \boldsymbol{x}_m\}$ be the prediction vector, a set of m extremal-contour points. The components of this vector are 2-D points and they are parameterized by the kinematic- and free-motion parameter vector $\boldsymbol{\Phi}$, i.e. $\boldsymbol{x}(\boldsymbol{\Phi})$. Similarly, let $\mathcal{Y} = \{\boldsymbol{y}_1, \dots, \boldsymbol{y}_i, \dots, \boldsymbol{y}_k\}$ be the observation vector—a set of contour points observed in the image. In order to estimate the motion parameters one has to compare these two sets through a metric and to minimize it over the motion variables. Therefore, the problem can be generally stated as the minimization of a multi-variate scalar function E of (1).

There are several ways of defining and measuring the distance between two sets of points, $\mathcal{Y}$ and $\mathcal{X}$. One way of measuring this distance is to sum over one-to-one pairings $(\boldsymbol{x}_j, \boldsymbol{y}_i)$:

$$E(\mathcal{Y}, \mathcal{X}(\boldsymbol{\Phi})) = \sum_i \sum_j \alpha_{ij} \|\boldsymbol{y}_i - \boldsymbol{x}_j(\boldsymbol{\Phi})\|^2 \quad (56)$$

where the *hidden* variables α_{ij} are the entries of an association matrix: $\alpha_{ij} = 1$ if the observable $\boldsymbol{y}_i$ matches the prediction $\boldsymbol{x}_j$, and $\alpha_{ij} = 0$ otherwise. Therefore one has to solve both for the hidden variables and for the motion parameters (David et al. 2004).

5.1 The Hausdorf Distance

Another way of measuring the distance between two point-sets is to use the *Hausdorff distance* (Huttenlocher et al. 1993; Sim et al. 1999) which does not make use of explicit point pairings:

$$H(\mathcal{Y}, \mathcal{X}) = \max(h(\mathcal{Y}, \mathcal{X}), h(\mathcal{X}, \mathcal{Y})) \quad (57)$$

where $h()$ is called the *directed* Hausdorff distance: $h(\mathcal{Y}, \mathcal{X}) = \max_i(\min_j(\|\boldsymbol{y}_i - \boldsymbol{x}_j\|))$. The function $h()$ identifies the point in $\mathcal{Y}$ which is the farthest from any point in $\mathcal{X}$. The Hausdorff distance is the maximum between $h(\mathcal{Y}, \mathcal{X})$ and $h(\mathcal{X}, \mathcal{Y})$ and hence it measures the degree of mismatch between two point sets *without making explicit pairings* of points in one set with points in the other set. This means that many points of $\mathcal{Y}$ may be assigned to the same point of $\mathcal{X}$.

5.2 The Chamfer Distance

If the max operator in the Hausdorff distance is replaced by the summation operator, we obtain the normalized *directed* (or non-symmetric) chamfer distance:

$$DCD(\mathcal{Y}, \mathcal{X}) = \frac{1}{k}\sum_{i=1}^{k} \min_j(\|\boldsymbol{y}_i - \boldsymbol{x}_j\|). \quad (58)$$

The directed chamfer distance, or *DCD*, is a positive function and has the properties of identity and of triangle inequality but not of symmetry. It also has the desirable property that it can be computed very efficiently. Indeed, the *DCD* can be computed from the binary image of the observed image contour set $\mathcal{Y}$ using the *chamfer-distance image* $C_{\mathcal{Y}}$ (Borgefors 1986; Gavrila and Philomin 1999). The subscript $\mathcal{Y}$ reminds that this image is associated with the set $\mathcal{Y}$ of observed edge points. For each image site (pixel) with integer-valued image coordinates u_1 and u_2, the chamfer-distance image $C_{\mathcal{Y}}(u_1, u_2)$ returns the real-valued distance from this pixel to the nearest contour point of $\mathcal{Y}$. Therefore one can evaluate the distance from a predicted extremal-contour point $\boldsymbol{x} \in \mathcal{X}$ to its closest image contour by evaluating the chamfer-distance image at $\boldsymbol{x}$ with real-valued image coordinates x_1 and x_2.

We denote by $[x]$ the integer part of a real number x. Let $u_1 = [x_1]$ and $u_2 = [x_2]$ be the integer parts, and $r_1 = x_1 - [x_1]$ and $r_2 = x_2 - [x_2]$ be the fractional parts of the coordinates of a predicted point $\boldsymbol{x}$. The chamfer distance at $\boldsymbol{x}$ can be obtained by bi-linear interpolation of the chamfer-distance image:

$$\begin{aligned} D(\mathcal{Y}, \boldsymbol{x}) = {} & (1-r_1)(1-r_2)C_{\mathcal{Y}}(u_1, u_2) \\ & + r_1(1-r_2)C_{\mathcal{Y}}(u_1+1, u_2) \\ & + (1-r_1)r_2C_{\mathcal{Y}}(u_1, u_2+1) \\ & + r_1r_2C_{\mathcal{Y}}(u_1+1, u_2+1). \end{aligned} \quad (59)$$

5.3 Minimizing the Chamfer Distance

The minimization problem defined by (1) can now be written as the sum of squares of the chamfer distances over the predicted model contours:

$$f(\boldsymbol{\Phi}) = \frac{1}{2}\sum_{j=1}^{m} D_j^2(\mathcal{Y}, \boldsymbol{x}_j(\boldsymbol{\Phi})) = \frac{1}{2}\sum_{j=1}^{m} D_j^2(\boldsymbol{\Phi}). \tag{60}$$

In order to minimize this function over the motion parameters, we take its second-order Taylor expansion as well as the Gauss-Newton approximation of the Hessian:

$$f(\boldsymbol{\Phi}+\boldsymbol{d}) = f(\boldsymbol{\Phi}) + \boldsymbol{d}^\top \mathbf{J}_D^\top \boldsymbol{D} + \frac{1}{2}\boldsymbol{d}^\top \mathbf{J}_D^\top \mathbf{J}_D \boldsymbol{d} + \cdots$$

where $\boldsymbol{D}^\top = (D_1 \ \ldots \ D_m)$ and $\mathbf{J}_D^\top = [d\boldsymbol{D}/d\boldsymbol{\Phi}]^\top$ is the $n \times m$ matrix:

$$\mathbf{J}_D^\top = \begin{bmatrix} \dfrac{dD_1}{d\boldsymbol{\Phi}} & \cdots & \dfrac{dD_m}{d\boldsymbol{\Phi}} \end{bmatrix}. \tag{61}$$

The Chamfer-Distance Gradient The embedding of the tracker into such an optimization framework requires an analytic expression for the gradient of the error function to be minimized. The derivative of the chamfer distance D_j with respect to the motion parameters is the following matrix product:

$$\frac{dD_j}{d\boldsymbol{\Phi}} = \left(\frac{dD_j}{d\boldsymbol{x}}\right)^\top \frac{d\boldsymbol{x}}{d\boldsymbol{\Phi}}.$$

By noticing that $d[x]/dx = 0$, we immediately obtain an expression for $dD_j/d\boldsymbol{x}$:

$$\begin{aligned}
\frac{\partial D_j}{dx_1} &= (1-r_2)(C_{\mathcal{Y}}(u_1+1,u_2) - C_{\mathcal{Y}}(u_1,u_2)) \\
&\quad + r_2(C_{\mathcal{Y}}(u_1+1,u_2+1) - C_{\mathcal{Y}}(u_1,u_2+1)), \\
\frac{\partial D_j}{dx_2} &= (r_1-1)(C_{\mathcal{Y}}(u_1+1,u_2) + C_{\mathcal{Y}}(u_1,u_2)) \\
&\quad + r_1(C_{\mathcal{Y}}(u_1+1,u_2+1) + C_{\mathcal{Y}}(u_1,u_2+1)).
\end{aligned}$$

We recall that $d\boldsymbol{x}/d\boldsymbol{\Phi} = \mathbf{J}_I(\mathbf{A}+\mathbf{B})\mathbf{J}_H$ is the extremal-contour Jacobian defined in (19).

Issues related to the minimization of the chamfer distance can be found in (Knossow et al. 2006). Here we analyse the practical conditions under which this minimization should be carried out. At each time instant, the tracker is initialized with the previously found solution and (60) must be minimized. This minimization problem needs one necessary condition, namely that the $n \times n$ Hessian matrix has full rank. The Jacobian $\mathbf{J}_D$ is of size $m \times n$ and we recall that n is the number of variables to be estimated (the motion parameters) and m is the number of predictions (extremal contour points). To compute the inverse of $\mathbf{J}_D^\top \mathbf{J}_D$ we must have $m \geq n$ with n independent matrix rows.

5.4 How Many Cameras?

Since each prediction accounts for one row in the Jacobian matrix, one must somehow insure that there are n "independent" predictions. If each body part is viewed as a rigid object in motion, then it has six degrees of freedom. A set of three non-collinear points constrains these degrees of freedom. Whenever there are one-to-one model-point-to-image-point assignments, a set of three points is sufficient to constrain all six degrees of freedom. In the case of the chamfer distance there are no such one-to-one assignments and each model point yields only one constraint. Therefore, when one uses the chamfer distance, the problem is underconstrained since three non-collinear points yield three constraints only. Within a kinematic chain, the root body-part has six degrees of freedom and each body-part has $6+p$ degrees of freedom. Fortunately the body-parts are linked together to form kinematic chains. Therefore, one sensible hypothesis is to assume that the points at hand are evenly distributed among the body parts.

The kinematic human-body model that we use is composed of 5 kinematic chains that share a common root body-part, 19 body-parts, and 54 degrees of freedom (48 rotational joints and 6 free-motion parameters). Therefore, with an average of 3 points per body-part, there are in principle enough constraints to solve the tracking problem. Notice that the root-body part can arbitrarily be chosen and there is no evidence that one body-part is more suitable than another body-part to be the root part.

In practice there are other difficulties and problems. Due to total and/or partial occlusions, not all the body-parts can be predicted visible in one image. Therefore, it is impossible to insure that all the degrees of freedom are actually measured in one image. Even if a point attached to a visible body-part is predicted in the image, it may not be present in the data and/or it may be badly extracted and located. Non-relevant edges that lie in the neighborhood of a predicted location contribute to the chamfer distance and therefore complicate the task of the minimization process.

One way to increase the robustness of the tracker it to make recourse to redundant data. The latter may be obtained by using several cameras, each camera providing an independent chamfer distance error function. Provided that the cameras are *calibrated and synchronized* the method described above can be simultaneously applied to all the cameras. There will be several Jacobian matrices of the form of (61) (one for each camera) and these matrices can be combined together into a unique Jacobian, provided that a common world reference frame is being used (Martin and

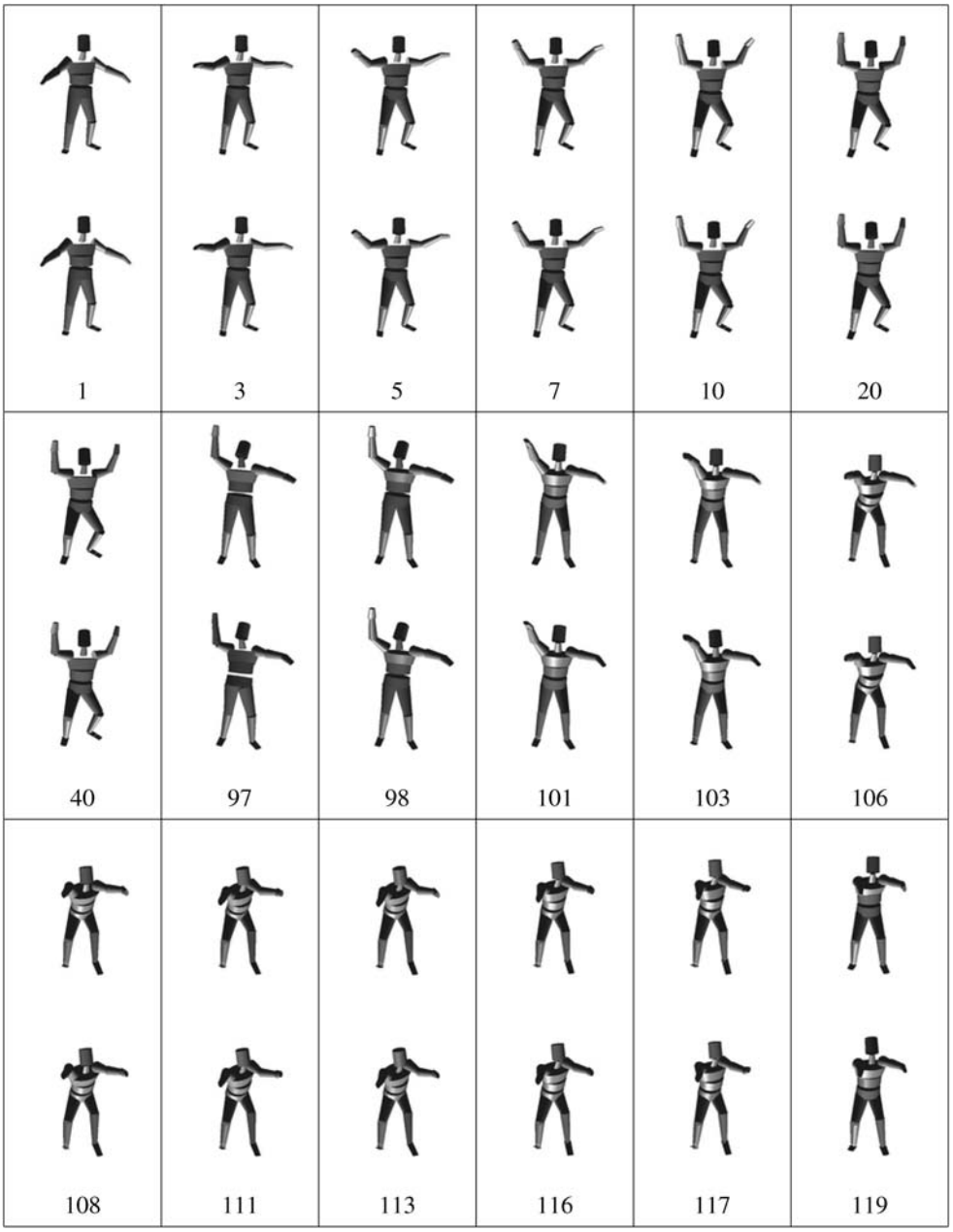


Fig. 4 This figure compares the ground truth (*first, third and fifth rows*) with the estimated poses (*second, fourth and sixth rows*). The ground-truth poses were used to simulate silhouette data to be used by the tracker

Horaud 2002). Therefore, by using several cameras one increases the number of predictions (columns in the Jacobian) without increasing the number of variables.

It is worthwhile to notice that the extremal contours viewed with one camera are different than the extremal contours viewed with another camera. Indeed, these two sets of contours correspond to different physical points onto the surface. One great advantage of using extremal contours in order to fit the model parameters with the data is that there is no need to establish matches across images taken with distinct cameras.

6 Experiments with Simulated and Real Data

The simulated data were produced using a motion capture system and an animation software. This outputs trajectories for the motion parameters of our human-body model. From these trajectories we generated a sequence of model motions. From each pose of the model we computed extremal contours for six images associated with six virtual cameras. We simulated a total of 120 frames for each image sequence.

Next, we applied our method to these contours. Figure 4 shows the simulated poses (top rows) as well as the esti-

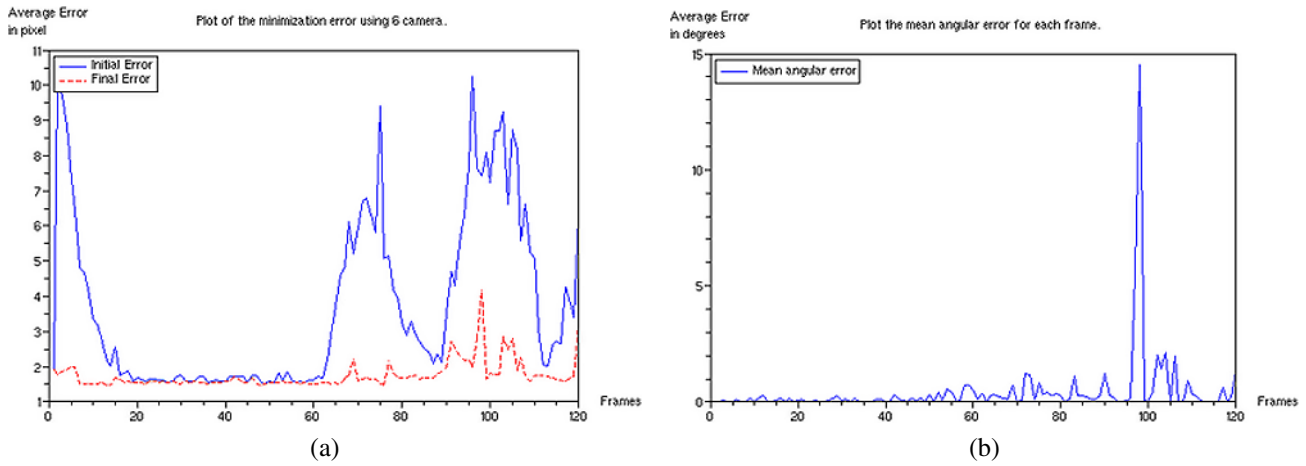


Fig. 5 **a** The error between the image contours and the projected extremal contours before minimization (*top curve*) and after minimization (*bottom curve*). **b** The average error between the simulated motion parameters and the estimated ones

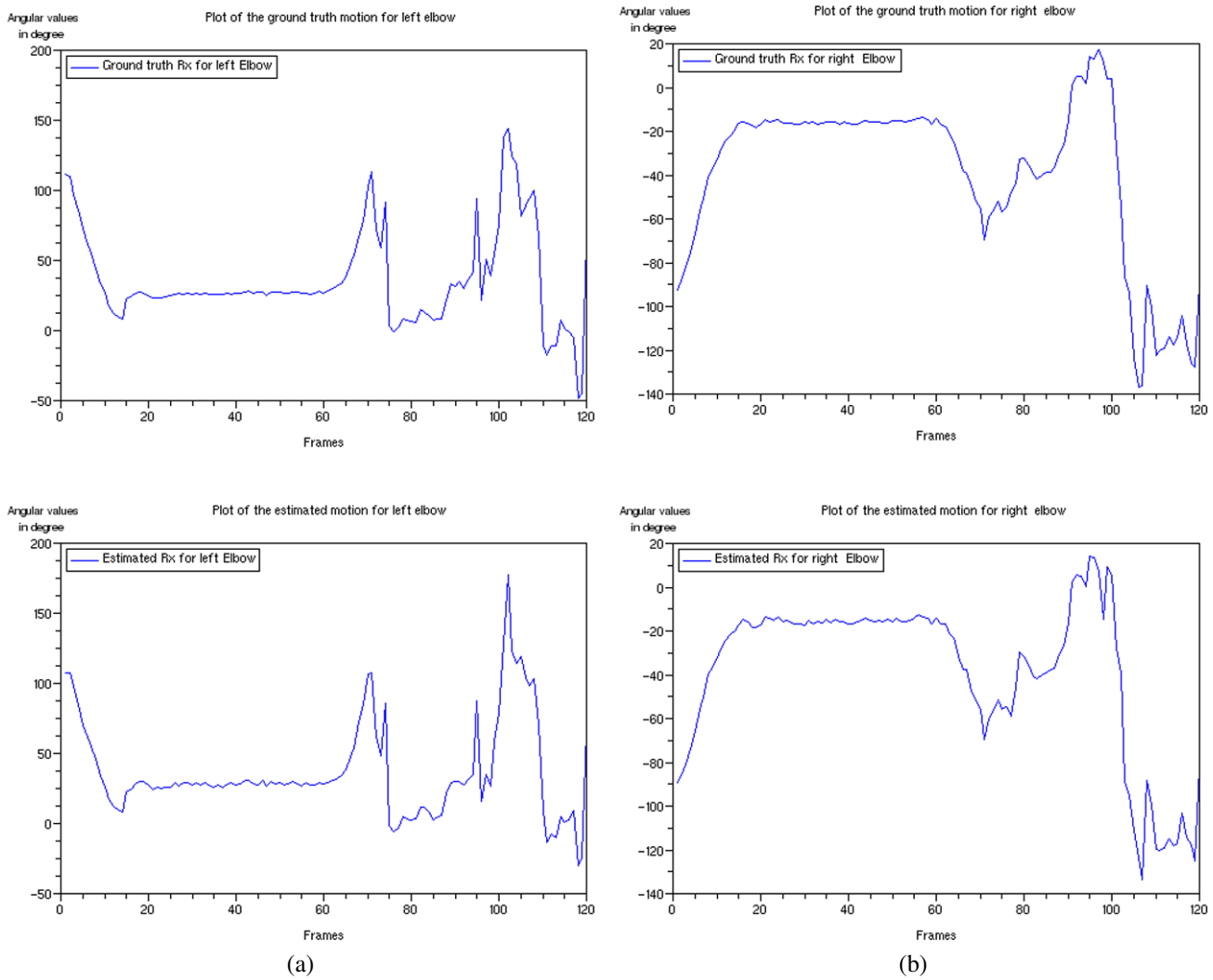
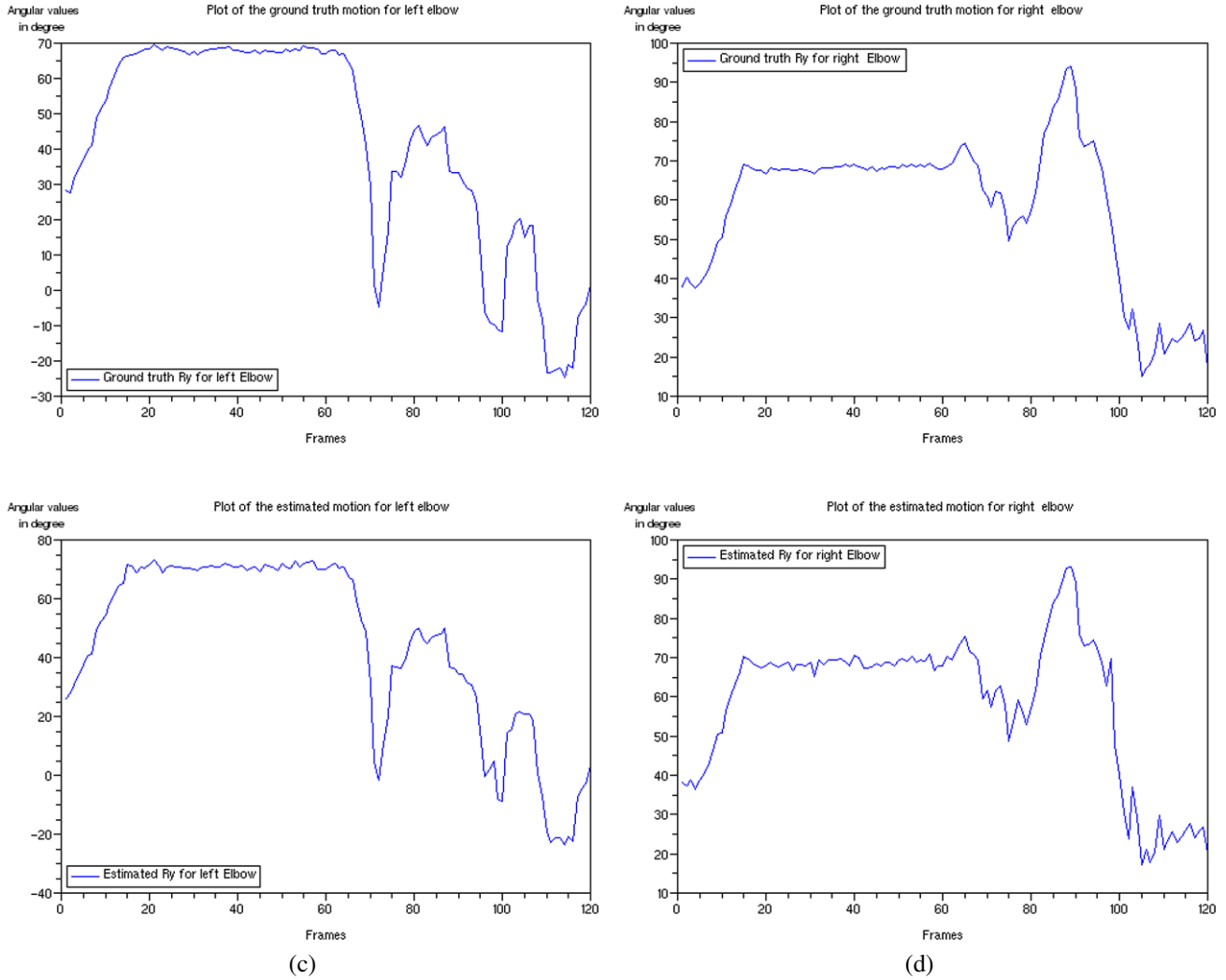


Fig. 6 Ground-truth and estimated joint-angle trajectories for the left and right elbows



(c) (d)

Fig. 6 (*continued*)

mated poses (bottom rows). Figures 5, 6, and 7 compare the results of our method with the ground truth. Figure 5-b plots the average error between the true motion parameters and the estimated ones. One may notice that the average error remains within 2^0, with the exception of frame 98 for which the average error is 15^0: this error is due to 180^0 ambiguity associated with one of the joints. Nevertheless, the tracker was able to recover from this large error. Figure 5a shows the initial error between the predicted contours and the image contours (top curve) as well as the error once the motion parameters were fitted using the minimization described in Sect. 5.3. The failure of the tracker at frame 98 corresponds to an error of 4–5 pixels. The initial mean error is 3.7 pixels whereas the final mean error is 1.5 pixels.

In more detail, Figs. 6 and 7 compare the estimated trajectories with the ground truth for the left and right elbows and for the left and right shoulders. Both the elbow and shoulder are modeled with two rotational degrees of freedom.

In order to track real human-body motions we used a setup composed of six cameras that were accurately calibrated and whose video outputs are finely synchronized. Fine synchronization (of the order of 10^{-6} s) and fast shutter speed (10^{-3} s) allow one to cope with fast motions. The camera setup is shown on Fig. 8. It consists in six Firewire cameras that deliver 600×800 uncompressed images at 30 frames per second.

We used two different persons, Ben and Erwan. These two persons have the same size and therefore we used the same roughly estimated parameters for the elliptical cones modeling the body parts.

We gathered three sets of data shown on Fig. 1 (Erwan-1), Fig. 11 (Ben) and Fig. 12 (Erwan-2). For each data set, the figures show the images associated with the first three cameras, the associated silhouettes, and the estimated pose of the model displayed from the viewpoint of the third camera. The extremal contours eventually fitted to the data are shown overlayed onto the raw images.

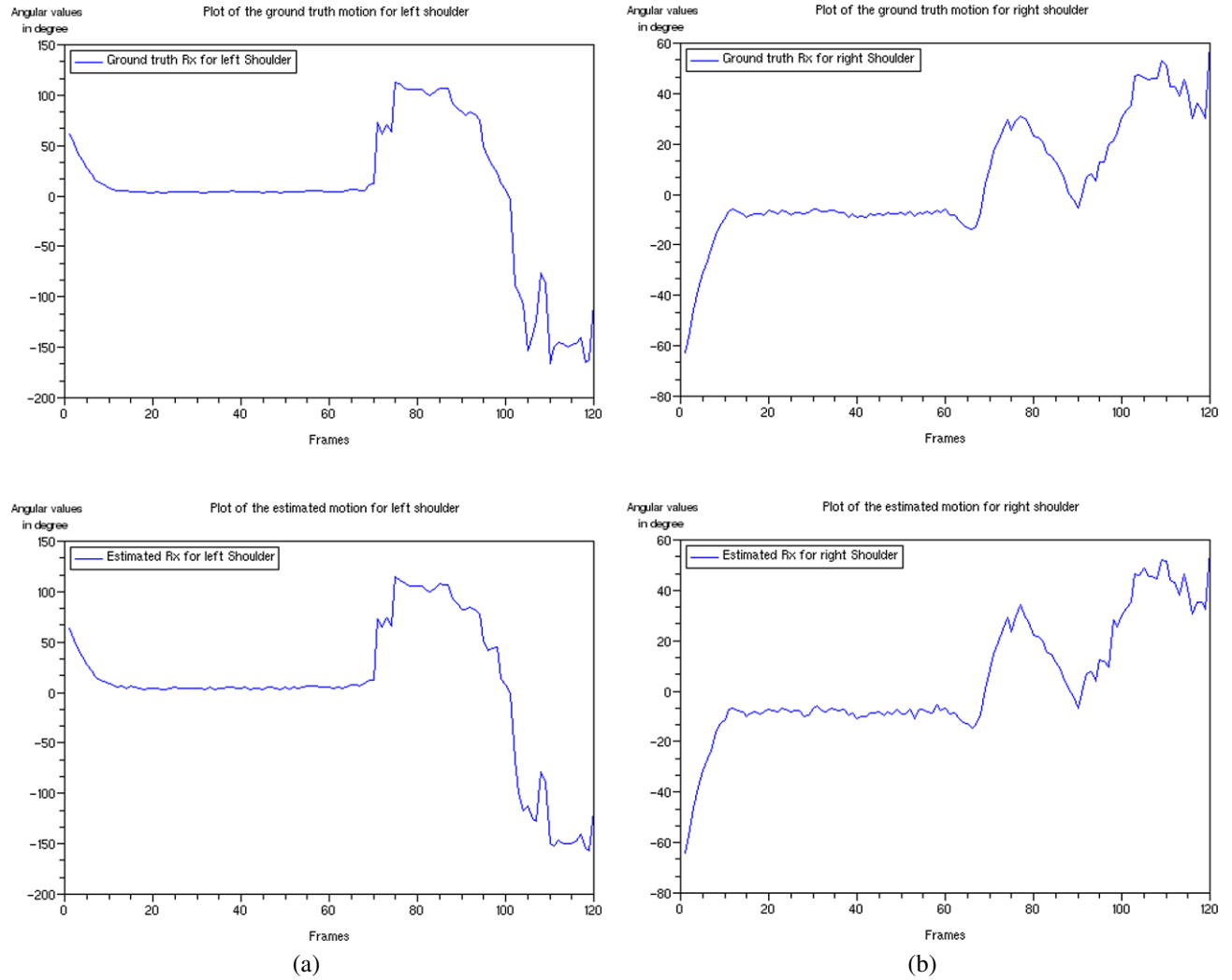


Fig. 7 Ground-truth and estimated joint-angle trajectories for the left and right shoulders

The tracking is initialized by incremental pose estimation of the body parts. We start with an initial guess (Fig. 9a) from which the pose of the root body part is first estimated (Fig. 9b). This is followed by the pose estimation of other body parts (Fig. 9c). The final kinematic pose found by this initialization process is shown on Fig. 9d. The first example, Erwan-1, has 250 frames, the second example, Ben, has 800 frames and the third example, Erwan-2, has 200 frames. Notice that the Erwan motions involve all the degrees of freedom of the articulated model, as well as the motion of the root body-part.

The efficiency of minimization-based trackers, as the one described here resides in number of iterations needed by the optimization algorithm to converge. In all the examples described in this paper we minimized an error function that has two terms: one term corresponds to the rigid motions of the body parts and the other terms corresponds to the sliding of the contour generator on the body-parts' surface. Under these circumstances, the tracker converges in 3 to 5 iterations and the RMS image error is, in this case, 1.5 pixels, Fig. 13 (bold plots). If the sliding-motion term is left out the efficiency of the tracker is substantially degraded because the optimizer needs twice more iterations for an RMS error of 2.3 pixels, Fig. 13 (dashed plots).

It is worthwhile to notice that we used a human model with the same measurements for the two persons (body-part parameters such as the size of the arms, feet, thighs, head, torso, etc.). More accurate model parameters (finely adjusted to each person) will result in smaller RMS errors.

6.1 Comparison with Marker-Based Motion Capture Data

One way to quantitatively evaluate the performance of markerless human tracking methods such as the one described in this paper, is to compare it with a marker-based motion capture system. Until recently it was believed that markerless motion capture systems cannot compete with marker-

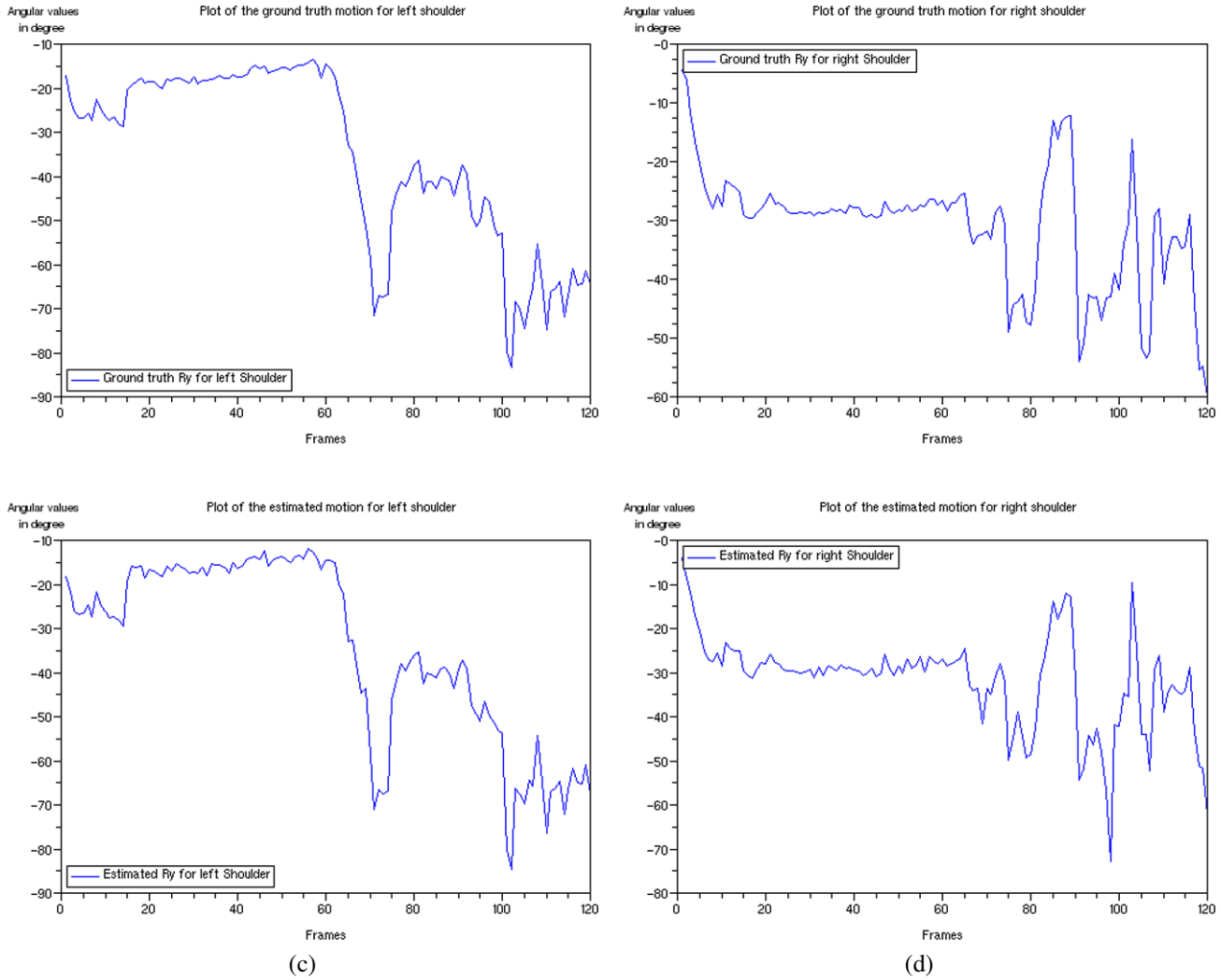


Fig. 7 (*continued*)

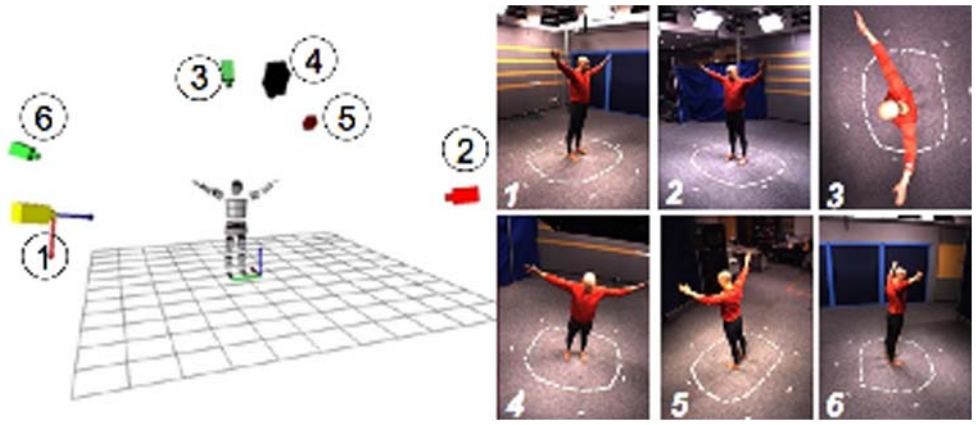


Fig. 8 The camera setup used in our experiments. The cameras are calibrated with respect to a global reference frame. The human body model is shown in it's reference position

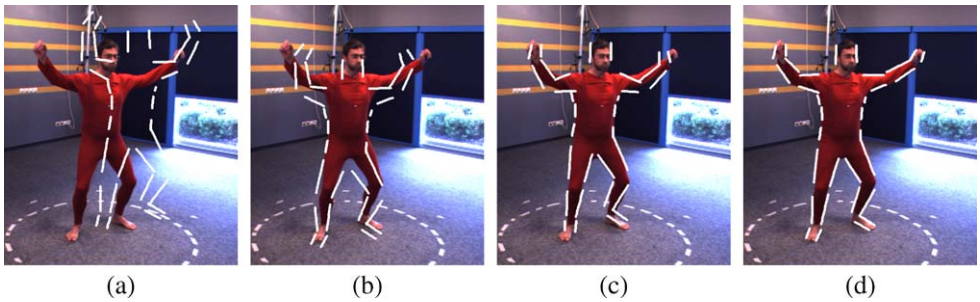


Fig. 9 Initialization: Starting from an initial guess, (**a**), the pose of the root body-part is first estimated, (**b**), followed by the incremental estimation of the remaining body parts, (**c**) and (**d**)

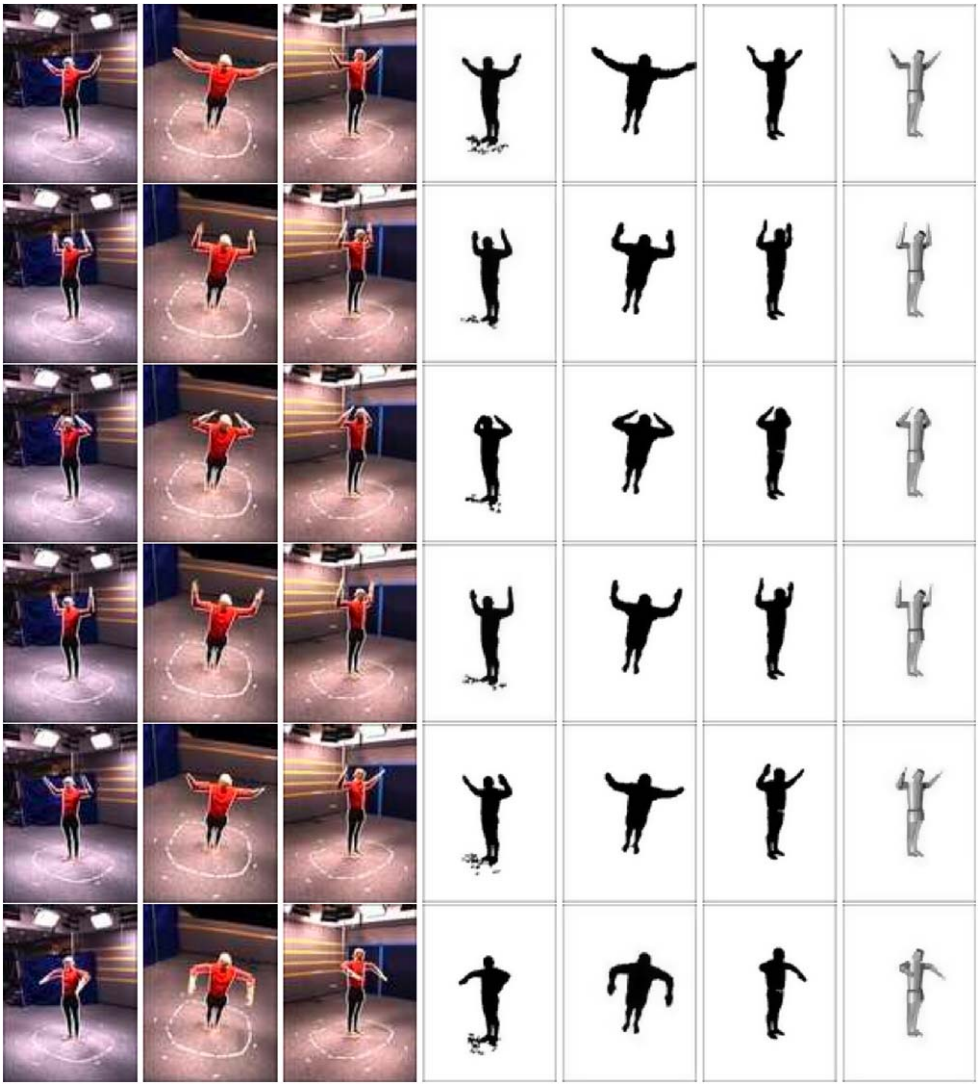


Fig. 10 The result of tracking Ben with 6 cameras over 800 frames (continues on the next figure)

based systems (Gleicher and Ferrier 2002). We performed the following experiment and evaluation which is similar in spirit with the work described in (Balan et al. 2005; Sigal and Black 2006).

We equipped a room with two camera systems, our 6-camera system and a VICON system using 8 cameras. We simultaneously gathered markerless and marker-based data with these two systems. While our system gathers 8-

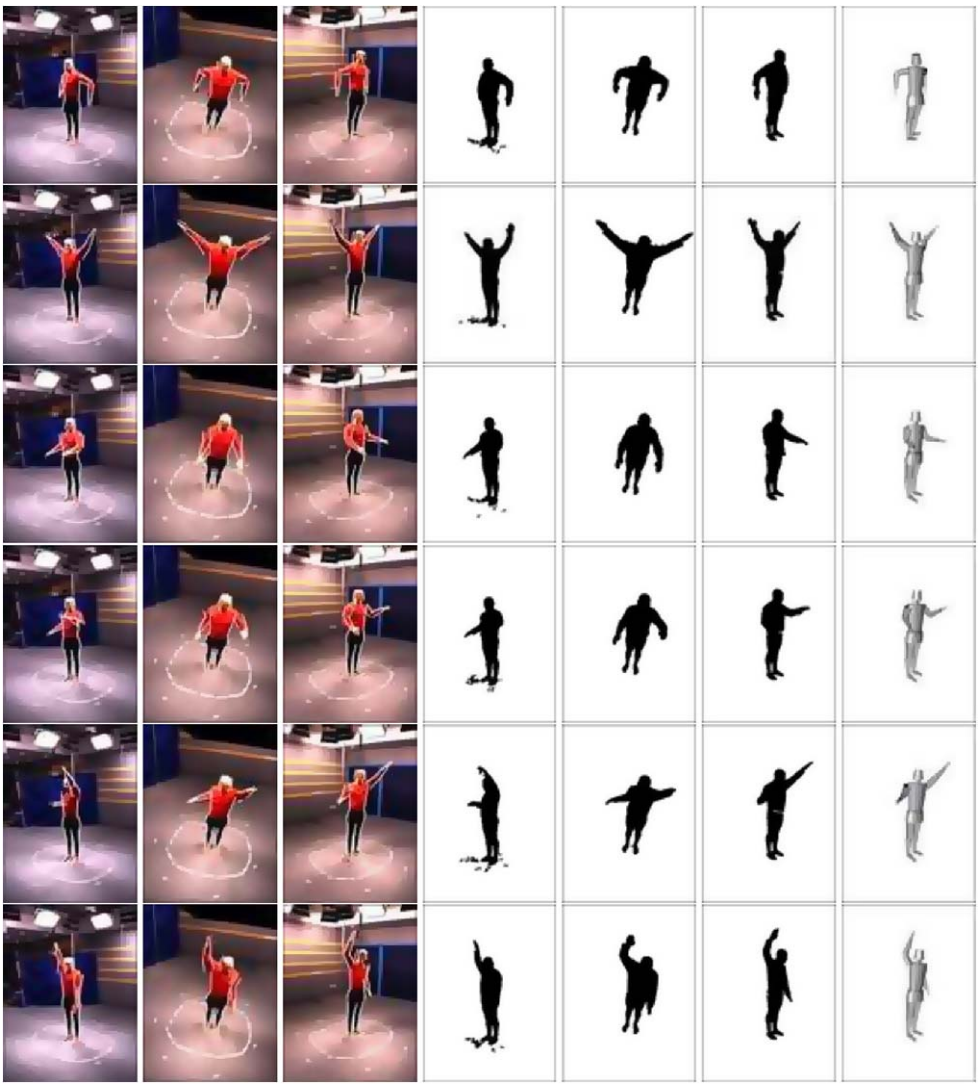


Fig. 11 The result of tracking Ben with 6 cameras over 800 frames (continued from the previous figure)

bit color images, the VICON system only gathers the image locations of the markers. Figure 14 shows two out of the six image sequences (first and second columns), the corresponding articulated poses found with our method (the third and fourth columns show the pose of the model from the viewpoints of the two cameras shown onto the left side of the figure). Both our algorithm and an algorithm based on the VICON data output joint trajectories. These trajectories are used to estimate the poses of a virtual character, as shown on the fifth and sixth columns. Character animation using both these types of data are available at http://perception.inrialpes.fr/~Knossow.

7 Conclusion

In this paper we proposed a contour-based method for tracking the motion of articulated objects. The main contributions of the paper are as follows. We derived an exact kinematic parameterization of the extremal contours produced by developable surfaces. We combined this parameterization with the zero-reference kinematic model that is well suited for representing the space of human motions, i.e., a combination of both the space of articulated motions (spanned by rotational joints) and the space of free motions (spanned by three rotations and three translations). We derived an analytical expression for the Jacobian linking joint- and free-motion velocities to extremal-contour velocities and we showed how this Jacobian matrix can be plugged into a non-linear minimization method. We made explicit two components of the Jacobian: a rigid-motion component and a sliding-motion component. The cost function uses the directed chamfer distance between extremal contours predicted by the model and image contours extracted from silhouettes. One major advantage of using the directed chamfer

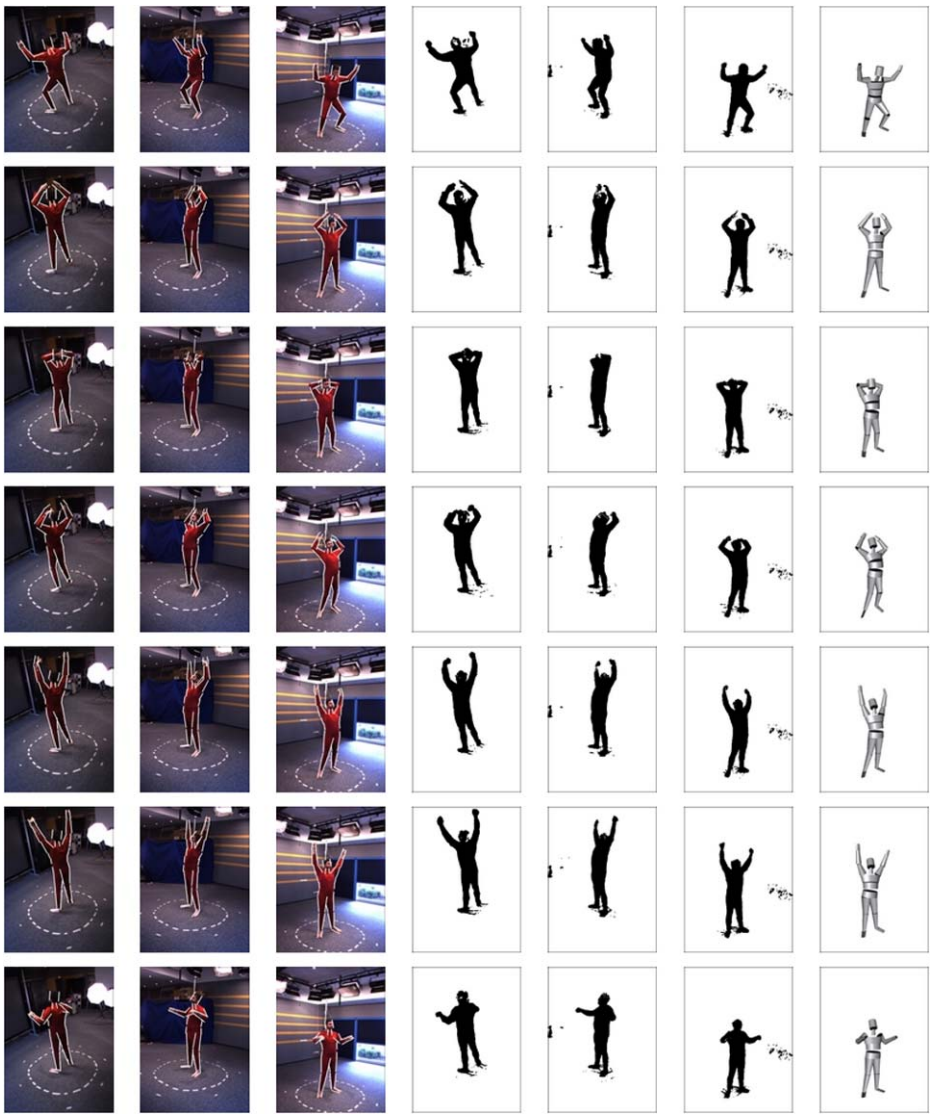


Fig. 12 The result of tracking Erwan-2 with 6 cameras over 250 frames

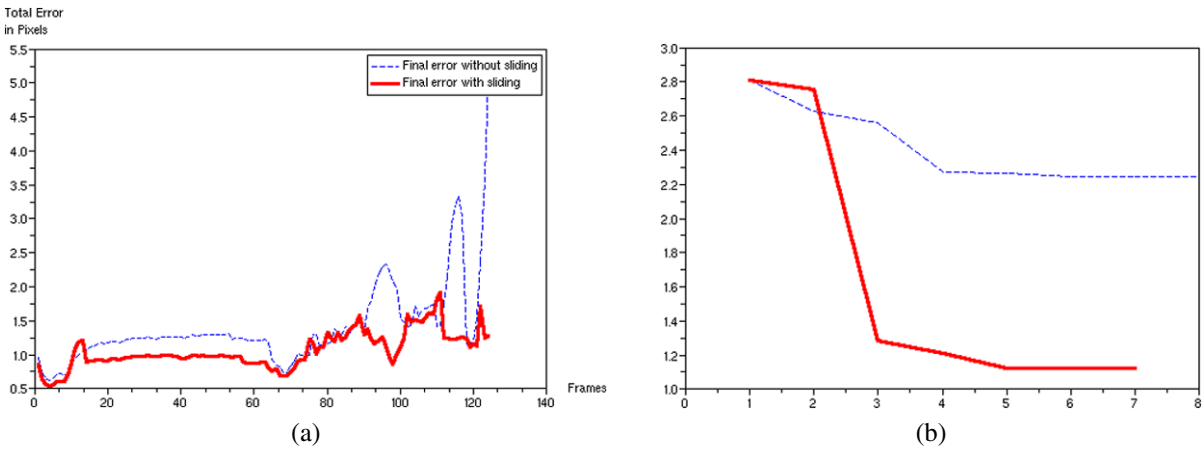


Fig. 13 These graphs show the advantage of using the sliding motion. **a** A comparison between the minimization error obtained using the sliding motion (*bold*) and without using it (*dashed*). **b** A comparison of the speed of convergence of the minimization method (number of iterations) with the sliding-motion component (*bold*) and without using it (*dashed*)

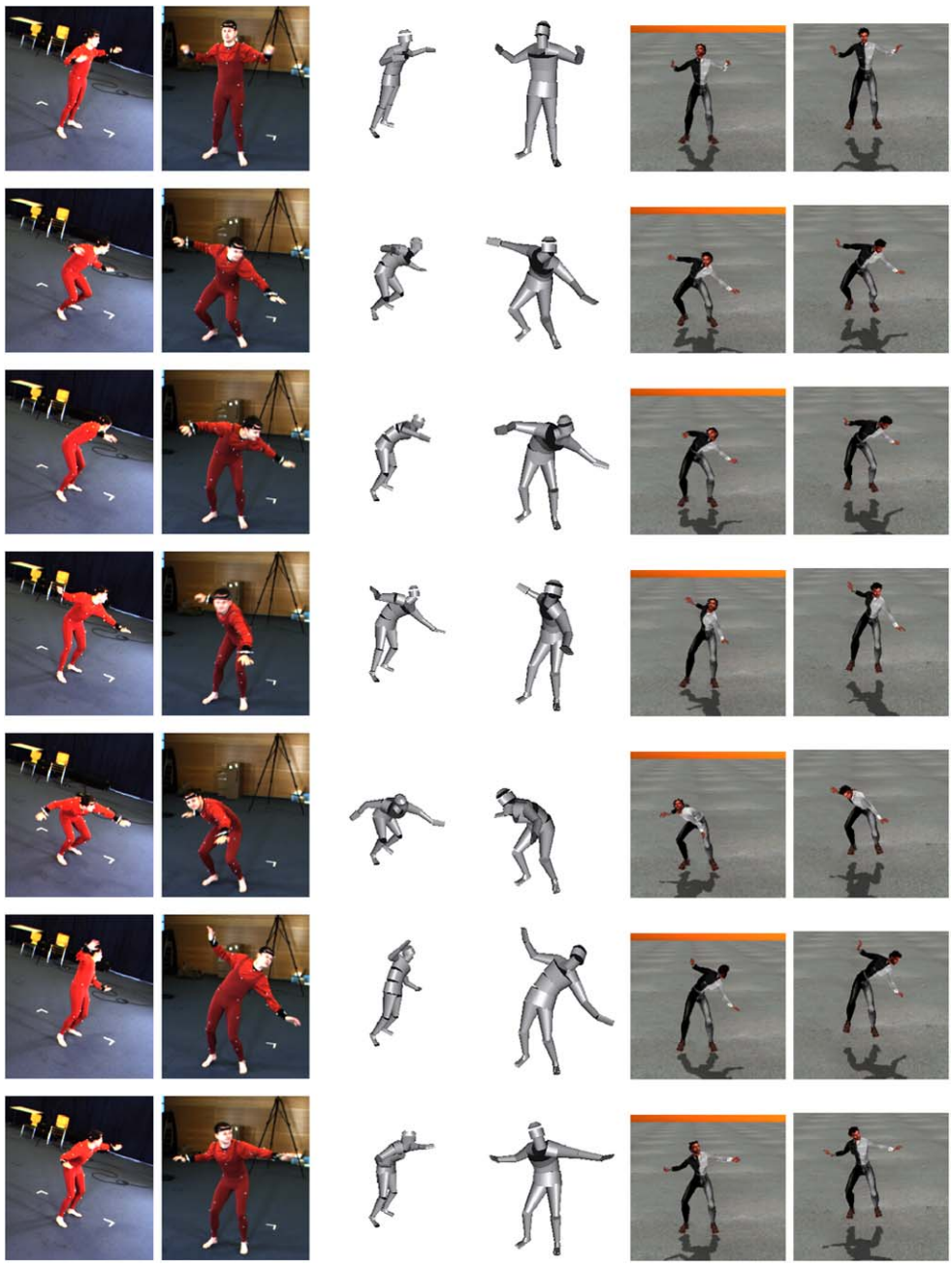


Fig. 14 This figure shows a *qualitative* comparison between marker-less and marker-based motion capture. The videos shown on to the *left* (together with four other videos which are not shown) produced the results shown on the *third, fourth and fifth columns*. The *last column* shows the result obtained from a method that uses a 8-camera VICON system to locate image markers

distance is that it does not require one-to-one assignments between image observations and model features.

Moreover, we analysed the conditions under which the minimization process can be carried out effectively, i.e., without failures due to numerical instabilities. Although, in principle, one camera may suffice, in practice it is desirable to have several images gathered simultaneously with several cameras. We carried out a large number of experiments with both simulated and real data. The tracker performed very well and is able to recover from badly estimated poses. We performed experiments with both simulated and real data gathered with six cameras. We compared the angle trajectories obtained with our method with trajectories obtained using a marker-based commercial system that uses eight cameras. We plan to compare more thoroughly our method with other methods within formal evaluation protocols.

In the future we plan to have a probabilistic look at the problem while maintaining the deterministic relation-

ship between extremal contours and moving articulated objects. One possibility is to consider the graphical model framework successfully applied to the pictorial recognition of articulated objects (Felzenswalb and Huttenlocher 2005), (Ronfard et al. 2002). Another possibility is to view each extremal contour as a thin and elongated cluster and to apply model-based clustering methods (Fraley and Raftery 2002). Both these approaches raise the problem of relating the parameters of the probability distribution function at hand with the kinematic parameterization proposed in this paper. It is worthwhile to notice that the clustering framework just mentioned is consistent with the chamfer distance which can be modified such that it accounts for a probabilistic association between a set of observations and a model.

References

Agarwal, A., & Triggs, W. (2006). Recovering 3D human pose from monocular images. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, *28*(1), 44–58.

Balan, A. O., Sigal, L., & Black, M. J. (2005). A quantitative evaluation of video-based 3D person tracking. In *PETS'05* (pp. 349–356).

Barrow, H. G., & Tenenbaum, J. M. (1981). Interpreting line drawings as three-dimensional surfaces. *Artificial Intelligence*, *17*(1–3), 75–116.

Borgefors, G. (1986). Distance transformation in digital images. *Computer Vision, Graphics, and Image Processing*, *34*(3), 344–371.

Bregler, C., Malik, J., & Pullen, K. (2004). Twist based acquisition and tracking of animal and human kinematics. *International Journal of Computer Vision*, *56*(3), 179–194.

Cheung, K. M., Baker, S., & Kanade, T. (2005a). Shape-from-silhouette across time, part I: theory and algorithms. *International Journal of Computer Vision*, *62*(3), 221–247.

Cheung, K. M., Baker, S., & Kanade, T. (2005b). Shape-from-silhouette across time, part II: applications to human modeling and markerless motion tracking. *International Journal of Computer Vision*, *63*(3), 225–245.

David, P., DeMenthon, D. F., Duraiswami, R., & Samet, H. (2004). Softposit: simultaneous pose and correspondence determination. *International Journal of Computer Vision*, *59*(3), 259–284.

Delamarre, Q., & Faugeras, O. (2001). 3D articulated models and multi-view tracking with physical forces. *Computer Vision and Image Understanding*, *81*(3), 328–357.

Deutscher, J., Blake, A., & Reid, I. (2000). Articulated body motion capture by annealed particle filtering. In *Computer vision and pattern recognition* (pp. 2126–2133).

Do Carmo, M. P. (1976). *Differential geometry of curves and surfaces*. New York: Prentice-Hall.

Drummond, T., & Cipolla, R. (2001). Real-time tracking of highly articulated structures in the presence of noisy measurements. In *ICCV* (pp. 315–320).

Felzenswalb, P., & Huttenlocher, D. (2005). Pictorial structures for object recognition. *International Journal of Computer Vision*, *61*(1), 55–79.

Forsyth, D. A., & Ponce, J. (2003). *Computer vision—a modern approach*. New Jersey: Prentice Hall.

Forsyth, D. A., Arikan, O., Ikemoto, L., O'Brien, J., & Ramanan, D. (2006). Computational studies of human motion, part 1: tracking and motion synthesis. *Foundations and Trends in Computer Graphics and Vision*, *1*(2), 77–254.

Fraley, C., & Raftery, A. E. (2002). Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association*, *97*, 611–631.

Gavrila, D. M. (1999). The visual analysis of human movement: a survey. *Computer Vision and Image Understanding*, *73*(1), 82–98.

Gavrila, D. M., & Davis, L. S. (1996). 3D model-based tracking of humans in action: a multi-view approach. In *Conference on computer vision and pattern recognition* (pp. 73–80), San Francisco, CA.

Gavrila, D. M., & Philomin, V. (1999). Real-time object detection for smart vehicles. In *IEEE Proceedings of the seventh international conference on computer vision* (pp. 87–93), Kerkyra, Greece.

Gleicher, G., & Ferrier, N. (2002). Evaluating video-based motion capture. In *Proceedings of the computer animation 2002* (pp. 75–80), Geneva, Switzerland, June 2002.

Huttenlocher, D. P., Klanderman, G. A., & Rucklidge, W. J. (1993). Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *15*(9), 850–863.

Kakadiaris, I., & Metaxas, D. (2000). Model-based estimation of 3D human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(12), 1453–1459.

Kehl, R., & Van Gool, L. J. (2006). Markerless tracking of complex human motions from multiple views. *Computer Vision and Image Understanding*, *103*(23), 190–209.

Knossow, D., Ronfard, R., Horaud, R., & Devernay, F. (2006). Tracking with the kinematics of extremal contours. In *Lecture notes in computer science*. *Computer vision—ACCV 2006* (pp. 664–673), Hyderabad, India, January 2006. Berlin: Springer.

Koenderink, J. (1990). *Solid shape*. Cambridge: The MIT Press.

Kreyzig, E. (1991). *Differential geometry*. New York: Dover. Reprint of a U. of Toronto 1963 edition.

Martin, F., & Horaud, R. (2002). Multiple camera tracking of rigid objects. *International Journal of Robotics Research*, *21*(2), 97–113.

McCarthy, J. M. (1990). *Introduction to theoretical kinematics*. Cambridge: MIT Press.

Mikic, I., Trivedi, M. M., Hunter, E., & Cosman, P. C. (2003). Human body model acquisition and tracking using voxel data. *International Journal of Computer Vision*, *53*(3), 199–223.

Moeslund, T. B., Hilton, A., & Krüger, V. (2006). A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, *104*(2), 90–126.

Mooring, B. W., Roth, Z. S., & Driels, M. R. (1991). *Fundamentals of manipulator calibration*. New York: Wiley.

Murray, R. M., Li, Z., & Sastry, S. S. (1994). *A mathematical introduction to robotic manipulation*. Ann Arbor: CRC Press.

Plaenkers, R., & Fua, P. (2003). Articulated soft objects for multi-view shape and motion capture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *25*(10), 1182–1187.

Ronfard, R., Schmid, C., & Triggs, W. (2002). Learning to parse pictures of people. In *Proceedings of the 7th European conference on computer vision* (Vol. 4, pp. 700–714), Copenhagen, Denmark, June 2002. Berlin: Springer.

Sigal, L., & Black, M. J. (2006). *Humaneva: synchronized video and motion capture dataset for evaluation of articulated human motion* (Technical Report CS-06-08). Department of Computer Science, Brown University, Providence, RI 02912, September 2006.

Sim, D. G., Kwon, O. K., & Park, R. H. (1999). Object matching algorithms using robust Hausdorff distance measures. *IEEE Transactions on Image Processing*, *8*(3), 425–429.

Sminchisescu, C., & Triggs, W. (2003). Kinematic jump processes for monocular 3D human tracking. In *International conference on computer vision and pattern recognition* (Vol. I, pp. 69–76), June 2003.

Sminchisescu, C., & Triggs, W. (2005). Building roadmaps of minima and transitions in visual models. *International Journal of Computer Vision*, *61*(1), 81–101.

Song, Y., Goncalves, L., & Perona, P. (2003). Unsupervised learning of human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *25*(7), 814–827.

Toyama, K., & Blake, A. (2002). Probabilistic tracking with exemplars in a metric space. *International Journal of Computer Vision*, *48*(1), 9–19.

CHAPITRE 3

Reconnaissance d'actions

Free-Viewpoint Action Recognition de Daniel Weinland, Rémi Ronfard et Edmond Boyer, Computer Vision and Image Understanding, Volume 104, Number 2-3, pages 249–257, November/December 2006.

Automatic Discovery of Action Taxonomies de Daniel Weinland, Rémi Ronfard et Edmond Boyer, IEEE Conference on Computer Vision and Pattern Recognition, pages 1639–1645 - 2006.

Action Recognition from Arbitrary Views de Daniel Weinland, Rémi Ronfard et Edmond Boyer, Action Recognition from Arbitrary Views using 3D Exemplars. International Conference on Computer Vision, Rio de Janeiro, Brazil, pages 1–7, 2007.

Available online at www.sciencedirect.com
ScienceDirect
Computer Vision and Image Understanding 104 (2006) 249–257
Computer Vision and Image Understanding
www.elsevier.com/locate/cviu

Free viewpoint action recognition using motion history volumes

Daniel Weinland *,[1], Remi Ronfard, Edmond Boyer

Perception-GRAVIR, INRIA Rhone-Alpes, 38334 Montbonnot Saint Martin, France

Received 31 January 2006; accepted 25 July 2006

Abstract

Action recognition is an important and challenging topic in computer vision, with many important applications including video surveillance, automated cinematography and understanding of social interaction. Yet, most current work in gesture or action interpretation remains rooted in view-dependent representations. This paper introduces *Motion History Volumes* (MHV) as a free-viewpoint representation for human actions in the case of multiple calibrated, and background-subtracted, video cameras. We present algorithms for computing, aligning and comparing MHVs of different actions performed by different people in a variety of viewpoints. Alignment and comparisons are performed efficiently using Fourier transforms in cylindrical coordinates around the vertical axis. Results indicate that this representation can be used to learn and recognize basic human action classes, independently of gender, body size and viewpoint.

Keywords: Action recognition; View invariance; Volumetric reconstruction

1. Introduction

Recognizing actions of human actors from video is an important topic in computer vision with many fundamental applications in video surveillance, video indexing and social sciences. According to Neumann et al. [1] and from a computational perspective, actions are best defined as four-dimensional patterns in space and in time. Video recordings of actions can similarly be defined as three-dimensional patterns in image-space and in time, resulting from the perspective projection of the world action onto the image plane at each time instant. Recognizing actions from a single video is however plagued with the unavoidable fact that parts of the action are hidden from the camera because of self-occlusions. That the human brain is able to recognize actions from a single viewpoint should not hide the fact that actions are firmly four-dimensional, and, furthermore, that the mental models of actions supporting recognition may also be four-dimensional.

In this paper, we investigate how to build spatio-temporal models of human actions that can support categorization and recognition of simple action classes, independently of viewpoint, actor gender and body sizes. We use multiple cameras and shape from silhouette techniques. We separate action recognition in two separate tasks. The first task is the extraction of motion descriptors from visual input, and the second task is the classification of the descriptors into various levels of action classes, from simple gestures and postures to primitive actions to higher levels of human activities, as pointed out by Kojima et al. [2]. That second task can be performed by learning statistical models of the temporal sequencing of motion descriptors. Popular methods for doing this are hidden Markov models and other stochastic grammars, e.g., stochastic parsing as proposed by Ivanov and Bobick [3]. In this paper, we focus on the extraction of motion descriptors from multiple cameras, and their classification into *primitive actions* such as raising and dropping hands and feet, sitting up and down, jumping, etc. To this aim, we introduce new motion descriptors based on *motion history volumes* which fuse action cues, as seen

* Corresponding author.
E-mail addresses: weinland@inrialpes.fr (D. Weinland), ronfard@inrialpes.fr (R. Ronfard), edmond.boyer@inrialpes.fr (E. Boyer).
[1] D. Weinland is supported by a grant from the European Community under the EST Marie-Curie Project Visitor.

1077-3142/$ - see front matter © 2006 Elsevier Inc. All rights reserved.
doi:10.1016/j.cviu.2006.07.013

from different viewpoints and over short time periods, into a single three-dimensional representation.

In previous work on motion descriptors, Green and Guan [4] use positions and velocities of human body parts, but such information is difficult to extract automatically during unrestricted human activities. Motion descriptors which can be extracted automatically, and which have been used for action recognition, are optical flows, as proposed by Efros et al. [5], motion templates in the seminal work of Bobick and Davis [6], and space-time volumes, introduced by Syeda-Mahmood et al. [7] or Yilmaz and Shah [8]. Such descriptors are not invariant to viewpoint, which can be partially resolved by multiplying the number of action classes by the number of possible viewpoints [6], relative motion directions [5], and point correspondences [7,8]. This results in a poorer categorization and an increased complexity.

In this research, we investigate the alternative possibility of building free-viewpoint class models from view-invariant motion descriptors. The key to our approach is the assumption that we need only consider variations in viewpoints around the central vertical axis of the human body. Within this assumption, we propose a representation based on Fourier analysis of motion history volumes in cylindrical coordinates. Fig. 1 explains our method for comparing two action sequences. We separately compute their visual hulls and accumulate them into motion history volumes. We transform the MHVs into cylindrical coordinates around their vertical axes, and extract view-invariant features in Fourier space. Such a representation fits nicely within the framework of Marr's 3D model [9] which has been advocated by linguist Jackendoff [10] as a useful tool for representing action categories in natural language.

The paper is organized as follows. First, we recall Davis and Bobick's definition of motion templates and extend it to three dimensions in Section 2. We present efficient descriptors for matching and aligning MHVs in Section 3. We present classification results in Section 4 and conclude in Section 5.

2. Definitions

In this section, we first recall 2D motion templates as introduced by Davis and Bobick in [6] to describe temporal actions. We then propose their generalization to 3D in order to remove the viewpoint dependence in an optimal fashion using calibrated cameras. Finally, we show how to perform temporal segmentation using the 3D MHVs.

2.1. Motion history images

Motion Energy Images (MEI) and Motion History Images (MHI) [6] were introduced to capture motion information in images. They encode, respectively, where motion occurred, and the history of motion occurrences, in the image. Pixel values are therefore binary values (MEI) encoding motion occurrence at a pixel, or multiple-values (MHI) encoding how recently motion occurred at a pixel. More formally, consider the binary-valued function $D(x,y,t)$, $D=1$ indicating motion at time t and location (x,y), then the MHI function is defined by:

$$h_\tau(x,y,t) = \begin{cases} \tau, & \text{if } D(x,y,t)=1, \\ \max(0, h_\tau(x,y,t-1)-1), & \text{otherwise,} \end{cases} \quad (1)$$

where τ is the maximum duration a motion is stored. The associated MEI can easily be computed by thresholding $h>0$.

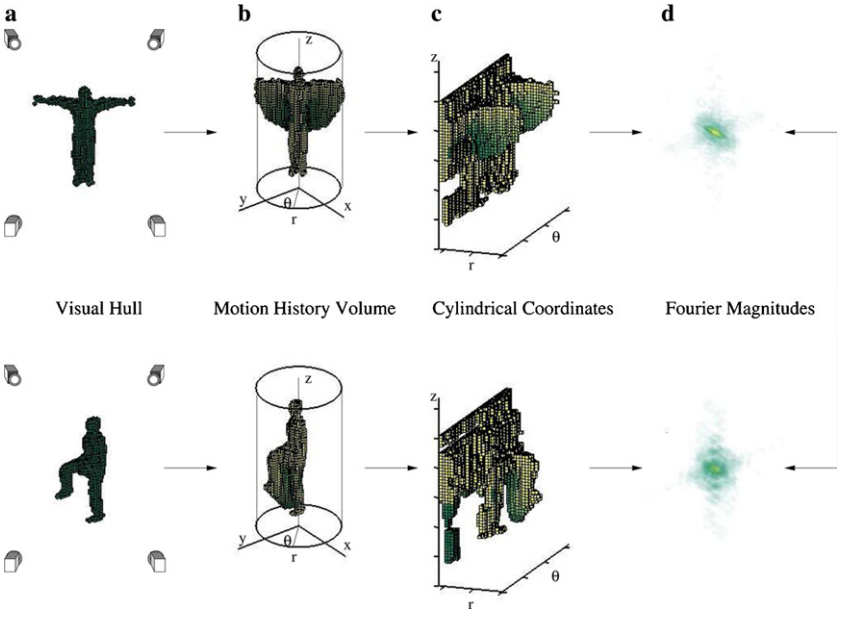


Fig. 1. The two actions are recorded by multiple cameras, spatially integrated into their visual hulls (a), and temporally integrated into motion history volumes (b) and (c). Invariant motion descriptors in Fourier space (d) are used for comparing the two actions.

The above motion templates are based on motion, i.e., $D(x,y,t)$ is a motion indicating function, however Bobick and Davis also suggest to compute templates based on occupancy, replacing $D(x,y,t)$ by the silhouette occupancy function. They argue that including the complete body makes templates more robust to incidental motions that occur during an action. Our experiments confirm that and show that occupancy provides robust cues for recognition, even if occupancy encodes not only motion but also shapes which may add difficulties when comparing movements, as illustrated in Fig. 2.

2.2. Motion history volumes

In this paper, we propose to extend 2D motion templates to 3D. The choice of a 3D representation has several advantages over a single, or multiple, 2D view representation:

- A 3D representation is a natural way to fuse multiple images information. Such representation is more informative than simple sets of 2D images since additional calibration information is taken into account.
- A 3D representation is more robust to the object's positions relative to the cameras as it replaces a possibly complex matching between learned views and the actual observations by a 3D alignment (see Section 2.3).
- A 3D representation allows different camera configurations.

Motion templates extends easily to 3D by considering the occupancy function $D(x,y,z,t)$ in 3D, where $D=1$ if (x,y,z) is occupied at time t and $D=0$ otherwise, and by considering voxels instead of pixels:

$$v_\tau(x,y,z,t) = \begin{cases} \tau, & \text{if } D(x,y,z,t)=1, \\ \max(0, h_\tau(x,y,z,t-1)-1), & \text{otherwise.} \end{cases} \quad (2)$$

In the rest of the paper, we will assume templates to be normalized and segmented with respect to the duration of an action:

$$v(x,y,z) = v_{\tau=t_{\max}-t_{\min}}(x,y,z,t_{\max})/(t_{\max}-t_{\min}) \quad (3)$$

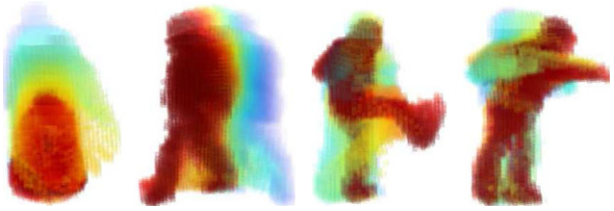


Fig. 3. Motion history volume examples: From left to right: "sit down"; "walk"; "kick"; "punch". Color values: red = current; ...; blue = maximum duration, encode time of last occupancy. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this paper.)

where $t_{\min}$ and $t_{\max}$ are start and end time of an action. Hence, motions loose dependencies on absolute speed and result all in the same length. Section 2.3 shows how we detect these boundaries using a motion energy based segmentation.

The input occupancy function $D(x,y,z,t)$ is estimated using silhouettes and thus, corresponds to the visual hull [12]. Visual hulls present several advantages, they are easy to compute and they yield robust 3D representations. Note however that, as for 2D motion templates, different body proportions may still result in very different templates. Fig. 3 shows examples for motion history volumes.

2.3. Temporal segmentation

Temporal segmentation consist in splitting a continuous sequence of motions into elementary segments. In this work, we use an automatic procedure that we recently introduced in [13]. It relies on the definition of motion boundaries as minima in motion energy, as originally proposed by Marr and Vaina [9]. Such minima correspond either to small rests between motions or to reversals in motion. As it turns out, an approximation of the global motion energy can be effectively computed using MHVs: Intuitively, instant motion can be encoded using MHVs over small time windows (typically 2–10 frames). Then the sum over all voxel values at time t will give a measure of the global motion energy at that time. Next, we search this energy for local minima, and recompute the MHVs based on the detected boundaries. For more details we refer to our work in [13].

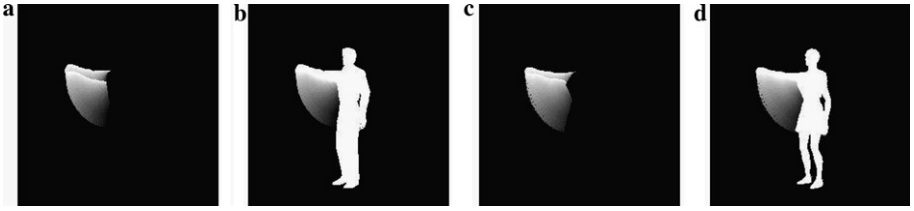


Fig. 2. Motion versus occupancy. Using motion only in image (a), we can roughly gather that someone is lifting one arm. Using the whole silhouette instead, in (b), makes it clear that the right arm is lifted. However the same movement executed by a woman, in (c), compares favorably with the man's action in (a), whereas the whole bodies comparisons between (b) and (d) is less evident.

3. Motion descriptors

Our objective is to compare body motions that are free in locations, orientations and sizes. This is not the case of motion templates, as defined in the previous section, since they encode space occupancy. The location and scale dependencies can be removed by centering, with respect to the center of mass, and scale normalizing, with respect to a unit variance, motion templates, as usual in shape matching. For the rotation, and following Davis and Bobick [6] who used the Hu moments [14] as rotation invariant descriptors, we could consider their simple 3D extensions by Sadjadi and Hall [15]. However, our experiments with these descriptors, based on first and second order moments, were unsuccessful in discriminating detailed actions. In addition, using higher order moments as in [16] is not easy in practice. Moreover, several works tend to show that moments are inappropriate feature descriptors, especially in the presence of noise, e.g., Shen [17]. In contrast, several works, such as that by Grace and Spann [18] and Heesch and Rueger [19], demonstrated better results using Fourier based features. Fourier based features are robust to noise and irregularities, and present the nice property to separate coarse global and fine local features in low and high frequency components. Moreover, they can be efficiently computed using fast Fourier-transforms (FFT). Our approach is therefore based on these features.

Invariance of the Fourier transform follows from the Fourier *shift theorem*: a function $f_0(x)$ and its translated counterpart $f_t(x) = f_0(x - x_0)$ only differ by a phase modulation after Fourier transformation:

$$F_t(k) = F_0(k)\mathrm{e}^{-\mathrm{j}2\pi k x_0}. \tag{4}$$

Hence, Fourier magnitudes $|F_t(k)|$ are shift invariant signal representations. The invariance property translates easily onto rotation by choosing coordinate systems that map rotation onto translation. Popular example is the Fourier–Mellin transform, e.g., Chen et al. [20], that uses log-polar coordinates for translation, scale, and rotation invariant image registration. Recent work in shape matching by Kazhdan et al. [21] proposes magnitudes of Fourier spherical harmonics as rotation invariant shape descriptors.

In a similar way, we use Fourier-magnitudes and cylindrical coordinates, centered on bodies, to express motion templates in a way invariant to locations and rotations around the z-axis. The overall choice is motivated by the assumption that similar actions only differ by rigid transformations composed of scale, translation, and rotation around the z-axis. Of course, this does not account for all similar actions of any body, but it appears to be reasonable in most situations. Furthermore, by restricting the Fourier-space representation to the lower frequencies, we also implicitly allow for additional degrees of freedom in object appearances and action executions. The following section details our implementation.

3.1. Invariant representation

We express the motion templates in a cylindrical coordinate-system:

$$v\left(\sqrt{x^2+y^2}, \tan^{-1}\left(\frac{y}{x}\right), z\right) \rightarrow v(r,\theta,z).$$

Thus, rotations around the z-axis results in cyclical translation shifts:

$$v(x\cos\theta_0 + y\sin\theta_0, -x\sin\theta_0 + y\cos\theta_0, z) \rightarrow v(r,\theta+\theta_0,z).$$

We center and scale-normalize the templates. In detail, if v is the volumetric cylindrical representation of a motion template, we assume all voxels that represent a time step, i.e., for which $v(r,\theta,z) > 0$, to be part of a point cloud. We compute the mean μ and variances σ_r and σ_z in z- and r-direction. The template is then shifted, so that $\mu = 0$, and scale normalized so that $\sigma_z = \sigma_r = 1$.

We choose to normalize in z and r direction, instead of a principal component based normalization, focusing on the main directions human differ on, and assuming scale effects dependent on positions to be rather small. This method may fail aligning, e.g., a person spreading its hand with a person dropping its hand, but gives good results for people performing similar actions, which is more important.

The absolute values $|V(r,k_\theta,z)|$ of the 1D Fourier-transform

$$V(r,k_\theta,z) = \int_{-\pi}^{\pi} v(r,\theta,z)\mathrm{e}^{-\mathrm{j}2\pi k_\theta\theta}\,\mathrm{d}\theta, \tag{5}$$

for each value of r and z, are invariant to rotation along θ.

See Fig. 4 for an illustration of the 1D-Fourier transform. Note that various combinations of the Fourier transform could be used here. For the 1D Fourier-transform the spatial order along z and r remains unaffected. One could say, a maximum of information in these directions is preserved.

An important property of the 1D-Fourier magnitudes is its *trivial ambiguity* with respect to the reversal of the signal. Consequently, motions that are symmetric to the z-axis (e.g., move left arm–move right arm) result in the same motion descriptors. This can be considered either as a loss in information or as a useful feature halving the space of symmetric motions. However, our practical experience shows that most high level descriptions of human actions do not depend on this separation.

In cases where it is important to resolve left/right ambiguities a slightly different descriptor can be used. One such descriptor is the magnitude $|V(k_r,k_\theta,k_z)|$ of the 3D-Fourier transform

$$V(k_r,k_\theta,k_z) = \int_{-\infty}^{\infty}\int_{-\pi}^{\pi}\int_{-\infty}^{\infty} v(r,\theta,z)\mathrm{e}^{-\mathrm{j}2\pi(k_r r + k_\theta\theta + k_z z)}\,\mathrm{d}r\,\mathrm{d}\theta\,\mathrm{d}z, \tag{6}$$

applied to the motion template v. This descriptor is only symmetric with respect to an inversion of all variables,

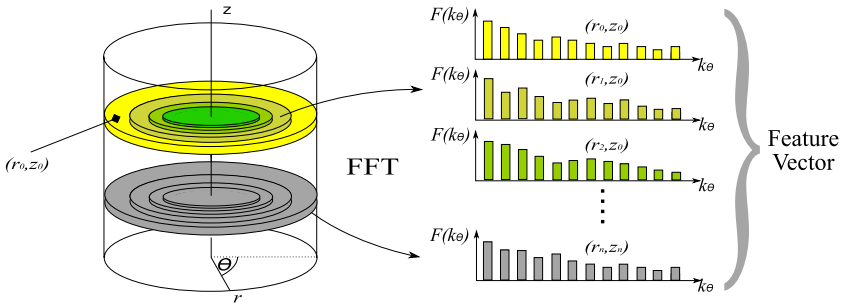


Fig. 4. 1D-Fourier transform in cylindrical coordinates. Fourier transforms over θ are computed for couples of values (r, z). Concatenation of the Fourier magnitudes for all r and z forms the final feature vector.

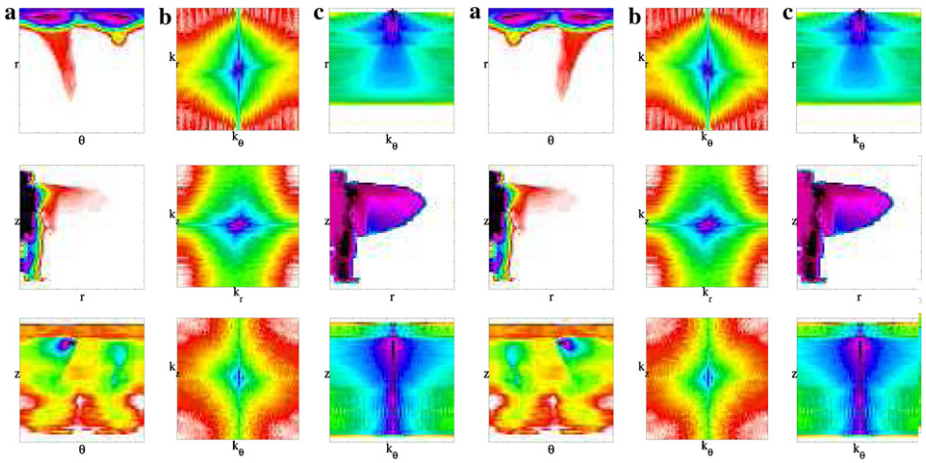


Fig. 5. Volume and spectra of sample motions "lift left arm" (left) and "lift right arm" (right): (a) cylindrical representation in (θ, r), (r, z), (θ, z) averaged over the third dimension for visualization purposes; (b) corresponding 3D-Fourier Spectra; (c) 1D-Fourier spectra. Note that the 3D descriptor treats both motions differently (i.e., top and bottom row (b)), while the 1D descriptors treats them the same.

i.e., humans standing upside-down, which does not happen very often in practice. While our previous work [22] used that descriptor Eq. (6) with success, the results were anyway inferior to those obtained with Eq. (5) and an invariance to left right symmetry proved to be beneficial in many classification cases. A visualization of both descriptors is shown in Fig. 5.

3.2. On invariance vs. exhaustive search

Although we cannot report experiments for lack of space, another significant result of our research is that viewpoint-invariant motion descriptors (Fourier magnitudes) are at least as efficient as methods based on exhaustive search (correlation), at least for comparing simple actions. Numerous experiments have shown that, although it is possible to precisely recover the relative orientations between history volumes using phase or normalized correlation in Fourier space [23], and compare the aligned volumes directly, this almost never improves the classification results. Using invariant motion descriptors is of course advantageous because we do not need to align training examples for learning a class model, or align test examples with all class prototypes for recognition.

4. Classification using motion descriptors

We have tested the presented descriptors and evaluated how discriminant they are with different actions, different bodies or different orientations. Our previous results [22] using a small dataset of only two persons already indicated the high potential of the descriptor. This paper presents results on an extended dataset, the so called *IXMAS* dataset. The dataset is introduced in Section 4.1, followed by classification results using dimensional reduction combined with Mahalanobis distance and linear discriminant analysis (LDA).

4.1. The IXMAS dataset

The Inria Xmas Motion Acquisition Sequences (IXMAS)[2] aim to form a dataset comparable to the current "state-of-the-art" in action recognition. It contains 11 actions, see Fig. 6 for instance, each performed three times by 10 actors (5 males/5 females). To demonstrate the

[2] The data is available on the perception website http://perception.inrialpes.fr in the "Data" section.

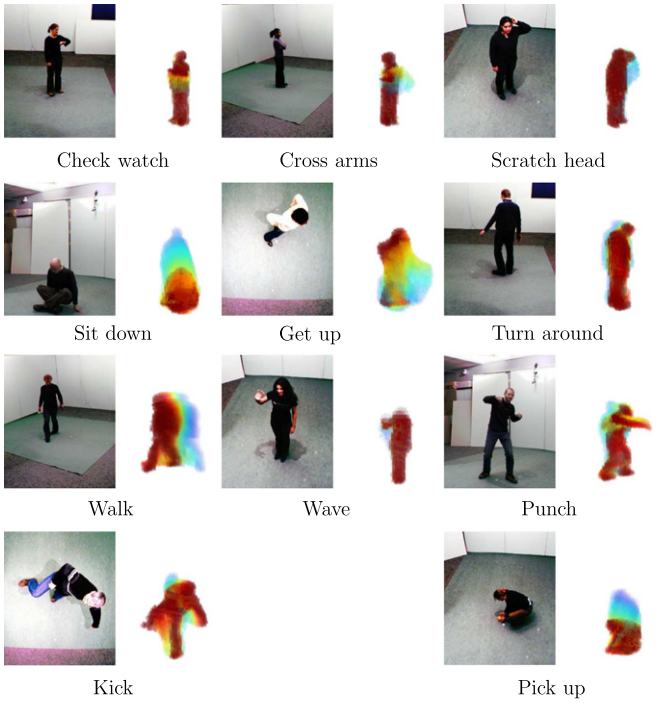


Fig. 6. Eleven actions, performed by 10 actors.

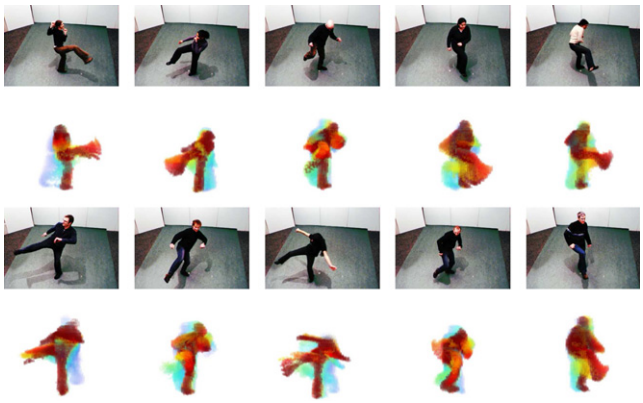


Fig. 7. Sample action "kick" performed by 10 actors.

view-invariance, the actors freely change their orientation for each acquisition and no further indications on how to perform the actions beside the labels were given, as illustrated in Fig. 7.

The acquisition was achieved using five standard Firewire cameras. Fig. 8 shows example views from the camera setup used during the acquisition. From the video we extract silhouettes using a standard background subtraction technique modeling each pixel as a Gaussian in RGB space. Then visual hulls are carved from a discrete space of voxels, where we carve each voxel that not projects into all of the silhouettes. However, there are no special requirements for the visual hull computation and even the simplest method showed to work perfectly with our approach. After mapping into cylindrical coordinates the representation has a resolution of $64 \times 64 \times 64$. Temporal segmentation was performed as described in Section 2.3. Note, that the temporal segmentations splits some of the actions into several elementary parts. To evaluate the descriptor on a selected dataset of primitive motions, we choose from each of the segments the one that best represents the motion. For example the action "check watch" is split into three parts: an upward motion of the arm—several seconds of resting in this position—releasing the arm. From these motions we only use the first for the class "check watch". Another example is the action "walk", that has been broken down into separate steps. Interestingly, in those examples, we were able to classify even moderately complex actions based on one segment only. However, classification of composite actions is a topic of future research.

Fig. 8. Example views of five cameras used during acquisition.

4.2. Classification using Mahalanobis distance and PCA

In initial experiments on a small dataset and with different distance measures (i.e., Euclidean distance, simplified Mahalanobis distance, and Mahalanobis distance + PCA, see also [22]), the combination of a principal component analysis (PCA) dimensional reduction plus Mahalanobis distance based normalization showed best results. Due to the small amount of training samples we only used one pooled covariance matrix for all classes. Interestingly, we found that the method extends well to larger datasets and even competes with linear discriminant analysis (LDA), as will be shown in Section 4.3.

PCA is a commonly used method for dimensional reduction. Data points are projected onto a subspace that is chosen to yield the reconstruction with minimum squared error. It has been shown that this subspace is spanned by the largest eigenvectors of the data's covariance Σ, and corresponds to the directions of maximum variance within the data. Further, by normalization with respect to the variance, an equally weighting of all components is achieved, similar to the classical use of Mahalanobis distances in classification, but here computed for one pooled covariance matrix.

Every action class in the data-set is represented by the mean value of the descriptors over the available population in the action training set. Any new action is then classified according to a Mahalanobis distance associated to a PCA based dimensional reduction of the data vectors. One pooled covariance matrix Σ based on the training samples of all classes $\mathbf{x}_i \in \mathbb{R}^d$, $i = 1, \ldots, n$ was computed:

$$\Sigma = \frac{1}{n} \sum_{i}^{n} (\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^{\top}, \tag{7}$$

where $\mathbf{m}$ represents the mean value over all training samples.

The Mahalanobis distance between feature vector $\mathbf{x}$ and a class mean $\mathbf{m}_i$ representing one action is:

Table 1
IXMAS data classification results. Results on PCA, PCA + Mahalanobis distance based normalization using one pooled covariance, and LDA are presented

Action	PCA (%)	Mahalanobis (%)	LDA (%)
Check watch	46.66	86.66	83.33
Cross arms	83.33	100.00	100.00
Scratch head	46.66	93.33	93.33
Sit down	93.33	93.33	93.33
Get up	83.33	93.33	90.00
Turn around	93.33	96.66	96.66
Walk	100.00	100.00	100.00
Wave hand	53.33	80.00	90.00
Punch	53.33	96.66	93.33
Kick	83.33	96.66	93.33
Pick up	66.66	90.00	83.33
Average rate	73.03	93.33	92.42

$$d(\mathbf{m}_i, \mathbf{x}) = (\mathbf{x} - \mathbf{m}_i)^\top V \Lambda^{-1} V^\top (\mathbf{x} - \mathbf{m}_i),$$

with Λ containing the k largest eigenvalues $\lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant \lambda_k$, $k \leqslant n-1$, and V the corresponding eigenvectors of Σ. Thus feature vectors are reduced to k principal components.

Following this principle, and reducing the initial descriptor Eq. (5) to $k = 329$ components an average classification rate of 93.33% was obtained with leave-one-out cross validation, where we successively used 9 of the actors to learn the motions and the 10th for testing. Note that in the original input space, as well as for a simple PCA reduction without covariance normalization the average rate is only 73.03%. Detailed results are given in Table 1.

4.3. Classification using linear discriminant analysis

For further data reduction, class specific knowledge becomes important in learning low dimensional representations. Instead of relying on the eigen-decomposition of one pooled covariance matrix, we use here a combination of PCA and Fisher linear discriminant analysis (LDA), see e.g., Swets and Weng [24], for automatic feature selection from high dimensional data.

First PCA is applied, $Y = V^\top X$, $V = [\mathbf{v}_1, \ldots, \mathbf{v}_m]$, to derive a $m \leqslant n - c$ dimensional representation of the data points x_i, $i = 1, \ldots, n$. The class-number c dependent limit is necessary to guaranty non-singularity of matrices in discriminant analysis.

Fisher discriminant analysis defines as within-scatter matrix:

$$S_w = \sum_i^c \sum_j^{n_i} (\mathbf{y}_j - \mathbf{m}_i)(\mathbf{y}_j - \mathbf{m}_i)^\top, \tag{8}$$

and between-scatter matrix:

$$S_b = \sum_i^c (\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^\top, \tag{9}$$

and aims at maximizing the between-scatter while minimizing the within-scatter, i.e., we search a projection W that maximize $\frac{\det(S_b)}{\det(S_W)}$. It has been proven that W equal to the largest eigenvectors of $S_w^{-1} S_b$ maximizes this ratio. Consequently, a second projection $Z = W^\top Y$, $W = [w_1, \ldots, w_k]$, $k \leqslant c - 1$ is applied to derive our final feature representation Z.

During classification each class is represented by its mean vector $\mathbf{m}_i$. Any new action $\mathbf{z}$ is then classified by summing Euclidean distances over the discriminant features and with respect to the closest action class:

$$d(\mathbf{m}_i, \mathbf{z}) = \|\mathbf{m}_i - \mathbf{z}\|^2. \tag{10}$$

In the experiments the magnitudes of the Fourier representation Eq. (5) are projected onto $k = 10$ discriminant features. Successively we use nine of the actors to learn the motions, the 10th is used for testing. The average rate of correct classifications is then 92.42%. Class specific results are shown in Table 1 and Fig. 9.

We note that we obtain much better results with the Mahalanobis distance, using the 329 largest components of the PCA decomposition, as compared to using the PCA components alone. LDA allows us to further reduce the number of features to 10, but otherwise does not further improve the overall classification results.

4.4. Motion history vs. motion energy and key frames

With the same dataset as before, we compare our MHV based descriptors with a combination of key poses and

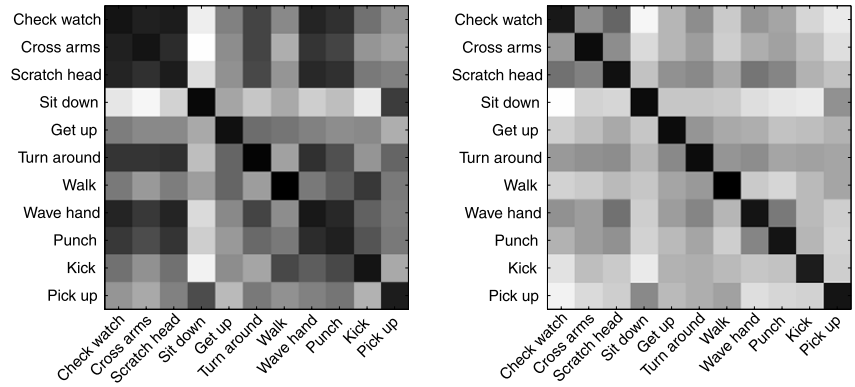


Fig. 9. Average class distance: (Left) before discriminant analysis. (Right) after discriminant analysis.

Table 2
IXMAS data classification results

Action (%)	MEV (%)	Key frame (%)	MHV (%)
Check watch	86.66	73.33	86.66
Cross arms	80.00	93.33	100.00
Scratch head	73.33	86.66	93.33
Sit down	70.00	93.33	93.33
Get up	46.66	53.33	93.33
Turn around	90.00	60.00	96.66
Walk	100.00	80.00	100.00
Wave hand	80.00	76.66	80.00
Punch	93.33	80.00	96.66
Kick	90.00	90.00	96.66
Pick up	70.00	96.66	90.00
Average rate	80.00	80.30	93.33

Results using the proposed MHVs are presented. For comparison we also include results using binary MEVs and key frame descriptors.

energy volumes. While Davis and Bobick suggested in the original paper the use of history and binary images, our experiments with motion volumes showed no improvement in using a combination of MHVs and the binary MEVs. We repeated the experiment described in Section 4.3, for MEVs. Using the binary information the recognition rate becomes 80.00% only. See Table 2 for detailed results. As can be expected: reverse actions, e.g., "sit down"–"get up", present lower scores with MEVs than with MHVs. The MHVs show also better performance in discriminating actions on more detailed scales, e.g., "scratch head"–"wave".

Also, to show that integration over time plays a fundamental role of information, we compare our descriptor with descriptors based on a single selected *key frame*. The idea of key frames is to represent a motion by one specific frame, see e.g., Carlson and Sullivan [25]. As invariant representation, we use the magnitudes of Eq. (5). For the purpose of this comparison we simply choose the last frame of each MHV computation as corresponding *key frame*. The average recognition rate becomes 80.30%. While motion intensive action, e.g., "walk"–"turn around" score much lower, a few pose expressive actions, e.g., "pick up", achieve a better score. This may indicate that not all actions should be described with the same features.

We conclude, that invariant Fourier descriptors of binary motion volumes and key frames are suitable for motion recognition as well. However, the use of additional motion information, as present in the motion history volumes, in both cases distinctly improves the recognition.

4.5. Classification on video sequences

The previous experiments show that the descriptor performs well in discriminating selected sets of learned motions. In this experiment we test the descriptor on unseen motion categories as they appear in realistic situations. For this purpose we work on the raw video sequences of the IXMAS dataset. In a first step the dataset is segmented into small motion primitives using the automatic

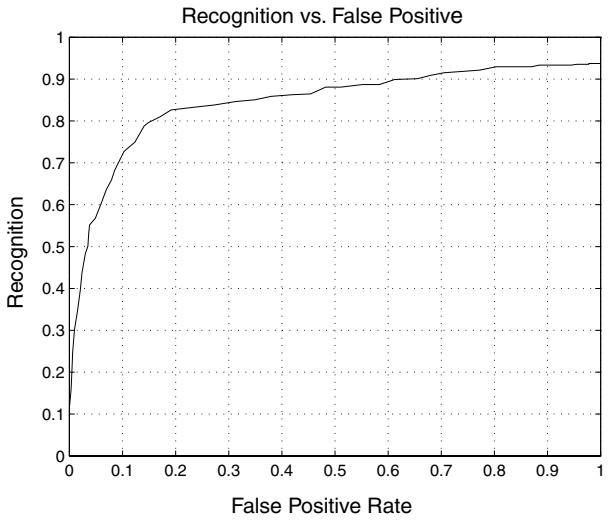


Fig. 10. Recognition on raw video sequences: plots recognition rate into 11 classes against false positive rate.

segmentation. Then each segment is either recognized as one of the 11 learned classes or rejected. As in the previous experiments, we work in PCA space spanned by the 11 sample motions and perform nearest-mean assignment. To decide for the "reject"-class we use a global threshold on the distance to the closest class.

The automatic segmentation of the videos results in 1188 MHVs, corresponding to approximately 23 min of video. In manual ground truth labeling we discover 495 known motions and 693 "reject"-motions. Note, that such a ground truth labeling is not always obvious. A good example is the "turn"-motion that was included in the experiments, but additional turn-like motions also appear as the actors where free to change position during the experiments. Moreover, it might be that an actor was accidentally checking his watch or scratching his head.

Testing in a leave-one-out manner, using all possible combinations of 9 actors for training and the remaining 10th for testing, we show a multi-class ROC curve, Fig. 10, plotting the average number of correctly classified samples, against the number of false positives. We found a maximal overall recognition rate (including correctly rejected motions) of 82.79%, for 14.08% false positives and 78.79% correctly classified motions. Fig. 11 shows the average distance between the "reject"-motions and the learned classes.

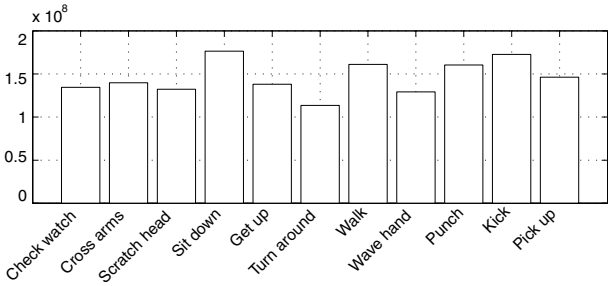


Fig. 11. Average distance between "reject"-samples and training classes.

The experiments demonstrate the ability of MHVs even to work with large amounts of data and under realistic situations (23 min of video, 1188 motion descriptors). The segmentation proved to almost always detect the important parts of motions; MHVs showed good quality in discriminating learned and unseen motions.

An obvious problem for the false detections, is the nearly infinite class of possible motions. Modeling unknown motions may require more than a single threshold and class, multiple classes and explicit learning on samples of unknown motions becomes important. Another problem we found is, that many motions can not be modeled by a single template. Small motions may seem very similar, but over time belong to very different actions. For example the turn around motion is split into several small steps that may easily be confused with a single side step. In such cases temporal networks over templates, as e.g., in an HMM approach, must be used to resolve these ambiguities. However, we leave this for future work.

5. Conclusion

Using a data set of 11 actions, we have been able to extract 3D motion descriptors that appear to support meaningful categorization of simple action classes performed by different actors, irrespective of viewpoint, gender and body sizes. Best results are obtained by discarding the phase in Fourier space and performing dimensionality reduction with a combination of PCA and LDA. Further, LDA allows a drastic dimension reduction (10 components). This suggests that our motion descriptor may be a useful presentation for view invariant recognition of an even larger class of primitive actions. Our current work is suited to segmentation of composite actions into primitives, and classification of sequences of the corresponding LDA coefficients.

References

[1] J. Neumann, C. Fermller, Y. Aloimonos, Animated heads: from 3d motion fields to action descriptions, in: DEFORM/AVATARS, 2000.
[2] A. Kojima, T. Tamura, K. Fukunaga, Natural language description of human activities from video images based on concept hierarchy of actions, International Journal of Computer Vision 50 (2) (2002) 171–184.
[3] Y.A. Ivanov, A.F. Bobick, Recognition of visual activities and interactions by stochastic parsing, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (8) (2000) 852–872.
[4] R.D. Green, L. Guan, Quantifying and recognizing human movement patterns from monocular video images-part I: a new framework for modeling human motion, IEEE Transactions on Circuits and Systems for Video Technology 14 (2) (2004) 179–190.
[5] A.A. Efros, A. Berg, G. Mori, J. Malik, Recognizing action at a distance, in: IEEE International Conference on Computer Vision, 2003.
[6] J.W. Davis, A.F. Bobick, The recognition of human movement using temporal templates, IEEE Transactions on Pattern Analysis and Machine Intelligence 23 (3) (2001) 257–267.
[7] T. Syeda-Mahmood, M. Vasilescu, S. Sethi, Recognition action events from multiple viewpoints, in: EventVideo0l, 2001, pp. 64–72.
[8] A. Yilmaz, M. Shah, Actions sketch: a novel action representation, in: IEEE Conference on Computer Vision and Pattern Recognition, 2005, pp. I: 984–989.
[9] D. Marr, L. Vaina, Representation and recognition of the movements of shapes, Proceedings of the Royal Society of London B 214 (1982) 501–524.
[10] R. Jackendoff, On beyond zebra: the relation of linguistic and visual information, Cognition 20 (1987) 89–114.
[12] A. Laurentini, The visual Hull concept for silhouette-based image understanding, IEEE Transactions on Pattern Analysis and Machine Intelligence 16 (2) (1994) 150–162.
[13] D. Weinland, R. Ronfard, E. Boyer, Automatic discovery of action taxonomies from multiple views, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2006. Available from: <http://perception.inrialpes./Publications/2006/WRB06>.
[14] M.-K. Hu, Visual pattern recognition by moment invariants, IRE Transactions on Information Theory IT-8 (1962) 179–187.
[15] F. Sadjadi, E. Hall, Three-dimensional moment invariants, IEEE Transactions on Pattern Analysis and Machine Intelligence 2 (2) (1980) 127–136.
[16] C. Lo, H. Don, 3-d moment forms: their construction and application to object identification and positioning, PAMI 11 (10) (1989) 1053–1064.
[17] D. Shen, H.H.-S. Ip, Discriminative wavelet shape descriptors for recognition of 2-d patterns, Pattern Recognition 32 (2) (1999) 151–165.
[18] A.E. Grace, M. Spann, A comparison between Fourier–Mellin descriptors and moment based features for invariant object recognition using neural networks, Pattern Recognition Letters 12 (10) (1991) 635–643.
[19] D. Heesch, S.M. Rueger, Combining features for content-based sketch retrieval – a comparative evaluation of retrieval performance, in: Proceedings of the 24th BCS-IRSG European Colloquium on IR Research, Springer-Verlag, London, UK, 2002, pp. 41–52.
[20] Q. Chen, M. Defrise, F. Deconinck, Symmetric phase-only matched filtering of Fourier–Mellin transforms for image registration and recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 16 (12) (1994) 1156–1168.
[21] M. Kazhdan, T. Funkhouser, S. Rusinkiewicz, Rotation invariant spherical harmonic representation of 3d shape descriptors, in: Symposium on Geometry Processing, 2003.
[22] D. Weinland, R. Ronfard, E. Boyer, Motion history volumes for free viewpoint action recognition, in: IEEE International Workshop on modeling People and Human Interaction, 2005. Available from: <http://perception.inrialpes.fr/Publications/2005/WRB05>.
[23] C.D. Kuglin, D.C. Hines, The phase correlation image alignment method, in: IEEE International Conference on Cybernetics and Society, 1975, pp. 163–165.
[24] D.L. Swets, J. Weng, Using discriminant eigen features for image retrieval, IEEE Transactions on Pattern Analysis and Machine Intelligence 18 (8) (1996) 831–836.
[25] S. Carlsson, J. Sullivan, Action recognition by shape matching to key frames, Workshop on Models versus Exemplars in Computer Vision, 2001.

Automatic Discovery of Action Taxonomies from Multiple Views

Daniel Weinland * Remi Ronfard Edmond Boyer
Project PERCEPTION, INRIA Rhone-Alpes,
38334 Montbonnot Saint Martin, France
{weinland, ronfard, eboyer}@inrialpes.fr

Abstract

We present a new method for segmenting actions into primitives and classifying them into a hierarchy of action classes. Our scheme learns action classes in an unsupervised manner using examples recorded by multiple cameras. Segmentation and clustering of action classes is based on a recently proposed motion descriptor which can be extracted efficiently from reconstructed volume sequences. Because our representation is independent of viewpoint, it results in segmentation and classification methods which are surprisingly efficient and robust. Our new method can be used as the first step in a semi-supervised action recognition system that will automatically break down training examples of people performing sequences of actions into primitive actions that can be discriminatingly classified and assembled into high-level recognizers.

1 Introduction

Recognizing actions of human actors from video is an important topic in computer vision with many fundamental applications in video surveillance, video indexing and social sciences. From a computational perspective, actions are best defined as four-dimensional patterns in space and in time [10]. Yet, much current research in computer vision ignores this fact and attempts to learn action models directly from monocular video [3, 6, 1]. In our work, we use multiple video cameras and shape-from-silhouette techniques to obtain four-dimensional recordings of action sequences. We compute new motion descriptors based on - *motion history volumes* - which fuse action cues, as seen from different viewpoints and over short time periods, into a single three dimensional representation. From that representation, we are able to segment the action streams into primitives and to cluster those primitives into a hierarchy of primitive action classes.

*D. Weinland is supported by a grant from the European Community under the EST Marie-Curie Project Visitor.

Our long-term goal is to automatically generate high-level descriptions of video sequences in terms of the actions that can be recognized or inferred from the given visual input. Actions generally fall under two distinct categories - composite actions which can be broken down into distinct temporal parts or segments, and primitive actions, which cannot be broken down further. In order to build a general action recognizer, we need the ability to break down a given sequence into primitive action segments, to label those segments into primitive actions using a vocabulary of learned action models, and to assemble the labeled segments into composite actions using concept hierarchies [8] or grammars [11].

In this work, we use a novel motion descriptor based on the *motion history volume* (MHV) which summarizes the action content of a short multi-view sequence without knowledge of body parts [15]. We automatically segment action sequences into primitive actions which can be represented by a single MHV and we cluster the resulting MHVs into a hierarchy of action classes, which allow us to recognize multiple occurrences of repeating actions. We are able to perform those two steps automatically, mainly because MHVs work in a volume space which considerably reduces the ambiguities traditionally associated with changes in viewpoints and occlusions even in multiple views.

As a concrete example, we asked two members of our lab to perform a sequence of simple actions, each repeated several times with different poses and styles, in front of 6 calibrated cameras. The resulting data set consists of unsegmented and unlabeled synchronized video sequences such as the one depicted in Figure 1. Using the new motion descriptor, we were able to segment (Section 5) and cluster (Section 6) such sequences into primitive actions, which we used as training examples for learning statistical classifiers. Such a semi-supervised scheme is important in practical terms because it facilitates the creation of large training sets for action recognition in the large.

Our method generates action taxonomies based on purely visual cues since we create higher-level action classes by abstracting two or more recorded actions which

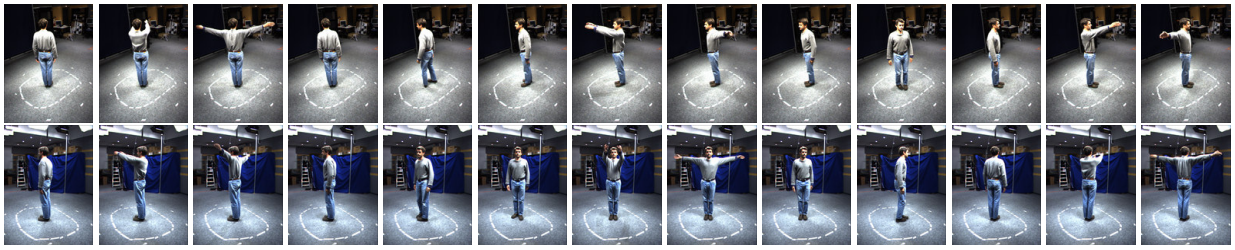


Figure 1. Example action sequence: Raise arms - rotate arms - turn left - raise arms - rotate arms - turn left - raise arms - rotate arms, seen from two different viewpoints. Such sequences are difficult to segment and label consistently from monocular cues, but are easily segmented and labeled using our view-independent motion descriptors.

look the same from all viewpoints (as measured by the differences in a metric space of motion descriptors extracted from their MHVs). We believe this is an important step towards building complete, semantic taxonomies of actions and plans.

The paper is organized as follows. We review related work in Section 2. We briefly review motion history volumes and associated view-independent motion descriptors in Sections 3 and 4. We describe our segmentation algorithm in Section 5 and our clustering algorithm in Section 6. Both algorithms are based on the motion descriptors introduced in Section 4. Finally in Section 7 we describe a semi-supervised action classification system which uses the proposed algorithms to automatically segment and label the training and test sequences, and report initial results obtained on a limited but realistic data set.

2 Related work

Segmentation and labeling of action sequences from *monocular video* is a difficult problem that has received considerable attention in recent years. Rittscher et al. learn dynamical models of actions from tracked contours and use them to segment new sequences [12]. Rui and Anandan perform an SVD decomposition of a long sequence of optical flow images and detect discontinuities in the trajectories of selected SVD components to segment video into *motion patterns* [13]. Zelnik-Manor and Irani cluster video sequences into *events* using normalized cuts on multiresolution sequences of spatio-temporal gradient magnitudes [16]. Brand and Kettnaker use unsupervised HMMs to perform simultaneous segmentation and clustering of actions from sequences of human silhouettes [3]. Wang et al. also use unsupervised HMMs to segment 2D hand motions and extract a vocabulary of musical conducting gestures, which allows them to describe video sequences optimally in the sense of minimum description length [14]. Feng and Cham compare methods for segmenting action sequences with or without body part correspondences and propose a hybrid scheme that can handle *ambiguous correspondences* [5]. All such methods work only with restricted variations in viewpoint, which make them ill-suited to cases such as of Figure 1 where each action is performed multiple times with vastly different poses.

Segmentation and labeling of action sequences from *multiple views* is a relatively little-studied area. Previous work assumes either that the cameras are uncalibrated (so that reconstruction is not possible) or that a full human body model can be recovered (so that reconstruction includes body part recognition and tracking). Thus, Marr and Vaina discuss the problem of segmenting the 3D movement of shapes and suggest the use of local minima of the 3D motion of human limbs as natural transitions between primitive movements [9]. Campbell et al. investigate several view-invariant features for action classification from face and hand tracking based on using multi-view stereo [4]. Davis and Bobick use *motion templates* in multiple views, but they assume uncalibrated cameras and are therefore unable to perform 3D reconstruction [2]. Similarly, Ogale et al. cluster action sequences in multiple views separately by detecting minima and maxima of optical flow inside silhouettes and matching the selected silhouettes using phase correlation [11]. This allows them to learn action grammars from examples recorded by multiple cameras, but their grammars remain viewpoint-dependent.

To the best of our knowledge, no previous work has attempted to perform segmentation and clustering from *volumetric* reconstructions. In this paper, we propose such a method, which extends monocular methods most naturally by introducing view-invariant motion descriptors built from silhouettes in multiple calibrated views. Compared with previous work, our method has the advantage that we perform all three steps of segmenting, clustering and classifying action sequences in 3D with a representation which is fully view-invariant, and is much simpler to recover than a full human body model.

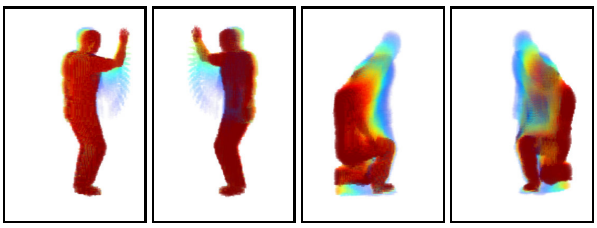


Figure 2. Example motion history volumes: "Lift arm" and "knee", rendered from different viewpoints. Colors: red = current, ..., blue = maximum duration, encode time of last occupancy.

3 Motion History Volumes

In this section, we present the 3D motion templates on which we ground our approach. These templates are a 3D generalization of the 2D motion templates introduced by Bobick and Davis in [2]. Both 2D and 3D templates are based on image silhouettes, which are binary valued functions indicating object occupancies in image projections.

Motion templates encode the history of motion occurrences. In 2D images, pixel values are therefore multiple-values recording how recently motion occurred at a pixel. The extension to 3D is straightforward by considering voxels instead of pixels, and the space occupancy function $D(x, y, z, t)$ over time steps t. Voxel values in the MHV at time t are then defined by:

$$v_\tau(x, y, z, t) = \begin{cases} \tau & \text{if } D(x, y, z, t) \\ \max(0, v_\tau(x, y, z, t-1) - 1) & \text{oth.} \end{cases} \quad (1)$$

where τ is the maximum duration a motion is stored.

The input occupancy function $D(x, y, z, t)$ is estimated using silhouettes and is defined by the visual hull at time t. Voxelic visual hulls are easy to compute and yield robust 3D representations. Note however that, as for 2D motion templates, different body proportions may still result in different templates. Figure 2 shows examples for motion history volumes.

4 Motion Descriptors

To compare or discriminate motions, we need to find a representation which is invariant to transformations in locations, orientations or sizes. To this purpose, we use both alignment and invariant descriptors based on motion templates and Fourier transform. The idea is first to center scale-normalized motion history volumes into a cylindrical coordinate system where the z-axis is aligned with the vertical direction. Hence, dependencies on scale and horizontal translations are removed. For rotations around the vertical axis, we use the fact that they correspond to translations in the cylindrical coordinate systems, and that a function $f_0(x)$ and its translated counterpart $f_t(x) = f_0(x - x_0)$ only differ by a phase modulation after Fourier transform:

$$F_t(k) = F_0(k)e^{-j2\pi k x_0}. \quad (2)$$

Thus absolute values of the Fourier transform are rotation invariant descriptors.

The choice made here is motivated by the assumption that similar actions only differ by rigid transformations composed of scale, translation, and rotation around the z-axis. Of course, this does not account for all similar actions of any body shape, but it appears to be reasonable in most situations. In addition, restricting the Fourier-space representation to the lower frequencies also implicitly allows for additional degrees of freedom in object appearances and action executions. Our experiments also show that Fourier magnitudes provide more discriminative information than correlation features when comparing actions. The following section details our exact implementation.

Alignment We express the motion templates in a cylindrical coordinate-system:

$$v(\sqrt{x^2 + y^2}, \tan^{-1}\left(\frac{y}{x}\right), z) \rightarrow v(r, \theta, z).$$

Thus rotations around the z-axis results in cyclical translation shifts:

$$v(x\cos\theta_0 + y\sin\theta_0, -x\sin\theta_0 + y\cos\theta_0, z) \rightarrow v(r, \theta + \theta_0, z).$$

We center and scale-normalize the templates. In detail, if v is the volumetric cylindrical representation of a motion template, we assume all voxels that represent a time step, i.e. for which $v_0(r, \theta, z) > 0$, to be part of a point cloud. We compute the mean μ and variances σ_r and σ_z in z- and r-direction. The template is then shifted, so that $\mu = 0$, and scale normalized so that $\sigma_z = \sigma_r = 1$. We choose to normalize in z and r direction, instead of a PCA based normalization, focusing on the main directions human differ on, and assuming scale effects dependent on positions to be rather small. This method may fail aligning e.g. a person spreading its hand with a person dropping its hand, but gives good results for people performing similar actions, which is more important.

Invariant descriptors In the new coordinate system we apply a 1D Fourier-transform over θ for each value r and z:

$$V(r, k_\theta, z) = \int_{-\pi}^{\pi} v(r, \theta, z)e^{-j2\pi k_\theta \theta} d\theta, \quad (3)$$

and take as invariant features the magnitudes:

$$f(r, k_\theta, z) = |V(r, k_\theta, z)|. \quad (4)$$

Note that various combination of the Fourier transform could be used here, for example magnitudes of the 3D Fourier-transform over all dimensions r, θ, z as we did in [15]. We use the Fourier transform over the single dimension θ to preserve exact spatial information in the remaining directions. Such spatial information appears to be important when segmenting motions into elementary actions. The counterpart is that the above descriptor (4) is ambiguous with axial symmetries along the z-axis, hence similar actions performed by the left or right body parts can be difficult to discriminate.

It should also be mentioned here that to preserve the properties of the Fourier transform (e.g. robustness to noise, separation in fine and coarse features) for all dimensions, an additional 2D Fourier-transform can be applied to $f(r, k_\theta, z)$ for r and z:

$$\hat{V}(\omega_r, k_\theta, \omega_z) = \iint_{-\infty}^{\infty} |V(r, k_\theta, z)| e^{-j2\pi(\omega_r r + \omega_z z)} dr dz. \quad (5)$$

5 Temporal Segmentation

Temporal segmentation consists in splitting a sequence of motions into elementary segments. It is a necessary preliminary step to higher level processing of motion sequences including classification and clustering. In supervised approaches, segments are usually manually labeled in an initial set of motion sequences, and further operations are achieved by correlating unknown motion sequences with these learned segments on a frame by frame basis, using possibly various temporal scales [2, 7]. In this paper, we do not assume such *a priori* knowledge and propose instead a simple but efficient approach to automatically segment 3D motion sequences.

Any temporal segmentation relies on the definition of elementary motion segments. There are two main approaches to segmentation: Energy minima can be used to detect reversal of motion direction, following an early proposal by Marr and Vaina [9]. Or discontinuities can be used to detect changes in the temporal pattern of motion [13]. From experiments we found energy minima more stable, i.e. similar action sequences are segmented more consistently. The function over time that we segment is then a global motion energy function. This function is an approximation of the global body velocity estimated using the motion history volumes. It is based on the observation that rest states correspond to instants where few motions only occur, and thus result in few voxels encoding motion in the MHV, when small temporal windows are considered. Therefore, segment detection simply consists in finding minima of the sum of voxel values in the MHV, assuming a small value for τ in 1. Figure 3 shows several examples of sequences segmented this way. As can be seen in the figure, detection of energy minima is fairly unambiguous in this examples.

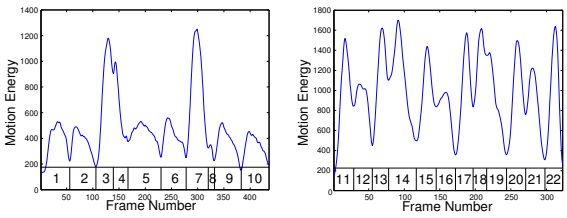


Figure 3. Motion Energy for action: Lift arms - rotate arms - lower arms and turn in new position. Executed three times by (left) female actor, (right) male actor. Local energy minima serve as segmentation criteria of sequences. Motion volumes for each segment are shown in Figure 4.

In the implementation we use a derivative of Gaussian filter and zero crossing to detect the minima. Parameter τ in equation (1) was set to constant 10 frames during all experiments. Temporal scale was not important for detection of all relevant segments. In practice, the minima detection appears to be very successful in segmenting motions, even for coupled motions, like moving torso and arms in parallel, local minima occur. Of course, this measure is still sensitive to small variations of velocity that can result in local minima. However, by allowing a possible over-segmentation the method will detect most of the motion segment boundaries.

6 Action taxonomies

Given a segmented action sequence, we would like to recognize multiple occurrences of the same primitive actions and to label the sequence accordingly. This capability will be important in the next section when we attempt to train classifiers for all primitive actions in a semi-supervised fashion.

We build an action taxonomy from a segmented sequence by hierarchically clustering the segments into classes. Initially, each segment is a single occurrence of its own action class, and is represented as a single point in the space of view-invariant motion descriptors of Section 4, which is a high-dimensional Euclidean space. We then apply a standard hierarchical clustering method to the segments. This creates a binary tree of action classes, where each class is now represented by a point cloud in the space of motion descriptors (see Figure 6).

We report experiments on two different datasets of increasing complexity. In each we segment the sequences as explained in section 5 and compute a single MHV per segment. This is illustrated in Figures 4 and 8. The experiments

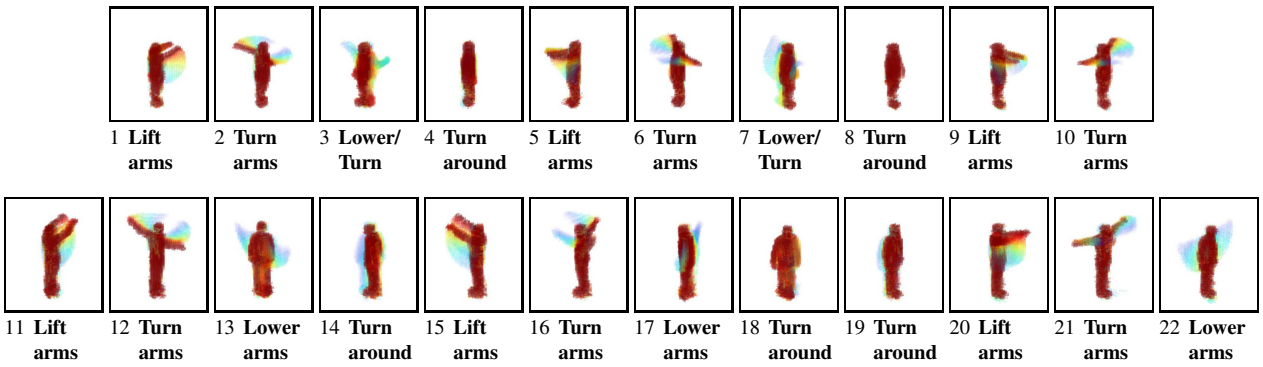


Figure 4. History volumes computed at segments of varying duration, and their clusters, using segmentation from Figure 3. (Top) female actor repeating three times: Lift arms ahead - rotate arms - lower arms and turn in new position. (Bottom) the same done by a male actor, from original sequence shown in Figure 1. The clusters are labeled manually for presentation purposes.

were conducted on MHVs obtained from 6 silhouettes extracted using a standard background subtraction method. The resulting motion templates were mapped into a discrete cylindrical coordinate representation of size $64 \times 64 \times 64$. Clustering was achieved using an agglomerative scheme, where the distance between objects is the Euclidean distance, and clusters were linked according to their furthest neighbor. The first dataset shows how actions performed by different persons, with different bodies, are handled by our system. The second dataset is a more realistic set of natural actions in arbitrary orders. Its interpretation is less straightforward, but it gives strong insights on the potential of our motion descriptors to yield consistent high-level interpretations.

6.1 Clustering on Primitive Actions

Here a dataset of 22 motion sequences performed by both a male and a female actor were considered. Segmented key actions are shown in Figure 5. The actors perform successively each action three times while changing their orientations in between. The automatic motion segmentation returns 203 motion volumes (100 for the woman, 103 for the man). We start by computing a dendrogram of all male segments, using Euclidean distances and furthest neighbor assignments. A good trade-off between motion variation within single clusters and multiple clusters having same labels is then to cut the hierarchy into 21 clusters. All segments inside these clusters are labeled according to the most obvious interpretation. From these labels, the 21 clusters are then labeled with respect to the most current actions which occurs in each cluster. Figure 6 shows the labeled dendrogram. Within these clusters, 7 (6.8%) actions were obviously assigned a wrong cluster, 4 actions give birth to single clusters, and one cluster is ambiguous (lower or lift arm sideways).

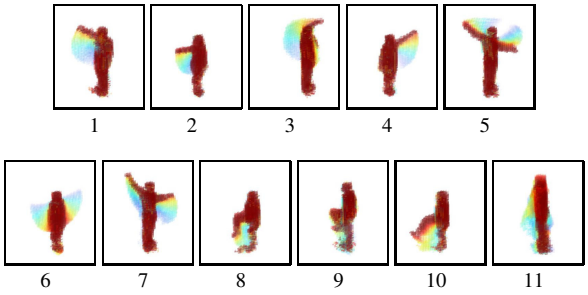


Figure 5. Perspective views of the motion history volumes computed for each action category. (1) lift right arm ahead. (2) lift right arm sideways. (3) lift left arm sideways ahead. (4) lift left arm sideways. (5) rotate both arms lifted. (6) lower both arms sideways. (7) lift both arms sideways. (8) lift right leg bend knee. (9) lift left leg bend knee. (10) lift right leg firm. (11) jump.

We next compute a hierarchy from the male and female data. The procedure is the same as in the previous experiment. Du to higher variations in the dataset the clusters result in a coarser action grouping. A good trade-off between motion variation within single clusters and multiple clusters having same labels is this time to cut the hierarchy into 9 clusters, as shown in Figure 7. With respect to this labeling only two actions are wrongly assigned.

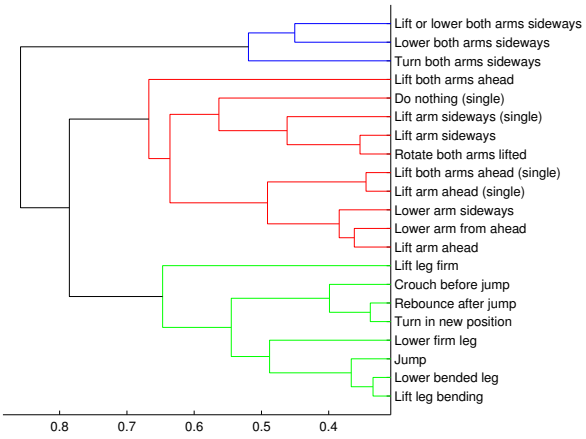


Figure 6. Hierarchical clustering of 103 male actions. 21 top nodes labeled with respect to the most occurring action.

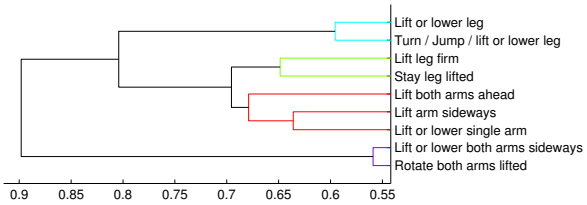


Figure 7. Hierarchical clustering of 203 male and female actions. 9 top nodes labeled with respect to the most occurring action.

6.2 Clustering on Composite Actions

In another clustering experiment we used a different dataset of actions with a much more complex semantics. Those sequences are pantomimes of various daily life actions such as catching a ball, picking up, stretching, laughing, etc. The segmentation and clustering methods were applied to each of these sequences. Figures 8 and 9 show the segmented motion templates and the hierarchy obtained for one such sequence. Again groups of higher level actions in Figure 9 have a simple interpretation such as lift or lower arms. Note also the group *rest in position* where segments without motion, typically between actions, have been consistently clustered.

7 Semi-supervised classification

In this section we use the MHV clusters as training data to learn discriminant classifiers for each of the discovered action classes. We use the motion templates that have been

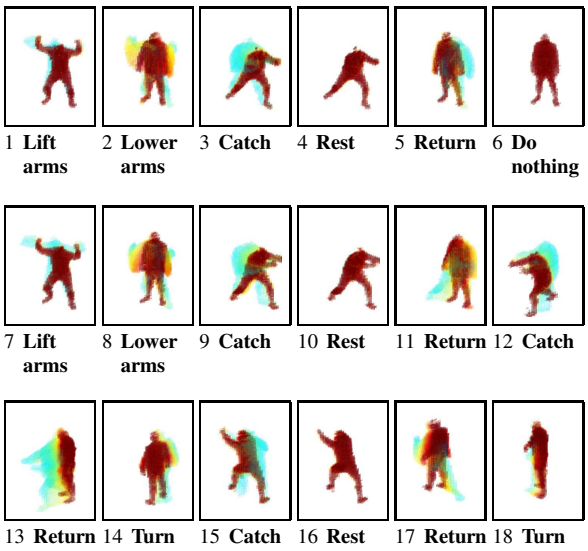


Figure 8. History Volumes for pantomime sequence "catching ball".

automatically extracted in the previous section, i.e. we use the 11 actions corresponding to the key labels in Figure 5. Each action is represented each by 3 samples per actor. One example per class is shown in Figure 5.

We split the set into two configurations: woman/man and man/woman. While simple, this test shows how the proposed descriptors discriminate actions with different bodies. Every action class in the data-set is represented by the mean value of the descriptors over the available population in the action training set. Any new action is then classified

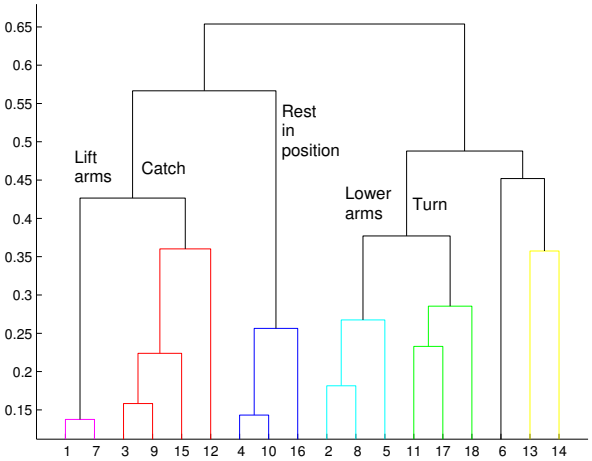


Figure 9. Hierarchical clustering of "catching ball" sequence.

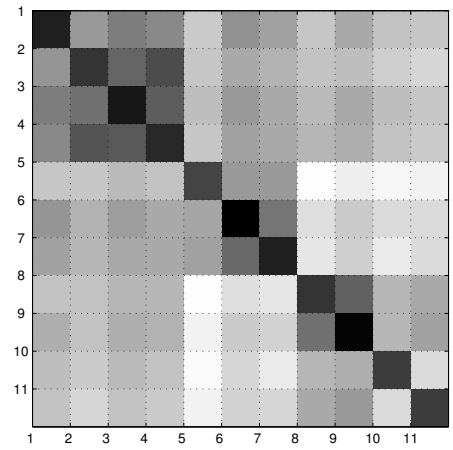


Figure 10. Average distances in feature space between male action classes and female samples. Actions see Figure 5.

according to a Mahalanobis distance associated to a PCA (Principal Component Analysis) based dimensional reduction of the data vectors.

One pooled covariance matrix Σ based on all training samples $\mathbf{x}_i \in \mathbb{R}^d$, $i = 1, \ldots, n$ was computed:

$$\Sigma = \frac{1}{n} \sum_{i}^{n} (\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^\top, \tag{6}$$

where $\mathbf{m}$ represents the mean value over all training samples.

The Mahalanobis distance between feature vector $\mathbf{x}$ and a class mean $\mathbf{m}_i$ representing one action is:

$$d(\mathbf{m}_i, \mathbf{x}) = (\mathbf{x} - \mathbf{m}_i)^\top V \Lambda^{-1} V^\top (\mathbf{x} - \mathbf{m}_i),$$

with Λ containing the k largest eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_k$, and V the corresponding eigenvectors of Σ. Thus feature vectors are reduced to k principal components.

In all tests x are the vectorized $6 \times 6 \times 6$ lowest complex valued frequencies of equation (5), that are further reduced to the 32 largest principal components. Independent whether the classes are learned from the male/female data, we achieve in both cases a classification rate of 100%. A confusion matrix of average distances is shown in Figure 10, with surprisingly good results even with respect to axial symmetry.

8 Conclusion

In this paper, we have introduced new methods for segmenting and clustering sequences of volumetric reconstructions of a human actor performing actions, without recognition or tracking of body parts. This has allowed us to learn classifiers for a small vocabulary of primitive actions, independently of style, gender and viewpoint. We have also applied our algorithms to discover meaningful hierarchies of action concepts in more complex composite sequences. We are currently using our new semi-supervised method to build training sets with more actions, actors and styles. In future work, we plan to use those techniques to learn statistical models of composite actions by simultaneously learning the component actions and their grammars.

References

[1] A. Agarwal and B. Triggs. Learning to track 3d human motion from silhouettes. In *ICML*, 2004.

[2] A. F. Bobick and J. W. Davis. The recognition of human movement using temporal templates. *PAMI*, 23(3):257–267, 2001.

[3] M. Brand and V. Kettnaker. Discovery and segmentation of activities in video. *PAMI*, 22(8):844–851, August 2000.

[4] L. W. Campbell, D. A. Becker, A. Azarbayejani, A. F. Bobick, and A. Pentland. Invariant features for 3-d gesture recognition. In *FG*, pages 157–163, October 1996.

[5] Z. Feng and T. Cham. Video-based human action classification with ambiguous correspondences. In *V4HCI*, 2005.

[6] R. Green and L. Guan. Quantifying and recognizing human movement patterns from monocular video images. *TCSV*, 14, February 2004.

[7] Y. Ke, R. Sukthankar, and M. Hebert. Efficient visual event detection using volumetric features. In *ICCV*, October 2005.

[8] A. Kojima, T. Tamura, and K. Fukunaga. Natural language description of human activities from video images based on concept hierarchy of actions. *IJCV*, 50(2):171–184, 2002.

[9] D. Marr and L. Vaina. Representation and recognition of the movements of shapes. *Proc. R. Soc. B*, 214:501–524, 1982.

[10] J. Neumann, C. Fermüller, and Y. Aloimonos. Animated heads: From 3d motion fields to action descriptions. In *DEFORM/AVATARS*, 2000.

[11] A. Ogale, A. Karapurkar, and Y. Aloimonos. View-invariant modeling and recognition of human actions using grammars. In *WDV*, 2005.

[12] J. Rittscher, A. Blake, A. Hoogs, and G. Stein. Mathematical modelling of animate and intentional motion. *Phil. Trans. R. Soc. B*, pages 475–490, 2003.

[13] Y. Rui and P. Anandan. Segmenting visual actions based on spatio-temporal motion patterns. In *CVPR*, pages 1111–1118, 2000.

[14] T. Wang, H. Shum, Y. Xu, and N. Zheng. Unsupervised analysis of human gestures. In *PCM*, pages 174–181, 2001.

[15] D. Weinland, R. Ronfard, and E. Boyer. Motion history volumes for free viewpoint action recognition. In *PHI*, 2005.

[16] L. Zelnik-Manor and M. Irani. Event-based video analysis. In *CVPR*, December 2001.

Action Recognition from Arbitrary Views using 3D Exemplars

Daniel Weinland*
LJK - INRIA
Grenoble, France
weinland@inrialpes.fr

Edmond Boyer
LJK - INRIA
Grenoble, France
eboyer@inrialpes.fr

Remi Ronfard
Artificialife Inc.
Montreal, Canada
remir@artificialife.com

Abstract

In this paper, we address the problem of learning compact, view-independent, realistic 3D models of human actions recorded with multiple cameras, for the purpose of recognizing those same actions from a single or few cameras, without prior knowledge about the relative orientations between the cameras and the subjects. To this aim, we propose a new framework where we model actions using three dimensional occupancy grids, built from multiple viewpoints, in an exemplar-based HMM. The novelty is, that a 3D reconstruction is not required during the recognition phase, instead learned 3D exemplars are used to produce 2D image information that is compared to the observations. Parameters that describe image projections are added as latent variables in the recognition process. In addition, the temporal Markov dependency applied to view parameters allows them to evolve during recognition as with a smoothly moving camera. The effectiveness of the framework is demonstrated with experiments on real datasets and with challenging recognition scenarios.

1. Introduction

We consider the problem of recognizing actions using *a priori* unknown camera configurations. Action recognition has received considerable attention over the past decades, as a result of the growing interest for automatic and advanced scene interpretations shown in several applications domains, *e.g.* video-surveillance or human machine interactions. In this field, two main directions have been followed. *Model based approaches, e.g.* [6, 20] assume a known parametric model, typically a kinematic model, and represent actions in a joint or parameter space. Unfortunately, recovering the parameters, *e.g.* the pose, of the model appears to be a difficult intermediate task without the help of landmarks.

*D. Weinland is supported by a grant from the European Community under the EST Marie-Curie Project Visitor.

In contrast, *template based* or *holistic* approaches, *e.g.* [3, 7, 2, 19], do not use such an intermediate representation and directly model actions using image information, silhouettes or optical flow for instance. Action templates are then spatio-temporal shapes either in a three-dimensional space, when a single camera is considered, or in a four dimensional space when multiple calibrated cameras are considered. In both cases, action recognition is achieved by comparing a motion template, built from observations, with learned models of the same type. This limits recognition to situations where observed and learned models are obtained using similar camera configurations.

In this work, we propose an approach that takes advantage of the template based methods but that does not constrain camera configurations during recognition. Instead, actions can be observed with any camera configuration, from single to multiple cameras, and from any viewpoint. Our main motivation is to be able to cope with unknown recognition scenarios without learning multiple and specific databases. This has particularly clear applications in video-surveillance where actions are often observed from a single and arbitrary viewpoint.

To this purpose, we propose an exemplar-based hidden Markov model (HMM) inspired by the works of Frey and Jojic [9] and Toyama and Blake [18]. This model accounts for dependencies between three dimensional exemplars, *i.e.* representative pose instances, and image cues, this over time sequences. Inference is then used to identify the action sequence that best explains the image observations. In particular, a nice feature is that observations from any calibrated view can be incorporated. In addition, explicitly modeling the transformation between exemplars and image cues allows such transformation to change over time during recognition.

The paper proceeds as follows. In Section 2 we review the state of the art in view-independent action recognition. In Section 3 we present an overview of the proposed approach. Details on the exemplar-based HMM design are given in Section 4. In Section 5 the exemplar selection and the model learning are explained. Section 6 details recogni-

tion. Experiments using a challenging dataset of 11 actions are presented in Section 7.

2. Related Work

In order to allow actions to be learned and recognized using different camera configurations, action descriptions must exhibit some view invariance. Campbell [5] describes 3D hand and head trajectories using view invariant coordinate representations. Fundamental matrices can also be used to compare 2D action representations from different views, as joint trajectories in [16, 20] or silhouettes in [17]. To achieve similar comparisons, Parameswaran and Chellappa [14] use projective invariants of coplanar landmark points on a human body. In a previous work [19] we compare 3D action representations based on visual hulls and propose invariant Fourier-descriptors that are computed from multiple-view reconstructions. These approaches have focused on representations in which view dependent information is removed, often at the cost of an impoverished action model and without adding full flexibility in camera configurations. This motivates the search for another solution.

In a different context, Frey and Jojic [9] show how to account for view transformations in a dynamic probabilistic model. In the same spirit, Toyama and Blake [18] extend the idea for tracking with powerful image distances, and Elgammal *et al.* [8] propose a nonparametric mixture extension that, however, applies to view-dependent action recognition. Our approach builds on a similar model and incorporates geometric transformations into the probabilistic modeling of an action.

It is worth to mention also the work of Brand[4] that uses HMMs and a direct mapping between a three dimensional joint space and silhouette observations for pose estimation. It shares some similarities with our approach since we also use HMMs to model temporal sequences of exemplars.

A very recent and interesting work is that of Lv and Nevatia [12]. Developed in parallel to our method, it shares the idea of projecting a set of learned 3D exemplars/key-poses into 2D to infer actions from arbitrary view. However we use a probabilistic model instead of the deterministic linked action graph introduced in [12], allowing therefore to naturally handle uncertainties inherent to actions performed by different people and with different styles.

3. Overview

We model an action as a sequence over a set of key-poses, the exemplars. Figure 1 shows two examples of observation sequences and the corresponding best matching exemplar sequences computed with our model.

Exemplars are represented in 3D as visual hulls that have been computed using a system of 5 calibrated cameras. The model does thus not rely on motion capture data, which is generally difficult to obtain.

The observation sequence comes in this example from a single camera and is represented trough silhouettes obtained from background subtraction. To match observation and exemplars, the visual hulls are projected into 2D and a match between the resulting silhouettes is computed. The recognition phase thus generates 2D from 3D and never has to infer 3D from a single view observation.

Modeling actions and views The matching between model and observation is represented in a probabilistic framework (Section 4). Consequently, and crucially, that neither the best matching exemplar sequence, nor the exact projection parameters need to be known. Instead a probability of all potential exemplar sequence and projection is computed. Using the classical HMM algorithms [15], such a probability can be efficiently computed under the following conditions: First, we use a small set of exemplars that is shared by all models. As we show in Section 5.1, a small set of exemplars is sufficient to describe a large variety of actions, if the exemplars are discriminative with respect to these actions. Second, we make a few reasonable assumptions on the parameters of the projective transformation, *i.e.* the camera calibration and position of a person can be robustly observed during recognition and only the orientation of a person around the vertical axis is unknown.

Exemplar selection and model learning Learning an action model consists of two steps: A set of exemplars is selected and shared by all actions models (Section 5.1); probabilities over these exemplars are learned individually for each action (Section 5.2).

When selecting the exemplars, we are interested in finding the subset of poses from the training sequences, that bests discriminates actions. To this purpose, we present in Section 5.1 a novel solution based on a method for feature subset selection, a *wrapper* [11].

Given a set of exemplars, the action specific probabilities are estimated using standard probability estimation techniques for HMMs, as described in Section 5.2. Interestingly, the learning of dynamics over a set of selected 3D exemplars can be performed either on 3D sequences of aligned visually hulls (Section 5.2.1), thus under ideal conditions, or simply from single view observations (Section 5.2.2). Hence 3D information is not mandatory for that step.

Classification Classification is performed using standard HMM algorithms, as described in Section 6.

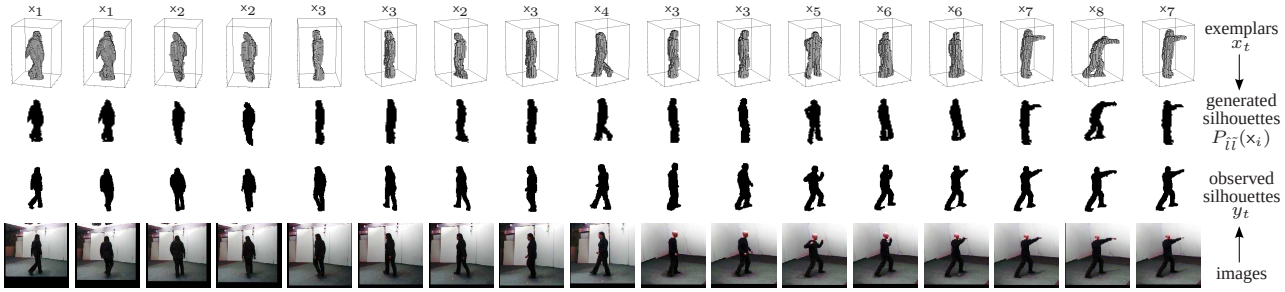


Figure 1. 2D observation sequences y_t ("walk in cycle" and "punch"), observed from different viewpoints and with unknown orientation of the persons, are explained trough 3D action models. The best matching exemplar sequence x_t and the best matching 2D projection $P_{\tilde{l}\hat{l}}(\mathsf{x}_i)$, as generated by the models, are displayed. Both models share a small set of exemplars (labeled on top).

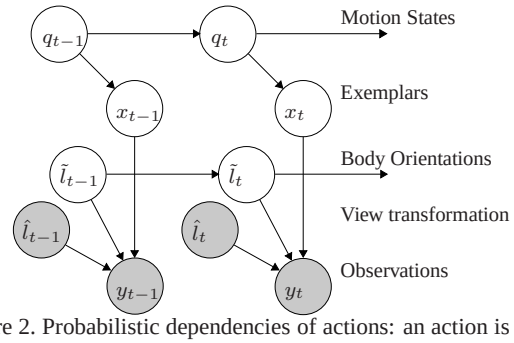


Figure 2. Probabilistic dependencies of actions: an action is modeled as a hidden state sequence Q, *e.g.* a motion sequence in a pose space. At each time step t, a 3D exemplar x_t, *i.e.* a visual hull, is drawn from the motion sequence Q. Observations y_t, *i.e.* silhouettes, result then from a geometric transformation of exemplars that is defined by 2 sets of parameters $\hat{l}$ and $\tilde{l}$. $\hat{l}$ are observed parameters, *e.g.* camera parameters determined in a preliminary step, and $\tilde{l}$ are latent parameters, *e.g.* body orientation determined during recognition. Shaded nodes in the graph correspond to observed variables.

4. Probabilistic Model of Actions and Views

Our representation for human action is a product of two independent random processes, one for the orientation of the subject relative to the camera, and the other for the view-independent, body-centered poses taken by the performer during the various stages of the action. The two processes are modeled in an exemplar based Markov model, shown in Figure 2, in the spirit of [9] and [18].

Hidden Motion States Dynamics in exemplar space are represented by a discrete N-state latent variable q that follows a first order Markov chain over time. Thus: $p(q_t|q_{t-1},\ldots,q_1) = p(q_t|q_{t-1})$, with $t \in [1\ldots T]$, and with the prior $p(q_1)$ at time $t = 1$. Though generally hidden, q can intuitively be interpreted as a quantization of the joint motion space into action-characteristic configurations.

Exemplars At each time t, a three dimensional body template x_t is drawn from $p(x_t|q_t)$. A crucial remark here is that these templates do not result from body models and joint configurations but are instead represented by a set of M exemplars: $\mathsf{X} = \{\mathsf{x}_{i\in[1\ldots M]}\}$, learned from three dimensional training sequences.

Note here that $p(x_t = \mathsf{x}_i|q_t)$ models the non-deterministic dependencies between motion states and body configuration. Thus motion states q are not deterministically linked to exemplars as in [12, 18], allowing therefore a single motion state q to be represented with different exemplars, to account for different body proportions, style, or clothes.

View Transformation and Observation To ensure independence with respect to the view projection onto the image plane: $P_{\tilde{l}\hat{l}}(\mathsf{x}) = \hat{P}[R_\theta, u]\mathsf{x}$, we condition observations y on parameters that represent this transformation. We differentiate view transformation parameters $\{\hat{l}_t\}$ that can be robustly observed (*i.e.* the camera matrix $\hat{P}$ and position u), and body pose parameters $\{\tilde{l}_t\}$ that are latent (*i.e.* the orientation around the vertical axis θ).

The resulting density $p(y_t|x_t, \hat{l}_t, \tilde{l}_t)$ is represented in form of a kernel function centered on the transformed exemplars $P_{\tilde{l}\hat{l}}(\mathsf{x}_i)$:

$$p(y_t|x_t = \mathsf{x}_i, \hat{l}_t, \tilde{l}_t) \propto \frac{1}{Z} \exp\big(-d(y_t, P_{\tilde{l}\hat{l}}(\mathsf{x}_i))/\sigma^2\big), \quad (1)$$

where d is a distance function between between the resulting silhouettes, *e.g.* the Euclidean distance (i.e. the number of pixels which are different), or a more specialized distance such as the chamfer distance [10]. (Note that both were giving similar results in our experiments.)

The temporal evolution of the latent transformation variables is modeled as a Markov process with transitions probabilities $p(\tilde{l}_t|\tilde{l}_{t-1})$, and a prior $p(\tilde{l}_1)$. This is equivalent to a temporal filtering of the transformation parameters where, interestingly, various assumptions could be made on the dynamic of these parameters: a static model or an autoregres-

sive model, or even a model taking into account dependencies between an action and view changes.

In our implementation all variables $\{\tilde{l}, \hat{l}\}$ are discretized. For instance, the orientation θ is discretized into L equally spaced angles within $[0, 2\pi]$ and u is discretized into a set of discrete positions. The temporal evolution of θ is modeled using a von Mises distribution: $p(\theta_t|\theta_{t-1}) \propto \exp(\kappa \cos(\theta_t - \theta_{t-1}))$, that can be seen as the circular equivalent of a normal distribution, and a uniform prior $p(\theta_1)$.

5. Learning

We learn separate action models λ_c for each action class $c \in \{1, \ldots, C\}$. A sequence of observations $Y = \{y_1, \ldots, y_T\}$ is then classified with respect to the maximum a posteriori (MAP) estimate:

$$g(Y) = \arg\max_c p(Y|\lambda_c)p(\lambda_c). \quad (2)$$

The set λ_c is composed of the probability transition matrices $p(q_t|q_{t-1}, c)$, $p(q_1|c)$ and $p(x_t|q_t, c)$, which are specific to the action c, as they represent the action's dynamics. In contrast, the observation probabilities $p(y_t|x_t, \hat{l}_t, \tilde{l}_t)$ are tied between classes, meaning that all actions $\{c = 1..C\}$ share a common exemplar set, *i.e.* $\mathsf{X}_c = \mathsf{X}$, and a unique variance $\sigma_c^2 = \sigma^2$. In the context of HMMs, such an architecture is known as a *tied-mixture* or *semi-continuous* HMM[1]. This architecture is particularly well adapted to action recognition since different actions naturally share similar poses. For example, many actions share a neutral rest position and some actions only differ by the sequential order of poses that composed them. In addition, sharing parameters dramatically reduces complexity during recognition, when every exemplar must be projected with respect to numerous latent orientations.

Learning consists then in two main operations: selecting the exemplar set that is shared by all models; learning the action specific probabilities. As we will see in the following, the two operations are tightly coupled. Selection uses learning to evaluate the discriminant quality of an candidate exemplar set, and learning probabilities relies on a selected set of exemplars. Both operations are detailed below.

5.1. Exemplar Selection

Identifying discriminative exemplars is an essential step of the learning process. Previous works use motion energy minima and maxima [12, 13], or k-means clustering (adapted to return exemplars) [18] to this end. However, there is no apparent relationship between such criteria and the action discriminant quality of the selected exemplars. In particular for the adapted k-means clustering [18] we observed experimentally, that clusters tend to consist of different poses performed by similar actors rather than similar poses performed by different actors. Consequently, selecting exemplars as poses with minimum within-cluster distance often leads to neutral and therefore non-discriminative poses.

In light of this, we propose a novel approach for exemplar selection, to better link the discriminant quality of exemplars and the selection. We therefore use a wrapper [11], a technique for discriminant feature subset selection. The idea behind a wrapper is to use the trained classifier (2) itself to evaluate how discriminative a candidate set of exemplars is. Thus a wrapper performs a greedy search over the full set of exemplars, where in each iteration classifiers are learned and evaluated for each possible subset considered.

The wrapper method we use is called "forward selection" [11], and proceeds as follows: Let $\mathcal{Y}$ denote a set of 3D visual hulls. Assume training sequences and test sequences for all actions $c \in \{1, \ldots, C\}$ are given.

1. Set $\mathsf{X} = \emptyset$.

2. Find $y^* \in \{\mathcal{Y} \setminus \mathsf{X}\}$, where a classifier g (trained on all actions) using exemplar set $\{\mathsf{X} \cup y^*\}$ has best recognition performance on the test-set. Add y^* to X.

3. Repeat step 2 until M visual hulls from $\mathcal{Y}$ have been added to X.

Note that the above procedure can only work when the exemplar set is shared by all action models. The selection thus starts by training a classifier for each singleton exemplar. The exemplar for which the classifier has best evaluation performance is selected, and the procedure is repeated for couples of exemplars, triples, *etc.*, until M exemplars have been selected. Note that training and evaluation of the classifier can be performed in 3D or 2D, as detailed in Section 5.2. In case that the training sequences are 3D, $\mathcal{Y}$ can simply be the training-set.

The approach is illustrated in Figures 3 and 4 where exemplars and the associated classification rates are shown. Figure 3 shows that the selected poses naturally represent *key-frames* or characteristic frames of an action.

5.2. Learning Dynamics

Given a set of exemplars, the action parameters $\lambda_{c\in\{1,\ldots,C\}}$: probabilities $p(q_t|q_{t-1}, c)$, $p(q_1|c)$ and $p(x_t|q_t, c)$, can be learned. Various strategies can be considered for that purpose. In the following, we sketch 2 of them: learning from 3D observations (sequences of visual hulls), and learning from 2D observations (image sequences). Note that in both cases, motion is learned in 3D over the set of 3D exemplars, obtained as described in section 5.1.

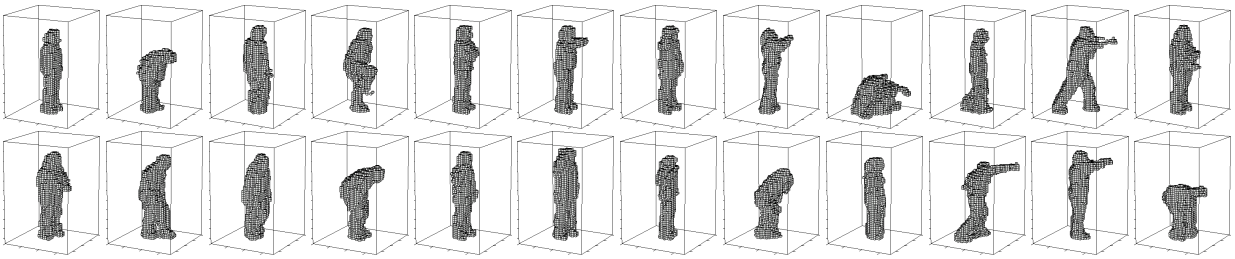


Figure 3. Selected exemplars: first 24 discriminative exemplars as returned by the forward selection. The dataset is composed of 11 actions performed by 10 actors. Recognition rates are shown in Figure 4.

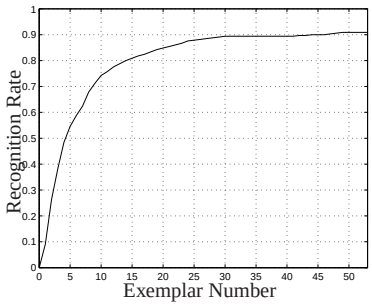


Figure 4. Recognition rate vs. number of selected exemplars.

5.2.1 Learning from 3D Observations

In this training scenario, several calibrated viewpoints are available, leading therefore to 3D visual hull sequences, and all actions are performed with the same orientation. In that case, motion dynamics are learned independently from any viewing transformation, thus $p(y_t|x_t, \hat{l}_t, \tilde{l}_t) = p(y_t|x_t)$ with y being 3D. Transformation parameters appear later during the recognition phase where both dynamics and viewing process are joined into a single model.

Each model λ_c is learned through a forward-backward algorithm that is similar to the standard algorithm for Gaussian mixture HMMs [15], except that the kernel parameters, that correspond to mean and variance of the Gaussians (*i.e.* X and σ), are not updated. Note that a similar forward-backward algorithm was already proposed in the context of exemplar based HMMs [8].

5.2.2 Learning from 2D Observations

In this scenario, dynamics in the exemplar 3D space are learned using 2D cues only. In that case, the situation is similar when either learning or recognizing. A nice feature here is that only a valid set of 3D exemplars is required, but no additional 3D reconstruction. This is particularly useful when large amounts of 2D observations are available but no 3D inference capabilities (*e.g.* 3D exemplars can be synthesized using a modeling software; the dynamics over these exemplars are learned form real observations).

View observations are not aligned and so the orientation variable $\tilde{l}$ is latent. Nevertheless, the number of latent states remains in practice small, (*i.e.* $L \times N$, with L being the number of discrete orientations $\tilde{l}$ and N the number of states q). The model can be learned by introducing a new variable $\acute{q} = (q, \tilde{l})$ of size $L \times N$ that encodes both state and orientation. Probabilities of this *extended* states are then simply defined as Cartesian products of the transition probabilities for q and $\tilde{l}$. Loops in the model are thus eliminated, and learning can be performed via the forward-backward algorithm introduced in 5.2.1.

6. Action Recognition from 2D Cues

A sequence of observations Y is classified using the MAP estimate (2). Such a probability can now be computed using the classical forward variable $\alpha(\acute{q}_t|\lambda_c) = p(y_1, \ldots, y_t, \acute{q}_t|\lambda_c)$ as explained in [15], where $\acute{q} = (q, \tilde{l})$ is a variable encoding state and orientation as explained in Section 5.2.2

Arbitrary viewpoints do not share similar parameters; in particular scales and metrics can be different. However, the kernel parameter σ^2 is uniquely defined, with the consequence that distances computed in equation (1) can be inconsistent when changing the viewpoint. To adjust σ^2 with respect to changes in these parameters, we introduce $\sigma^2_{\tilde{l}} = s_{\tilde{l}}\sigma^2$. Ideally, $\sigma^2_{\tilde{l}}$ should be estimated using test data. In practice, the following simple approximation of $\sigma^2_{\tilde{l}}$ appears to give satisfactory results with the distance functions we are considering:

$$s_{\tilde{l}} = \frac{1}{M}\sum_{i=1}^{M} \frac{\frac{1}{L}\sum_{\tilde{l}=1}^{L} ||P_{\tilde{l}\tilde{l}}(\mathsf{x}_i)||^2}{||\mathsf{x}_i||^2}. \qquad (3)$$

Another remark is that observations from multiple calibrated cameras can easily be incorporated. Assuming multiple view observations $\{y_t^1, \ldots, y_t^K\}$ at time t, we can write their joint conditional probability as:

$$p(y_t^1, \ldots, y_t^K|x_t, \hat{l}_t, \tilde{l}_t) \propto \prod_{y_t^k}^{K} p(y_t^k|x_t, \hat{l}_t, \tilde{l}_t). \qquad (4)$$

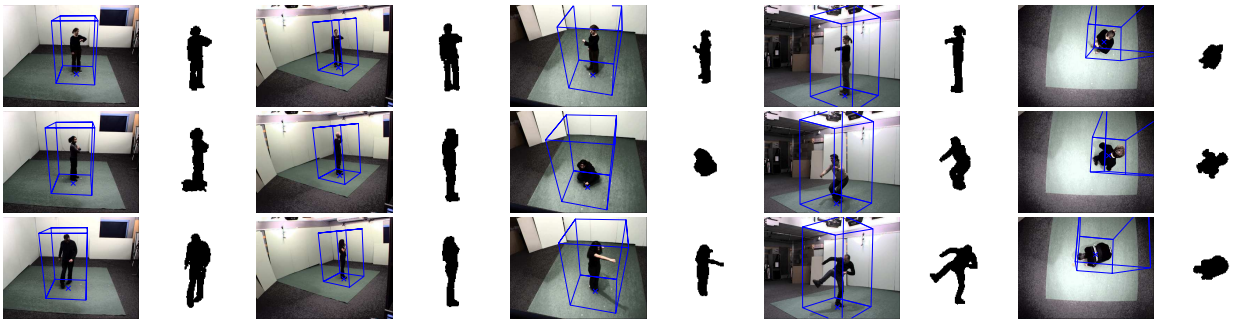


Figure 5. Camera setup and extracted silhouettes: (Top) the action "watch clock" from the 5 different camera views. (Middle and bottom) sample actions: "cross arms", "scratch head", "sit down", "get up", "turn", "walk", "wave", "punch", "kick", and "pick up". Volumetric exemplars are mapped onto the estimated interest regions indicated by blue box.

7. Experiments

Experiments were conducted on our publicly available dataset[1], the IXMAS dataset. We choose 11 actions, performed by 10 actors, each 3 times, and viewed by 5 calibrated cameras (see Figure 5). In this dataset, actor orientations are arbitrary since no specific instruction was given during the acquisition. The 3D sequences are segmented into elementary segments using our approach proposed in [19].

Note, that the same dataset was used in [12] in a similar context. However, results are reported only for a single sequence (out of three) per actor. This sequence has been selected to give best results, thus making a direct comparison difficult.

Our experimental scheme is as follows: 9 of the actors are used for exemplar selection and model learning, the remaining actor is then used for testing. We repeat this procedure by permuting the test-actor and compute the average recognition rate. Examplar selection is performed on subsampled sequences (*i.e.* 2.5 frames/s) to save computational costs. Example results for exemplars are shown in Figure 3. The number M of examplars was empirically set to 52 . Parameter learning and testing is performed using all frames in the database. Action are modeled with 2 states, which appears to be adequate since most segmented actions cover short time periods. Voxel grids are of size: $64 \times 64 \times 64$ and image ROIs: 64×64. The rotation around the vertical axis is discretized into 64 equally spaced values. Consequently, each frame is matched to 52×64 exemplar projections. The ground plane is clustered into 4 positions.

7.1. Learning in 3D

In these experiments, learning is performed in 3D (as explained in 5.2.1). Recognition is then performed on 2D views with arbitrary actor orientations. Recognition rates per camera are given in Figure 6(a), the corresponding views are shown in Figure 5.

cameras	2 4	3 5	1 3 5	1 2 3 5	1 2 3 4
%	81.3	61.6	70.2	75.9	81.3

Table 1. Recognition rates with camera combinations. For comparisons, a full 3D recognition considering 3D manually aligned models as observations, instead of 2D silhouettes, yields 91.11%.

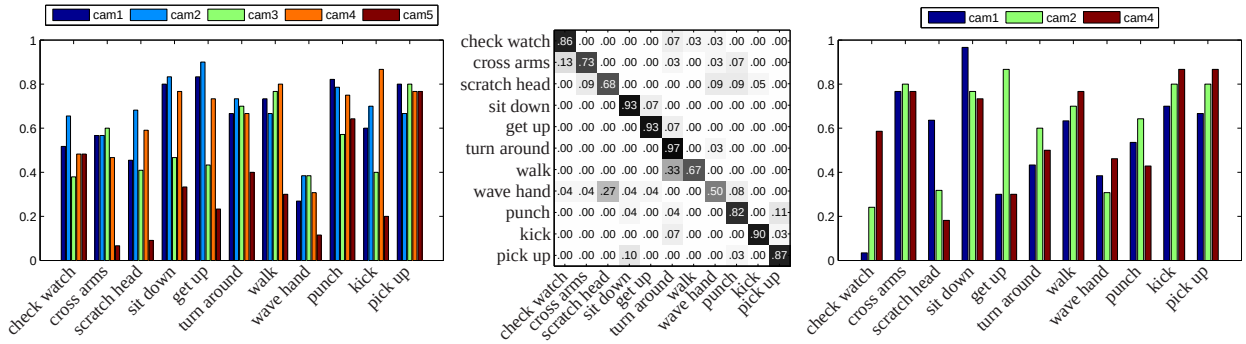
Unsurprisingly, the best recognition rates are obtained with fronto-parallel views (cameras 2 and 4). The top camera (camera 5) scores worst. For this camera, we observe that: the silhouette information is not discriminative; the perspective distortion results in strong bias in distances; estimating the position of the actor is difficult. All these having a strong impact on the recognition performance.

In the next experiment, several views were used in conjunction to test camera combinations. First, 2 view combinations were experimented. Camera 2 and 4 give the best recognition rate at 81.27%. Those 2 cameras are both approximately fronto-parallel and perpendicular one another. Figure 6(b) shows the resulting confusion matrix for this specific setup. Adding further cameras did not improve results. We also try other camera combinations (Table 1). For instance, combining the two cameras with the worst recognition results (camera 3 and 5) raises the recognition rate to 61.59%.

7.2. Learning from single views

In this experiment, learning is performed using single cameras (as explained in Section 5.2.2). Observations during learning and recognition are thus not aligned. The exemplars considered are the same than in the previous section. Learning from a single view is obviously prone to ambiguities, especially when the number of training samples is limited. We thus restricted the experiments to the 3 best cameras with respect to the previous experiments. Figure 6(c) shows the recognition results per action class

[1]The data-set is available on the Perception website http://perception.inrialpes.fr in the "Data" section.



(a) (b) (c)

Figure 6. (a) Recognition rates when learning in 3D and recognizing in 2D. The average rates per camera are $\{65.4, 70.0, 54.3, 66.0, 33.6\}$. (b) Confusion matrix for recognition using cameras 2 and 4. Note that actions performed with the hand are confused, *e.g.* "wave" and "scratch head" as well as "walk" and "turn". (c) Recognition rates when learning and recognizing in 2D.

and per camera. Compared to the previous scenario, recognition rates drop drastically, as a consequence of learning from non-aligned data and single view observations. Surprisingly, some of the actions, *e.g.* "cross arms", "kick" still get very acceptable recognition rates, as well as "sit down" and "pick up" that would normally be confused. The average rate for camera 1 is 55.24%, 63.49% for camera 2 and 60.00% for camera 4.

8. Conclusion

This paper presented a new framework for view independent action recognition. The main contribution is a probabilistic 3D exemplar model that can generate arbitrary 2D view observations. It results in a versatile recognition method that adapts to various camera configurations. The approach was evaluated on a dataset of 11 actions and with different challenging scenarios. The best results where obtained with a pair of fronto-parallel perpendicular cameras, validating the fact that actions can be recognized from view arbitrary viewpoints.

References

[1] J. R. Bellegarda and D. Nahamoo. Tied mixture continuous parameter modeling for speech recognition. *ASSP*, 38:2033–2045, 1990. 4

[2] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. In *ICCV*, pages 1395–1402, 2005. 1

[3] A. Bobick and J. Davis. Real-time recognition of activity using temporal templates. In *WACV*, pages 39–42, 1996. 1

[4] M. Brand. Shadow puppetry. In *ICCV*, pages 1237–1244, 1999. 2

[5] L. W. Campbell, D. A. Becker, A. Azarbayejani, A. F. Bobick, and A. Pentland. Invariant features for 3-d gesture recognition. In *FG*, pages 157–163, 1996. 2

[6] L. W. Campbell and A. F. Bobick. Recognition of human body motion using phase space constraints. In *ICCV*, pages 624–630, 1995. 1

[7] A. A. Efros, A. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *ICCV*, pages 726–733, 2003. 1

[8] A. M. Elgammal, V. D. Shet, Y. Yacoob, and L. S. Davis. Learning dynamics for exemplar-based gesture recognition . In *CVPR*, pages 571–578, 2003. 2, 5

[9] B. J. Frey and N. Jojic. Learning graphical models of images, videos and their spatial transformations. In *UAI*, pages 184–191, 2000. 1, 2, 3

[10] D. Gavrila and V. Philomin. Real-time object detection for smart vehicles. In *ICCV*, pages 87–93, 1999. 3

[11] G. H. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. In *ICML*, pages 121–129, 1994. 2, 4

[12] F. Lv and R. Nevatia. Single view human action recognition using key pose matching and viterbi path searching. In *CVPR*, 2007. 2, 3, 4, 6

[13] A. Ogale, A. Karapurkar, G. Guerra-Filho, and Y. Aloimonos. View-invariant identification of pose sequences for action recognition. In *VACE*, 2004. 4

[14] V. Parameswaran and R. Chellappa. View invariance for human action recognition. *IJCV*, 66(1):83–101, 2006. 2

[15] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. pages 267–296, 1990. 2, 5

[16] C. Rao, A. Yilmaz, and M. Shah. View-invariant representation and recognition of actions. *IJCV*, 50(2):203–226, 2002. 2

[17] T. Syeda-Mahmood, M. Vasilescu, and S. Sethi. Recognizing action events from multiple viewpoints. In *EventVideo01*, pages 64–72, 2001. 2

[18] K. Toyama and A. Blake. Probabilistic tracking in a metric space. In *ICCV*, pages 50–59, 2001. 1, 2, 3, 4

[19] D. Weinland, R. Ronfard, and E. Boyer. Free viewpoint action recognition using motion history volumes. *CVIU*, 104(2-3):249–257, 2006. 1, 2, 6

[20] A. Yilmaz and M. Shah. Recognizing human actions in videos acquired by uncalibrated moving cameras. In *ICCV*, pages 150–157, 2005. 1, 2

CHAPITRE 4

Autres travaux

Relaxation d'images de classification et modèles de la physique statistique de Marc Sigelle et Rémi Ronfard, Traitement du Signal, Volume 9, Number 6, 1992.

Region-Based Strategies for Active Contour Models de Rémi Ronfard, International Journal of Computer Vision, Volume 13, Number 2, pages 229-251, October 1994.

Triangulating Multiply-Connected Polygons - A simple yet efficient algorithm de Rémi Ronfard et Jarek Rossignac, Computer Graphics Forum, Volume 13, Issue 3, pages 281 - 292, August 1994.

Full-Range Approximation of Triangulated Polyhedra de Rémi Ronfard et Jarek Rossignac, Computer Graphics Forum, Volume 15 Issue 3 , pages 67-76, August 1996.

Implicit Simplicial Models for adaptive curve reconstruction de Gabriel Taubin et Rémi Ronfard, IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 18, Number 3, page 321-325, 1996.

Detail-Preserving Variational Surface Design de Ioana Boier-Martin, Rémi Ronfard et Fausto Bernardini, Journal of Computing and Information Science in Engineering, Volume 5, Number 2, page 104–110, June 2005.

Modèles de Potts et relaxation d'images de labels par champs de Markov

Potts Models and Image Labelling Relaxation by Random Markov Fields

Marc SIGELLE
E.N.S.T.
46, rue Barrault,
75634 Paris Cedex 13

Marc Sigelle est né à Paris le 18 mars 1954. Diplômé de l'École Polytechnique en 1975 et de l'École Nationale Supérieure des Télécommunications en 1977, il a travaillé au Centre National d'Études des Télécommunications où il a mené des travaux de recherche en Physique et en Informatique. Il est à Télécom Paris depuis 1989, où il mène des travaux sur les Champs de Markov en Traitement d'Images.

Rémi RONFARD

DASSAULT
Systèmes Modeleur géométrique
38 avenue Charles de Gaulle,
92150 Suresnes

Rémi Ronfard est né à Cambrai le 25 juin 1963. Diplômé de l'École des Mines de Paris en 1986, il a travaillé en tant qu'attaché au Centre de Télédétection de l'École des Mines à Sophia Antipolis. Il a soutenu une thèse de Doctorat en février 1991 sur l'extraction de contours dans les images multi-spectrales et en couleur. Il travaille actuellement au département modeleur Géométrique de Dassault Systèmes.

RÉSUMÉ

Nous montrons dans cet article la relation profonde entre certains modèles d'énergie provenant de la Physique Statistique utilisés et les modèles utilisés en champ de Markov pour l'étiquetage d'images. Nous présentons comme application une méthode markovienne de relaxation et d'amélioration d'images préclassifiées. On définit pour cela une fonction énergie ne dépendant que des labels et de leur valeur initiale, la connaissance *a priori* sur l'image provenant de la matrice de confusion déduite des échantillons de référence utilisés pour la classification initiale. La fonction à minimiser inclut divers termes assurant la régularité spatiale des labels, la croissance ou la disparition de certaines classes. Cette méthode permet en particulier de reclasser contextuellement les pixels d'une classe de rejet. Enfin, nous présentons des résultats obtenus sur des images multispectrales en télédétection et en géologie, où nous comparons les résultats des modes conditionnels itérés et du recuit simulé. La méthode n'opérant que sur un processus label s'avère être très performante.

MOTS CLÉS

Images, classification, relaxation, champs de Markov, physique statistique.

ABSTRACT

We show in this paper the deep relationship between classic models from Statistical Physics and Markovian Random Fields models used in image labelling. We present as an application a markovian relaxation method for enhancement and relaxation of previously classified images. An energy function is defined, which depends only on the labels and on their initial value. The main a priori *pixel knowledge results from the confusion matrix of the reference samples used for initial classification. The energy to be minimized includes also terms ensuring simultaneous spatial label regularity, growth of some classes and disparition of some others. The method allows for example to reclassify previous rejection class pixels in their spatial environment. Last we present some results on Remote Sensing multispectral and geological ore images, comparing the performances of Iterated Conditional Modes (ICM) and Simulated Annealing (SA). Very low CPU time was obtained due to the principle of the method, working on labels instead of gray levels.*

KEY WORDS

Images, classification, relaxation Markov Random fields, statistical physics.

1. Introduction

Nous présentons dans cet article une méthode à base de Champs de Markov pour le traitement général d'une classe de problèmes de relaxation de labels dans les images [1], [2], [3], [4]. Nous décrivons dans ce but un cadre théorique issu de la Physique Statistique permettant de généraliser le modèle d'Ising (restauration d'images binaires) à la restauration d'images multi-classes connu sous le nom de modèle de Potts en Physique [5]. L'approche usuelle utilisée en segmentation markovienne fait intervenir la référence aux données originales [6], [7], que ce soit une image mono- ou multispectrale. La méthode exposée ici s'applique dans le cas où l'on dispose de peu de connaissances *a priori* spécifiques, et où les seuls éléments d'information proviennent d'une classification initiale des données, qui sera prise comme référence. Dans cette approche, l'algorithme d'étiquetage initial reste inchangé, et la méthode de relaxation par Champs de Markov intervient comme seconde étape permettant de « retoucher » la classification initiale en fonction des vesoins requis. Le déroulement de chaque étape du processus global peut ainsi être contrôlé et modifié si nécessaire. De plus, l'algorithme de relaxation n'opérant que sur un processus label entraîne un faible temps calcul. Nous espérons que le cadre ainsi décrit fournisse une approche cohérente à ce genre de problèmes.

Le paragraphe 2 décrit le problème de classification contextuelle étudié et sa décomposition en deux étapes. Les paragraphes 3 et 4 présentent le cadre théorique employé ainsi que ses relations avec la théorie des réseaux de spins en Physique Statistique. Dans le paragraphe 5 nous discutons les paramètres du modèle et donnons un aperçu de leur estimation à partir de données issues de la classification initiale et d'autres connaissances spécifiques *a priori* lorsqu'elles sont disponibles. Nous présentons enfin dans le paragraphe 6 un ensemble de résultats obtenus sur des images de télédétection en classification multispectrale et sur des images géologiques de fond de mines, puis décrivons les performances de la méthode.

2. Relaxation stochastique de données classifiées de façon optimale

Notons X le processus *pixel* (par exemple une image Landsat multispectrale) et L le processus *label* (par exemple une classification thématique du terrain). Nous proposons ici le schéma suivant de classification en deux étapes :

$$X \quad \rightarrow \quad L_0 \quad \rightarrow \quad L$$

première classification (A) relaxation markovienne (B)

$$\text{Max } P(X/L_0)\,P(L_0) \quad \text{Max } P(L_0/L)\,P(L)\,.$$

Lorsque la première classification (A) est bayésienne (ce qui sera le cas envisagé dans la suite), l'étiquetage initial L_0 est obtenu en maximisant $P(X/L_0)\,P(L_0)$, les probabilités conditionnelles $P(X/L_0)$ étant connues à partir des histogrammes multispectraux d'échantillons de référence convenablement choisis. Le processus de relaxation stochastique (B) permet de reclasser les pixels situés aux frontières entre régions ainsi que les pixels non classés dans l'étape (A). Les pixels frontières sont reclassés de façon contextuelle grâce au Champ de Markov $P(L)$. En ce qui concerne le traitement des pixels de la classe de rejet, deux possibilités se présentent :

— l'échantillonnage initial n'est pas complet en nombre de classes : ces pixels constituent donc une nouvelle classe, dont la régularité spatiale sera assurée au même titre que les autres régions par le processus (B) ;

— l'échantillonnage initial est complet : ces pixels doivent alors être reclassés dans les autres régions, et ceci de façon contextuelle.

Il est clair que dans ce schéma, aucune référence aux probabilités initiales de classification n'est effectuée pendant la relaxation, de sorte que l'attache aux données doit être précisée de façon fiable à partir de l'étiquetage initial. Nous suggérons d'utiliser pour cela les résultats de la première classification sur les échantillons de référence eux-mêmes, par une utilisation non conventionnelle de la matrice de confusion qui en résulte (voir § 5).

3. Modèle d'Ising et régularisation d'images binaires

Le modèle d'Ising a été développé en 1925 dans le cadre de la théorie du ferromagnétisme pour rendre compte des propriétés de transition de phase ordre-désordre dans les solides. Dans le modèle du réseau plan isotrope et 4-connexe, un spin en un site s du réseau interagit avec ses 4-voisins (notés r) par une constante de couplage J et avec un champ magnétique B de sorte que l'énergie du site s conditionnellement à ses voisins s'écrit

$$(1) \qquad W_s = -\,J \sum_r \sigma_r\,\sigma_s - B\sigma_s \quad (\sigma_s = \pm 1)\,.$$

L'énergie totale E du réseau est la somme des termes d'interaction site-site (cliques d'ordre 2) et des termes d'interaction site-champ magnétique (cliques d'ordre 1). Une forme équivalente de la fonction énergie est la forme de Heisenberg

$$(2) \qquad E = \frac{J}{2} \sum_{(r,\,s)} \|\sigma_r - \sigma_s\|^2 + \frac{B}{2} \sum_r \|\sigma_r - b\|^2$$

ou encore

$$(3) \qquad E = \frac{J}{2} \sum_{(r,\,s)} \|\hat{u}_r - \hat{u}_s\|^2 + \frac{B}{2} \sum_r \|\hat{u}_r - \hat{b}\|^2$$

la première somme portant sur les cliques d'ordre 2, la deuxième sur les cliques d'ordre 1, où les $\hat{u}_r$ sont les vecteurs unitaires associés aux directions des spins en chaque site et $\hat{b}$ le vecteur unitaire associé au sens du champ magnétique.

Rappelons maintenant la démonstration de Carnevali, Coletti et Patarnello [8] établissant le lien entre régularisation d'une image binaire et modèle d'Ising. Elle est la source des extensions que nous avons effectuées pour les images multi-classes. Considérons une image à régulariser $f^0(x, y)$ comme continue dérivable dans un domaine de $\mathbf{R}^2$, on cherche $f(x, y)$ solution du problème

$$\min E = E_1 + E_2$$

où

$$(4)\quad E_1 = \lambda \, \|f - f^0\|^2 = \lambda \iint_{\text{Image}} (f(x, y) - f^0(x, y))^2 dx\, dy$$

est le terme d'écart à l'image originale

$$(5)\quad E_2 = \mu \, \|\nabla f\|^2 = \mu \iint_{\text{Image}} \left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2 dx\, dy$$

est le terme de régularisation d'image adopté (norme L^2 du gradient de l'intensité). E est l'énergie totale de l'image. En discrétisant sur l'image, il vient

$$(6)\quad E_1 = \lambda \sum_{i,j} \|f_{i,j} - f^0_{i,j}\|^2$$

$$(7)\quad E_2 = \mu \left(\sum_{i,j} \|f_{i,j} - f_{i+1,j}\|^2 + \|f_{i,j} - f_{i,j+1}\|^2 \right).$$

(Les effets de bord peuvent être pris en compte en raccordant l'image de façon torique ou en l'encadrant d'un bord fixe.)

Lorsque l'image est binaire : $f_{i,j} = 0$ ou 1 $\forall i, j$ on associe à chaque pixel sa « variable de spin » $\sigma_{i,j}$

$$(8)\quad f_{i,j} = \frac{1 + \sigma_{i,j}}{2} \quad (\sigma_{i,j} = \pm 1).$$

Les équations (4-5) deviennent

$$(9)\quad E = E_1 + E_2$$

$$(10)\quad E_1 = \frac{\lambda}{4} \sum_{i,j} \|\sigma_{i,j} - \sigma^0_{i,j}\|^2$$

$$(11)\quad E_2 = \frac{\mu}{4} \left(\sum_{i,j} \|\sigma_{i,j} - \sigma_{i+1,j}\|^2 + \|\sigma_{i,j} - \sigma_{i,j+1}\|^2 \right).$$

Comparant à (3), on trouve que l'énergie totale E est celle d'un modèle d'Ising 2D isotrope de constante de couplage entre sites $J = \frac{\mu}{2}$ en présence d'un champ magnétique de module constant $B = \frac{\lambda}{2}$ mais inhomogène et dont le sens en chaque pixel dépend de sa valeur originale binaire (c'est-à-dire un champ magnétique presque « homogène par morceaux » si l'image de départ n'est pas trop bruitée).

N.B. : Une série de tests menés sur des images binaires donne un choix des paramètres $J = 2$, $B = 3$, température initiale $T_0 = 2$, en accord avec les valeurs trouvées par Carnevali *et al.* [8]. Nous verrons dans la suite comment généraliser ce choix aux images multi-classes.

4. Restauration d'images multi-classes et modèle de Potts

Le potentiel de cliques d'ordre 2 le plus couramment utilisé dans les méthodes de régularisation d'images de classes est [4], [6], [7], [9], [11], [12]

$$(12)\quad V(r, s) = \begin{cases} -\beta & \text{si} \quad \ell_r = \ell_s \\ +\beta & \text{si} \quad \ell_r \neq \ell_s \end{cases}$$

(on supposera la 4(connexité dans la suite de l'exposé).

Il s'agit là à une constante près du modèle dit (standard) de Potts en Physique Statistique

$$(13)\quad V(r, s) = -K\delta(\ell_r, \ell_s) \; (K = 2\beta)$$

δ est le symbole de Kronecker défini par

$$\delta(x_1, x_2) = 1 \quad \text{si} \quad x_1 = x_2, \quad 0 \quad \text{sinon}$$

Ce modèle a été abondamment étudié pour ses propriétés de Transitions de Phase [5], qui sont maintenant bien connues et qui dépendent fortement du nombre de classes q. Le modèle de Potts coïncide avec le modèle d'Ising lorsque $q = 2$, en vertu de la formule

$$\delta(\ell_r, \ell_s) = \frac{1 + \sigma_r \sigma_s}{2}$$

(d'où il résulte $K_{\text{Potts}} = 2 J_{\text{Ising}}$), les σ_i étant les variables de spin ± 1 associées aux valeurs (binaires) des pixels. Il en est donc une extension naturelle lorsque q est quelconque. Nous allons maintenant montrer comment étendre les résultats du paragraphe à partir de ce formalisme. Partons de la formule suivante [5]

$$(14)\quad \delta(m, n) = \frac{1}{q} (1 + (q - 1)\, \hat{u}_m \cdot \hat{u}_m)$$

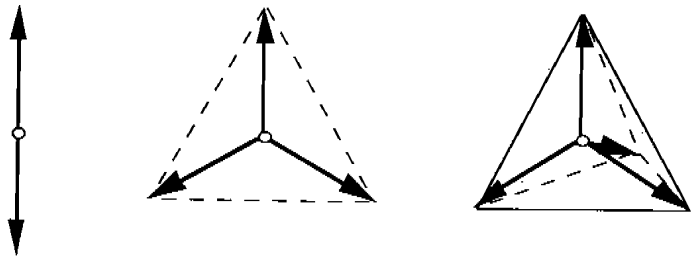
— où m et n sont deux entiers appartenant à l'intervalle $[0 \ldots q - 1]$,

— où les $\hat{u}_i$ sont q vecteurs unitaires « régulièrement répartis » au sommet d'un hypertétraèdre de R^{q-1} (voir *fig.* 1), c'est-à-dire tels que

$$(15)\quad \sum_{k=0}^{q-1} \hat{u}_k = \vec{0} \quad \text{et} \quad \hat{u}_m \cdot \hat{u}_n = -\frac{1}{q - 1} \quad \text{si} \quad m \neq n.$$

Le cas $q = 2$ est bien celui du modèle d'Ising : on obtient deux valeurs de spin de direction opposée. Les vecteurs $\hat{u}_i$ étant unitaires, la fonctionnelle énergie obtenue en (9-10-11) s'étend de façon naturelle sous la forme suivante

$$(16)\quad E = E_1 + E_2$$



q = 2 q = 3 q = 4

Fig. 1. — Le modèle de Potts pour plusieurs valeurs du nombre de classes (d'après [5]).

$$(17)\quad E_1 = \lambda \sum_{i,j} \left\| \hat{u}_{i,j} - \hat{u}^0_{i,j} \right\|^2$$

$$(18)\quad E_2 = \mu \left(\sum_{i,j} \left\| \hat{u}_{i,j} - \hat{u}_{i+1,j} \right\|^2 + \left\| \hat{u}_{i,j} - \hat{u}_{i,j+1} \right\|^2 \right).$$

Le contenu scalaire de l'image $f(x, y)$ a été remplacé par une expression vectorielle dans $\mathbf{R}^{q-1}$, et la forme d'énergie obtenue est une généralisation à $q-1$ dimensions du modèle de Heisenberg. L'énergie conditionnelle en un site s prend la forme

$$(19)\quad W_s = -K \sum_r \delta(\ell_r, \ell_s) - B\delta(\ell_s, \ell^0_s)$$

où

$$(20)\quad \lambda = \frac{q-1}{q}\frac{B}{2} \quad \text{et} \quad \mu = \frac{q-1}{q}\frac{K}{2}$$

les notations étant celles des formules (4) et (5).

La forme (19), qui est probablement le modèle le plus simple associé à la restauration d'images de classes, est indépendante de toute « relation d'ordre » entre labels distincts ou de toutes connaissances *a priori* sur les régions (proximité, etc...). Nous développons au paragraphe suivant une extension de cette formulation afin de permettre leur prise en compte.

NB. : De la même façon que $K_{Potts} = 2\,J_{Ising}$, on a $B_{Potts} = 2\,B_{Ising}$.

Cela nous a incité à adopter dans nos applications K = 4, B = 6, température initiale $T_0 = 4$.

Interprétation physique du potentiel de Derin

Besag, Derin, Kelly et Karssemeijer [4], [9], [10], [11], [12] incluent dans leur modèle d'énergie le potentiel suivant (d'ordre 1)

$$(21)\quad V_c(n) = \alpha_n \quad \text{pour} \quad n \in [0 \ldots q-1]$$

qui permet de favoriser ou défavoriser l'apparition de certaines classes, ce que nous allons retrouver à partir du formalisme précédent. En effet, on peut écrire

$$(22)\quad V_c(n) = \sum_{k=0}^{q-1} \alpha_k\, \delta(k, n)$$

c'est-à-dire, en employant les expressions vectorielles développées plus haut

$$(23)\quad V_c(n) = \frac{q-1}{q} \sum_{k=0}^{q-1} \alpha_k \hat{u}_k \cdot \hat{u}_n + W = -\vec{C} \cdot \hat{u}_n + W$$

où W est une constante scalaire et $\vec{C}$ un vecteur de $\mathbf{R}^{q-1}$, *entièrement analogue à un champ magnétique constant*

$$(24)\quad \vec{C} = -\frac{q-1}{q} \sum_{k=0}^{q-1} \alpha\, \hat{u}_k \nu = \hat{c}$$

où $\nu = \left\| \vec{C} \right\|$. L'énergie totale correspond alors au modèle de Potts en présence d'un champ magnétique constant. Les pixels (« spins ») vont « s'orienter » selon la direction de ce champ, c'est-à-dire selon les labels k associés aux composantes α_k les plus faibles. Cette modélisation est cohérente : ainsi lorsque les α_k sont égaux, $\vec{C}$ est le vecteur nul $\left(\sum_{k=0}^{q-1} \hat{u}_k = \vec{0} \right)$. Effectivement aucune classe n'a été particularisée. Notons que le terme supplémentaire associé au potentiel de Derin apparaissant dans la fonctionnelle énergie est

$$(25)\quad E_3 = \frac{\nu}{2} \sum_{i,j} \left\| \hat{u}_{i,j} - \hat{c} \right\|^2.$$

Dans un de nos applications, nous avons pris en compte un terme de champ $\vec{C}$ dirigé selon un vecteur $\hat{u}_L$ particulier ($L \in [0 \ldots q-1]$), afin d'augmenter ou de diminuer la probabilité de présence de la classe L dans l'image à restaurer. La forme de l'énergie conditionnelle au pixel s devient

$$(26)\quad U = -B \sum_s \delta(\ell_s, \ell^0_s) + \alpha_L \sum_s \delta(\ell_s, L) - K \sum_{(r,s)} \delta(\ell_r, \ell_s).$$

Tous ces développements montrent donc le lien profond avec la « Physique des Transitions de Phase » pour les modèles envisagés dans ce travail.

5. Extension à la restauration d'images de classes

On peut toujours mettre la probabilité *a posteriori* à maximiser $P(L_0/L)\,P(L)$ sous forme d'une distribution de Gibbs « composée » :

$$(27)\quad P(L_0/L)\,P(L) = \frac{1}{Z} e^{-U(L_0/L) - U(L)}.$$

En effet le terme contextuel P(L) peut déjà être supposé champ de Markov, c'est-à-dire vérifiant la propriété de Gibbs. Quant à la conditionnelle $P(L_0/L)$, on peut également la mettre sous forme de distribution de Gibbs sous l'hypothèse de stationnarité et d'indépendance des pixels

conditonnellement à leurs valeurs initiales. Il est alors facile de généraliser le modèle d'énergie développé au paragraphe précédent en définissant les « éléments de matrice » suivants :

— le terme d'énergie d'ordre 1 est

$$U(\ell_s, \ell_s^0) = -\sum_{ij} B_{ij}\,\delta(i, \ell_s^0)\,\delta(j, \ell_s) \tag{28}$$

— le terme d'énergie d'ordre 2 est

$$U(\ell_r, \ell_s) = -\sum_{ij} K_{ij}\,\delta(i, \ell_r)\,\delta(j, \ell_s) \tag{29}$$

— [K] est une matrice générale décrivant l'interaction entre sites, dont les paramètres sont en général difficiles à évaluer et au prix de méthodes complexes (ne serait-ce que pour le modèle de Potts lui-même). Nous choisirons par la suite une représentation diagonale « scalaire »

$$[K] = -K[\mathbb{1}]$$

où [$\mathbb{1}$] désigne la matrice identité. La seule connaissance *a priori* supplémentaire sur la configuration spatiale des régions que nous avons incluse dans [K] concerne le cas de deux régions i et j non adjacentes. Le coefficient K_{ij} associé prend alors une valeur négative de grande amplitude ;

— [B] est une matrice dont chacun des éléments B_{ij} contrôle la probabilité de transition de l'étiquetage initial $L_0 = i$ vers l'étiquetage relaxé $L = j$. Ses coefficients peuvent être déduits de la classification initiale. En effet, soient S_j des échantillons de référence parfaitement classifiés, de taille n_j. On peut en déduire la probabilité conditionnelle d'un étiquetage initial i étant donné l'étiquetage « vrai » j :

$$P_{ij} = P(L_0 = i/L = j) = \frac{n_{ij}}{n_j} \tag{30}$$

où n_{ij} est l'occurrence de la classe i dans l'échantillon j. La matrice [P] n'est autre que la transposée de la matrice de confusion calculée sur les échantillons de référence eux-mêmes. En effet, la matrice de confusion associée à ces échantillons donne, ligne par ligne, la probabilité de classification initiale obtenue pour chacun des échantillons supposés parfaits (voir une application au § 6.4). De plus, le processus de relaxation (B) maximisant la probabilité *a posteriori* $P(L_0/L)\,P(L)$, et en supposant l'hypothèse d'ergodicité ainsi que l'indépendance des pixels vis-à-vis de $P(L_0/L)$, on peut déduire les coefficients B_{ij} :

$$P(\ell_s^0 = i/\ell_s = j) = \frac{\exp(B_{ij})}{Z_j} \approx P_{ij} \tag{31}$$

où les coefficients Z_j assurent la normalisation des probabilités conditionnelles P_{ij}

$$\sum_i P_{ij} = 1\,. \tag{32}$$

Les B_{ij} sont définis à une constante additive près. Dans ce contexte nous pouvons maintenant formuler la contribution des divers termes d'énergie $-\alpha_L\,\delta(1, L)$ comme suit :

$$[B] = -\sum_{L=0}^{q-1} \alpha_L \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & \cdot & 0 \\ 0 & 1 & 0 & \cdot & 0 \\ 0 & 1 & 0 & \cdot & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}. \tag{33}$$

En effet, pour chaque valeur de L, on a $U(\ell_s, \ell_s^0) = -\alpha_L$ lorsque $\ell_s = L$ (0 sinon) indépendamment du label initial ℓ_s^0.

6. Implémentation et résultats

1. *Conditions générales d'expérimentation*

— L'image est raccordée de façon « torique » de façon à obtenir la même cardinalité du voisinage pour tout point de l'image. De façon à accélérer le déroulement de l'algorithme, nous avons mis en place les procédures suivantes.

— Les matrices [K] et [B] sont à coefficients entiers, aux coefficients de pondération respectifs K et B près, de sorte que la probabilité de transition en tout pixel peut se mettre sous la forme :

$$P = \min\left[\exp\left(\frac{KM + BN}{T}\right), 1.0\right] \tag{34}$$

où M et N sont des entiers finis. On peut donc connaître à température fixée l'ensemble des probabilités de transition possibles, sans avoir à les recalculer en chaque pixel.

— De la même façon, nous avons implanté une liste circulaire (2 048 éléments par exemple) prédéfinie d'« objets aléatoires », contenant à la fois des labels et des nombres générés de façon aléatoire. A l'examen de chaque pixel, l'élément courant de la liste indique une transition possible, ainsi qu'un nombre aléatoire qui est comparé à la probabilité P définie par (34). Lorsque le pixel suivant est examiné on passe à l'élément de liste suivant, et ainsi de suite pendant l'ensemble des itérations.

Les paramètres de l'algorithme de recuit simulé que nous avons employés sont les suivants :

— La température initiale est $T_0 = 4$. Elle décroît ensuite d'un facteur $\rho = 5\,\%$ à chaque itération, c'est-à-dire à chaque balayage d'image.

— La convergence est supposée atteinte quand le nombre de pixels ayant changé de valeur entre deux itérations est inférieur à $\frac{1}{160}$ (taille de l'image).

— En général, de 30 à 50 itérations suffisent pour assurer la convergence pour des images comprenant un faible nombre de vclasses (de 4 à 7). Une image de taille 256×256 nécessite donc un temps CPU moyen de 1,6 s par itération sur VAX 8550 et de 6 s sur PC-AT 286 (équipé d'une carte de visualisation en temps réel), d'où un temps CPU total compris entre 45 s et 1 mm 20 s sur VAX et entre 3 et 5 mn sur PC respectivement (i.e. environ 4 fois plus de temps pour des images 512×512).

Nous avons comparé les résultats issus de la méthode du recuit simulé et des Modes Conditionnels Itérés (ICM). Le recuit simulé s'est révélé plus efficace et mieux adapté à notre problème, car la classification initiale peut être proche d'un minimum local de la fonction énergie, surtout lorsque la région à reclasser est importante (voir *fig.* 2 : images LANDSAT). En effet, dans ce cas la contribution des termes d'attache aux données et d'interaction pixel-pixel est prépondérante de sorte que cette région ne peut être facilement reclassifiée par l'ICM.

2. *Reclassification d'images LANDSAT*

Dans notre premier exemple, une image multispectrale LANDSAT de la région de Toulon avait été classifiée précédemment en sept régions : sols nus — culture — sites urbains — feuillus — conifères — eau — classe de rejet (*fig.*2.1). La classe de rejet correspond au label L = 0 (en noir sur la *fig.* 2). On note qu'elle forme presque une région en elle-même. la matrice d'interaction de cliques d'ordre 2 est $[K] = -4.0\,[\mathbb{1}]$ dans tous les cas. Deux matrices [B] ont été utilisées dans nos expériences (coefficient de pondération dans la fonction énergie B = 1.0)

$$\begin{bmatrix} 6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 6 & 0 & \cdot & \cdot & \cdot & 0 \\ 0 & 0 & 6 & 0 & \cdot & \cdot & 0 \\ 0 & \cdot & 0 & 6 & 0 & \cdot & 0 \\ 0 & \cdot & \cdot & 0 & 6 & 0 & 0 \\ 0 & \cdot & \cdot & \cdot & 0 & 6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 6 \end{bmatrix}$$

$[B]_1$

sans reclassification

$$\begin{bmatrix} -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ -8 & 6 & 0 & \cdot & \cdot & \cdot & 0 \\ -8 & 0 & 6 & 0 & \cdot & \cdot & 0 \\ -8 & 0 & 0 & 6 & 0 & \cdot & 0 \\ -8 & 0 & \cdot & 0 & 6 & 0 & 0 \\ -8 & 0 & \cdot & \cdot & 0 & 6 & 0 \\ -8 & 0 & 0 & 0 & 0 & 0 & 6 \end{bmatrix}$$

$[B]_2$

reclassification de la classe 0

Les matrices [B] utilisées dans l'étude d'images LANDSAT . Les résultats correspondants sont indiqués en figure 2.

— la figure 2.2 a été obtenue pour la fonctionnelle d'énergie décrite en (26) avec la matrice $[B]_1$ par recuit simulé. Les régions correspondant aux diverses classes ont été « homogénéisées » de façon « uniforme », indépendamment de leur classe. L'image obtenue peut déjà être utilisée de façon exploitable ;

— la figure 2.3 a été obtenue par ICM pour la même fonctionnelle d'énergie avec la matrice $[B]_2$ (correspondant au label de rejet L = 0). La convergence a été obtenue en 10 itérations. Comme précisé auparavant, on n'obtient pas une reclassification totale des pixels de rejet. C'est ici l'un des cas où le recuit simulé s'avère indispensable pour obtenir l'optimum désiré.

— la figure 2.4 a été obtenue avec la matrice $[B]_2$ par recuit simulé. L'ensemble des pixels noirs a été reclassifié de façon contextuelle dans les autres classes, ce qui est en accord avec la connaissance *a priori* que l'on avait de l'image (6 classes seulement). On remarque que les autres classes ont été peu modifiées par rapport à la figure 2.2, ce qui indique la stabilité de ce type de méthode. La validité du résultat reste à être étayée par comparaison avec une carte thématique réelle.

3. *Reclassification d'images de roches*

La série de figures 3 et 4 présente des résultats d'expérimentations effectuées sur des images de roches prises en imagerie vidéo couleur.

L'image étudiée comprend 6 classes

0 ⇒ Fond	(En noir sur l'image)
1 ⇒ Quartz	(En sombre sur l'image)
2 ⇒ Schistes	(En blanc sur l'image)
3 ⇒ Blende	(En gris foncé sur l'image)
4 ⇒ Oxydes	(En gris clair sur l'image)
5 ⇒ Classe de rejet	(En noir sur l'image) .

Un des buts de cette étude était d'estimer la teneur réelle en blende dans l'image initiale (classe 3). Il s'agissait comme dans l'exemple précédent de reclasser les pixels initialement rejetés, dont le taux considérable dans cette image (37,18 %) est dû à deux raisons :

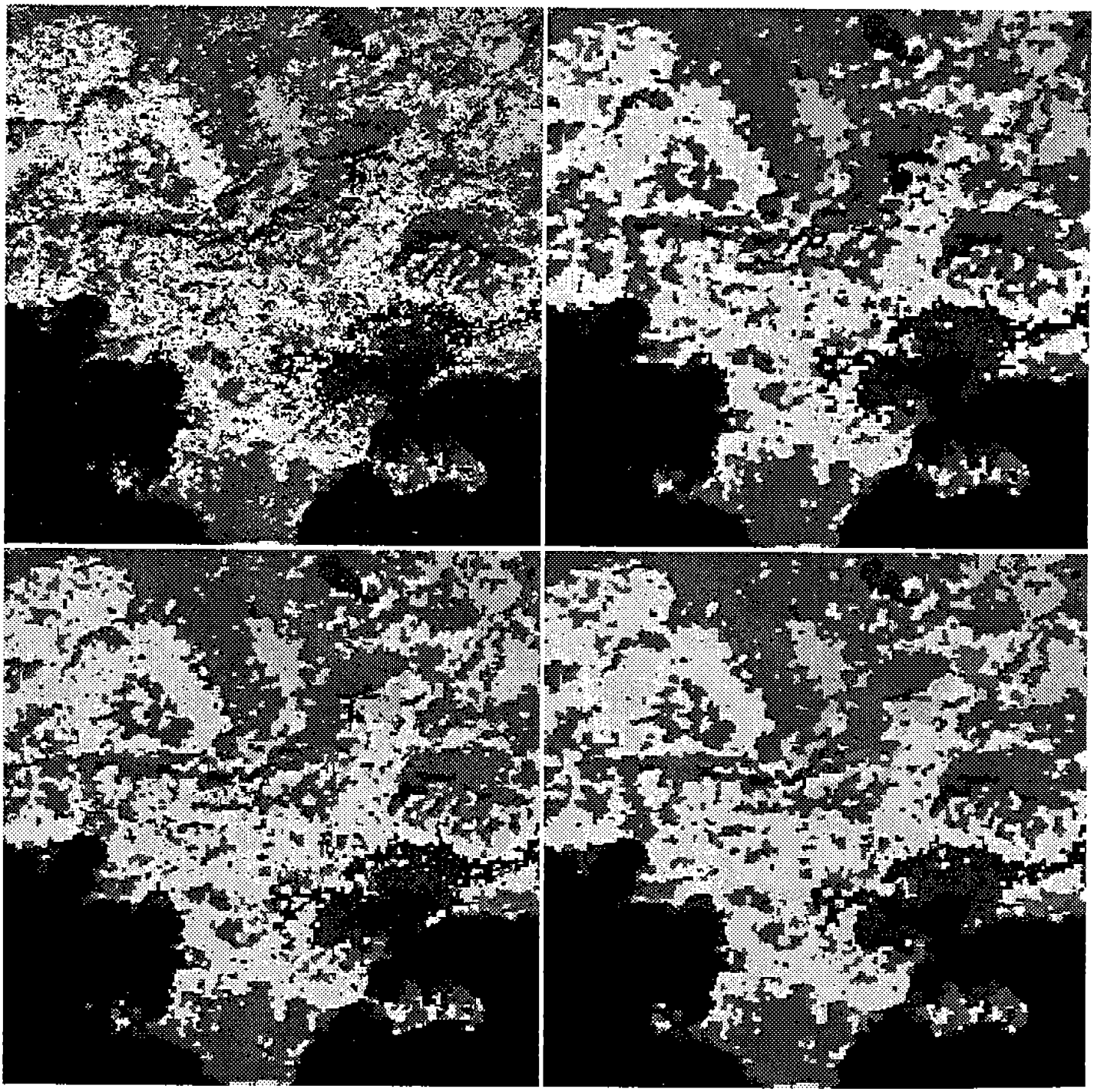
— Les distributions spectrales des minerais présents sur les trois canaux vidéo couleur se recouvrent considérablement, d'où un seuil de rejet élevé dans la classification initiale.

— L'image de teinte initiale est très texturée. Les points de fort gradient ont alors été inclus dans la classe de rejet pour permettre à l'algorithme d'homogénéiser la distribution spatiale des classes. Dans cette étude, elle est la suivante :

	classe 1	2	3	4	5
1	80,05	2,08	0	0,35	17,51
2	5,14	93,76	0	0	1,09
échantillon 3	21,35	10,82	64,24	2,64	0,94
4	25,20	10,87	3,25	52,24	7,53
5	0	0	0	0	0

La matrice de confusion sur les données de la classification initiale (les probabilités sont exprimées en %).

On note une « corrélation » considérable entre les diverses classes, que l'on peut plus ou moins prendre en compte dans l'algorithme de reclassification. Le lien avec la matrice [B] s'effectue, comme exposé précédemment, grâce à (31). Nous avons testé notre méthode avec les matrices [B] suivantes (le coefficient de pondération dans



1	2
3	4

Fig. 2.
1) Classification bayésienne multispectrale d'une image LANDSAT de la région de Toulon.
2) Reclassification de cette image par recuit simulé avec la matrice $[B]_1$ (voir texte).
3) Reclassification de cette image par Conditionnels Modes Itérés avec la matrice $[B]_2$.
4) Reclassification de cette image par recuit simulé avec la matrice $[B]_2$ (voir texte). Les pixels précédemment non classifiés (pixels noirs) ont disparu.

Fig. 3. — Etude d'images géologiques. Classification initiale d'une image de roche (voir texte).

la fonction énergie est B = 0,6)

$$[B]_3 = \begin{bmatrix} 10 & 4 & 7 & 7 & 0 \\ 2 & 10 & 5 & 5 & 0 \\ 0 & 0 & 9 & 3 & 0 \\ 0 & 0 & 2 & 9 & 0 \\ 6 & 0 & 0 & 4 & 0 \end{bmatrix} \qquad [B]_4 = \begin{bmatrix} 10 & 4 & 7 & 7 & 0 \\ 2 & 10 & 5 & 5 & 0 \\ 0 & 0 & 9 & 3 & 0 \\ 0 & 0 & 2 & 9 & 0 \\ 7 & 7 & 7 & 7 & 0 \end{bmatrix}$$

$$[B]_5 = \begin{bmatrix} 10 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 \\ 0 & 0 & 9 & 0 & 0 \\ 0 & 0 & 0 & 9 & 0 \\ 7 & 7 & 7 & 7 & 0 \end{bmatrix} \qquad [B]_6 = \begin{bmatrix} 10 & 4 & 7 & 7 & -5 \\ 2 & 0 & 5 & 5 & -5 \\ 0 & 0 & 9 & 3 & -5 \\ 0 & 0 & 2 & 9 & -5 \\ 6 & 0 & 0 & 4 & -5 \end{bmatrix}$$

Les différentes matrices [B] utilisées dans l'étude d'images de roches (La classe « fond » n'est pas incluse).

— Les matrices $[B]_3$ et $[B]_6$ prennent effectivement en compte la confusion entre classes. Les pixels de rejet seront reclassés par préférence vers les classes 1 (Schistes) et 4 (Oxydes) (voir dernière ligne de la matrice). La matrice $[B]_6$ ne diffère de $[B]_3$ que par un terme de type α_L, L = 5, assurant la reclassification totale de la classe de rejet. Les résultats sont indiqués respectivement sur les figures 3.1 et 3.4.

— Les matrices $[B]_4$ et $[B]_5$ assurent une répartition « uniforme » de la classe de rejet vers les autres classes. Le traitement avec $[B]_4$ permet une reclassification gouvernée par la « sous-matrice » de confusion associée aux vraies classes. Le traitement avec $[B]_5$ effectue au contraire une approximation « diagonale » de la matrice de confusion et assure ainsi une reclassification selon les proportions des vraies classes dans la classification initiale. Les résultats sont indiqués respectivement sur les figures 3.2 et 3.3.

Les taux de présence des minerais dans les diverses images sont les suivants :

	Initiale	$[B]_3$	$[B]_4$	$[B]_5$	$[B]_6$
quartz	15.64	37.74	21.71	26.05	39.70
schistes	33.64	41.85	53.25	54.05	41.30
blende	5.38	5.93	12.27	9.12	5.90
oxydes	8.16	12.72	12.77	10.78	13.10
rejet	37.18	1.76	0.0	0.0	0.0

La variation de ces taux de présence d'un cas à l'autre est cohérente avec le choix correspondant de la matrice [B]. Cette étude reste toutefois une approche d'école demandant à être étayée auprès de spécialistes du domaine.

7. Conclusion

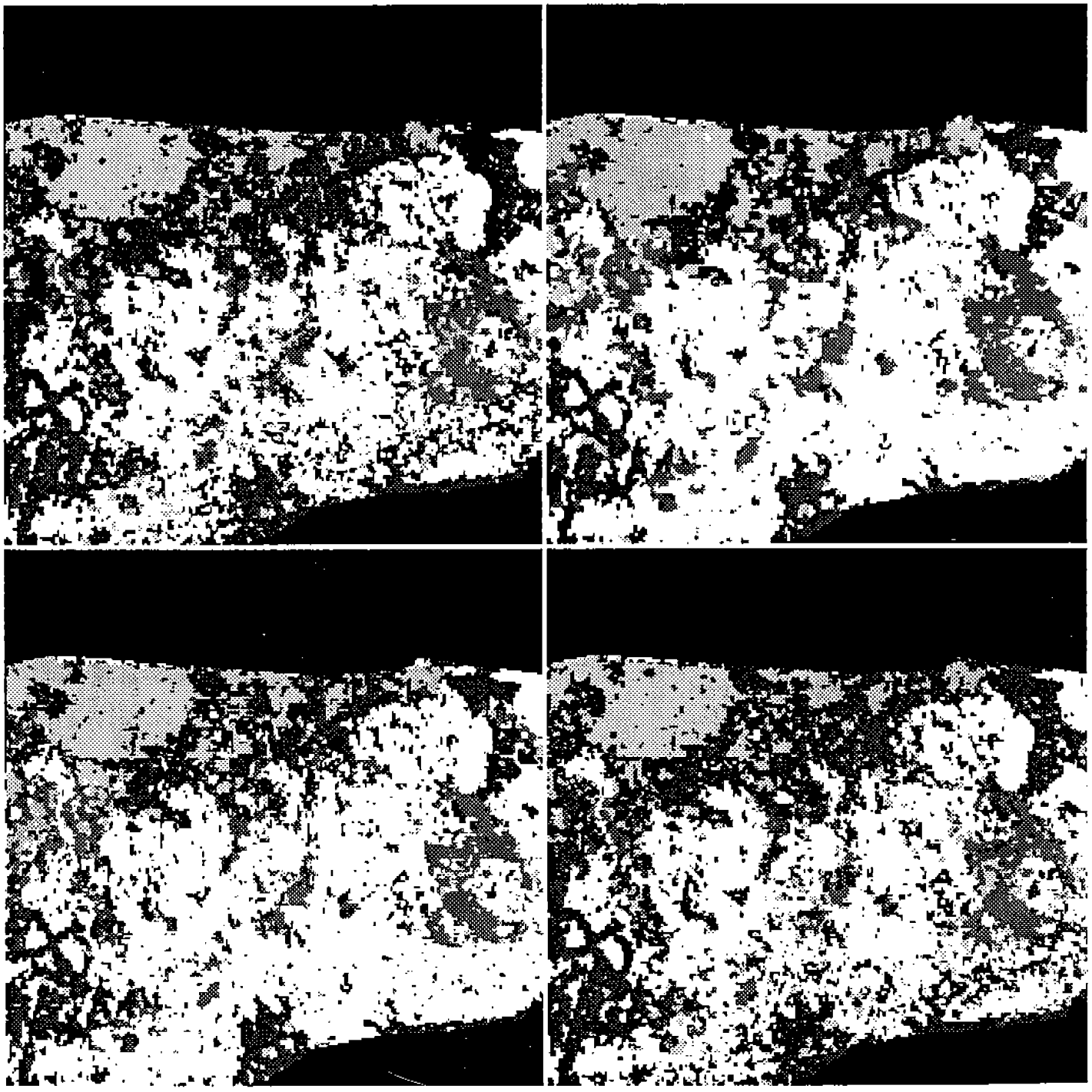
En conclusion, nous avons mis en place une méthode efficace et fiable de relaxation d'images de classes qui pourrait en particulier être en classification multispectrale. Le modèle énergétique décrit dans cet article pourrait être amélioré de façon profitable en y incluant la connaissance des probabiliés ponctuelles mises en jeu dans la première classification. Il en résulterait un processus de Markov non stationnaire [12]. Parmi les problèmes théoriques liés à l'algorithme du recuit simulé qui restent à aborder, on peut citer les suivants :

— *a*) Estimer la température de départ du recuit simulé en fonction de la « qualité » de l'image originale (basse si l'image est peu bruitée, forte sinon), ainsi que la température finale à atteindre (en fonction d'un taux final de contours par exemple).

— *b*) Examiner le comortement de l'image au voisinage d'une température critique. On sait en particulier qu'en 4 connexité et en champ magnétique nul, l'énergie moyenne par site possède une discontinuité à la température critique quand le nombre de classe q est supérieur à 4 ! Les propriétés critiques sont connues pour toute valeur de q, excepté $q = 3$ [5]. Ainsi la température critique vérifie

$$\exp \frac{K}{T_c} = 1 + \sqrt{q}\,. \tag{35}$$

L'énergie par site au point critique, ainsi que son saut lorsque $q \geqslant 4$ peuvent être calculés analytiquement [5].



1	2
3	4

Fig. 4.
Étude d'une image de roche.
1) Reclassification de cette image par recuit simulé avec la matrice $[B]_3$.
2) Reclassification de cette image par recuit simulé avec la matrice $[B]_4$ (voir texte).
3) Reclassification de cette image par recuit simulé avec la matrice $[B]_5$ (voir texte).
4) Reclassification de cette image par recuit simulé avec la matrice $[B]_6$ (voir texte).

Ces caractéristiques critiques peuvent d'ailleurs facilement être retrouvées par l'étude d'une série d'échantillonneurs de Gibbs effectués à diverses températures. Il serait nécessaire de généraliser ces résultats en présence d'un « champ magnétique » uniforme (ou uniforme par morceaux dans le cas de la restauration), ainsi que de les étendre au cas de la 8-connexité. l'ensemble des recherches menées dans ce domaine fait apparaître l'importance du nombre de classes *a priori* pour l'analyse d'images par des méthodes markoviennes [13] !

Manuscrit reçu le 18 novembre 1991.

Remerciements

Les auteurs tiennent à remercier l'ensemble du Département Image de Télécom Paris pour les fructueuses discussions et les nombreux conseils qui ont mené à la réalisation de ce travail.

BIBLIOGRAPHIE

[1] A. Rosenfeld, R. A. Hummel and S. W. Zucker, « Scene Labelling by Relaxation Operation », *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 6, p. 420-433, 1976.

[2] J. Kittler and J. Föglein, « Contextual classification of multispectral data », *Image and Vision Computing*, Vol. 2, No. 1, p. 13-29, 1984.

[3] S. Geman and D. Geman, « Stochastic relaxation, Gibbs Distribution, and the Bayesian restoration of Images », *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 6, No. 6, p. 721-741, 1984.

[4] N. Karssemeijer, « A Relaxation mathod for Image Segmentation using a Spatially Dependant Stochastic Model », *Pattern recognition Letters*, Vol. 11, No. 1, p. 13-23, 1990.

[5] F. Y. Wu, « The Potts Model », *Review of Modern Physics*, Vol. 54, No. 1, 1982.

[6] B. Chalmond, « An iterative Gibbsian technique for reconstruction of m-ary images », *Pattern recognition*, Vol. 22, No. 6, p. 747-762, 1989.

[7] D. Geman, s. geman, C. Graffigne and P. Dong, « Boundary Detection by Constrained Optimization », *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 7, p. 609-628, 1990.

[8] P. Carnevali, L. Coletti and S. Patarnello, « Image processing by Simulated Annealing », *IBM Journal of Research and Development*, Vol. 29, No. 6, p. 569-579, 1985.

[9] J. Besag, « On the Statistical Analysis of Dirty Picures », *Journal of the Royal Statistical Society*, Vol. B-48, p. 259-302, 1986.

[10] Derin and H. Elliott, « Modeling and Segmentation of Noisy and Textured Images using Gibbs Random Fields », *IEEE Transactions on pattern Analysis and Machine Intelligence*, Vol. 9, No. 1, p. 39-55, 1987.

[11] P. A. Kelly, H. Derin and K. D. Hartt, « Adaptative Segmentation of Speckled Images Using a Hierarchical random Field Model », *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. 26, No. 10, p. 1628-1641, 1988.

[12] N. Karssemeijer, « A Statistical Method for Automatic Labelling of Tissues in Medical images », *Machine Vision and Applications*, Vol. 3, No. 2, p. 75-86, 1990.

[13] T. N. Pappas and N. S. Jayant, « An Adaptive Clustering Algorithm for Image Segmentation », *ICASSP Conference Glasgow*, p. 1667-1669, 1989.

International Journal of Computer Vision, 13:2, 229–251 (1994)

Region-Based Strategies for Active Contour Models

RÉMI RONFARD* remi@ina.fr
Centre d'Imagerie et Télédetection, Ecole des Mines de Paris, Sophia Antipolis, 06 700, Valbonne, France, and Télécom Paris, Laboratoire Image, 46, rue Barrault, 75 013 Paris, France

Received January, 1992. Revised February and July, 1993.

Abstract
The variational method has been introduced by Kass et al. (1987) in the field of object contour modeling, as an alternative to the more traditional edge detection–edge thinning–edge sorting sequence. Since the method is based on a pre-processing of the image to yield an edge map, it shares the limitations of the edge detectors it uses. In this paper, we propose a modified variational scheme for contour modeling, which uses no edge detection step, but local computations instead—only around contour neighborhoods—as well as an "anticipating" strategy that enhances the modeling activity of deformable contour curves. Many of the concepts used were originally introduced to study the local structure of discontinuity, in a theoretical and formal statement by Leclerc & Zucker (1987), but never in a practical situation such as this one. The first part of the paper introduces a region-based energy criterion for active contours, and gives an examination of its implications, as compared to the gradient edge map energy of snakes. Then, a simplified optimization scheme is presented, accounting for internal and external energy in separate steps. This leads to a complete treatment, which is described in the last sections of the paper (4 and 5). The optimization technique used here is mostly heuristic, and is thus presented without a formal proof, but is believed to fill a gap between snakes and other useful image representations, such as split-and-merge regions or mixed line-labels image fields.

1 Introduction

1.1 The Contour Modeling Problem in Image Analysis

This paper addresses the problem of automatically creating geometric models for the external boundaries of objects in a 2D image grid. We call the geometric representation a "contour" of the object, and reserve the term "boundary" for its pixel location in the image. A contour representation of objects can be useful for image understanding in 2D or 3D.

The contour modeling problem has traditionally received two opposed approaches: region-based approaches derive a contour representation from a segmentation of the image into well-defined regions, while edge-based methods use a continuous approximation of the original image function, so that boundary points can be characterized by a differential property (image gradient or curvature) and a contour representation be fitted to the boundary points. In their simpler versions, both methods use a point-wise criterium to decide if a given pixel belongs inside an object, outside all objects or at an object boundary. In the region-based approach, a pixel belongs to the boundary if it is in the object region and has neighbors in the background. In the edge-based approach, a pixel belongs to the boundary if it passes a numerical test (e.g. local maximum of the image gradient). In this early boundary detection step, such methods do not take into account the fact that those boundary points really constitute a closed geometric contour, with usually strong continuity and smoothness properties. The next step consists of an approximation method, which strives to find an optimal contour going through all boundary points, but has no interaction with the first step. The deficiency of those methods lies in the absence of top-down mechanisms, such

*Current address: Computervision, 14 Crosby Drive, Bedford, MA 02144, Phone: (617) 275-1800

that boundary detection could be guided by contour constraints.

1.2 Active Contour Models: Approaches and Previous Work

As an alternative to the traditional approach presented in the previous section, the methods of variational calculus (Prenter 1989) have been used by Kass, Witkins and Terzopoulos (1987) in the field of contour modeling. The resulting contour models were named "active contour models" or "snakes." The snakes method provides a way to constraint the points that are tested as boundaries, so that they constitute a parametric curve (or, more simply, an N-sided polygon). Starting from a user-defined curve, an energy minimization algorithm is used to deform the contour model until it fits objects boundaries. The method is gradient-based, and the criterium that characterizes boundary points is summed up over the whole contour to provide the goodness-of-fit measure (or external energy). A smoothness criterium (or internal energy) is also added to guarantee good convergence properties and robustness. Those internal energies provide a very nice framework for top-down processes as mentioned above, and can be theoretically founded on regularization theory.

Active contour models provide a very appealing and successful alternative to the more contrived sequence of boundary points detection and contour curve approximation. They have received much attention in the last few years, and have been improved significantly, notably by Fua & Leclerc (1990), Menet et al. (1990), Amini et al. (1990) and Cohen (1991). They have been applied to image understanding problems in 2D (Fua & Leclerc 1990) or 3D situations (Nitzberg & Mumford 1990), for tracking objects over time (Kass et al. 1987; Cohen 1991), or for infering 3D structure from the deformation of apparent contours in a sequence of images (Cipolla & Blake 1990). Our interest in this approach was motivated by the need for a fast, interactive tool to assist image interpretation and morphometry in scientific applications, such as medical imaging and remote sensing. Of particular interest to us was the ability to focus on a given object of interest, specified by the user, among large sets of data. However, active contours all use an edge-based definition for object boundaries, and we felt the need to extend them to region-based definitions, which are more appropriate for color or remote sensing image analysis problems, and also provide important clues even in more traditional image analysis problems.

1.3 Proposed Approach

In this paper, we propose algorithms and strategies that generalize the variational approach of active contour models to region-based image analysis. In Fig. 1, we illustrate the action of external forces acting on an energy-minimizing contour model. Figure 1A shows the effect of an edge-based energy criterium, as used in most active contour models. A now classic problem with this approach stems from the fact that the image and gradient functions are not very well-behaved (Cohen 1991; Leitner et al. 1991). Inside regions, both derivatives of either the image or the gradient function vanish, therefore providing no clue to the energy-minimizing process. Around boundaries, a more subtle situation arises, where one derivative (normal to the boundary curve) also vanishes. On a theoretical basis, one should resort to higher-order derivatives (e.g. image curvature) or piece-wise continuous image models (preferably with explicit image discontinuities) to correctly model the boundaries (Leitner et al. 1991). In practical terms, most recent active contour models turn the difficulty by pre-processing the edge-data, e.g. through the use of a distance function.

In Fig. 1B, a region-based criterium is illustrated. Instead of a point-wise edge criterium, we use statistical models of the object region (enclosed by the contour model) and background region. If a homogeneous region against a homogeneous background is anticipated, the forces are defined in the following way. All contour points with a neighborhood that fits the object model are pushed outside by centrifugal forces. Conversely, all contour points with a neighborhood that fits the background model are pulled inside by centripetal forces. Both situations are depicted in Fig. 1B, and it is easily seen that this conjectures an external force field $F(M)$, defined for all image points M on the contour curve, and aligned to the normal N to the contour curve (oriented from object to background in Fig. 1). The force magnitude should also be proportional to the difference of statistical fits to object and background. A convenient notation for this is the following:

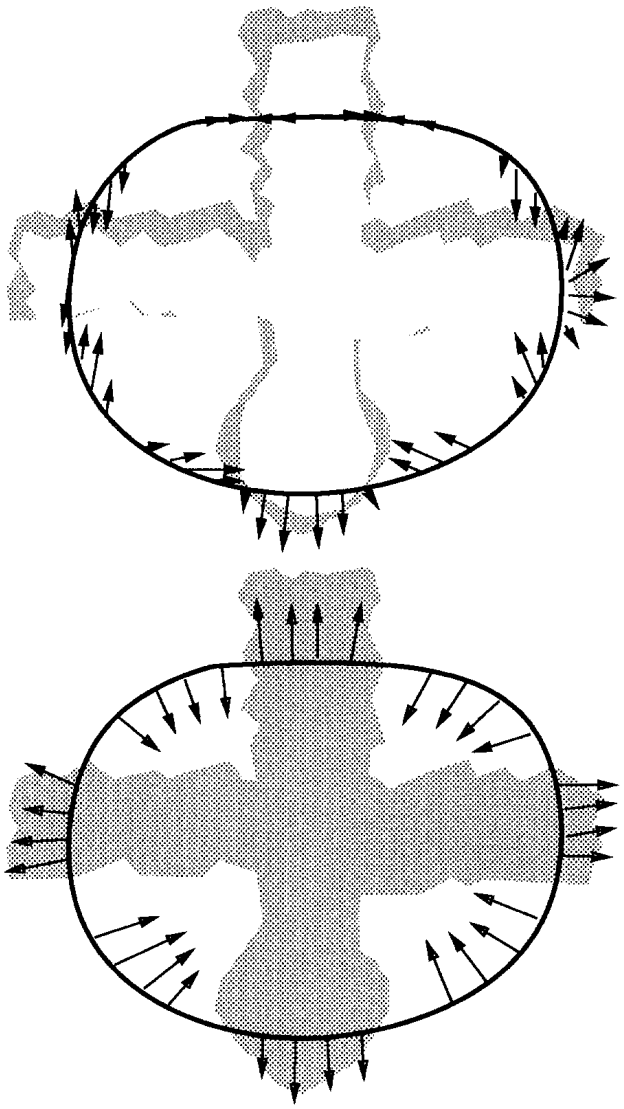


Fig. 1. Edge-based and region-based external forces acting on a contour model. (1A) Edge-based forces. An edge-map is estimated from the original image, and the x- and y-derivatives of the edge-map function are evaluated in every sampled point M. (1B) Region-based forces. Statistical models for the object and background regions are estimated from the original image values and current contour geometry. Statistical fits are evaluated in every sampled point M.

$$F(M) = [\ \text{object}(M) - \text{background}(M)\ ]\ N(M) \quad (1)$$

Contrary to edge-based models such as snakes, this external energy cannot easily be derived from a potential energy, because it is only defined along the contour curve, and is therefore not a point-wise function in the image plane. This paper proposes an approach that closely follows the intuitive view of Fig. 1B, while providing a more formal definition of the energies and forces involved, as well as optimization strategies suited to experiment with them. The resulting algorithm, named "anticipating snake," eventually captures the essence of the original snake approach, but in a very different computational setting, owing much to region-contour interaction methods such as simulated annealing relaxation (Geman et al. 1990) and anisotropic diffusion methods

Thus, in the following section of the paper, we will introduce a contrast measure based on a region statistical image model, which will allow us to use region energies to drive a contour model, and an investigation of the local aspects of variational edge detection methods will be presented, showing the advantages and failings of both edge-oriented and region-oriented contour models. Then, the optimization procedure and heuristics necessary to fit contour curves to object boundaries following our local, region-based scheme will be presented in the third and fourth section of the paper. This will include an important discussion on scale-change issues, and it will be shown how region-based active models can be devised to use a scale heuristic (section 3) and a diffusion heuristic (section 4) while they are being optimized. The last section of the paper will allow us to present and discuss an implementation and some results of the method, with application to different kinds of images.

2 Edge and Region Based Contour Models

2.1 Minimum Principles for Contour Models

Using a minimum principle to define the loci of object contours in an image is appealing, because it corresponds to the intuition of the gestalt definition of shape, which is perceived as a stable, minimum configuration of sensed data. Accordingly, minimum principles have been used extensively in vision research to reconstruct shape from lower-level image data. Active contours are a special case of such reconstruction problems. In order to set those models in a general framework, we will refer to the notations of Fig. 2 throughout this paper, whether dealing with snakes or anticipating snakes. Our general formulation is as follows. Given an image and a hypothetical object contour $M(s)$, we define a set of interaction forces on every point in the contour, so that every deformation of the contour can be quantified with an energy transition δW, basically equal to the work of the interaction forces during deformation. Then, any

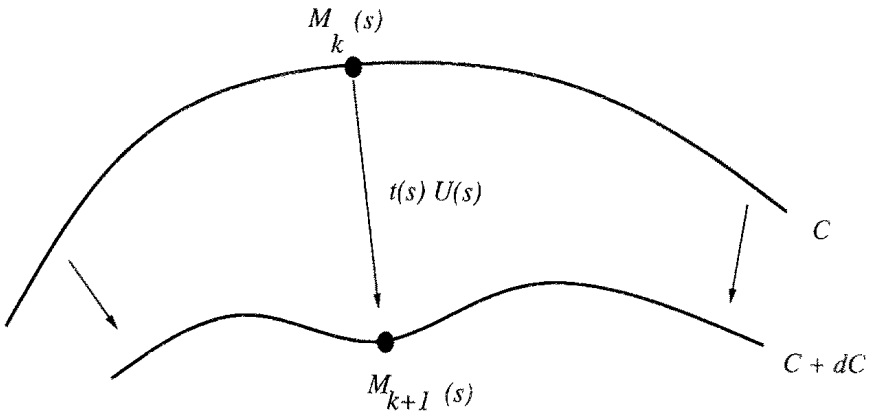


Fig. 2. Deformation of a contour model. Points on the contour are deformed from $M^k(s)$ at time k to $M^{k+1}(s)$ at time $(k+1)$. The deformation is in direction $U(s)$ and has amplitude $t(s)$. The contour model is defined by a set of forces $F(s)$ such that $\delta W = t(s)F(s).U(s)$.

particular contour will be a solution of the contour model if it is a local energy minimum, i.e.:

$$\delta W \geq 0 \text{ for all deformations } \delta M(s) \qquad (2)$$

Inversely, when we set out to define a contour model to solve an application problem, we must insure that actual object boundaries are solutions of (2)—this is our first minimum principle. In addition, we should favor models for which all solutions of (2) are indeed object boundaries (no false local minima). This constitutes a second minimum principle. Devising an active contour model, or equivalently a set of interaction forces, meeting the requirements of those two minimum principles is a tremendous task in the general case. If the problem is somewhat restricted, e.g. by use of "proper" initializations from which solutions can be obtained, or with rigid constraints, then useful models can be devised. In the rest of this section, we will first review the method of snakes, which is one such restricted method of practical value, with external forces driven by the image gradient. From there, we will then introduce our region-based method of *anticipating snakes.*

2.2 *Snakes: A Gradient-Based Contour Model*

2.2.1 Overview of the Snakes Method. Capturing the local structure of discontinuities is a difficult process, as illustrated by Leclerc & Zucker (1987), because many different situations can arise. One key definition for local edge modeling is that of the maximal step edge normal to the direction n at point (x, y), where G_n is the image gradient taken in the direction n (Haralick 1984):

$$\frac{\partial G_n}{\partial n} = 0 \qquad (3)$$

Unfortunately, this equation is usually not easily solved, so that a more tractable version must be adopted instead, using the total norm of the gradient $G(x, y) = ||I(x, y)||$:

$$\frac{\partial G}{\partial x} = \frac{\partial G}{\partial y} = 0 \qquad (4)$$

Direct resolution of equations (3) or (4) is possible in the form of a parametric solution curve $M(s)$, given an exact initial position $M(s_0)$ on the actual object boundary. This point of view has been advocated recently, because it solves exactly for the precise boundary, in cases when precision is the focus of image treatment—e.g. for medical applications (Cinquin et al. 1990). But in many other image vision problems, where robustness is the focus of interest, one must resort to regularization techniques to handle equations (3) or (4) efficiently.

The variational method has been introduced to cope with the difficult numerical problems encountered while trying to solve equations (3) or (4) on a local basis only. The interpretation of equations (3) or (4) becomes that of a maximal contrast condition i.e. minimization of gradient-based energies

$$E_{int} - G_n^2\left(M, \frac{\partial M}{\partial s}\right) \text{ or } E_{int} - G^2(M) \qquad (5)$$

relative to small variations or deformations of the parametric curve $M(s)$. The internal energy E_{int} which appears in this minimum principle has the interpretation of a regularizing term such as curvature

or arc length. The original snakes algorithm (Kass et al. 1987) uses a linear combination

$$\alpha||\frac{\partial M}{\partial s}||^2 + \beta||\frac{\partial^2 M}{\partial s^2}||^2 \tag{6}$$

which can be expressed using a finite difference scheme in the form of a rigidity/elasticity matrix B_{ij} if the contour curve is sampled into N points, so that the function $M(s)$ can be approximated by the $2N$-dimensional vector $M_i = [(x_i, y_i)i = 1..N]$.

The minimum principle imposes that the energy integral over the whole solution curve $M(s)$ be minimal: as such, it has no immediate local interpretation, although the Euler-Lagrange method can be used to transform the global, compound energy minimization back into a differential equation in M, $\frac{\partial M}{\partial s}$ and $\frac{\partial^2 M}{\partial s^2}$ (Kass et al. 1987). Using equations (5) and (6), and the notations of Kass et al. (1987) and Szeliski & Terzopoulos (1989), a finite difference scheme can be used to express the total energy as a quadratic form of the (discretized) position vector M (see appendix):

$$E = \frac{1}{2}\left[{}^tMBM - {}^tM\nabla G^2\right] \tag{7}$$

A natural solution for variational contour optimization is the classical gradient descent strategy. Clearly, the gradient of expression (7) is $BM - \frac{1}{2}\nabla G^2$, so that a step from M^k to M^{k+1} can be taken at time T^k such that

$$M^{k+1} = M^k - t\left[BM^k - \frac{1}{2}\nabla G^2\right] \tag{8}$$

Equation (8) introduces a constant step-size t which controls the rate of deformation of the algorithm, and plays an important role in all active contour optimization schemes. A variation on this theme is suggested by (Kass et al. 1987), resulting in a two-step, semi-implicit scheme with faster convergence, because t can then be chosen arbitrarily large while equation (4) imposes the condition $t < \frac{1}{min(B)}$. This yields a powerful treatment of internal energies (as should be expected), but leaves many pratical issues unanswered as far as external energies are concerned (those aspects of the snakes method have been detailed and discussed in Fua & Leclerc (1990) and Amini et al. (1990).

2.3 Anticipating Snakes: A Region-Based Contour Model

2.3.1 Region Statistics and Image Models. Because the method of snakes relies entirely on its potential energy $E(x, y)$, its domain of application has remained limited. One limitation is that it is sometimes not possible to provide such a function, because of image non-stationnarity. Another limitation is that regional criteria such as color or texture cannot easily be integrated into a potential function (unless a segmentation of the image can be provided). Variational methods have been described, which incorporate gradient and region criteria into a single energy function (Mumford & Shah 1985; Grossberg 1987; Geman & Geman 1984; Marroquin et al. 1987; Geman et al. 1990; Shah 1990). Those methods differ fundamentally from snakes, because they attempt to model the image intensity function, as well as its object boundaries. Although their theoretical importance is enormous, none has led to significant practical solution, most notably because of mathematical difficulties documented in e.g. Mumford & Shah (1989). In this paper, we take a less rigorous approach, drawing mainly on heuristic solutions, in order to show that region criteria can indeed be used to guide a contour model, with a quality of results comparable to that of snakes or related models. In our anticipating snakes method, we replace energy criterions such as (5) by other photometry functions, taking into account the local partition of the image into an object region and a background region. An intuitive interpretation of the approach will first be presented, and then expanded to a complete mathematical treatment suitable for use in the rest of the paper.

The basic idea is that a given closed contour model $C = \{M(s), s \text{ in } [0, 1]\}$ partitions the image plane into an inside (or object) region and an outside (or background) region which, along with a statistical model, may be fitted to the image data. Thus, an alternative to gradient-based schemes will consists in choosing local changes from $M^k(s)$ to $M^{k+1}(s)$ in a given direction $U(s)$, with amplitude $t(s)$, such that the new partition improves the fit. An intuitive implementation of this approach would use separate statistical models $I_{object}(x, y)$ and $I_{background}(x, y)$. Those models would be tested against the given image data $I(x, y)$, based on a mean-square-error criterion:

$$MSE = \int_{object} ||I_{object}(x, y) - I(x, y)||^2 dxdy$$

$$+ \int_{background} ||I_{background}(x, y) - I(x, y)||^2 dxdy \quad (9)$$

While (9) readily takes on the interpretation of an energy, it still cannot satisfy our minimum principles, because *any* arbitrary contour can be a local minimum in homogeneous parts of the image. We therefore need a more constrained energy definition, so that energy minima occur only at actual image boundaries.

2.3.2 A Region-Based Energy Model for Active Contours. In this section, we will use an image contrast measure best known in the context of split-and-merge methods as the fusion energy of the regions (or their Ward distance) (Beaulieu & Goldberg 1989). Using the notations and conventions of Fig. 3, we define the region energy $W^{region}(R_k)$ of a given region R_k as in Leclerc & Zucker (1987), Beaulieu & Goldberg (1989), i.e. the image functions $I(x, y)$ (intensities, colors or textures) are approximated as a linear combination of basis functions $K_i(x, y)$, using a least squares scheme:

$$I(x, y) = \int_{i=1}^{n} a_{ik} K_i(x, y) + \Delta_k I(x, y) \quad (10)$$

The region energy is defined as the sum of squared errors $\Delta_k I(x, y)$, with subscript k to remind us that the error-of-fit function $\Delta_k I$ depends on the region described:

$$W^{region}(R_k) = \left[I R_k || \Delta_k I(x, y)||^2 dxdy \right] \quad (11)$$

This scheme can be devised to fit any possible set of basis functions $K_i(x, y)$, according to the same least squares rule. We then proceed to define the contour energy of a closed curve C, bounding an internal region R_{in} and an external region R_{out}, as depicted in Fig. 4A. We want to derive the contour energy from neighborhood region energies such as in (6). A simple solution to this classical problem is to introduce the union $R_{in} + R_{out}$. We can then write the classical Ward distance between regions R_{in} and R_{out}

$$D[R_{in}, R_{out}] = W^{region}(R_{in} + R_{out}) - W^{region}(R_{in}) - W^{region}(R_{out}) \quad (12)$$

This is defined by Beaulieu & Goldberg (1989) as

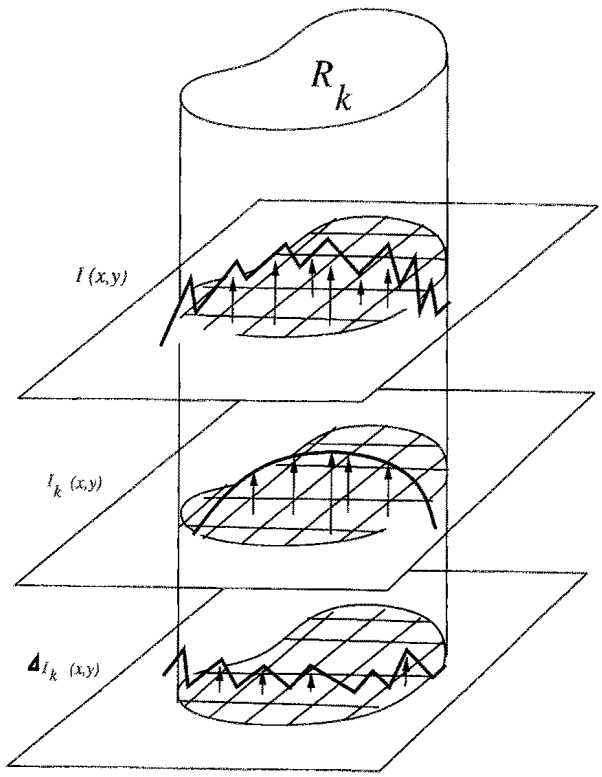


Fig. 3. Notations for image regions. Image intensity $I(x, y)$ is approximated by a region model $I_k(x, y)$ specific to region R_k. The error defines a new image function $\Delta I_k(x, y)$.

the energy needed to merge the two regions R_{in} and R_{out}. We are thus entitled to interpret it as the energy needed to disrupt the bounding contour C between the two regions, and we set $W^{contour}(C) = D[R_{in}, R_{out}]$. Such a contour energy can be interpreted as the amount of region energy absorbed (or explained) by the contour curve C at steady-state.

We now consider a deformed version $C + \delta C$ of the original contour, and proceed to compute the variation δW of the contour energy during deformation. To make further developments easier, we introduce two smaller regions δR_{in} and δR_{out}, corresponding to outwards and inwards deformations respectively, as in Fig. 4B. Note that the inside (respectively outside) region is now $R_{in} + \delta R_{out}$ (respectively $R_{out} + \delta R_{in}$) just before deformation, and $R_{in} + \delta R_{in}$ ($R_{out} + \delta R_{out}$) just after deformation. Since the union $R_{in} + R_{out} + \delta R_{in} + \delta R_{out}$ is not changed by the deformation, we only need consider the negative terms in (7). We thus write the contour energies as

$$W^{contour}(C) = W_0 - W^{region}(R_{in} + \delta R_{out}) - W^{region}(R_{out} + \delta R_{in})$$

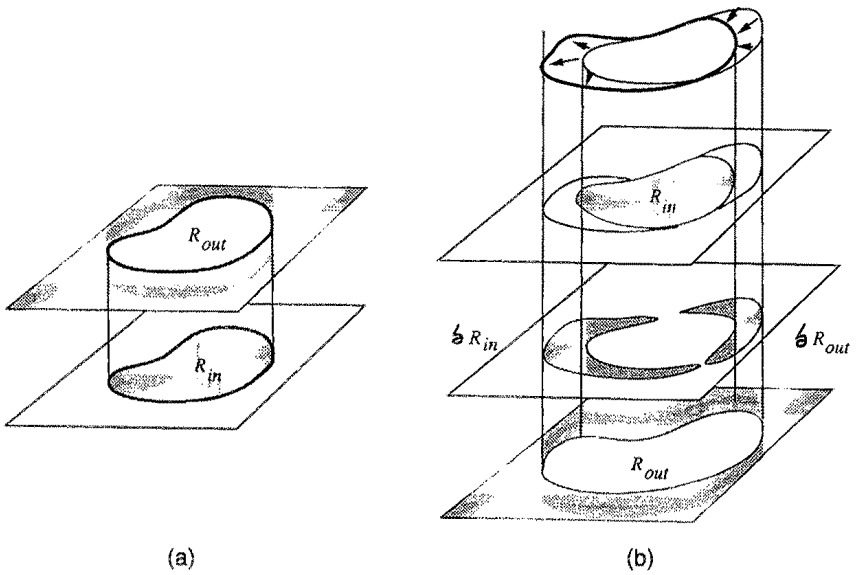


Fig. 4. Regions around a contour model. (4A) Inside and outside regions for a contour model at steady-state. (4B) Inside and outside regions during a deformation. R_{in} and R_{out} are defined as those regions which remain entirely inside or outside the curve during all deformations. The actual regions inside and outside the curve differ from those by δR_{in} and δR_{out}.

$$W^{contour}(C+\delta C) = W_0 - W^{region}(R_{in}+\delta R_{in}) - W^{region}(R_{out}+\delta R_{out}) \quad (13)$$

The energy for unions of disjoint regions can be further developed into a sum of the individual region energies and their Ward distance, thus

$$W^{region}(A+B) = W^{region}(A) + W^{region}(B) + D(A,B) \quad (14)$$

Using (14), we can transform (13) so that all individual region energies are cancelled out when we compute the difference $\delta W = W^{contour}(C+\delta C) - W^{contour}(C)$, and only Ward distances remain, thus

$$\delta W = [D(R_{in}, \delta R_{in}) - D(R_{out}, \delta R_{in})] + [D(R_{out}, \delta R_{out}) - D(R_{in}, \delta R_{out})] \quad (15)$$

When applied locally, this expression becomes even simpler, because local deformations will be either expanding ($\delta R_{out} = 0$) or retracting ($\delta R_{in} = 0$), so that only one of the two terms in (15) need to be evaluated locally. More specifically, we now study the case when an infinitesimal deformation $\delta M(s)$ is applied to a point $M(s)$ along the curve. The situation is depicted in Fig. 5 for the two possible cases of expansion and retraction. From (15) we can write the force $F(s)$ acting on the point $M(s)$, such that

$$\delta W = F(s).\delta M(s) \quad (16)$$

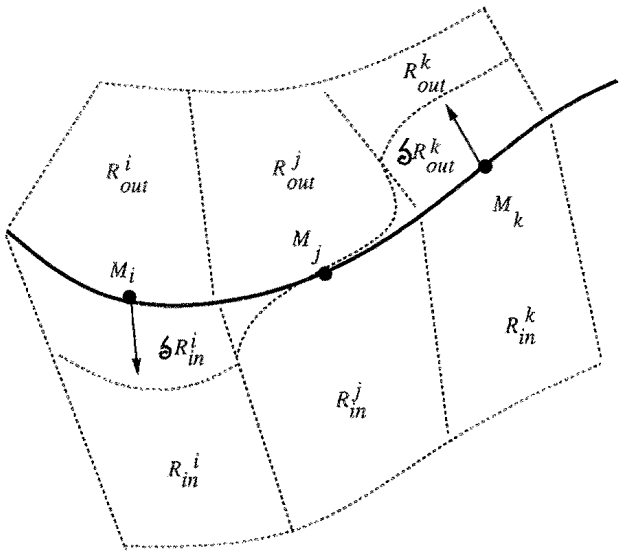


Fig. 5. Expansion and retraction of a contour model. Local interior and exterior regions are shown for each case.

In the case of a retraction, we find that

$$F(s) = [D(R_{out}, \delta R) - D(R_{in}, \delta R)]\frac{\delta M(s)}{||\delta M(s)||^2} \quad (17)$$

is a solution of (16), as can be easily checked out. Similarly, a solution of (16) for the case of an expansion is

$$F(s) = [D(R_{in}, \delta R) - D(R_{out}, \delta R)] \frac{\delta M(s)}{||\delta M(s)||^2} \quad (18)$$

Both cases can be unified if points $M(s)$ are constrained to deform along the oriented normal $N(s)$ to the contour. Assuming that the orientation is toward the outside, as in Fig. 7, and that the deformation is in the form $\delta M(s) = t(s)N(s)$, then (17) and (18) simplify to the single case

$$F(s) = [D(R_{in}, \delta R) - D(R_{out}, \delta R)] \frac{N(s)}{|t(s)|} \quad (19)$$

which is the exact form that was anticipated in (1) for the idealized region-based approach in the introduction to this paper (see Fig. 1B). This expression is very important for our active contour scheme, because it makes explicit the energy variations as a function of deformation, in a computationally attractive way. In this paper, we will use the notation $\frac{\partial W}{\partial C}$ to denote the pseudo-force defined by (19). This is to reminds us that the pseudo-force is not an external field derived from a potential energy, but really a function of the contour curve and pixel values in its image neighborhood. We still refer to $F(s)$ or $\frac{\partial W}{\partial C}(s)$ as a force, because it is used as a force. Since an active contour will usually be discretized and approximated by a piece-wise linear curve, we next review how this changes the computation of energies and forces.

2.3.3 The Discretized Anticipating Snake Model. The previous section has shown that the energy variations and forces acting on a contour model cannot be expressed analytically as simple functions of the position of curve points $M(s)$ and their tangent vectors, but involves computing the deformation regions as well. Therefore, we cannot use the Euler-Lagrange method to transform the energy-minimizing problem into a differential equation. This, of course, is due to the fact that the region derivatives (or Ward distances) depend on the image data in a more intricate fashion than in the case of snakes. Accordingly, it is preferable to use a discretized version of the contour, and express the local forces in this framework, much in the same way as Kass et al. (1987) derived their simpler equations using a finite difference approximation.

Our next step will be to partition the object and background neighborhoods of the contour into small local neighborhoods R_i, following a discretization of the contour curve itself, as in Fig. 6. For the purpose

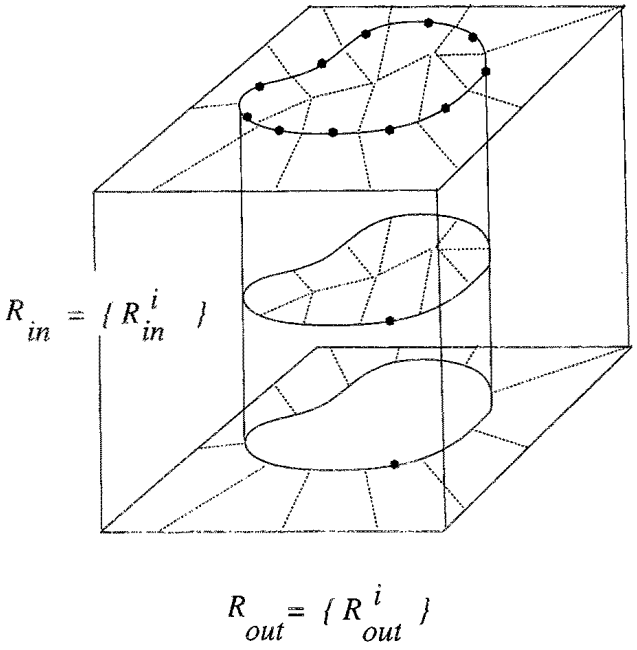


Fig. 6. Discrete regions of deformation. All deformations of point $M(i)$ are confined to its inside and outside regions.

of prototyping our approach, we used a neighborhood structure described in Fig. 7 and section 3 below. We now write the energy as a sum of contributions from every contour point:

$$W^{contour} = W_0 + \sum_i W^i_{INT}(M^i) - \sum_i [W^{region}(R^i_{in}) + W^{region}(R^i_{out})] \quad (20)$$

where $W^i_{INT}(M^i)$ is an internal, elastic energy, as defined in Kass et al. (1987). This approximates the real energy field whenever local neighborhoods partition the entire image plane, as in Fig. 6. Smoothness and rigidity terms can be transformed into finite differences as in (3), (16), Kass et al. (1987). External forces can still be computed as in (19), but using only terms belonging to one particular point M^i, so that the local dissipated energy during a deformation δM^i amounts to:

$$\left[\frac{\partial W^i}{\partial C} + \sum_{j \neq i} B_{ij} M^j \right] \delta M^i \quad (21)$$

Practical implementations of (21) can vary greatly, e.g. region statistics can be modeled separately for each point M^i, hardware can be used to produce the goodness-of-fit functions, or the image function itself can be approximated continuously and calculations

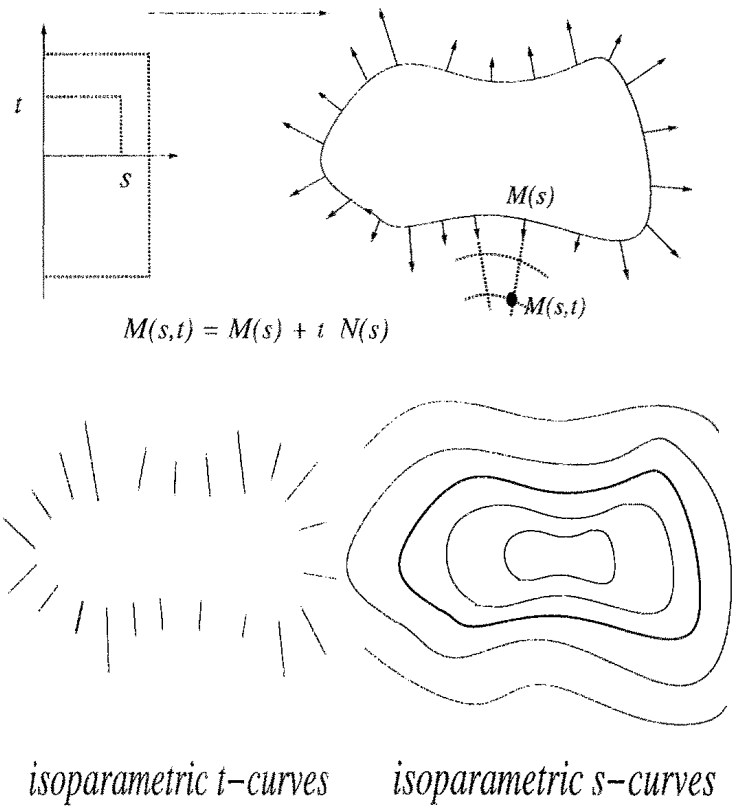


Fig. 7. Local coordinates. Choosing the normal direction for all deformations defines a local system of coordinates (s, t). Curves of constant t are the directions of deformation. Curves of constant s are iso-deformation curves. The real deformation will be $M(s) + t(s)N(s)$.

performed analytically. In this paper, we present the first (and simpler) solution.

2.4 *Internal Energy and Spline Models*

Following Kass et al. (1987), we have written explicitly the internal energy terms in (7) and (21). We are now going to cancel those terms out, and use simpler kinds of contour models with external forces only. The reason is the following. In the regularization approach to snakes, a key issue lies in the choice of a common scale unit between internal and external energy terms (Ronfard 1990). Internal energies were first introduced by Kass et al. (1987), drawing on the theory of approximating spline functions (Laurent 1972), which have classically handled the difficulty by use of cross-correlation methods (Sharahray & Anderson 1989). No such solution extends to the case of equations (17) or (22), because of the strongly nonlinear image forces (Ronfard 1990). This is therefore a very difficult problem, and one that this work does not try to address.

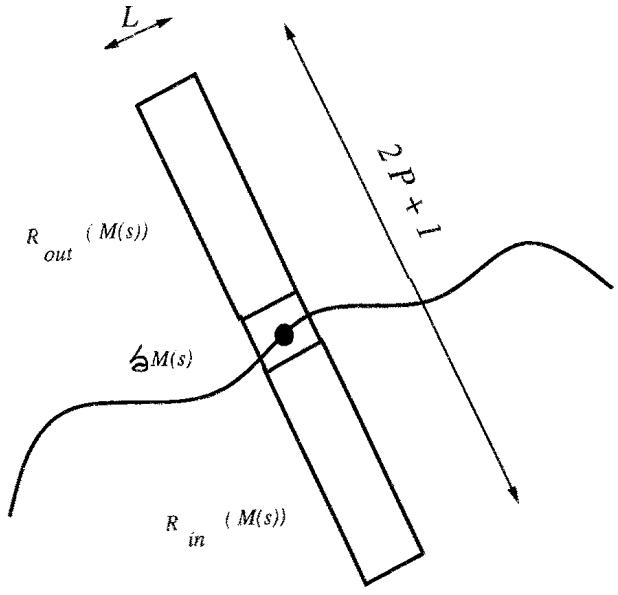


Fig. 8. A simple local structure for variational edge detection. We approximate local coordinates (s, t) with a discrete set $s = 1 \ldots NL$, $t = -P \ldots P$, i.e. a local image grid (s, t).

In the following, we will use instead a simplified version of contour models, similar to the B-snakes

described in Menet et al. (1990) or the spline-snakes in Cinquin et al. (1990). In essence, those spline models imbed the internal energy into a B-spline representation. Internal energy becomes an implicit function of the sampling rate $h = S/N$ along the contour. Control points or knot points can still undergo deformation, under the influence of external interaction forces alone, and other intermediate points are approximated as a B-spline curve fit to the deformed control or knot polygon (see section 5.2 for more details on this). The same basic procedure has been reported also by Cipolla & Blake (1990) as well as Menet et al. (1990), Cinquin et al. (1990) and Ronfard (1990), so that we will resort to it without further justification in this paper, as a powerful and efficient short-cut for illustration of our energy model.

3 Local Depth-Adapting Algorithm

3.1 A Simple Neighborhood Structure for Anticipating Snakes

3.1.1 Energy and forces. The snakes equation (8) establishes a global evolution rule for all sampled points on an active contour model. The first part of this paper has been devoted to present a different approach for active contours, consisting in the choice of arbitrary directions of deformation and local evolution rules, one for each point in the contour curve. We now present algorithms and heuristic methods useful for an active contour model based on equation (19). Using the discrete form of equations introduced in this paper makes all derivatives and forces depend explicitly on image scale (the characteristic size of image neighborhoods around the contour curve). The choice of any particular image scale can be adapted, either geometrically (to fit local image characteristics) or temporally (to improve the efficiency and robustness of the optimization process). The adaptive nature of the algorithms presented in this section makes them particularly suited for interpretation-guided, semi-interactive segmentation, as will be illustrated later.

For the purpose of clarity, we will focus in this paper on the very simple neighborhood structure shown in Fig. 7 and Fig. 8, which can be easily and efficiently implemented. It consists in separate, non-overlapping L-pixel-wide bands centered on sampled points $M(s)$. These structures are $(2P+1)$ pixel-deep and directed along the normal to the curve $N(s)$. All image-contour interactions at point $M(s)$ thus originate from a neighborhood $M(s) + t(s)N(s)$, in the narrow image strip following the estimated local normal $N(s)$, as in Fig. 8. The normal vector $N(s)$ may be obtained from the B-spline expansion of the contour curve, provided the sampling is not too fine. In practice, better estimates will be obtained if a 1-D gaussian-derivative kernel is convolved with the sampled contour $M(s)$ $(s = 1 \ldots N)$. This is important to insure that the mapping from (s, t) to a neighborhood of the curve C in the image plane is one-to-one. This also allows finer sampling intervals on the contour curve, independently of its original B-spline representation (Ronfard 1991).

We compute image-contour interactions from the neighborhood structure in Fig. 8 in a straightforward fashion, using the simplest form of equation (10), i.e. with a piece-wise stationary image model, so that the error-of-fit functions $\Delta_k I$ are simply the average-corrected image values. In this case, the region energies are simply regional image variances

$$W^{region}(R_k) = \int_{R_k} ||I(x, y) - \langle I \rangle_k||^2 dxdy \tag{22}$$

and the contour energy is just the classical (stationary) Ward distance between regions R_{in} and R_{out}, i.e.

$$W^{contour}(C) = D[R_{in}, R_{out}] \tag{23}$$

The general expression for the image forces acting on the contour curve during a step-wise deformation $t(s) \rightarrow t(s) + \Delta t(s)$ then becomes

$$\frac{\partial W}{\partial C} \approx \int_0^1 (D_{in}[M(s)] - D_{out}[M(s)]) \frac{ds}{|\Delta t(s)|} \tag{24}$$

where $D_{in}(M(s))(resp.D_{out}(M(s)))$ denotes the Ward distance between $R_{in}(M(s))(resp.R_{out}(M(s)))$ and the L-pixel deep region $\delta M(s)$ around $M(s)$, as shown in Fig. 8. This is the simplest possible implementation of equation (21), and it has the interesting properties that the local deformation state of the contour curve is completely described by the parameter $t(s)$, and image forces are local to the neighborhood structure $M(s) + t(s)N(s)$, as prescribed.

3.1.2 Computational Framework. We can now present a computionally simple variation scheme, based on the above discussion, as a candidate to solve the active contour optimization problem. For each iter-

ation step of the procedure, we have the following sequence:

- Compute normals $N(s)$ for all sampling points $M(s)$, $s = 1 \ldots N$, and build the image statistics in the neighborhood image regions.
- Starting from undeformed points $M(s) = A(s, 0)$ take as many steps as possible in either one of the deformation directions $M(s)+t(s)N(s)$ or $M(s)-tN(s)$, $t(s) = 0 \ldots P$, as long as the power of the external forces remain positive

$$D_{in}[M(s)] - D_{out}[M(s)])\Delta t(s) > 0 \qquad (25)$$

- Compute a new contour from all deformed sampled points, using them as control points for an regular cubic B-spline curve. Such a curve will *not* interpolate sampled points, but merely approximate them with the best-looking piece-wise cubic curve, in a certain sense (Laurent 1972). This is sufficient for our purpose, because we re-sample contour points in every iteration.

In order to proceed successfully with such a simple scheme, several issues have to be dealt with. First, a number of parameters remain unspecified, i.e. the maximum depth P allowed for step-wise deformations in a single iteration, the sampling rate N of points along the curve, the gaussian variance σ used to estimate smooth normals along the curve. Along with neighborhood width L, those parameters determine all scale choices for our model (see Fig. 9). In contrast to the original snakes (see Fig. 10), those parameters are all local to the iterative scheme, and can therefore be adapted more easily than a filter size for edge detection. Indeed, experience has shown that all scale parameters should be modified during the optimization process. Thus, heuristic methods for determining N and P in each step of the procedure will be presented here (we introduce no such refinement for σ, which we in fact consider as a function of N and P), while the last section of the paper will be devoted to the width scale parameter L.

A different line of problems arise from the choice of the Ward distance. We use the sign of $\Delta t(s)$ $(D_{in}[M(s)] - D_{out}[M(s)])$ in order to determine in which direction the curve should be deformed. It would be comforting to know that this expression vanishes on boundary points. This is not the case, however, and section 3.3 discusses transformed distance functions with a better behaviour in that respect.

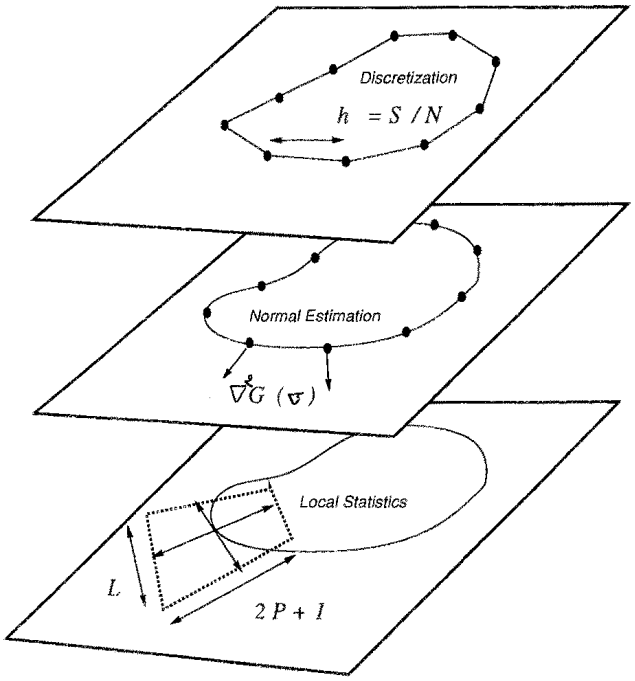


Fig. 9. Scale factors in the anticipating snakes model. Scale choices are needed in (1) to discretize the contour; in (2) to estimate smoothed normals along the contour; in (3) to decide image neighborhood size (depth $2P + 1$ and width L).

3.2 Local Depth-Adapting Strategy

3.2.1 Scale-Space Strategies. The point in this section is to adapt neighborhood size P (see Fig. 8 and Fig. 9) to the scale at which the object-to-background contrast is maximal. A compromise must therefore be found between smaller neighborhoods (for which the norms have no statistical significance) and larger neighborhoods (where more than two regions are in the scope of $M(s) + t(s)N(s)$). The second term of the alternative represents the most difficult situation, because all the equations used to model edges assume two regions (possibly identical) only. This makes the traditional coarse to fine, scale-space approach advocated by Kass et al. (1987) very dangerous, since the local forces acting from the image at coarser scales may become unfounded and misleading, and no verification can be made at finer scales.

This failing is illustrated by the dominant role played in such cases by the automatic internal forces of snakes or other related models (elastic forces, pressure). In such cases, active contours are driven away from their target object boundaries, without being given any chance to recover them (since only finer scales will subsequently be examined, to improve—not correct—the first coarser optimization steps). Fol-

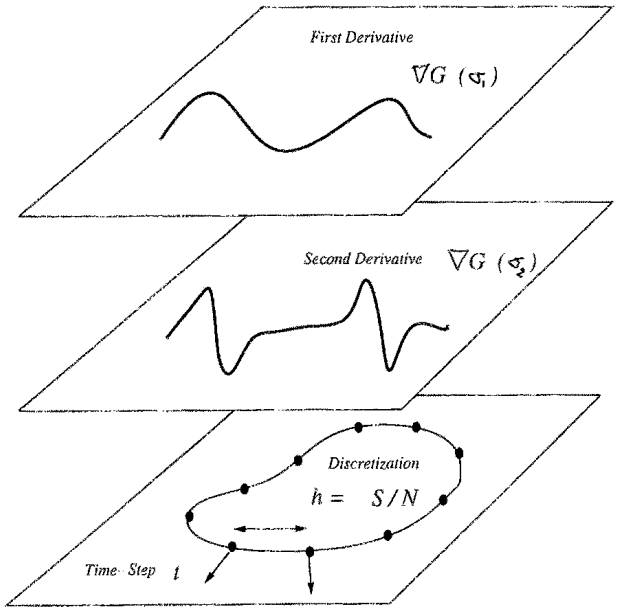


Fig. 10. Scale factors in the snakes model. Scale choices are needed in (1) to compute the image gradient; in (2) to compute the gradient derivatives in x and y; in (3) to discretize the contour and choose uniform step t.

lowing Leclerc & Zucker (1987), we have investigated the opposite, fine to coarse strategy. This approach starts with very small neighborhood structures ($P = 1, 2, 3 \ldots$) even though a high level of noise may be present at such scale. Noise in earlier stages of optimization can be dealt with because the contour curve is regularized, and the steps taken in each iteration are very small ($t_{max} < P$). Then P is increased while optimizing, until all neighborhoods become statistically significant. The optimal deformation state is then easily obtained in a few iterations.

3.2.2 Depth-Adapting Algorithm. This strategy captures the intuitive idea behind active contour models, that optimization should use local image analysis whenever available (in close neighborhoods of object boundaries) and internal cohesion forces when the image profiles are locally flat. The fine to coarse approach also offers an efficient heuristic for increasing the dependency of the variational procedure on image characteristics, since object boundaries are only a finite distance away from the initial contour position, and will be reached in a finite number of steps—without oscillations. This of course assumes a control mechanism for stopping at interfering objects appearing in the background, as the scope of the neighborhood structures extends. All of the above observations can now be summed up in the following depth-adapting algorithm:

[1] Start with an initial B-spline curve, sampled uniformly after arc-length s. Then choose an initial depth parameter $P = P_0$ (preferably $d < P_0 < D$ where d is the min-distance from the initializer to the target object and D is the min-distance between objects in the image).

[2] Iterate the basic steps : fit all control points in the B-spline basis, compute normals $N(s)$ at control points $M(s)$, extract $(2P + 1)$ image pixels in the direction of $N(s)$ for every s, using a fast line drawing algorithm such as Bressenham's, move all control points step-wise from $M(s)$ to $M(s) + tN(s)$, $-P < t < P$, according to the sign of the dissipated energy. This is simply the work of the external pseudo-force from equation (19).

[3] Increment P, and compare energy levels obtained in step [2] with those of the increased neighborhoods, allowing only lower energies at coarser scales. All points with increasing energies at this stage should become inactive or attached.

[4] Iterate steps [2] and [3] until no more control points are active.

An application of this algorithm to a brain tomography scan is presented in Fig. 11. The initial curve is an interactively designed B-spline, sampled every five pixels in a 256 x 256 image. Ten iterations were performed before convergence, with increased depth ranging from five to fifteen pixels on each side of the curve. Since all objects in this image are isolated and offer a roughly constant contrast to their background, the algorithm performs well in this case.

3.3 Controlling Stability

When computing the difference of the Ward distances $D_{in}[M(s)]$ and $D_{out}[M(s)]$, we simply check for its sign, in order to determine in which direction the curve should be deformed. This raises a difficult problem for stability issues, since deformations will always occur, regardless of the magnitude of the energy term. We would like to have a threshold value here to tell us when the external forces acting on the curve are so small as too be negligible. This is not possible within the Ward distance formulation, however, because it does not capture the intuition that external forces smoothly vanish around real object

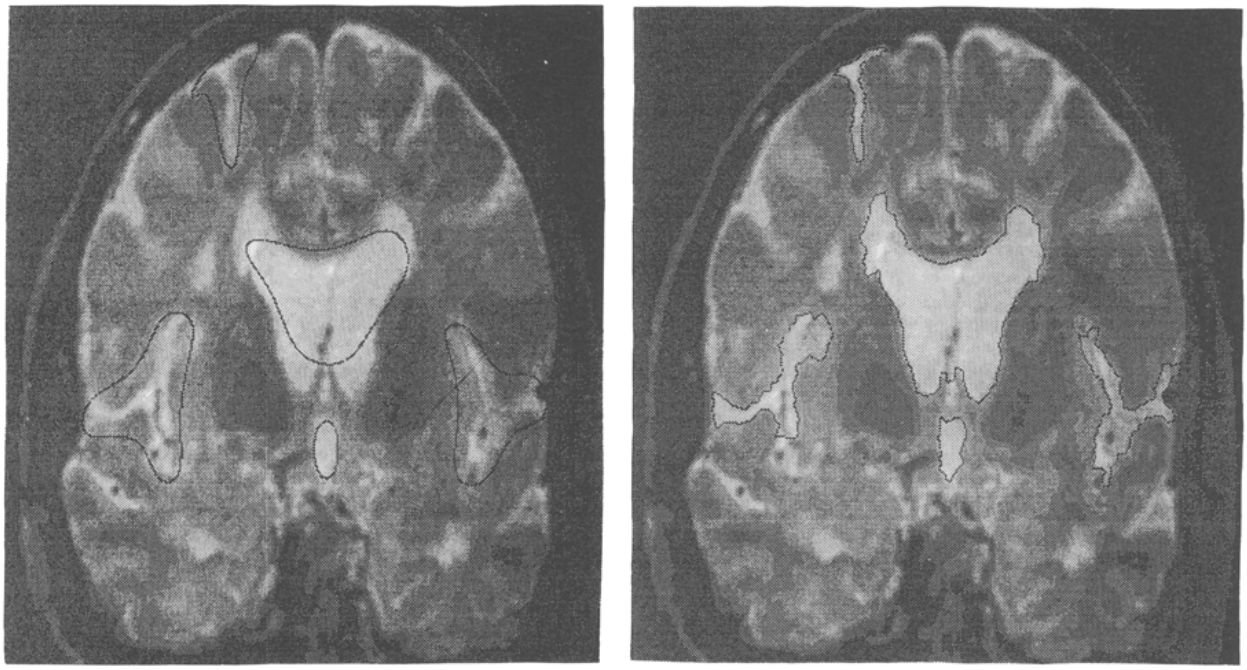


Fig. 11. Application of depth-adapting algorithm to medical imaging segmentation.

boundaries. Instead, it is easily seen that the force magnitude $D_{in} - D_{out}$ may be as great as the Ward distance $[R_{in}, R_{out}]$ in the vicinity of a highly contrasted edge, whereas regions without edges can have very low $D_{in} - D_{out}$ altogether. In such cases, the threshold value would have all optimal points move in and out around an object boundary, while those points that are far from optimum remain in a fixed (and erroneous) position. In order to avoid this undesirable behaviour, we can use the sign and magnitude of other functions of the involved Ward distances. For example, we can use

$$\frac{D_{in}[M(s)] - D_{out}[M(s)]}{D[R_{in}R_{out}]} \tag{26}$$

which appears as a formal derivative of the logarithm of the Ward distance, and enhances the forces in regions where the overall contrast is low, so that points in those areas can be moved away. Another possibility is

$$[D_{in}[M(s)] - D_{out}[M(s)]]e^{-kD[R_{in}R_{out}]} \tag{27}$$

which can be interpreted (loosely) as the derivative of the exponential function of the Ward distance. This also has the desirable effect of boosting the forces in regions away from real boundaries, while having them vanish at higher-contrast edges. Since the exponential in (27) is better-behaved than the ratio in (26), we have used (27) in our implementation.

The use of the exponential further brings a nice statistical interpretation, since $e^{-kD[R_{in}R_{out}]}$ is the Boltzmann distribution for energies $D[R_{in}R_{out}]$, meaning that it is (up to some factor) the probability of the image intensity at M, given its neighbor pixels and the hypothesis of the contour curve C passing through M. Let's denote this conditional probability $P(I(M)|C \text{ in } M)$. It is classically related to the probability of the curve C passing in M, conditionally to the observed intensity $I(M)$, by the Bayes rule which we write here:

$$P(C \text{ in } M|I(M)) = \frac{P(I(M)|C \text{ in } M)P(C \text{ in } M)}{P(I(M))} \tag{28}$$

so that, taking both prior probabilities $P(C \text{ in } M)$ and $P(I(M))$ to be uniform, our evolution rules can be shown to solve for the most probable contour curve position, given the image intensity function $I(M)$ (Ronfard 1991).

Expressions (26) and (27) are very useful to enhance the performances of our variational scheme. We thus substitute one of the expressions (26) or (27) into the computation of image interaction forces. This alteration of our variational definition for optimal im-

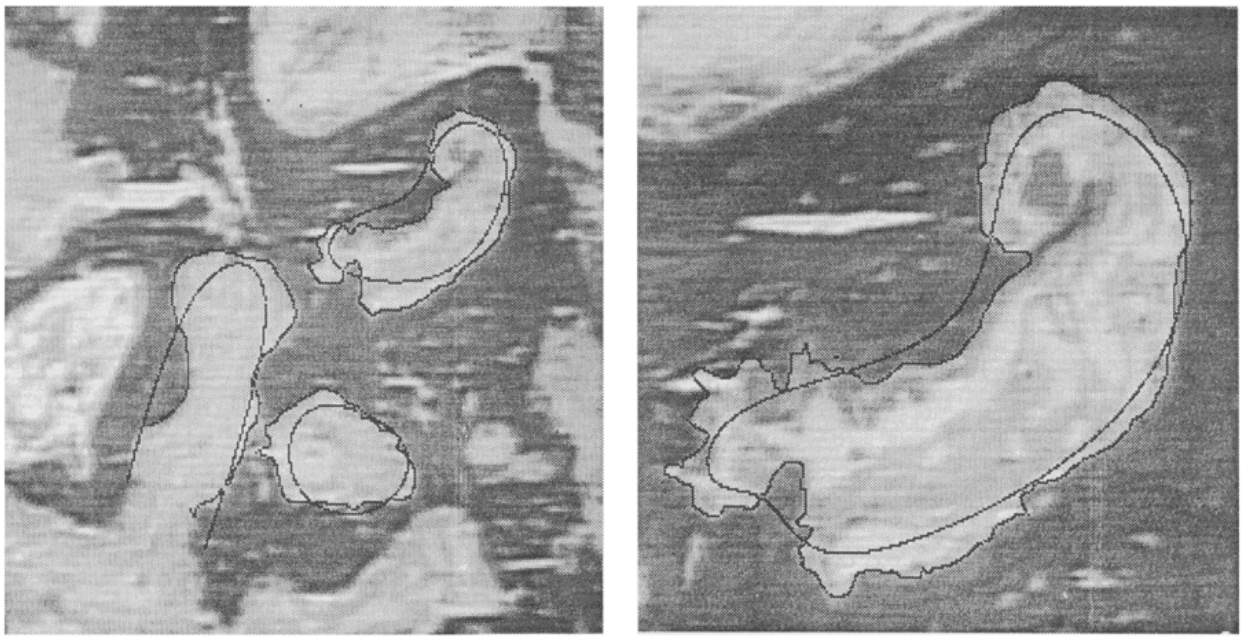


Fig. 12. Application of depth-adapting algorithm to histological morphometry.

age contours makes it remarkably similar to probabilist, non-determinist schemes such as described in Geman et al. (1990), where the exponentials of a quadratic contrast function play a central role.

Our second example (presented in Fig. 12) is a case where direct application of the depth-adapting algorithm fails if we do not use (27), because of the texture in the lighter background. Those bone biopsy images were digitized and contours extracted at interactive rate from a photo-microscope. Because the internal structure in the background has a lower contrast, using (27) results in a correct segmentation in very few iterations. Sampled points on the bone boundary are easily stabilized, since their energy transitions are scaled down by the exponential. Sampled points falling in the background are then more easily moved to the boundary, as the search depth is increased to reach higher contrasted edges.

4 Adaptive Diffusion Algorithm

4.1 Failings and Extensions of the Depth-Adapting Algorithm

Contours obtained with the algorithm presented in section 3 are usually comparable in precision and quality to other snakes (Kass et al. 1987), balloons (Cohen 1991) or model-driven detectors (Fua & Leclerc 1990), although it is difficult to substantiate such an affirmation. One critical element in our approach is the choice of the initial depth and discretization scales. Also, a snake can be used to fill-in missing data from the edge-map, or even occluded contours from 3D-objects, through its internal energy. Our method does not have this feature. More importantly, it usually fails in cases when too many object boundaries are present (experiments not presented), because each local part of the curve sticks to its own optimum, with very little global interaction between different sampling points, except in the spline fitting step.

This section introduces a different algorithm, based on a diffusion method, rather than a purely local scale-adapting method. More precisely, we show how the framework that we have established enables us to control some sort of consistency between locally optimized neighborhoods $M(s) + t(s)N(s)$ along the curve parameter s, so that alien boundary parts can be dismissed while consistent parts are inforced. This is intended as a solution to some of the initialization problems, as well as an interesting methodological shift from a purely local scheme, to a more global approach. While snakes and most other active models have used internal energy to provide global control, we choose to propagate the external energy, in a sense that will be made clear later.

4.2 A Diffusion Heuristic

The depth-adapting algorithm that we have presented and illustrated in our previous sections can be described as a set of automata, placed at regular intervals along the contour model, which detect possible object boundaries in a given direction, and move to those anticipated boundaries. Those moves determine the global deformation of the contour model. Each move is based on the Ward distance between the automaton and its local estimates of intensities or colors inside and outside the contour curve. We have made no attempt so far toward a cooperation between adjacent automata, and this of course results in a rather poor modeling power, since the global behaviour of our contour model only depends on (1) the nearest edge element to each automaton in the direction of its normal, and (2) the shape of the contour curve, because of the spline-fitting step which tends to pull its points towards its centers of curvatures. Without an intuition of what object shapes should be, this does not provide a robust detection method, except in regions where only one object is present.

In order to obtain a more satisfying model, especially in cases where several objects are present around the initial contour model, we are faced with the alternative, either to allow multiple contour models, i.e. deformable models consisting of several closed contours (one for each object), or to enhance a single contour model, so that it can discriminate between different object boundaries, and converge toward a single object. We will not discuss further the first approach, because it seems difficult to restrict it to the simpler cases of contour models having disjoint interiors, without introducing nested structures (Koenderink & van Doorn 1979) or overlapping contours (Nitzberg & Mumford 1990). In both those cases, contour models become much more complex. In the second approach, we only need to enforce new constraints on the contour model, e.g. impose that the regions bounded by the contour curve be of homogeneous intensity or color, at least in the neighborhood of the contour model. This will cause the contour model to favour consistent boundaries, and selectively converge towards a single object. We now have to find an efficient way of enforcing such constraints.

This will be easier to illustrate in terms of a decision network. Instead of a set of independent automata, we now want a network of such automata, so that each one can base its dynamics on estimates of intensities or colors provided by its nearest neighbors as well. As an example, Fig. 13 shows how local estimates $\langle I \rangle_{in}$ and $\langle I \rangle_{out}$ can be propagated along the contour curve, and new estimated values $\langle I^* \rangle_{in}$ and $\langle I^* \rangle_{out}$ can be formed from this mechanism. Since each automaton computes its own estimates, it is a simple matter to connect them so that they share those values. In this section of the paper, we will show how such a simple diffusion mechanism can be used to enhance the modeling power of our anticipating snake model.

In order to compute the second-order estimates $\langle I^* \rangle_{in}$ and $\langle I^* \rangle_{out}$, we use combinations of the first-order estimated image intensities $\langle I \rangle_{in}$ and $\langle I \rangle_{out}$, typically using a one-dimensional gaussian weight function $g(j)$ as shown in Fig. 13 and Fig. 14

$$\langle I^* \rangle_{in}^{i} = \sum_{j=-L}^{L} g(j) \langle I \rangle_{in}^{i+j}$$

and

$$\langle I^* \rangle_{out}^{i} = \sum_{j=-L}^{L} g(j) \langle I \rangle_{out}^{i+j} \tag{29}$$

In (29), indices i and j are in the range of the discrete arc length s across C. This yields a representation of the neighborhood structure extended in width as two layers of inter-connected cells, as depicted in Fig. 14 below, while retaining the local decision structure described above. The input cells are image pixel values, extracted in the estimated normal direction $N(s)$. The first layer of cells then computes local estimates of the image values in half-neighborhoods, while the second layer averages those values along the curve parameter. Finally, output cells perform nonlinear transforms and local comparisons to determine the directions of deformation, much in the same way as before.

4.3 Adaptive Diffusion

With this new definition for our local neighborhood structure, we have introduced a new parameter L, which controls the width of the propagation process, i.e. typically the number of inter-connected points on the contour curve, or more intuitively, the width of the second-order neighborhood structure itself. Just as in our previous sections, we need to decide for a control strategy for this new parameter. This will be

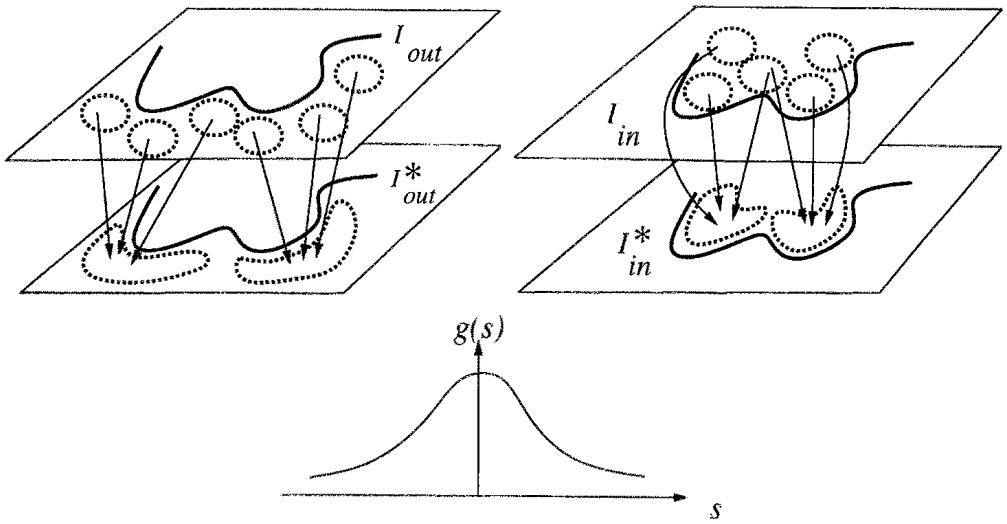


Fig. 13. Extending image neighborhood width. The simultaneous diffusion of intensity on both sides of the contour extends neighborhoods in width.

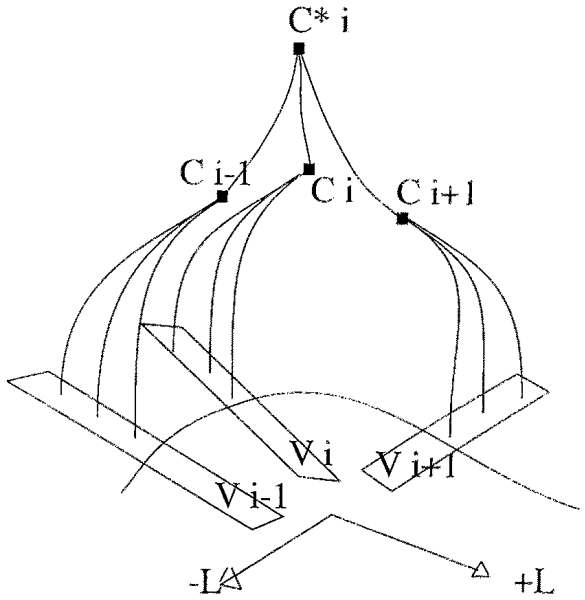


Fig. 14. Interconnection of neighborhood structures. Local color values are weighted sums of the estimated colors for each neighborhood.

based on the following experimental observations:

- substituting $\langle I^* \rangle_{in}$ and $\langle I^* \rangle_{out}$ to the original values in equations (23) and (25) improves the convergence qualities of the anticipating snakes algorithm around object boundaries (where first-encountered boundaries act as seeds for width propagation of the deformations),
- away from object boundaries, extending the width of image neighborhoods yields confusions and unpredictable results, mainly because the differences between inside and outside neighborhoods are small and obscured by random, tangent heterogeneities.

We thus have to introduce a more adaptive propagation scheme, so that we can use the seed values at object boundaries to extend in the width dimension, while ignoring this dimension altogether when no such information is available. Such adaptive treatments are reminiscent of biologically motivated computational models, such as retinex theory (Land 1977) or boundary contour–feature contour interactions (Grossberg 1987).

The retinex scheme computes perceived lightness or color using path integrals away from boundaries—a scheme later shown to be equivalent to a diffusion equation with boundary conditions (Blake & Brelstaff 1987). This original idea was recently re-discovered and extended as anisotropic diffusion i.e. diffusion of image intensity values, with coefficients depending on local image gradients—a scheme which allows filling-in to take place away from object boundaries, which act as diffusion barriers. It should be noted that our own extending neighborhood structures must follow the same kind of rule, but in a different perspective, since all unidentified neighborhoods should progressively be filled-in by image values diffusing from distinct object and background seed half-neighborhoods (the contour curve itself acting as a diffusion barrier).

Interaction between extending neighborhood structures (allowing image intensities to fill-in) and op-

timizing contour curves (which extend and maintain optimal local contrasts) can best be described in the light of the neural theory of interacting features and boundaries, as defined in Grossberg (1987). S. Grossberg's model of visual perception introduces two competing forces (completion of boundaries and featural filling-in) which are both necessary and sufficient to explain, in his view, most pre-attentive object perception data. Feature contours are generated by orientation-insensitive, direction-of-contrast sensitive cells, while boundary contours are detected by direction-of-contrast insensitive, orientation-sensitive cells.

This is an important distinction, which we are going to use a lot in this section. The process of computing a Ward distance along the normal to the contour curve, as in our previous section, is orientation-sensitive, but not sensitive to direction-of-contrast, i.e. it responses to all discontinuities tangent to the contour curve, regardless of the absolute intensities or colors inside and outside the contour. It would be useful to devise direction-of-contrast sensitive detectors as well, in order to guide the deformable contour towards a continuous boundary, i.e. with a constant or slowly varying direction-of-contrast. Of course, this is difficult, unless we are given the object and background colors, or we can anticipate those colors, based on larger, extended neighborhoods.

In order to achieve this goal in the framework of our anticipating snake algorithm, we propose to use an intensity or color diffusion scheme with non-constant coefficients, each coefficient $h(s)$ at arc-length s being a monotonous increasing function of the image contrast along the normal $N(s)$ at s. This is a quite different assumption compared to anisotropic diffusion, because the diffusion along our curve parameter s is controlled by the contrast in the orthogonal direction t. Therefore, the more contrast we find between the inside and outside colors at s, the more we diffuse those colors. When the diffused colors are used by each automaton to compute its motion, we get a process which has become sensitive in the direction-of-contrast. For highly contrasted points, this does not change the computation of energy and forces much, and those points will still maximize their own local contrast, regardless of direction-of-contrast. On the other hand, for lower contrast points, local color values are outweighted by the diffused values from highly contrasted points, which impose a given direction of contrast. Therefore those points will maximize their Ward distance to the diffused colors "from within" and "from without," as if we had set the functions object() and background() in equation (1) for those points, or equivalently the intensity or color charateristics of regions R_{in} and R_{out} in equation (19). This clearly results in a direction-of-contrast process for those points.

Drawing upon this basic idea, we therefore have implemented a simple diffusion process with a diffusion coefficient $h(s)$ which is a function of arc length. This consists of the same two-layer structure as Fig. 14, but with an additional term $h(s)$. We defer the precise definition of $h(s)$ to section 4.5 and simply introduce it here with the following diffusion equation:

$$\langle I^* \rangle^i = \langle I \rangle^i + \sum_{j=-L}^{L} g(j) h_{i+j} (\langle I \rangle^{i+j} - \langle I \rangle^i) \tag{30}$$

The interpretation of $h(s)$ at $s = i + j$ in this equation is that of a local contrast measure at point $M(s)$. Using $h(s)$ results in a selective diffusion from highly contrasted areas to undifferentiated regions, as in Fig. 15. More precisely, equation (30) results from the following line of reasoning: $\langle I^* \rangle^i$ is expected to take values ranging between the local depth value $\langle I \rangle^i$ and the surrounding value I^i_{surr}:

$$I^i_{surr} = \frac{\sum_{j=-L}^{L} g(j) h_{(i+j)} \langle I \rangle^{i+j}}{\sum_{j=-L}^{L} g_j h_{i+j}} \tag{31}$$

Therefore we have $\langle I^* \rangle^i = (1-H)\langle I \rangle^i + H I^i_{surr}$. This simplifies to equation (30) if H is chosen to be the averaged contrast $\sum_{j=-L}^{L} g(j) h_{(i+j)}$ around M_i. Substituting $\langle I^* \rangle^i$ from equation (30) into the anticipating snake algorithm, and increasing the neighborhood width L at every iteration, a very efficient and stable scheme was obtained, as illustrated in Fig. 16 and Fig. 17. The critical—and somehow more technical—point in this version of the algorithm lies in the estimation of the $h(s)$ coefficients—which play a similar role here as the line-process energies in Geman et al. (1990).

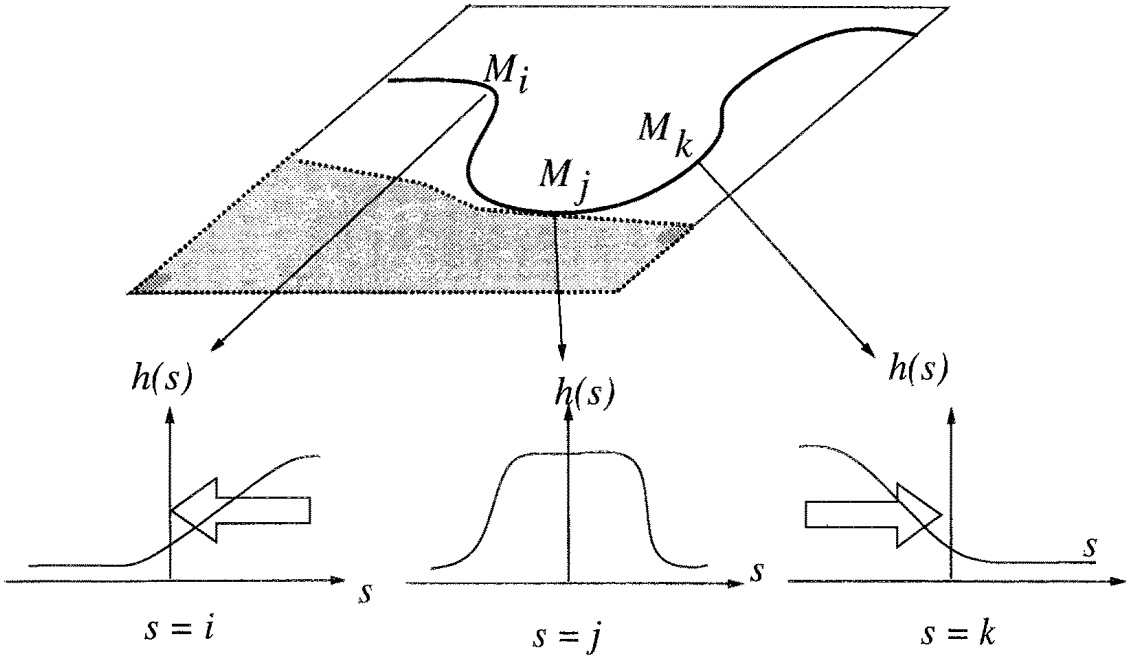


Fig. 15. Diffusion coefficient influence. Diffusion coefficients are related to image contrast across the contour curve. This has the effect of diffusing contrasted colors into undifferentiated regions.

Fig. 16. Application of diffusion algorithm to real-scene image.

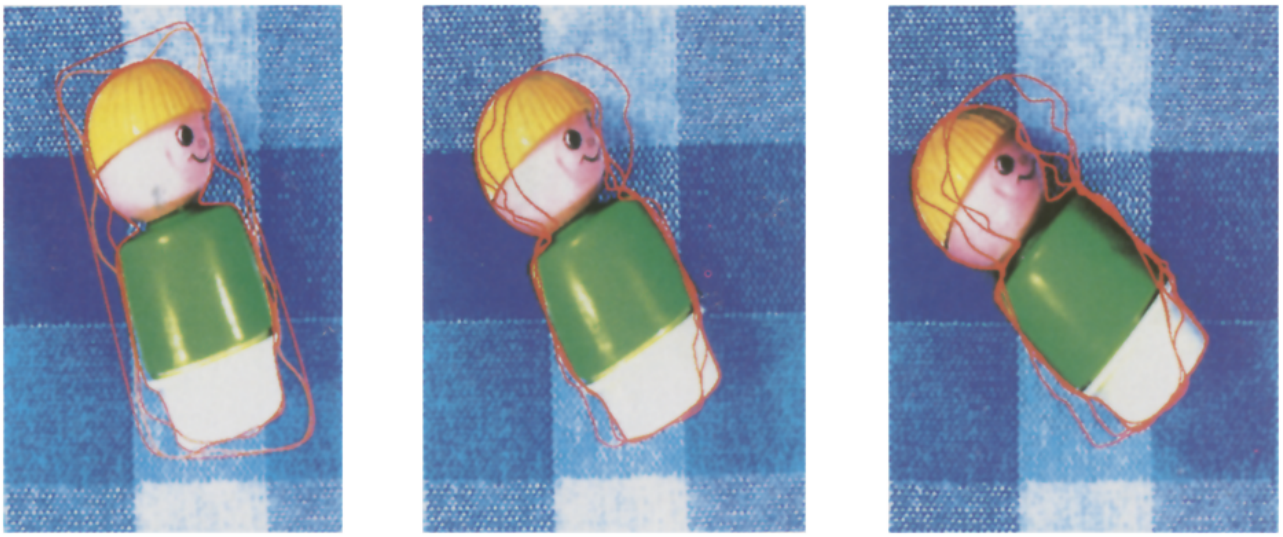


Fig. 17. Steps in diffusion algorithm and object tracking application.

4.4 Adaptive Diffusion Algorithm

In order to illustrate the use of our diffusion heuristic, we have implemented the following extension of our anticipating snakes algorithm.

[1] Starting from an initial contour curve, we perform a first series of iterations following the depth procedure of our previous section, until a limit depth value P is obtained.
[2] We use variances from all half-neighborhoods to estimate the contrasts $h(s)$ and build the two layers of neighborhoods described above.
[3] We compute fusion forces as in equation (15), using the second layer outputs $\langle I^* \rangle^i$ for all neighborhoods, and apply the depth deformation rules from $M(s)$ to $M(s) + t(s)N(s)$.
[4] We increment L and iterate [2] and [3], until an equilibrium is reached.

It should be noted that, since expression (30) acts as a smoothing, regularizing term, no explicit stopping criterion is needed (contrary to the depth-adapting algorithm). In fact, it is possible to rely (as in the original snakes approach) on a much simpler rule, i.e. we end the process when no more deformations occur. Figure 16 presents an example of initial and stabilized contour curves in a color image (shown here in grey levels). The Ward distance is derived from a visual metric in the vector-space of color triplets. All objects have been optimized separately. The contours shown are sampled every five pixels. The depth parameter was varied from 10 to 20 pixels in the first ten iterations. Then the width parameter was increased from 5 to 15 points in the last ten iterations (i.e. every sampled point received information from up to 30 neighbors). The displayed results use as many as 80 to 100 sampling points here. It should be noted that the final scale is large enough for different objects to overlap, meaning that we really obtain a solution which is an optimum among all possible present object boundaries. Figure 17 shows further interesting application results, in the case of a sequence of images. The images in the sequence have been chosen so that the average deplacement corresponds to the depth parameter used, of approximately 20 pixels. Again, this shows a good modeling power, discriminating easily between object boundaries and the surrounding texture. Intermediate states of the contour model are shown in those three images, illustrating only the adaptive-diffusion part of the algorithm (Fig. 17). Except for the first frame, where we have provided an initial curve manually, contours obtained in a given frame were used directly to initialize the next frame (although no real-time implementation has been attempted).

4.5 Computing the Diffusion Coefficients

We cannot use the Ward distance in order to compute the diffusion coefficients, because $D(I_1, I_2)$ has the

dimension of a square norm of image values and $h(s)$ must be a dimensionless coefficient in [0,1]. Non-linear transforms such as (27) and (28) result in poor numerical behaviour when averaged by the gaussian function g_j. Our experience here is that better contrast estimates can in fact be obtained using a statistical test for identical variances in the first layer of the neighborhood structure (Fig. 4). This follows a theoretical suggestion by Leclerc & Zucker (1987).

Letting V_1 be the total variance in the complete neighborhood $M(s) + tN(s)$ and V_2 be the sum of separately estimated variances inside and outside the contour curve at point $M(s)$, we test the ratio $\frac{V_1}{V_2}$, which we assume to follow an incomplete β-function (Press et al. 1988). The result of this test is the probability $p(s)$ of the observed ratio, under the hypothesis that V_1 and V_2 were drawn from a homogeneous population. The local contrast $h(s)$ can thus be estimated as $1-p(s)$. The test Fisher-Snedecor has already been used to study the local structure of discontinuities and detect edges (Leclerc & Zucker 1987) and it proves both consistent and reasonably efficient in our case as well. It extends naturally to more complex image cases than illustrated here, e.g. color or multi-spectral imagery (Ronfard 1991).

5 Results and Discussion

5.1 Scale Changes: Comparison of Depth-Adapting and Diffusion Algorithms

A remarkable feature of the two algorithms presented here is their capacity to change scales during optimization. Our treatment of scale changes in depth closely follows Leclerc & Zucker (1987) but in a more favourable context, because of error-correcting iterations and regularization. As a consequence, a more ambitious treatment of scale changes was possible, as captured by our width extending scheme. However, extensions in depth or in width both have merits and failings, which will now be illustrated and discussed.

Extending in depth has been illustrated in Fig. 1 and Fig. 8. The intuitive nature of the extension process is that all pixels in a neighborhood of the contour curve interact with their projections on the curve, the forces being repulsive, proportional to the similarity of pixel values on and around the curve, and limited in range to the depth parameter P (Fig. 8) All such forces result in a contour-image interaction which drives the optimization process throughout.

This scheme can be very successful (e.g. Fig. 11) as long as the expected boundaries are absolute contrast maxima even at a reasonably large scale P. When this is not the case, our extending scheme will favour higher contrasts in the distance (within P) and lose track of real boundaries. Controlling this situation has proved extremely difficult in the absence of a more global information. On the other hand, cases that can be correctly modeled by depth neighborhoods are optimized without oscillations, in a near-optimal number of steps (typically less than the Haussdorf distance between initial curve and solution). Furthermore, the resulting curve can be refined to quasi pixel-size resolution (Fig. 11 shows final curve sampled every 3 pixels).

Thus, our depth-adapting strategy is best used when image contrast is high and varies smoothly along the anticipated boundary. Otherwise, smoothing image values around optimizing contour is necessary. Adaptive diffusion provides important clues on how to solve those cases. The process has been represented in Fig. 13 as distributed forces acting on the contour curve from neighborhood pixels on the curve.

The diffusion process that we use to compute local pixel colors presents major advantages over both traditional snake methods and our local depth-adapting algorithm, when an overall direction of contrast is perceptible, because it is able to correct itself locally, using neighborhood information without assumptions on the geometry of expected object boundary (as in Fig. 16 and Fig. 17). On the other hand, it is more difficult to control, and not as precise as the simpler depth-adapting algorithm, because it changes image values and therefore leads to smoother contour solutions than the real image boundaries. An implementation using more elaborate neighborhood structures (such as a triangulation of the image) would be very useful for further study of the adaptive diffusion concept.

5.2 Merits and Failings of the Anticipating Snakes Method

The framework presented in this paper addresses only the case of feature contours (step-edges), and cannot easily be made to recognize "roof-edges" or bound-

aries between objects of similar colors. In the case of step-edges, our diffusion algorithm retains most of the available image contrast information, while allowing non-stationnarity (as opposed to pure region segmentation based on the same energy function). The use of an energy measure derived from local comparisons of quadratic error-of-fit functions was borrowed from split-and-merge methods, for which it is well known that smooth, regular contours are usually difficult to obtain. Our anticipating snakes approach could provide a criterion for those cases, making contour extraction much more reliable, while retaining the robustness qualities of region analysis. It also supports scale changes more easily than other related schemes, which is an important factor for stability of our results.

Figures 11 and 12 present results obtained with the depth algorithm, on tomography scan and microscopy images. Figure 16 shows results of the diffusion algorithm on a fairly complex video image. An application of the diffusion algorithm to object tracking in an image sequence is shown in (Fig. 17). In simpler cases such as Fig. 11 or Fig. 12, convergence is obtained in ten to fifteen iterations, with an initial depth of about five pixels, and the algorithm runs in time linear with the number of sampled points. If the diffusion heuristic is used in those images, the number of iterations is reduced, but the running time increases because of the more drastic computations there. However, all examples shown were obtained in times compatible with interactive use (typically under 2 seconds on a 12 MHz microcomputer for 30 sampled points).

The combination of contour-oriented control structures and region-oriented energy measures is powerful, yet many problems will remain unsolved as long as contours are optimized one at a time. There are cases when our anticipating snakes split into separate curves, and it would be interesting to proceed with such twin processes (as a matter of fact, such situations are detected, and corrected by simply deleting all loops but the largest one). Multiple-contour optimization could also prove valuable in order to generate initial curves at different positions in the image—sharing neighborhood values when necessary, thus cooperating into a global segmentation procedure such as Geman et al. (1990), but with an explicit contour shape representation, as in Mumford & Shaw (1985; 1989).

Other difficult issues in the approach that we present here include the choice of an optimal sampling rate, efficient control of the diffusion heuristic after several iterations, and the trade-off between local depth-adapting and diffusion strategies in the course of optimization. Such issues should be discussed in the light of more specific, domain-dependent image analysis application of the method.

6 Conclusion

This paper was motivated by the need to explore variational conditions and algorithms resting on local region analysis, instead of pre-processed edge maps, to handle cases when such maps are not available or too costly. Based on such local region criteria, we have presented a coherent treatment of image-contour forces and variations, as well as strategies for application in a practical active contour system, which compares well with other existing systems, especially in the case of over-segmented images. This is remarkable since we essentially took a very simplistic approach to optimization, control and convergence issues. We argue that it is due to the fact that the region-based energy used here is a better representation for optimal shapes than are image gradients in those cases. Our energy definitions have been presented with statistical and perceptual interpretations, and can be suggested—along with their variational procedures and heuristics—to enforce explicit shape representations in other areas of image analysis i.e. simulated annealing image segmentation and retinex/diffusion schemes.

In the more specific domain of active contour models, our first contribution has been to introduce a general formulation for region-based models, using local error-of-fit functions to build an energy criterium suitable for optimization. This formulation has been illustrated using piece-wise constant image models only, and should benefit from higher-order models when available. Our second contribution has been to make use of adaptive neighborhood structures and diffusion processes, in order to obtain a more robust active contour modeling scheme in the case of very busy images with many objects. The same heuristics could easily be transposed to the case of edge-based contour models as well, hopefully with the same benefits.

This work was performed while the author was a research assistant with Ecole des Mines de Paris, as well as being hosted by Laboratoire Image, Telecom Paris. The author would like to acknowledge the help and support of J.M. Monget, M. Albuisson and

L. Wald, at Ecole des Mines, H. Maitre, F. Schmidt and M. Sigelle at Telecom, as well as valuable critical examination and discussion on earlier versions of this work by Ph. Cinquin.

References

A. A. Amini, T. E. Weymouth & R. C. Jain, Using dynamic programming for solving variational problems in vision, IEEE trans. Pattern Analysis &Machine Intelligence, vol. 12, no. 9, 1990.

J. M. Beaulieu & M. Goldberg, Hierarchy in picture image segmentation—a step-wise optimization approach, IEEE trans. Pattern Analysis & Machine Intelligence, vol. 11, no. 2, pp. 150–163, 1989.

M. O. Berger, Snake growing, Lecture Notes in Computer Sciences, vol. 427, edited by O. Faugeras, pp. 571–572, Springer-Verlag, 1990.

A. Blake & G. Brelstaff, Computing lightness, Pattern.Recognition Letters, vol. 5, pp. 129–138, 1987.

P. Cinquin, F. Leitner, I. Marque & S. Lavallée, 2D and 3D segmentation methods based on differential equations and spline-snakes, Proc. Conference on Curves & Surfaces, Chamonix, France, 1990.

R. Cipolla & A. Blake, The dynamic analysis of apparent contours, 3rd International Conference on Computer Vision, Osaka, 1990.

Cohen, On active contour models & balloons, Computer Vision, Graphics & Image Processing, Image Understanding, vol. 53, n. 2, pp. 211–218, March 1991.

P. Fua & Y. Leclerc, Model driven edge detection, Machine Vision & Applications, vol. 3, pp. 45–56, 1990.

M. Gage, On an area-preserving evolution equation for plane curves, Contemporary Mathematics, vol. 51, pp. 51–62, 1986.

S. Geman & D. Geman, Stochastic relaxation, Gibbs distributions and the restoration of images, IEEE trans. Pattern Analysis & Machine Intelligence, vol. 6, no. 6, 1984.

S. Geman, D. Geman, C. Graffigne & P. Dong : Boundary detection by constrained optimization, IEEE trans. Pattern Analysis & Machine Intelligence, vol. 12, no. 7, pp. 609–627, 1990.

S. Grossberg, Cortical dynamics of 3-dimensional form, color and brightness perception—monocular theory, Perception & Psychophysics, vol. 41, no. 2, pp. 87–116, 1987.

R. M. Haralick, Digital step-edges from zero-crossings of second directional derivatives, IEEE trans. Pattern Analysis & Machine Intelligence, vol. 1, pp. 58–68, 1984.

M. Kass, A. Witkins & D. Terzopoulos, Snakes—active contour models, International Journal of Computer Vision, vol. 1, no. 4, pp. 321–330, 1987.

B. B. Kimia, A. Tannenbaum & S. W. Zucker, Toward a computational theory of shape : an overview, Lecture Notes in Computer Sciences, vol. 427, pp. 402–407, Springer-Verlag, 1990.

J. J. Koenderink & A. J. van Doorn, The Structure of Two-dimensional Scalar Fields with Applications to Vision, Biological Cybernetics, vol. 33, pp. 151–158, 1979.

E. H. Land, The retinex theory of color vision, Scientific American, vol. 237, no. 6, pp. 108–129, 1977.

P. J. Laurent, Approximation et optimisation, Hermann, Paris, 1972.

Y. G. Leclerc & S. W. Zucker, The local structure of image discontinuities in one dimension, IEEE trans. Pattern Analysis & Machine Intelligence, vol. 9, no. 3, pp. 341–355, 1987.

F. Leitner, F. Berthommier, T. Coll, I. Marque, D. Francillard, Ph. Cinquin & J. Demongeot, Neural networks, differential systems and segmentation of medical images, in From pixels to features II, ed. by H. Burkhardt, Y. Neuvo & J. C. Simon, Elsevier Science Publishers, pp. 253–274, 1991.

Marroquin, Mitter & Poggio, Probabilistic solution of ill-posed problems in computional vision. 1987

S. Menet, P. Saint-Marc & G. Medioni, Active contour models: Overview, Implementation and applications, proc. IEEE Conference on Systems, Man & Cybernetics, Los Angeles, California, USA, 1990.

D. Mumford & J. Shah, Boundary detection by minimizing functionals, in Proc. International Conference on Computer Vision and Pattern Recognition, pp. 22–26, San Franciso, 1985.

D. Mumford & J. Shah, Optimal approximations by piece-wise smooth functions and associated variational problems, Communications on Pure and Applied Mathematics, vol. 42, pp. 577–685, 1989.

M. Nitzberg & D. Mumford, The 3.1-D sketch, 3rd International Conference on Computer Vision, Osaka, 1990.

P. M. Prenter, Splines and Variational Methods, Wiley Classics Library, 1975, reprinted in 1989.

W. H. Press, B. P. Flannery, S. A. Teukolsky & W. T. Vetterling, Numerical Recipes in C, Cambridge University Press, 1988.

R. Ronfard, Contours & boundaries in color images, proc. of SPIE Conference on Visual Communication & Image Processing, Lausanne, 1990.

R. Ronfard, Principles for variational edge detection in multi-spectral and color images, PhD thesis, Ecole des Mines de Paris, Feb. 1991.

J. Shah, Parameter estimation, multi-scale representation and algorithms for energy-minimizing segmentations, in Proc. 10th International Conference on Pattern Recognition, pp. 815–819, Atlantic City, 1990.

B. Sharahray & D. J. Anderson, Optimal estimation of contour properties by cross-validated regularization, IEEE trans. on Pattern Analysis & Machine Intelligence, vol. 11, n. 8, Jan. 1989.

M. Sigelle & R. Ronfard, "Relaxation of previously classified images by a Markov Field technique and its relationship with statistical physics," proc. of the 7th Scandinavian Conference on Image Analysis, IAPR, Aalborg, Denmark, August 1991.

R. Szeliski & D. Terzopoulos, From splines to fractals, Computer Graphics, vol. 23, no. 3, pp. 51–60, 1989.

Appendix A Variational Principles and Equations

We assume that the exact energy expression is :

$$E = \int_0^1 \left(\alpha || \frac{\partial M}{\partial s} ||^2 + \beta || \frac{\partial^2 M}{\partial s^2} ||^2 - G^2(M) \right) ds$$

and that it can be approximated by a sum of local energy fields $E = \sum_{i=1}^{n} E_i$:

$$E = \alpha \sum_{i=1}^{n} || \frac{\partial M_i}{\partial s} ||^2 + \beta \sum_{i=1}^{n} || \frac{\partial^2 M_i}{\partial s^2} ||^2 - \sum_{i=1}^{n} G^2(M_i)$$

All terms in this expression can be approximated by finite differences:

$$\left\|\frac{\partial M_i}{\partial s}\right\|^2 = (x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2$$

$$\left\|\frac{\partial^2 M_i}{\partial s^2}\right\|^2 = (x_{i+1} - 2x_i + x_{i-1})^2 + (y_{i+1} - 2y_i + y_{i-1})^2$$

If we now differentiate, we obtain

$$\frac{\partial E}{\partial x_i} = \sum_{j=1}^{n} B_{ij} x_j - \frac{\partial G^2}{\partial x_i}$$

and

$$\frac{\partial E}{\partial y_i} = \sum_{j=1}^{n} B_{ij} y_j - \frac{\partial G^2}{\partial y_i}$$

in which the coefficients in B are simple sums of α and β.

If we assume $E(M)$ to be a quadratic function, we can write (using Euler formula for homogeneous functions):

$$2E = \sum_{i=1}^{n} \frac{\partial E}{\partial x_i} x_i + \sum_{i=1}^{n} \frac{\partial E}{\partial y_i} y_i$$

which easily develops into

$$E = \frac{1}{2}\left[{}^tXBX - {}^tX\frac{\partial G^2}{\partial X} + {}^tYBY - {}^tY\frac{\partial G^2}{\partial Y}\right]$$

Equation (7) in the paper is but a short-hand notation for this latter expression, in which we recognize potential energy terms ${}^tXBX + {}^tYBY$ as well as artificial, quasi-static Gibbs function coefficients ${}^tX\frac{\partial G^2}{\partial X} + {}^tY\frac{\partial G^2}{\partial Y}$.

When we use our region-based formalism, the latter, image-driven energy terms appear only as variations ${}^t\delta X\frac{\partial G^2}{\partial X} + {}^t\delta Y\frac{\partial G^2}{\partial Y}$ which are interpreted (more meaningfully in our opinion) as the amount of dissipated energy due to the image force vector ∇G^2.

EUROGRAPHICS '94 / M. Daehlen and L. Kjelldahl (Guest Editors), Blackwell Publishers

Volume 13, (1994), number 3

Triangulating multiply-connected polygons: A simple, yet efficient algorithm.

Remi P. Ronfard *Jarek R. Rossignac*
IBM T.J.Watson Research Center
P.O.Box 704, Yorktown Heights, NY 10598

Abstract

We present a new, simple, yet efficient algorithm for triangulating multiply-connected polygons. The algorithm requires sorting only local concave minima (sags). The order in which triangles are created mimics a flooding process of the interior of the polygon. At each stage, the algorithm analyses the positions and neighborhoods of two vertices only, and possibly checks for active sags, so as to determine which of five possible actions to take. Actions are based on a local decomposition of the polygon into monotonic regions, or *gorges* (raise the water level in the current gorge, spill into an adjacent gorge, jump to the other bank of a filled gorge, divide a gorge into two, and fill a gorge to its top). The implementation is extremely simple and numerically robust for a large class of polygons. It has been tested on millions of cases as a preprocessing step of a walkthrough and inspection program for complex mechanical and architectural scenes. Extensive experimental results indicate that the observed complexity in terms of the number of vertices, remains under $O(N^{\frac{3}{2}})$ in all cases.

1 Introduction

Triangulation is a fundamental operation in computational geometry and has been studied extensively in an attempt to reduce its algorithmic complexity, i.e. the running time of triangulation as a function of the number of vertices N in the polygon [7] [5] [6] [1]. Triangulation is particularly important in geometric modeling and in graphics. In these applications, the performance of triangulation algorithms cannot be evaluated solely in terms of their algorithmic *worst-case* complexity. More important is the average *expected* behaviour for polygons with a *reasonable* number of edges. Several efficient triangulation algorithms have been proposed for polygons that are *simply-connected* (without holes) [3]. Unfortunately, many computer-aided design systems produce multiply-connected faces that need to be triangulated for efficient rendering and for other downstream applications.

We present in this paper an efficient algorithm, called *flooding*, which works for both simply and multiply connected polygons. The remainder of this section defines the domain, i.e., the class of polygons properly triangulated by the flooding algorithm and their representation accepted as input format by the algorithm. Section 2 positions the proposed solution in the context of previous publications. Section 3 introduces a suitable vocabulary for presenting the

algorithm, in terms of the metaphor of *flooding* a polygon with water. Section 4 describes the algorithm (at an intuitive level) in terms of a finite state machine with only 6 possible states and five transition operations. Section 5 provides implementation details. Section 6 discusses the correctness of the algorithm, and addresses the handling of degenerate (invalid) input models or of topological errors resulting from numerical inaccuracies. Finally, the algorithmic complexity is analyzed in Section 7.

We define a polygon to be a bounded, connected subset of the plane, such that it is equal to the interior of its closure and that it has a finite piecewise linear (nowhere dense) boundary. The boundary may be decomposed into a finite set of *vertices* and *edges* as follows. Vertices are the points with non-linear neighborhoods. Edges are the maximally connected components of the boundary minus its vertices.

The boundary of a polygon may be partitionned into maximally connected components called loops. There exists a unique circular order of all the edges within a loop. The order is obvious at manifold vertices. The edge orientation defines the start-vertices and end-vertices. The *next* edge is the one starting at the end-vertex of the current edge. At non-manifold vertices there are two or more such edges. We chose as *next* the first one encountered while rotating in a clockwise manner around the end-vertex. Given the above convention, a polygon has a unique representation (as a set of loops, each loop being a circular ordering of vertex references) which is always valid. In this paper, we address the issue of triangulating a polygon represented using our conventions. The triangulation produces a list of triangles T and internal edges I such that:

a) Internal edges of I are relatively open line segments contained in the polygon and connect vertices of the polygon (triangulation does not add artificial vertices),

b) Internal edges and triangles partition the generalized polygon (they are pairwise disjoint and their union is the entire polygon).

2 Previous work

Triangulation may be performed by first decomposing the polygon into simpler parts and then by triangulating each part with an efficient, special-purpose method. As an example, polygon decomposition into convex parts [9] yields an efficient method of triangulation, because *any* vertex in a convex polygon can be joined by internal lines to *all* other non-adjacent vertices to produce a correct triangulation. Another interesting approach consists in adding horizontal internal edges, so as to decompose the polygon into a union of trapezoids. Since trapezoids are trivially triangulated, this again provides an efficient approach [a] [4], which has been recently improved to a linear complexity algorithm [1]. A more general (monotonic) decomposition method has been described by Lee and Preparata [3], as a special case of what they call *regularization* of a planar graph. The regularization algorithm finds a set of monotonic regions partitioning both the interior and the exterior of any simple polygon in $N \log N$ time and N space. An efficient triangulation of monotonic polygons has been described by Garey and his colleagues in 1978 [5] and is also discussed in [4].

The new flooding algorithm applies to all types of polygons, whether simply or multiply connected. Flooding generalizes the monotonic decomposition of [3] and the monotonic tri-

angulation of [5], but does not require building their complex auxiliary data-structures. It is similar in spirit to the sweep-line algorithm of [6], but sweeps only one gorge at a time.

3 The flood metaphor.

A coordinate system (a direction, the *vertical*, and an origin) is chosen and used to define the *height* for vertices. A polygon is *monotonic* if its boundary has exactly one maximum and one minimum. A polygon may be decomposed into monotonic regions: its *gorges*. (Such a decomposition is in general not unique.) Since a gorge is monotonic, its boundary can be split into two parts, its *left and right banks*.

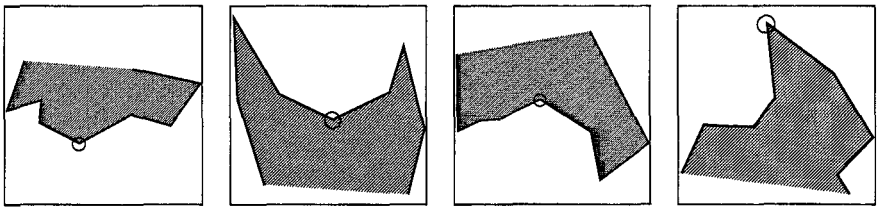


Figure 1: Non-monotonic vertex types.

As illustrated in figure 1, there are four types of special vertices: a convex local minimum is called a *bottom*, a concave local minimum is called a *sag*, a concave local maximum is called a *top*, and a convex local maximum is called a *ceiling*. Note that no distinction needs to be given to non-manifold vertices. A *wedge* is a chain of adjacent triangles sharing a common vertex, called the apex of the wedge.

The internal data-structure used by the flooding algorithm is an array of vertices and a list of loops, each stored as a doubly-linked list of vertex references. As the algorithm progresses, loops may be split or merged, but there is one *active loop* at any moment.

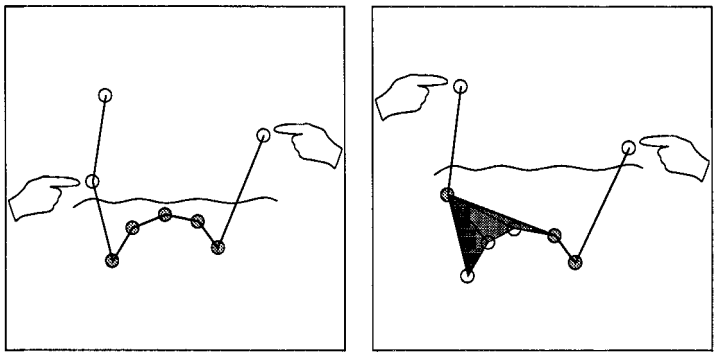
The *active chain* is a connected subset of the active loop. Its starting and ending vertices are called respectively *Left* and *Right*. The vertex preceding Left is called NextLeft. The vertex following Right is called NextRight. (NextLeft and NextRight are indicated by fingers of left and right hands in the following figures.) The active chain always forms a concave line and is bounded by a bottom vertex of the active loop. The Right and Left vertices are on the opposite sides of a gorge. Initially, the active chain contains only one vertex, the lowest bottom in the external loop of the polygon, which is the Left *and* the Right vertex of the active chain.

In the simplest situation, the flooding algorithm fills that gorge by removing bottom triangles and by extending the active chain to its neighbor edges. Basically, the height of NextLeft and NextRight are compared. Suppose, without loss of generality that NextLeft is lower. The vertices of the active chain that are visible from NextLeft (and connected to NextLeft, by either external or internal edges) define a wedge of triangles to be removed from the polygon. The active loop is adjusted and NextLeft becomes Left. This operation

is called a raise, and can be iterated in a monotonic polygon, which it will triangulate in $O(N)$ time [3] [5] [6] [7]. The main contribution of our method consists in the four other operations (bridge,fill,spill and jump), which together with the raise, will triangulate an arbitrary triangle, in a similar fashion.

4 High-level description of the algorithm.

In the pre-processingstage, we extract all sags and sort them in height. (Equal height conflicts are resolved using lexicographics order of both the X and Y coordinates of the vertices.) The sorted list of sags reduces the cost of searching for the lowest sags inside the current gorge.



Before a raise After a raise

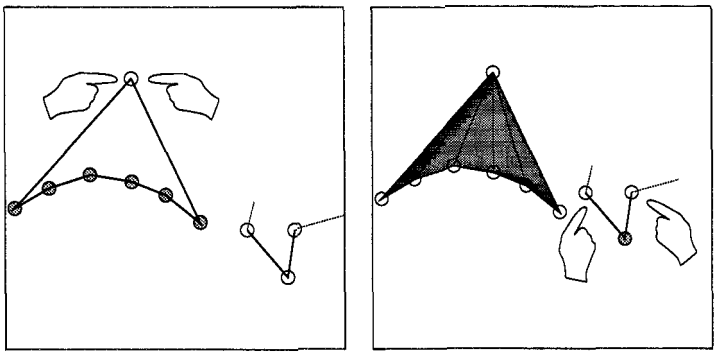
Figure 2: Raise.

Each sag will eventually be connected to a top or a regular vertex by an internal edge. Such a bridge operation may divide the active loop or connect it to another loop creating two gorges. The algorithm will pursue (and fill) one of these gorges. The bottom of the other gorge will be put on a to-do list of pending bottoms. When the top of a gorge is reached (the gorge is full), the flooding algorithm checks the list for bottoms of unprocessed gorges. (Note that a pending bottom added to the to-do list during a bridge operation can also be processed as a result of spilling around a hole, in which case it will not require further actions.)

Each vertex has pointers to its immediate right and left neighbours in a loop, plus an additional jump pointer, initialized to null. The purpose of jump is to point directly to the other bank of a previously filled gorge, that was momentarily put on hold to pursue a spill.

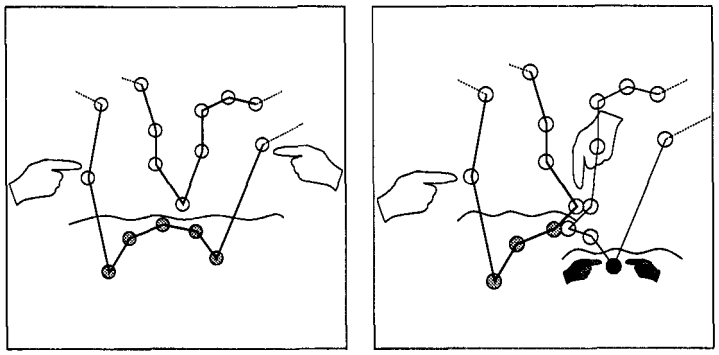
Flooding may be viewed as afinite-state automaton with only the following five transitions (i.e., actions):

- **Raise.** Assume (without loss of generality) that NextLeft is below NextRight. If NextLeft is above Left (the edge goes up and Left is not a top), we search the sorted list of sags for the lowest sag above Left and Right and below NextLeft in that gorge. If no such sag exists, one can triangulate a portion of the polygon by constructing interior edges between NextLeft and the vertices of the active chain visible from NextLeft (cf. fig. 2).



Before a *fill* After a *fill*

Figure 3: Fill.



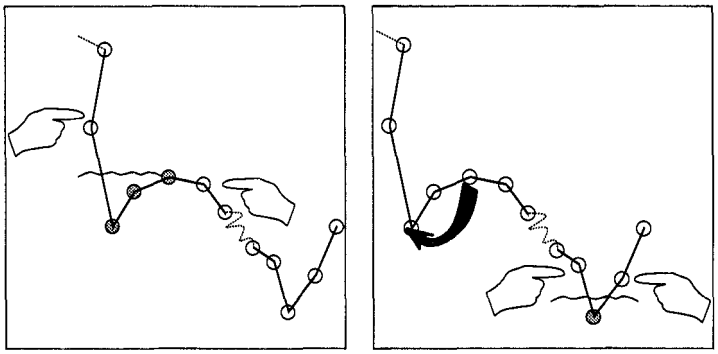
Before a *bridge* After a *bridge*

Figure 4: Bridge.

- **Fill.** If NextLeft and NextRight are the same, internal edges can be created from this vertex to all non-adjacent vertices in the loop triangulating the remainder of the polygon (figure 3). After a *fill*, the Left and Right pointers are reset to the next non-processed pending bottom.

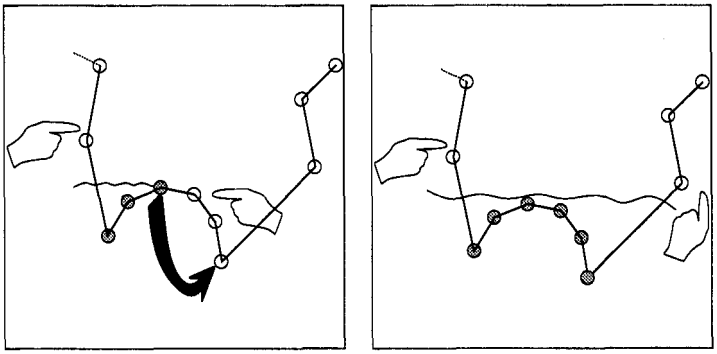
- **Bridge.** If **a** sag is found during the above check, we create a *bridge* of two internal edges from the sag to the highest vertex in the active chain. After a Bridge, the chain is either split into two subchains, one of which remains active, while the other one remains attached to a new *pending bottom* (cf fig. 4).

- **Spill.** If Left (respectively Right) is a top, and has a null *jump* pointer), we set its jump pointer to Right (respectively Left) on the opposite bank of the gorge for when we are done filling the adjacent gorge, and then follow the chain towards the left (respectively right), searching for the next bottom *not matched* by a sag encountered along the search path. This new bottom becomes the start of a new gorge-flooding process (it becomes



Before a *spill* After a *spill*

Figure 5: Spill.



Before a *jump* After a *jump*

Figure 6: Jump.

the Left and the Righ of a new active chain, see figure 5). The requirement for skipping as many bottoms as sags encountered during the search guarantees that we will not attempt to back-track a spill, which would create an infinite loop.

- **Jump.** If the Left (respectively Right) pointer is a top, and its *jump* pointer is not null, we jump to the other bank (i.e., reset it to the value in the jump pointer), as seen on figure 6.

The entire flooding process is illustrated in figures 7 and 8, which show one example of triangulation, at several important intermediate steps of the algorithm (figure 7), with all triangles numbered according to the order in which they have been flooded (in figure 8).

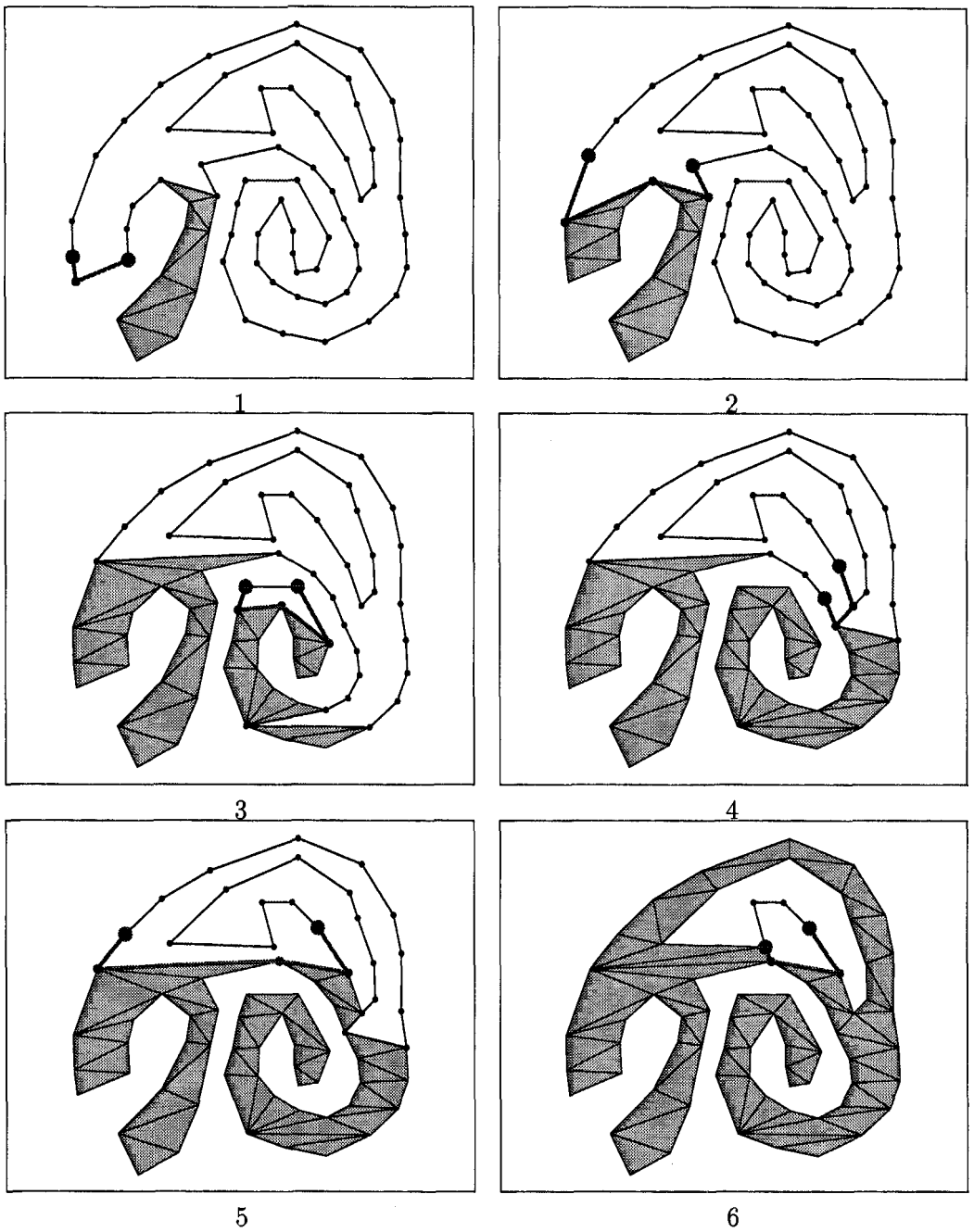


Figure 7: Intermediate steps. Each figure shows the result of a series of actions, followed by numbers indicating which triangles have been created and removed: (1) Raise 0-7, Spill; (2) Raise 8-12, Jump; (3) Raise 13-18, Bridge, Raise 19-26, Spill, Raise 27-30, Jump. (4) Raise 31-33, Fill. Raise 34-41, Bridge; (5) Raise 42-45, Jump; (6) Bridge, Raise 46-52, Spill, Raise 53-62, Jump, Raise 63-64, Fill.

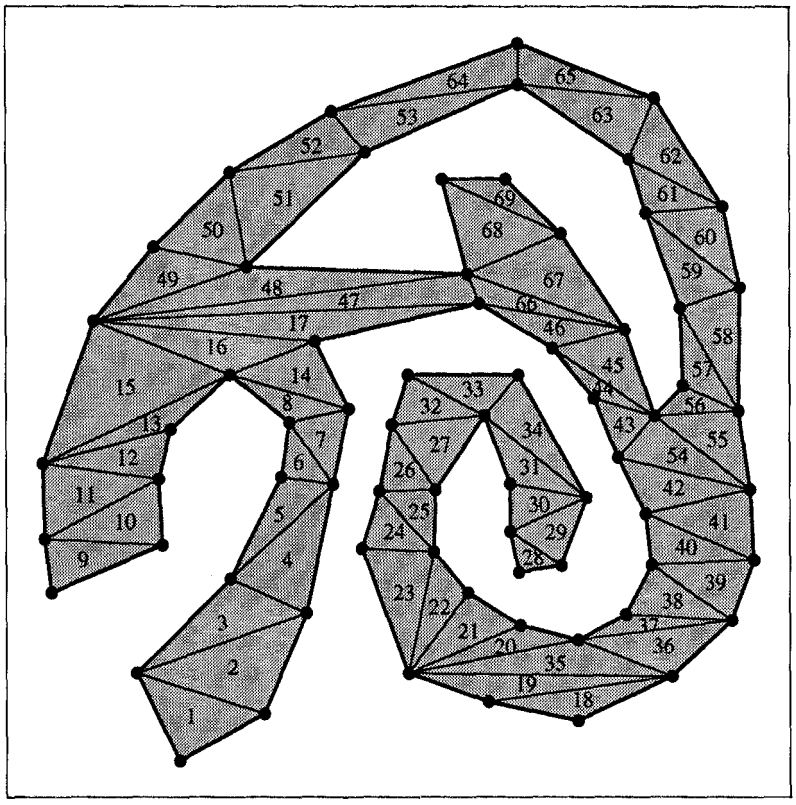


Figure 8: Flooding triangulation example. The numbers indicate the order in which triangles have been removed.

5 Implementation details

The control mechanism for our finite-state implementation of the method is based on the decision tree summarized in figure 9. In order to control numerical stability and handle degenerate cases, the number of geometric routines has been kept at a minimum. Our implementation consists of just three geometric predicates, called *convex, below* and *inside*.

Convex. A predicate $convex(p_1, p_2, p_3)$ is used to decide whether vertex p_2 between p_1 and p_3 is convex. Given the orientation of p_1, p_2 and p_3 in their loop, this solves the question whether triangle (p_1, p_2, p_3) lies inside or outside the polygon.

Below. A predicate $below(p_1, p_2)$ is used to compare the heights of a pair of vertices, with the convention that vertices with equal height use the other coordinate for comparison (this defines a lexicographic ordering of vertices, based on both their X and Y coordinates in the plane; we can still represent this ordering as a *height* if we give an imaginary tilt on all horizontal lines).

Inside. A predicate *inside(S,A,B)* is needed to determine whether a given vertex S (a sag) lies inside the quadrangle formed by the Left and Right pointers and their fingers NextLeft and NextRight (fig. 4). This is obviously a sufficient condition for a sag S to be inside the active gorge. For sags whose heights have been bracketed between the current water level L_1 and the next water level L_2, this will be a necessary condition also.

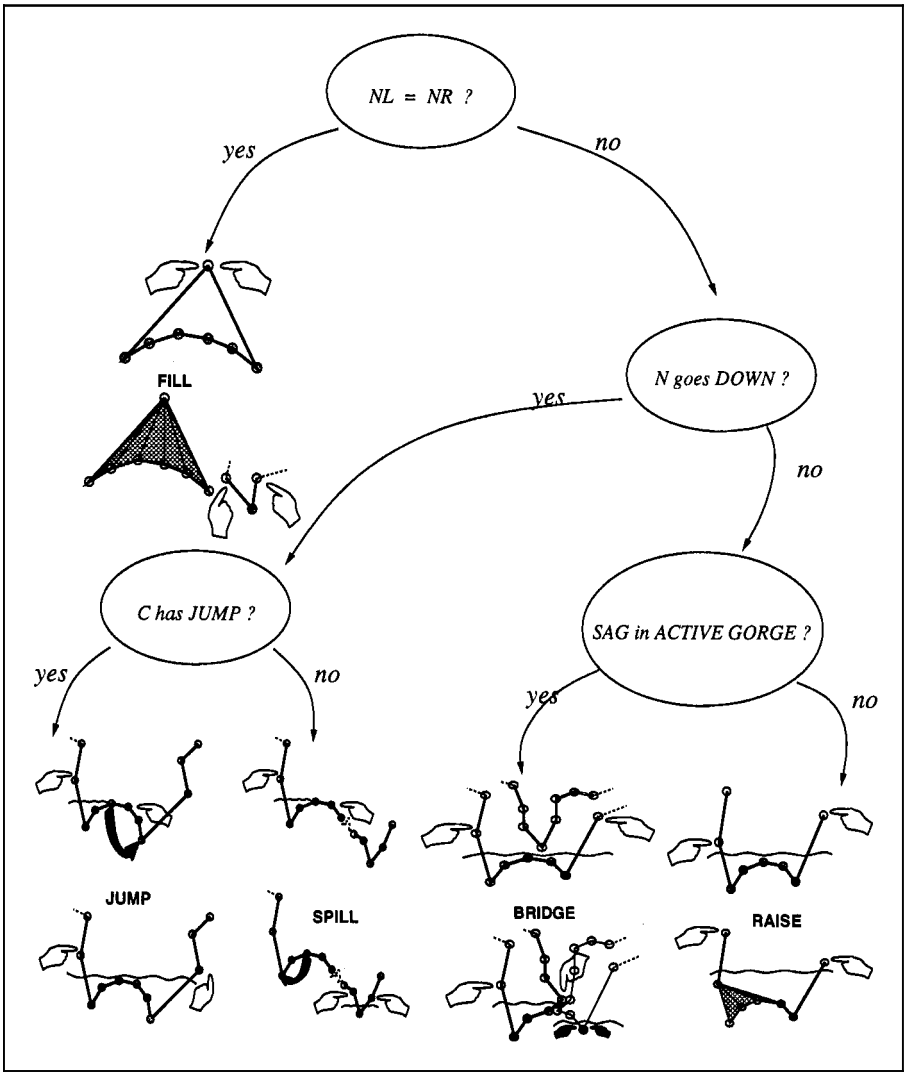


Figure 9: Decision tree for flooding actions. Next is the lowest of NextLeft (NL) and NextRight (NR); C is the corresponding left or right end of chain (Left or Right).

6 Correctness of the algorithm.

Although a formal proof of our algorithm is difficult, we can nevertheless show that if the polygon is valid, then (a) the algorithm produces a valid set of triangles, and (b) the resulting triangulation covers the input polygon completely. In order to prove (a), we first remark that all vertices between the Left and Right pointers (if any) are concave. Only the Left and Right ends of the active chain can ever be convex, which is why we claim that our algorithm generalizes Garey et al. [5]. As a result, the local configuration in the active chain is always similar to a monotonic polygon, which we can triangulate correctly as long as no other edges are contained in it. Because we detect sags, this can never happen, and all triangles are therefore *valid*. Secondly, as we triangulate a polygon, we update the boundary of the non-triangulated region, by *chopping* off triangles as we progress. We now must explain how this process eventually leads to a set of *empty* regions, and thus prove (b). Intuitively, the reason is that there is only one way of closing the active chain, which is to go through *all* vertices in the initial bottom'sloop. It is then easy to prove that when this happens, the remaining region enclosed will be completely triangulated by the final *fill*. If the initial bottom'sloop is monotonic, i.e. without tops nor enclosed sags, this intuition is easily proved, as in [5]. If the loop encloses sags, but has no tops, it will be divided into several components, each of them monotonic, and therefore each of them completely triangulated. The more difficult case comes with *tops* and spilling. The active chain is temporarily disrupted after a *spill*, but this is necessarily followed by a corresponding *jump* from the opposite direction, because each top has exactly two different banks. Although this is quite difficult to prove formally, it intuitively shows that the disrupted chain is always restored, and therefore no vertices are left behind (see [8] for more detail). As a consequence, flooding will produce a valid triangulation when loops are valid. When this is not the case, our finite-state machine implementation provides at least to direct ways to track down errors and degeneracies: forbidden transitions and forbidden states (see fig. 10).

	monotonic	ceiling	sag	top	bottom
monotonic	R,B,F	X	X	X	R,B,F
ceiling	X	X	X	X	X
sag	X	X	X	X	X
top	S,J	X	X	X	S,J
bottom	R,B,F	X	X	S,J	R,B,F

Figure 10: Vertex combinations and states. States are raise (R), fill (F), bridge (B), spill (S), jump (J) and forbidden (X).

Because of its finite state implementation, our algorithm either ends successfully or an error is detected (forbidden state or transition). As an example, consider invalid polygons represented in fig. 11. In 11-A, the Right and Left pointers are identical, and classified as a *sag* because the polygon intersects itself. In 11-B, the Left pointer is classified as a *ceiling*, not as a top, so that spilling should not occur, and an error is detected for the same reason.

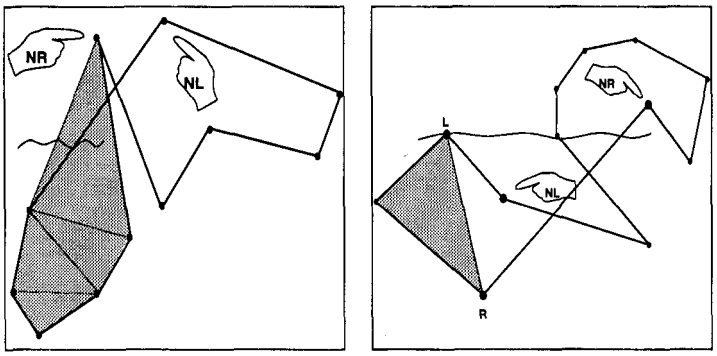


Figure 11: Invalid polygons and error detection. A,B: Invalid polygons.

In practice, we triangulate both the polygon and its complement, for verification purposes. We leave as an open conjecture that this detects all invalid or degenerate polygons.

7 Complexity analysis

Let us first describe worst-case costs for all relevant steps of the algorithm. Identifying bottoms and sags requires two *below* and one *convex* predicates to be evaluated. This is therefore linear in the total number N of vertices. Sorting sags can be done in $S\log(S)$ time, where S is the number of sags. Finding the next sag is linear in the number of sags, because sags are sorted. For each gorge, the worst case is when all sags between the current and next water levels have to be tested, and they are all outside. The *inside* test is then performed a maximum number of SG times in all, where G is the number of gorges. As a conclusion, the asymptotic behaviour of the flooding algorithm can be predicted to be $O(N)+O(SG)$, therefore $O(N^2)$ in the worst case.

In practice, we have observed almost-linear behaviour in many different cases, and an actual worst-case complexity of $O(N^{\frac{3}{2}})$. We base this claim on a series of experiments, described in [8], in which we fitted a log-linear model to the running times T vs. number of vertices N, in a variety of practical cases, i.e. $T = k N^A$ where A is the exponent. Our findings are that the exponent varies between 0.9 and 1.4 in all cases (see [8]) . It should also be noted that the connectivity type (number of holes) does not in itself add to the complexity, because holes are taken into account naturally (as sags) in our framework.

8 Conclusion

We have presented an algorithm for triangulation of a general class of multiply connected polygons, and described a compact implementation of it as a finite state machine. The implementation is remarkably robust because numerical errors generate *impossible* transitions, which can be checked and reported at no cost. The asymptotic behaviour of our approach

depends on the relative number of monotonic and *special* vertices (sags, tops, ceilings and bottoms). If the proportion of special vertices decreases with the total size of the polygon, as is usually the case in practical situations (smooth, faceted surfaces, manufactured objects), the method becomes linear in the total number of vertices. On average, it has been found to be $O(N^{\frac{3}{2}})$, although its theoretical worst-case behaviour can be predicted to be $O(N^2)$. One limitation in our approach is that self-intersecting polygons can only be reported as *invalid input,* but not triangulated. On the other hand, within the class of non-intersecting polygons, we are able to deal efficiently with an arbitrary number of holes, in a very natural way.

9 Acknowledgements

We would like to thank V. Srinivasan and V.T. Rajan for useful comments on this work.

References

[1] B. Chazelle. Triangulating a simple polygon in linear time. *Discrete Computational Geometry,* 6:485–524, 1991.

[2] B. Chazelle and J. Incerpi. Triangulation and shape complexity. *ACM trans. on Graphics,* 3:135–152, 1984.

[3] Lee D.T. and Preparata F.P. Location of a point in a planar subdivision and its applications. *SIAM J. on Computers,* 6:594–606, 1977.

[4] **A.** Fournier and D.Y. Montuno. Triangulating simple polygons and equivalent problems. *ACM trans. on Graphics,* 3(2):153–174, 1984.

[5] M.R. Garey, D.S. Johnson, F.P. Preparata, and R.E. Tarjan. Triangulating a simple polygon. *Information Processing Letters,* 7:175–179, 1978.

[6] S. Hertel and K. Melhorn. Fast triangulation of simple polygons. In *Conference on Foundations of Computer Science Theory,* pages 207–218, New-York, 1983.

[7] J. O'Rourke.*Art gallery theorems and algorithms.* Oxford University Press, New York, 1987.

[8] R. Ronfard and J. Rossignac. Triangulating multiply-connected polygons: A simple, yet efficient algorithm. Technical report, IBM Research, Yorktown Heights, 1994.

[9] S.B. Tor and A.E. Middleditch. Convex decomposition of simple polygons. *ACM trans. on Graphi c* ~3(4):244–265, 1984.

EUROGRAPHICS '96 / J. Rossignac and F. Sillion (Guest Editors), Blackwell Publishers

Volume 15, (1996), Number 3

Full-range approximation of triangulated polyhedra.

Rémi Ronfard [1] and Jarek Rossignac

IBM T.J.Watson Research Center, P.0.Box 704, Yorktown Heights. NY 10598

Abstract

We propose a new algorithm for automatically computing approximations of a given polyhedral object at different levels of details. The application for this algorithm is the display of very complex scenes. where many objects are seen with a range of varying levels of detail. Our approach is similar to the region-merging method used for image segmentation. We iteratively collapse edges, based on a measure of the geometric deviation from the initial shape. When edges are merged in the right order, this strategy produces a continuum of valid approximations of the original object, which can be used for faster rendering at vastly different scales.

Keywords: Visualisation, Polyhedral Surfaces, Levels of Detail, Region Merging.

1 Introduction

This paper describes a new method for automatically generating several levels of details of a polyhedral object, whose faces have been triangulated [9]. Our method is applicable to architectural walk-through, assembly mock-up: virtual reality, medical simulation, planning and monitoring. The difficulty in those applications is that a wide range of viewing conditions must be accommodated, so that 'the available levels-of-details should span several orders of magnitude [5, 12].

There are two sides to all simplification methods. On one side, it is desirable to be able to simply iterate the process until a desired level of detail is reached (in terms of the number of elements in the approximation): without having to guess for global parameters. Thus, incremental methods remove triangles from a 3D object, based on a measure of how much the shape changes locally in each move [15, 13. 16, 14]. On the other side, global control of the simplification error is also useful, and has been emphasized by recent authors [8, 2, 7, 6, 4].

Our method takes an intermediate view of the problem. It is based on local, incremental operations, but keeps track of the initial object, by tracing a *history* of all vertex moves. As a result, we can control approximation levels by prescribing geometric tolerances. and still get the benefits of working incrementally.

The previous work of Rossignac and Borrel [11]is based on space-partionning and clustering techniques in 3D space. By allowing topological changes. that method yield much higher compression ratios than most other previous work, at an extremely low computational cost.

[1]Current address: Institut National de l'Audiovisuel. 4.ave. de l'Europe. 94 366 Bry-sur-Marne, France.

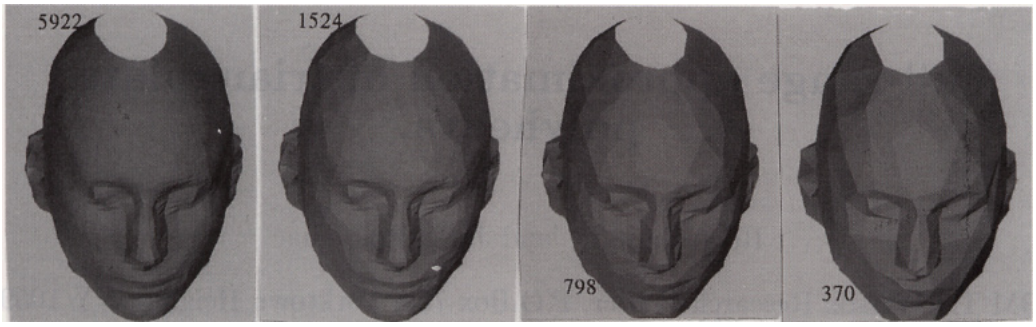


Figure 1: Surface of a human face. extracted from MR image data, with three levels of simplification. and the number of triangles for each level.

but with too little control over the quality of the approximation. In our new method, vertices are merged in a much more controlled way. based on a step by step evaluation of the approximation error.

The multiresolution approach advocated by Eck et al. [3] gives a suitable theoretical framework to surface simplification based on *wavelets* but it is not yet clear that it is practical. because it cannot deal with surface discontinuities or topological simplifications.

The paper is organized as follows. First, we present a general overview of the algorithm in section 2. Then, we discuss our definition of the approximation error, in section 3. Implementation issues are discussed in section 4. with a focus on data structures and topology issues. We end with a discussion of results and performance analysis.

2 Algorithmic foundations of the method.

Our method is based on a merging algorithm, which removes edges one at a time from a polyhedral object. Each edge removal is applied by moving the first, vertex into the second vertex of the edge, as shown in fig. 3. We view this operation as a *region.* merge. because all triangles around the merged vertices are modified. We associate a cost with each merge, which can be precomputed and stored for all edges. The cost function itself will be described in section 3. We also maintain the topological links between vertices, edges and triangles throughout the simplification process. As a general overview, we can outline our method as follows:

Pre-processing. Build the topology of the object, and compute the costs associated with all edges. Insert edges into a priority queue: so the the edge with the lowest, cost is readily available at all times.

Iteration. Merge edges one at a time, by moving their first vertex at the location of the second vertex, ant3 update the topological data structure accordingly.

Relaxation. After each merge, update the geometric location of the remaining vertex, based on its new neighborhood. Update the values of the cost function for all edges that have been affected. and update the priority queue accordingly.

Topology reconstruction. When the number of vertices, or the total approximation error. reach pre-specified levels. intermediate approximations of the object are extracted. and the whole process can be iterated until the full range of approximation levels have been obtained.

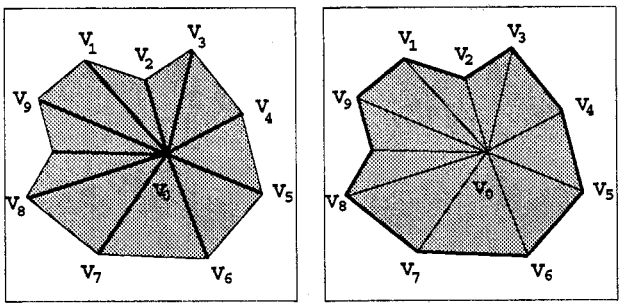


Figure 2: Star and crown of a vertex. A: Edges and faces touching a vertex are its *star.* B: Edges bounding the star of a vertex are its *crown.*

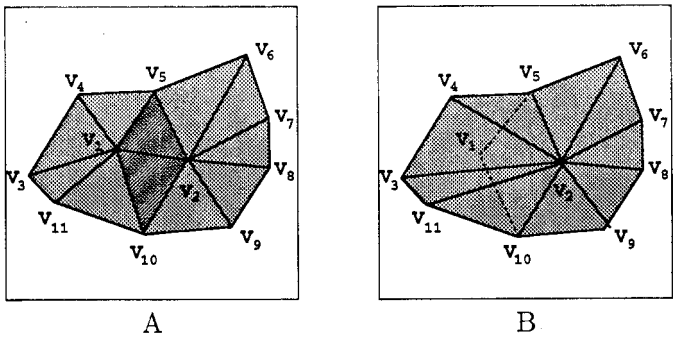


Figure 3: Merging vertices and their stars: A: Initial configuration around edge. B: Final configuration around collapsed vertex.

3 Geometric foundations of the method.

Local tesselation error.

We call the region composed of all edges and triangles around a given vertex its *star.* By definition, the neighborhood of a vertex is composed of edges and triangles in its star, as depicted in fig. 2. In the same figure, the *crown* of a vertex is introduced, consisting of the boundary of its star (i.e. the set of edges adjacent to but not part of. its star.).

Our single topological operation for simplifying polyhedral shapes consists in merging the stars of two adjacent, vertices. Fig. 3 shows a typical example of the region-merging operator, where one vertex V_1 is merged into another vertex V_2.

We evaluate the cost, of merging vertices, with the maximum of two functions of those vertices, LGE and LTE. The local geometric error LGE is the variation of a geometric error

G during the merge (a measure of the distance between the initial arid approximated shapes). The local tesselation error LTE is a penalty function, used to keep the triangular mesh valid and smooth. With reference to fig. 3, the merge involves deleting vertex V_1. edges V_1V_2. V_1V_5 and V_1V_{10}. as well triangles $V_1V_2V_5$ and $V_1V_2V_{10}$. Four more triangles are modified. and a cost LTE is associated with the amount, of rotation undergone by their normal vectors. In addition, vertex V_1 moves at a distance $\|V_1V_2\|$ from its previous location. thus increasing the global geometric error in the approximation by an amount LGE.

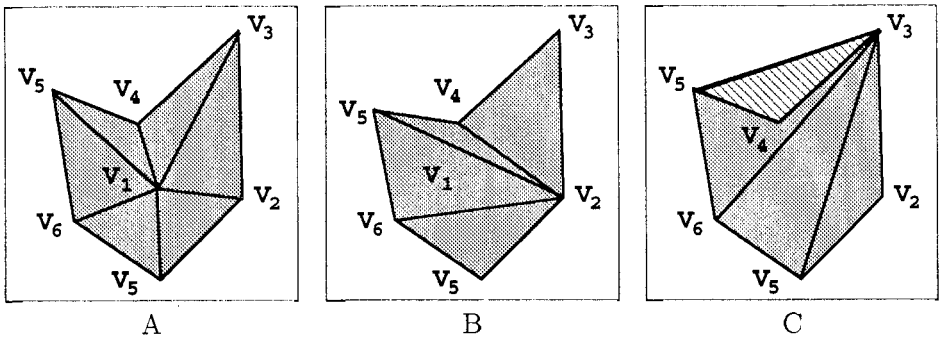


Figure 4: Enforcing validity constraints on a triangular mesh. A: Initial configuration. B: $V_1 \rightarrow V_2$ is an authorized move: all triangle normals keep their orientation. C: $V_1 \rightarrow V_3$ is a forbidden move: one triangle ($V_1 V_4 V_5$)has reversed its orientation.

A triangular mesh defined on a plane or a surface is a valid tesselation when no edges intersect. and no triangles overlap on that plane or surface. In our case, the surface is not explicitly given, but the validity of the mesh can still be approximately evaluated, if we assume that the initial shape is a valid mesh. When we merge vertices, we deform triangles in their star. For each triangle V_i, V_j, V_k. the deformation can be represented by the rotation of the normal vector to the plane of the triangle. and measured with just one angle A_{ijk} between 0 and π.

The validity and the quality of the mesh are violated when a triangle reverses its orientation. i.e. when it turns around the surface of the object with an angle equal to π. In the planar case. an example of this situtation is given in fig. 4. To prevent such cases, and to keep the mesh as balanced and smooth as possible, we impose the penalty function:

$$LTE(V_1, V_2) = K \max_{i,j,k \in star(V_1) i,j,k \notin star(V_2)} A_{ijk}$$

With a large coefficient K, this function can be used to prevent *flipping* triangles tangent to the surface of the object,. if we use a data structure allowing us to pre-compute LTE (without merging). Color plates 7 and 8*show the mesh at different, stages of the simplification process, on two simple examples.

Local geometric error.

We have devised a novel geometric error function, based on the distance of vertices as they move perpendicular to the surface of the initial object,. We use this geometric error as a cost function in combination with the penalty function, which puts constraints on how vertices move tangent to the surface. As a result! all vertices that have been merged with costs inferior to a given tolerance remain within this tolerance of the original shape (see [10]).

* See page C-462 for figures 7 and 8.

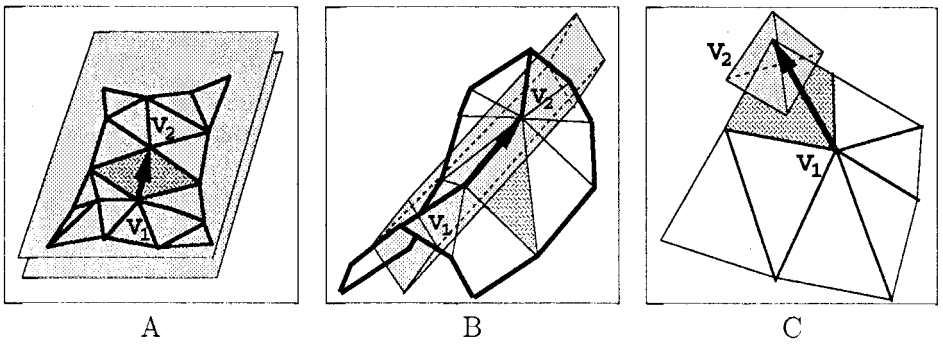


Figure 5: Special cases for simplification. The shaded regions materialize zones with zero, one or two degrees of freedom around vertex V_2. A: Merge almost coplanar vertices, two degrees of freedom. B: Merge vertices aligned on a sharp edge. one degree of freedom. C: Merge vertices into a sharp corner, zero degree of freedom.

Intuitively, the different, ways of merging vertices without changing the shape of the object are (a) to merge vertices whose regions are entirely co-planar: (b) to merge vertices along prominent edges. where the dihedral angle is greatest: (c) to merge vertices into prominent corners (see fig. 5). In order to deal with those three cases simultaneously. we measure the deviation of the merged vertices from their *tangent planes*, i.e. a finite set of planes around each vertex. We start with a simple observation: in the original object,. every vertex is a solution of a linear system of equations, obtained with the plane equations for all the triangles in its star. Let us write this system, with π_k denoting any such plane:

$$\pi_k(V_i) = a_k x_i + b_k y_i + c_k z_i + d_k = 0$$

If we move a vertex V_i to an arbitrary new position $(x.y.z)$, we can associate costs C_k proportional to the distances from those planes to the new position, i.e. $C_k = \pi_k(x, y, z)$.

As a generalization, we associate an arbitrary set of planes, called its *zone*, to each vertex in the object. Initially, those planes are picked from the star of the vertex. with a total cost of zero. If we now move one vertex, there will be a corresponding deviation $\pi_k(V_i)$ for each plane in its zone. If we *merge* vertices, we will also update their zones.

In order to maintain a constraint such as $\pi_k^2(V_i) \leq \epsilon$, we need to confine the vertex V_i in a *slab* around plane π_k. In order to maintain *all* the constraints in its zone, the vertex must be confined in a *region* which is the intersection of all the slabs in the zone. The exact shape of this region depends on the neighborhood of the vertex to be merged. There are three possible cases. In the case of a planar neighborhood, the region is just a single slab with an infinite extentbecause there is only one equation (see fig. 5-A). In the case of fig. 5-B, the region is an infinite tube around a sharp edge, because there are only two different equations in the zone. In most other cases, however, the vertex is really confined in a finite region, because three or more equations are in its zone. This is the case of the corner in fig. 5-C.

We define the geometric error function G around a vertex V_i as the maximum distance from V_i to planes in the zone of V_i. When vertex V_1 is merged into vertex V_2. the new region inherits plane equations from the zones of the two merged vertices. The local geometric error LGE is the variation of the error, therefore $G^{after} = \max\{G^{before}, LGE(V_1, V_2)\}$. Because we merge vertices by rank of increasing geometric errors: the maximum error after the merge is necessarily one of the new constraints in $zone(V_1)$. Therefore, an equivalent definition for LGE is simply:

$$LGE(V_1, V_2) = \max_{j \in zone(V_1)} \pi_j^2(V_2)$$

In other words. LGE is the largest new constraint associated with the vertex being merged. Note that we always choose the final vertex position to be V_2 itself, i.e. we only use existing vertices at this point. Using a combination of LGE and LTE as a cost function. we can merge edges with increasing costs, first, in the smoother regions of the object,. then along sharp edges. and finally into corners in the object (see examples showing the triangular mesh in fig. 7 and fig. 8 in the color plates).

4 Implementation details.

Vertex-based data structure.

We have devised a vertex-based data structure with easy access to edges around a local center. enabling us to compute the local cost functions as efficiently as possible. The local tesselation error LTE involves edges in the crown of a vertex. The local geometric error LGE involves edges in its star, as well as equations in its zone.

We therefore represent each vertex as a triplet: its star, its crown and its zone. The star contains a list of edges incident to the vertex. which means that we maintain a complete representation of the graph of vertices and edges. In addition, the crown specifies an imbedding of that graph on a particular surface: each edge in the crown corresponds to one triangle on the surface. The zone contains the history of the approximation, and relates its vertices to their ancestors in the original object. The construction of this dynamic data structure is quite straight-forward. and is described in much more detail in our researach report [10].

Edge ordering based on local errors.

In order to merge edges with increasing costs in the right order. we maintain an auxillary data structure, in the form of a heap of directed edges. We associate a priority to each directed edge, which is just the opposite of its cost function. We use the heap as a priority queue, such that (a) the first element in the queue always has the highest priority, and (b) we can efficiently remove and update edges when their priority changes. The heap is a particularly efficient implementation of the priority queue, as a balanced. partially ordered tree [1]. The heap is easily updated when an edge priority changes. by *bubbling* up (if the priority increases) or down (if the priority decreases). The heap is initialized by reading the triangulation data sequentially. Its elements are updated as we merge edges, as we will now explain in more details.

In each move, we update the local of the dynamic graph structure around the merged vertex, as well as the heap, and the position (coordinates) of the merged vertex. The heap updating operations consist in removing some edges (two opposite directed edges at a time), and reordering the heap based on new priorities, for those edges that have been modified but not deleted. Edges pointing from V_2 must be recomputed, because of the additional constraints inherited from V_1 . Edges pointing toward V_2 need only be recomputed if we also update the position of V_2 at. this point.

Relaxation.

After each merge, it is possible to optimize the coordinates of the central vertex. based on the (fixed) coordinates of its new neighbors. The general idea is to minimize an energy function

that will make the triangular mesh as smooth and balanced as possible, while keeping the geometric errors as small as possible.

An interesting choice for the relaxation energy is taken directly from the geometric error function. except that we now use the sum of errors. rather than the maximum error. i.e.

$$E_R(V_2) = \sum_{j \in zone(V_1)} \pi_j^2(V_2)$$

Deforming the object locally by moving V_2 into its *minimum* local error, enhances the method at all scales. We must acknowledge that, a more elegant solution would incorporate that optimization into the estimate of the merging costs. i.e. we should use E_R as our local geometric error, and the optimized value of V_2 in our cost function. But, this would be terribly inefficient. and we prefer to apply those two steps separately. As a heuristic, we have found that the two cost functions worked together quite nicely. because the optimized value for V_2 was never very far from its initial position. and the prediction used for LTE and LGE remained correct.

Local topology changes.

Collapsing an edge (V_1, V_2) can change the topology of the object. For instance. a through-hole may become filled-up (in topological terms, *handles* may be removed. e.g. a torus may be transformed into a sphere). Vertices sharing an edge with both V_1 and V_2 play a very important role in this step of our algorithm. Their number characterizes the local topological structure of the object, (zero at a wireframe edge. one at the border of an open surface, two inside a closed surface, three or more on a surface with self-intersections). Our vertex-based data structure allows all those cases to be represented: which is a key to changing the topological genus of the shape, i.e. removing holes, handles, and separating components [2, 7, 6]. Our dynamic data structure can tolerate such changes, because it is based on vertices, and vertices can be incident to arbitrary numbers of edges and triangles. In many cases, topological simplification is in fact acceptable, and we have used it to achieve the higher compression ratios in most of the examples shown here.

While we are able to correctly reconstruct approximations in presence of topological changes, we do not know how to re-organize constraints. For instance, if several components are separated, which constraints should be associated wth each component, ? In those cases. an appropriate enhancement of our method would consist in resolving the new structure, so that the constraints for each of several new components could be separated, and the constraints which came from filled-up holes could be eliminated. This appears to be a difficult issue in general, and is not a part of our current implementation. As a result, we keep too many constraints in those cases, and the approximation process artificially slows down in those regions.

5 Results and comments.

As an input, the algorithm is given a number L of levels and a geometric tolerance for each level (ϵ_1 , ϵ_2 ,..., ϵ_L). Each tolerance can be expressed as a percentage of the object size. or as a multiple of the initial costs found during the pre-computing step. As an alternative, the numbers of triangles in each level may be pre-specified ($N_1 = N_2$, ... , N_L).

A typical example is shown fig. 6 (top). with a maximum compression ratio of 1:80, leading to just under a hundred triangles. Subtle topological changes occur between the second and third approximation levels; holes disappear, antennas retract. eventually leading to the almost minimal representation of the fourth level of the figure.

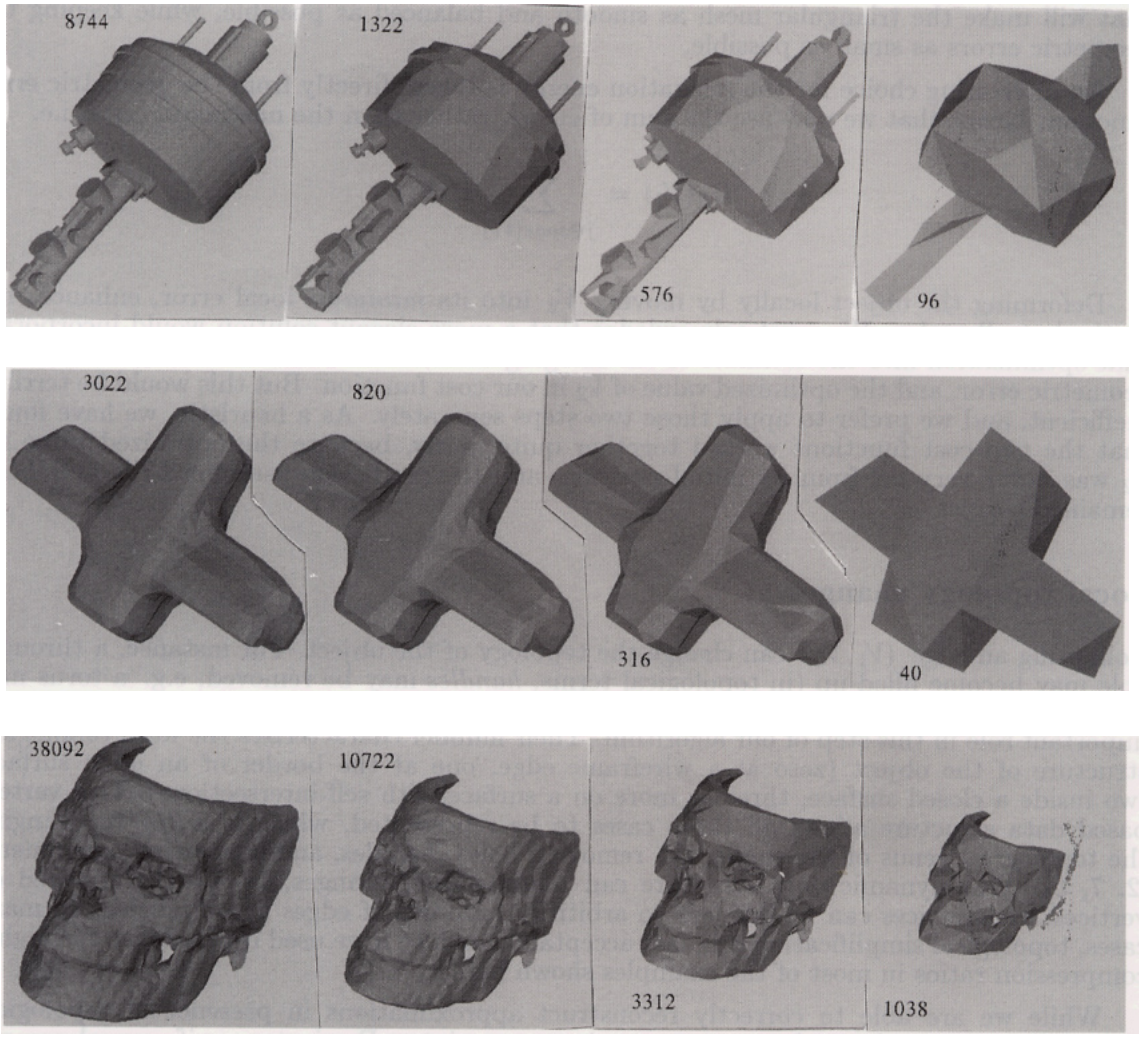


Figure 6: Top: Power brake assembly. Middle: Radiation iso-dose surface. Bottom: Skull surface extracted from X-ray scanner image data. Three levels of approximation for each example, and corresponding numbers of triangles.

We also show results obtained with iso-surface data (courtesy of G. Turk [15]) in fig. 6 (middle). Our results are comparable to previously published work, only with a much wider range of approximation levels. Examples in fig. 1 and 6 (bottom) are applications of our method to medical data. Iso-surface extraction methods of that sort typically generate results with hundreds of thousands of polygons, and efficient data reduction methods are therefore particularly important for dynamic simulations. surgery planning, and other related applications. Although our approach is heuristically based on only a partial measure of the approximation error. we are able to control the tolerances at each level. so that no vertices move further away from their initial configuration than 0.1. 0.5 and 1.0 per cent of the image size. As can be seen in fig. 6 (bottom), very good approximations can be obtained at intermediate compression ratios (1:10 to 1:20), while much coarser approximation result at further stages. The initial shapes were extracted from MR and CAT-scan data as iso-surfaces

(courtesy of A. Guéziec).

We estimate that the complexity of the method is $N_0 \log^2 \frac{N_0}{N_L}$ for bringing the number of vertices down from N_0 to N_L triangles. In our analysis. we consider the heap operations, which are approximately logarithmic, and the tests on all the equations in the vertex zones. Because equations are inherited. their number is constant thoughout the simplification process. hence the result. In practice, the computing times for all examples run from a few seconds to several minutes on a workstation.

6 Conclusion.

We have presented a new method for simplifying three-dimensional shapes, based on an particular measure of the approximation error, that we derive from simple plane equations. Our approach allows us to reach very high compression ratios in a variety of cases, while keeping the general appearance of the original shapes. It has been noted by other researchers that an important part of obtaining minimal approximations of shapes was the elimination of small features. No previous method has addressed that issue successfully. Our method shows that feature elimination can be achieved. partly at least: with purely geometric reasonning (i.e., without any detection or even understanding of the eliminated features), if we allow the topology ofthe shape to be modified.

References

[1] A.V. Aho and J.D. Ullman. *Foundations of Computer Sciemce.* Computer Science Press, 1992.

[2] T. DeRose, H. Hoppe, T. Duchamp, W. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. *Computer Graphics (Proc. SIGGRAPH)*, pages 71–78. 1992.

[3] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsberry, and W. Stuetzle. Multiresolution analysis of arbitrary meshes. *Computer Graphics (Proc. SIGGRAPH)*, pages 173–182, 1995.

[4] A. Guéziec. Surface simplification with variable tolerance. In *Second Annual Symposium on Medical Robotics and Computer Assisted Surgery*, 1995.

[5] P.S. Heckbert and M. Garland. Multiresolution modeling for fast rendering. *proc. Graphic Interface*, pages 43–50. 1994.

[6] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, W. McDonald, J. Schweitzer, and W. Stuetzle. Piecewise smooth surface reconstruction. *Computer Graphics (Proc. SIGGRAPH)*, pages 295-302. 1994.

[7] H. Hoppe, T. DeRose, T. Duchamp, W. McDonald, and W. Stuetzle. Mesh optimization. *Computer Graphics (Proc. SIGGRAPH)*, pages 19–26. 1993.

[8] T. Phillips, R. Cannon, and A. Rosenfeld. Decomposition and approximation of three-dimensional solids. *Computer Vision, Graphics, and Image Processing*, 33:307–317, 1986.

[9] R. Ronfard and J. R. Rossignac. Triangulating multiply-connected polygons: A simple yet efficient algorithm. In *Computer Graphics Forum, proceedings of Eurographics '94.* Oslo, Norway, September 1994.

[10] R. Ronfard and J. R. Rossignac. Full-range approximation of triangulated polyhedra. Technical Report 20423, IBM T.J. Watson Research Center. Yorktown Heights, New York, 1996.

[11] J. Rossignac and P. Borrel Multi-resolution 3d approximations for rendering complex scenes. *Modeling in Computer Graphics.* pages 455–465. 1993.

[12] J.R. Rossignac and M. Novak, Research issues in model-based visualization of complex data-sets. *IEEE Computer Graphics and Applications*, pages 83–35, March 1994.

[13] L. Schroeder, J.A. Zarge, and W.E. Lorensen. Decimation of triangle meshes. *Computer Graphics,* 26(2):65-69, 1992.

[14] W. Schroeder, K. Martin, and B. Lorensen. *The visualisation toolkit, an object-oriented approach to 3D graphics.* Prentice Hall. 1985.

[15] G. Turk. Re-tiling polygonal surfaces. *Computer Graphics,* 26(2):53-64, 1992.

[16] A. Varshney. *Hierarchical geometric approximation.* PhD thesis, University of North Carolina. Chapel Hill, 1994.

Short Papers

Implicit Simplicial Models for Adaptive Curve Reconstruction

Gabriel Taubin and Remi Ronfard

Abstract—Parametric deformable models have been extensively and very successfully used for reconstructing free-form curves and surfaces, and for tracking nonrigid deformations, but they require previous knowledge of the topological type of the data, and good initial curve or surface estimates. With deformable models, it is also computationally expensive to check for and to prevent self-intersections while tracking deformations. The *Implicit Simplicial Models* that we introduce in this paper are implicit curves and surfaces defined by piece-wise linear functions. This representation allows for local deformations, control of the topological type, and prevention of self-intersections during deformations. As a first application, we also describe in this paper an algorithm for two-dimensional curve reconstruction from unorganized sets of data points. The topology, the number of connected components, and the geometry of the data are all estimated using an adaptive space subdivision approach. The main four components of the algorithm are topology estimation, curve fitting, adaptive space subdivision, and mesh relaxation.

Index Terms—Curve fitting, topology estimation, shape recovery, geometric modeling.

1 Introduction

The reconstruction of curves and surfaces from unorganized sets of data points is an important problem in computer vision. Curves and surfaces can be represented parametrically or implicitly, and depending on the final application, one representation is more suitable than the other. Since parametric curves and surfaces, such as splines [1], allow for a high degree of local control, they are very good for modeling free-form objects, but the topological type of these curves and surfaces is determined by the topology of the domain, and it is difficult to check for, and to prevent self-intersections. The now popular deformable models [4], [6], [13] are all parametric. There has been some recent work on reconstructing surfaces of unknown topology [3], [8], but no new representation is introduced to control the topology and to prevent self-intersections during deformations.

Arbitrary topology can be achieved with implicit curves and surfaces. The *Implicit Simplicial Models* that we introduce in this paper are polygonal curves and polyhedral surfaces not represented as lists of vertices and planar faces, but defined implicitly by piece-wise linear functions. This representation allows for local deformations, control of the topological type, and prevention of self-intersections during deformations. A piece-wise linear function is determined by a simplicial tessellation of its domain, and by the values of the function at the vertices of the mesh. The function is linear in each one of the domain cells. The usual representation of a polygonal curve or polyhedral surface as a list of vertices and a list of flat faces can be recovered in time proportional to the number of faces. Irregular meshes allow for adaptive reconstruction algorithms, and for hierarchies of curves and surface approximations of different resolutions.

While the *topology* of an implicit simplicial model is determined by the combinatorial structure of the domain mesh, and by the signs of the values of the piece-wise linear function at the vertices of the mesh, the *geometry* is determined by the magnitudes of the values of the piece-wise linear function at the vertices of the mesh. Small scale constant topology deformations can be achieved by changing the magnitudes of the implicit function at the vertices of the mesh keeping the signs fixed. Large scale constant topology deformations are obtained by also deforming the underlying mesh without inverting the orientation of the cells.

This paper builds upon previous work on algebraic curve and surface fitting [9], [10], [11], [12], and some ideas from [7], where an algorithm for adaptively reconstructing piece-wise algebraic curves and surfaces defined on triangular and tetrahedral meshes is described.

While the reconstruction algorithm is applied only to two-dimensional curves here, the representation is valid in any dimension, and the general structure of the reconstruction algorithm can be generalized to higher dimension as well. However, since results were not available at the time this paper was written, we will discuss how to extend the reconstruction algorithm of this paper to surfaces, and how to track deformations, in future reports.

2 Implicit Simplicial Curves

The data structure commonly used to represent a simplicial curve is a pair of lists, a list of vertices, and a list of straight line segments. This representation is good for rendering the curve in time proportional to the number of segments, but it is not very good for checking for, least for imposing, topological or geometric constraints. Even if we start with a valid simplicial curve, deformations can introduce self-intersections, and there is no simple way to check for them. A straightforward check for self-intersections requires time proportional to the square of the number of segments. The main problem with this representation is that the conditions for a set of line segments to define a valid simplicial curve are global.

An implicit simplicial curve is the set of zeros of a piece-wise linear function of two variables, which is defined by a planar triangular mesh and the values of the function at the vertices of the mesh. Thus, we will represent an implicit simplicial curve as a set of three lists $C = \{V, T, F\}$. A list of vertices $V = \{v_1, \ldots, v_{n_V}\}$, a list of triangles $T = \{t_1, \ldots, t_{n_T}\}$, and a list of function values at the vertices of the mesh $F = \{F_1, \ldots, F_{n_V}\}$. Since we will also need later on an explicit representation for the edges of the mesh, we will write $\Sigma = \{V, E, T\}$ for the domain mesh, where $E = \{e_1, \ldots, e_{n_E}\}$ is the list of edges. The piece-wise linear function defining the implicit simplicial curve is

$$f = \sum_{i=1}^{n_V} F_i x_i = FX, \tag{2.1}$$

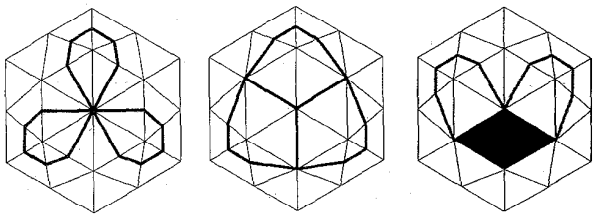
where F is seen as a row vector, $X = \{x_1, \ldots, x_{n_V}\}^t$, and x_i is the unique piece-wise linear function subordinated to the mesh Σ

• G. Taubin is with IBM T.J.Watson Research Center, P.O.Box 704, Yorktown Heights, NY 10598. E-mail: taubin@watson.ibm.com.
• R. Ronfard is with Institut National de l'Audiovisuel, 94366 Bry sur Marne Cedex, France. He was at the IBM T.J.Watson Research Center when this work was performed. E-mail: remi@ina.fr.

Manuscript received June 1, 1993; revised May 30, 1995. Recommended for acceptance by Eklundh.
For information on obtaining reprints of this article, please send e-mail to: transactions@computer.org, and reference IEEECS Log Number P95153.

which satisfies the following equation

$$x_i(v_j) = \begin{cases} 1 & \text{if } j = i \\ 0 & \text{if } j \neq i \end{cases}.$$



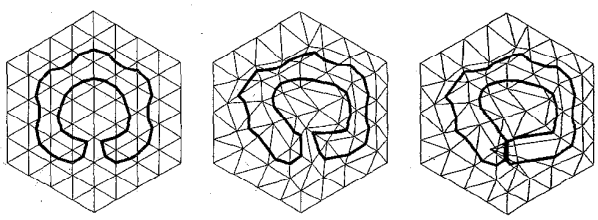
(a) (b) (c)

Fig. 1. Singular cases. (a): function is zero at one vertex. (b): Function is zero on three edges. (c): Function is zero on two neighboring triangles.

To prevent singular cases, such as those shown in Fig. 1, we will constrain the values of the piece-wise linear function at the vertices of the mesh to be non-zero. We will also require the domain mesh Σ to be positively oriented. A mesh $\Sigma = \{V, E, T\}$ is positively oriented if all the determinants

$$|V_t| = \begin{vmatrix} 1 & 1 & 1 \\ v_{i,1} & v_{j,1} & v_{k,1} \\ v_{i,2} & v_{j,2} & v_{k,2} \end{vmatrix}$$

associated with the triangles $t = \{v_i, v_j, v_k\}$ of the mesh, are positive, where $v_{i,1}$, $v_{i,2}$ are the coordinates of the vertex v_i of the mesh. Note that if the order of the vertices is interchanged in a triangle $t = \{v_j, v_i, v_k\}$ the sign of the determinant $|V_t|$ changes.



(a) (b) (c)

Fig. 2. Topology preserving deformations. (a): Original implicit simplicial curve. (b): Deformed mesh with constant external boundary. (c): Invalid mesh deformation can produce self-intersections.

Topology preserving deformations of a simplicial implicit curve are obtained automatically by deforming the domain mesh maintaining its orientation, *warping* the space around the curve, and eventually changing the magnitudes of the function values at the vertices of the mesh, but not their signs. If a mesh is deformed preserving its orientation, the outside boundary of the mesh might change shape, but preserves its topology. In our reconstruction algorithms we will impose a stronger constraint. We will keep the outside boundary of the mesh constant.

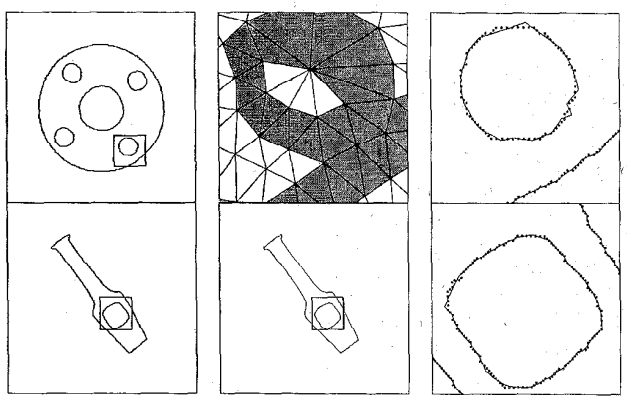
Once the mesh Σ is fixed, the *topology* of an implicit simplicial curve is fully determined by the *signs* of the piece-wise linear function f at the vertices of the mesh, $\{\sigma_1, \ldots, \sigma_{n_V}\}$, $\sigma_i = sign(F_i) \in \{-1, 1\}$, while the *geometry* of the curve is determined by the *magnitudes* of the same function values $\{|F_1|, \ldots, |F_{n_V}|\}$.

3 An Algorithm for Curve Reconstruction

In this section we describe an algorithm to reconstruct an implicit simplicial curve from an unorganized set of points in the plane $D = \{p_1, \ldots, p_{n_D}\}$. The only assumption is that the data points belong to, or are close to, a non-singular curve (without self-intersections), and are roghly uniformly distributed along the curve. The topology and the number of connected components are unknown in advance, and estimated by the algorithm. The result is a triangular mesh covering a neighborhood of the data set, and a regular piece-wise linear function represented by its values at the vertices of the mesh. Fig. 3 shows the global structure of the algorithm, and Fig. 4 shows typical examples of curves reconstructed with this algorithm.

```
procedure FitImplicitSimplicialCurve
  InitializeMesh
  for level ← 0 to max-level step 1 do
    EstimateTopology
    EstimateGeometry
    if MaximumFittingError < ϵ
      return
    else
      AdaptivelySubdivideMesh
      RelaxMesh
```

Fig. 3. Global structure of the implicit simplicial curve reconstruction algorithm.



(a) (b) (c)

Fig. 4. Multiply connected objects and simplicial curve reconstruction of boundaries, using the algorithm of Section 3—(a): data set; (b): reconstructed implicit simplicial curve; (c): details of a and b.

The algorithm follows a strict top-down approach, minimizing the amount of storage required. A very simple mesh is used at the beginning of the algorithm covering a region containing all the data points. In our experiments, we have used triangulated squares and regular hexagons as initial meshes, and in general, the hexagonal geometries produce better results. However, the initial mesh should be tailored to the application. Once a mesh is fixed, an implicit simplicial curve subordinated to the current mesh is reconstructed from the data by a least squares fitting algorithm, by first estimating the topology, and then the geometry of the curve. After the curve fitting step, the maximum fitting error within each triangle is measured. If all the triangles meet the prespecified tolerance, the algorithm finishes. Otherwise, the triangles where the tolerance is not met, are subdivided, along with a few other that are necessary to maintain a valid mesh. The mesh is then relaxed

to prevent the vertices of the subdivided mesh to be too close to the data, and to improve the aspect ratio of the triangles. This step is essential for the success of the algorithm. Once this new mesh is fixed, the loop is traversed again, until the tolerance test is satisfied by all triangles.

Now we proceed to describe in detail the main building blocks of the curve reconstruction algorithm: implicit curve fitting, topology estimation, testing, adaptive subdivision, and mesh relaxation.

3.1 Implicit Curve Fitting

We formulate this step of the algorithm building upon previous work on algebraic curve and surface fitting [9], [10], [11], [12]. Given a finite set of two dimensional data points $D = \{p_1, \ldots, p_{n_D}\}$, we cast the problem of fitting an implicit curve $Z(f) = \{p: f(p) = 0\}$ to the data set D as globally minimizing the mean square distance from the data points to the curve $Z(f)$, as a function of the vector of parameters F, the values of the piecewise linear function on the vertices of the mesh. For a piece-wise linear function, the mean square distance has the following explicit expression

$$\Delta_1(F) = \sum_{t\in T} \frac{n_{D_t}}{n_D} \sum_{p\in D_t} \frac{f_t(p)^2}{\|\nabla f_t(p)\|^2} = \sum_{t\in T} \frac{n_{D_t}}{n_D} \frac{F_t M_t F_t^t}{F_t N_t F_t^t}, \tag{3.1}$$

where D_t is the subset of data points that belong to the triangle $t = (v_i, v_j, v_k)$, n_{D_t} is the number of points in D_t, $f_t(p) = F_i x_i(p) + F_j x_j(p) + F_k x_k(p)$ is the restriction of $f(p)$ to the triangle t, $F_t = (F_i, F_j, F_k)$, $X_t = (x_i, x_j, x_k)$,

$$M_t = \frac{1}{n_{D_t}} \sum_{p\in D_t} \left[X_t(p) X_t(p)^t\right], N_t = \frac{1}{n_{D_t}} \sum_{p\in D_t} \left[DX_t(p) DX_t(p)^t\right], \tag{3.2}$$

and DX_t is the Jacobian of X_t

In fact, since X_t is a linear function, its Jacobian is constant, and the matrix N_t is only a function of the vertices of the triangle, and not a function of the data points inside it.

3.2 Topology Estimation

In the absence of prior knowledge about the solution, it is difficult to minimize (3.1), because the system to be solved becomes singular when a nodal value F_i approaches zero. On the other hand, if we can estimate the signs of the nodal values, i.e., the topology of the solution, we can minimize (3.1) locally, with a program such as LBFGS [5], which is designed for large unconstrained nonlinear minimization.

The piece-wise linear function determined by the coefficients F_i constitutes an approximate *inside-outside* function for the data (up to a global sign inversion). When an inside-outside function is directly available from the data, the topology estimation step is therefore not necessary. In all other cases, we can still estimate the inside-outside function, with combinatorial optimization methods. We do this by independently fitting straight lines in all non-empty triangles, and *counting sign changes*. More specifically, for each triangle $t = \{v_i, v_j, v_k\}$, i.e., such that the set D_t is not empty, we fit a straight line to the data set D_t in the least squares sense by minimizing the local mean square error

$$\frac{L_t M_t L_t^t}{L_t N_t L_t^t},$$

where M_t and N_t are the matrices of (3.2), obtaining its minimum value, the *local error of fit* ϵ_t, and its minimizer $L_t = (L_{t,i}, L_{t,j}, L_{t,k})$. This is done for each triangle independently of the data in other triangles. This can be done in closed form and involves solving a 2×2 eigenvalue problem. In general, the three components of L_t are nonzero. Let us denote by $\sigma_{t,i}, \sigma_{t,j}, \sigma_{t,k}$ the signs of $L_{t,i}, L_{t,j}, L_{t,k}$, i.e., $\sigma_{t,i}$ is equal to 1 or -1 depending on whether $L_{t,i}$ is positive or negative.

A good estimate for the signs of the global coefficients F_i can then be obtained by minimizing a sum over all triangles:

$$\Delta_2(F) = \frac{1}{n_T} \sum_{t\in T, t=\{v_i,v_j,v_k\}} \left(\sigma_{t,i}\sigma_{t,j}F_iF_j + \sigma_{t,i}\sigma_{t,k}F_iF_k + \sigma_{t,j}\sigma_{t,k}F_jF_k\right)\left(\frac{\epsilon_{\max} - \epsilon_t}{\epsilon_{\max} - \epsilon_{\min}}\right) \tag{3.3}$$

constraining the function values at the vertices of the mesh to be either -1 or 1

$$F_1, \ldots, F_{n_V} \in \{-1, 1\}. \tag{3.4}$$

Expression (3.3) involves a measure of the *goodness-of-fit* for each triangle, where $\epsilon_{\max} = \max\{\epsilon_{t_1}, \ldots, \epsilon_{t_{n_T}}\}$, and a similar definition for $\epsilon_{\min}$. Thus, the global coefficients F_i change signs only when there is enough evidence from the local fits in their neighborhood. Of course, we also have to define a goodness of fit and a fitting vector L_t for each empty triangle, because otherwise the problem could be underconstrained. For an empty triangle t we set $L_t = \{1, 1, 1\}$ with a high confidence value $e_t = e_{empty}$. By rearranging terms, we obtain

$$\Delta_2(F) = \sum_{e\in E} H_e F_i F_j \tag{3.5}$$

with the sum ranging over all the edges $e = \{v_i, v_j\}$ of the triangulation. The edge weight H_e corresponding to the edge $e=\{v_i,v_j\}$ is easily obtained from (3.3)

$$H_e = -\sum_{t:e\in t} \sigma_{t,i}\sigma_{t,j}\left(\frac{\epsilon_{\max} - \epsilon_t}{\epsilon_{\max} - \epsilon_{\min}}\right) \tag{3.6}$$

with the sum extended over all the triangles that contain e as an edge, The minimization of the quadratic expression (3.5) in $\{-1, 1\}$ is exactly the Ising model, for which simulated annealing schemes are well documented. We have found that simulated annealing based on (3.5) gives good results. This approximation is also faster, and more robust, than the more obvious choice of using (3.1) directly at this stage. Some results of the topology estimation step are presented in Figs. 5 and 6.

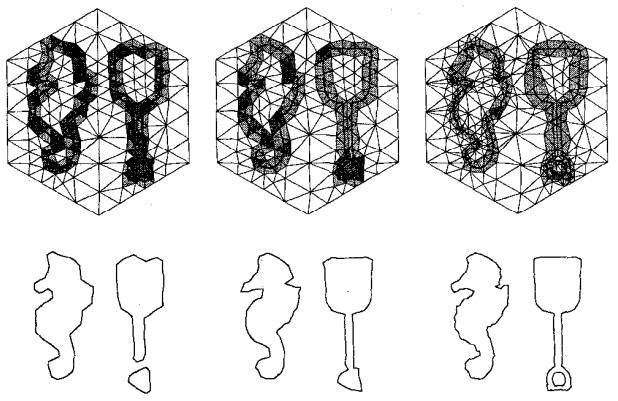


Fig. 5. Example of topology estimation step.

3.3 Testing

Testing the quality of the fit is necessary to determine when the algorithm should stop, and otherwise, which triangles need to be subdivided. After the minimization of (3.1) for the current mesh, for each triangle t we compute

$$\delta_t = \max_{p \in D_t} \frac{f_t(p)^2}{\|\nabla f_t(p)\|^2}.$$

If $\delta_t > \delta$ we mark the triangle for subdivision. If no triangle is marked, the algorithm stops. Otherwise we continue. In practice, this test is generally sufficient, although problems occur when the number of data points inside a triangle becomes too small, usually around high curvature regions, or in cases of sparse or very noisy data points.

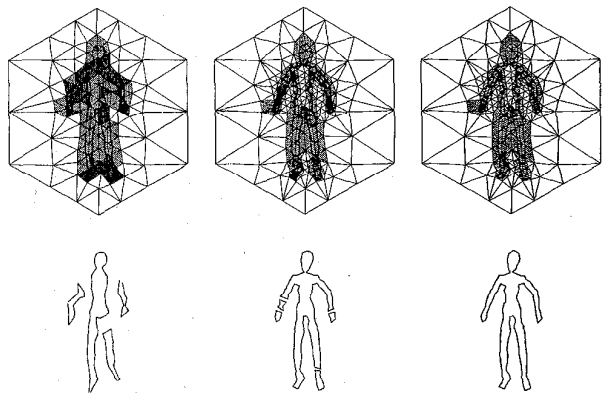
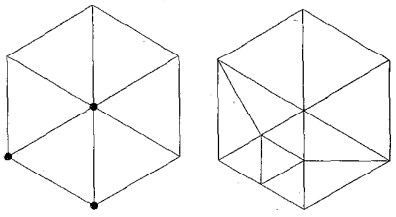


Fig. 6. Example of topology estimation step.

3.4 Adaptive Subdivision

At this point we have a valid triangular mesh with some triangles marked for subdivision. It turns out that to maintain a valid mesh, other unmarked triangles may have to be subdivided as well. A simple method to automate the process involves three steps [2], [14]. In the first step the vertices of each triangle marked for subdivision are marked. In the second step a new vertex is created at the midpoint of each edge which has the two vertices marked. In the third step, illustrated in Fig. 7, the triangles are subdivided according to how many vertices are marked. A triangle with the three vertices marked is subdivided into four triangles, a triangle with two vertices marked is subdivided into two triangles, and triangles with one or no marked vertices are not subdivided.



(a) (b)

Fig. 7. Subdivision rules. Marked vertices are represented with a circle. (a): Mesh with some vertices marked. (b): Mesh a after subdivision.

3.5 Mesh Relaxation

Since the desired final result is a regular implicit simplicial curve, we cannot allow the function to be close to zero at any vertex of the triangulation. If a vertex is allowed to get close to the data points, the function value at that vertex will tend to be small with respect to the values a the other vertices, or even zero. Contrary to what is done in other related algorithms based on triangular meshes, where the mesh is relaxed by pulling the vertices close to the data points [7], our mesh relaxation process *pushes* the vertices *away from the data*. It is not only that we want the vertices to be far away from the data, but we also want the data inside each triangle to be well approximated by a straight line connecting the midpoints of two edges. At least this is approximately what happens when the algorithm stops, but we would like to try to impose this condition at each level of subdivision. So, the data has to pull the vertices away, but at the same time, for each triangle the distance from the three vertices to the data must be as equal as possible. Also, the mesh has to remain a valid mesh, and the triangles have to remain as equilateral as possible, because otherwise, the local nonlinear minimization algorithm gets plagued with all sort of numerical problems.

We have decided to base our mesh relaxation algorithm on an energy minimization scheme, and instead of solving ordinary differential equations, we perform an approximate minimization based on gradient descent.

The mesh energy $U = \kappa_D U_D + \kappa_E U_E$ has two components, the *data energy* U_D, and the *edge energy* U_E. The two constants κ_D and κ_E must be positive. In our current implementation $\kappa_E = 0.25$ and $\kappa_D = 1.0$.

The data energy pushes the vertices as far away and equidistantly as possible from data points in each triangle. The edge energy pulls vertices together and tends to make equilateral triangles, which regularizes the mesh relaxation process. In addition, we constrain the boundary of the mesh to remain constant. This can be achieved by fixing the boundary vertices in their initial positions, and allowing the vertices laying on the boundary edges to move only along the edges they belong to. All of this can be done with linear constraints on some of the vertices. If a vertex v of the mesh belongs to a boundary edge e, then v must satisfy the linear equation of the line containing the edge $C_e(v) = 0$. Since a boundary vertex belongs to two non-parallel edges, making it satisfy the two constraints is equivalent to keeping it fixed.

The data energy is defined as follows

$$U_D = \rho^2 \sum_{e \in E} \left(\phi(v_i) - \phi(v_j)\right)^2, \tag{3.7}$$

with the inner sum extended over all the edges $e = \{v_i, v_j\}$, and where

$$\phi(v_i) = \frac{1}{n_D} \sum_{t: v_i \in t} \sum_{p_j \in D_t} \frac{1}{\|v_i - p_j\|}, \tag{3.8}$$

with the sum extended over all the triangles that contain v_i as a vertex, and all the data points inside these triangles. The constant ρ is the diameter of the mesh, and the reason to include the factor ρ^2 here is to make the data energy scale invariant. Scale invariance is important only to be able to define the mesh energy as a linear combination of the data energy and the edge energy, independently of the scale of the problem.

The edge energy is defined as follows

$$U_E = \frac{1}{\rho^2} \sum_{e \in E} \nu_i \nu_j \|v_i - v_j\|^2, \tag{3.9}$$

where $e = \{v_i, v_j\}$, and $\nu_1, \ldots, \nu_{n_V}$ are positive constants defined as follows. The *degree* of a vertex is the number of edges incident to the vertex, or equivalently, the number of vertices connected to the

former one through an edge of the triangulation. The *mean degree* of a mesh is the mean value of the degrees of its vertices. In our implementation we have defined the constant ν_i as the ratio of the degree of the vertex v_i over the mean degree of the mesh, but other values provide similar results. For example, making $\nu_i = 1$ for all i is also a good choice. The factor ρ^2 is included in the denominator to make the edge energy scale invariant.

Special care should be taken in choosing the energy minimization algorithm, because since each time the vertices are moved, the data set has to be repartitioned, evaluating the energy function is potentially expensive. In principle, after a mesh deformation each point must be tested against each triangle for membership, but for small deformations each data point most likely will remain in the same triangle, or will move to one of the three neighbors.

4 Conclusions

We have introduced implicit simplicial models as a new representation for piece-wise linear curves and surfaces. We have shown that this new representation allows for a complete and efficient control of the topology of the curve or surface, and has most of the good properties of more traditional deformable models, and algebraic curves and surfaces. Implicit simplicial models can be used to model free-form curves and surfaces, but at the same time they provide an inside-outside function defined in a large neighborhood of the curve or surface. This inside-outside function can be constructed as an approximation to the distance from an arbitrary point to the curve or surface. At the same time, implicit simplicial curves and surfaces have explicit local parameterizations, which are good for other purposes. As a first application, we have described a two dimensional curve reconstruction algorithm from unorganized data sets which can be extended with almost no modification to an algorithm for surface reconstruction. We believe that a number of graphics and vision problems can be solved either more robustly, more generally, or both using implicial simplicial models, as for example surface reconstruction, tracking of surface deformations and adaptive isosurface construction, to mention just a few applications that we intend to demonstrate in future reports.

References

[1] G.E. Farin, *Curves and Surfaces for Computer Aided Geometric Design: A Practical Guide*. New York: Academic Press, 1988.

[2] M. Hall and J. Warren, "Adaptive polygonalization of implicitly defined surfaces," *IEEE Computer Graphics and Applications*, pp. 33–42, Nov. 1990.

[3] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Surface reconstruction from unorganized points," *Computer Graphics*, pp. 71–78, July 1992. (*Proc. SIGGRAPH '92.*)

[4] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *Int'l J. Computer Vision*, vol. 1, no. 4, pp. 321–331, 1988.

[5] D. Liu and J. Nocedal, "On the limited memory bfgs method for large scale optimization," *Mathematical Programming*, vol. B45, pp. 503–528, 1989.

[6] D. Metaxas and D. Terzopoulos, "Dynamic deformation of solid primitives with constraints," *Computer Graphics*, vol. 26, no. 2, pp. 309–312, July 1992.

[7] D. Moore and J. Warren, "Approximation of dense scattered data using algebraic surfaces," Technical Report Rice COMP TR90-135, Department of Computer Science, Rice University, Houston, Oct. 1990.

[8] R.S. Szeliski and D. Tonnesen, "Surface modeling with oriented particle systems," *Computer Graphics*, pp. 185–194, July 1992. (*Proc. SIGGRAPH '92.*)

[9] G. Taubin, "Nonplanar curve and surface estimation in 3-space," *Proc. IEEE Conf. Robotics and Automation*, pp. 644–645, May 1988.

[10] G. Taubin, "Estimation of planar curves, surfaces and nonplanar space curves defined by implicit equations, with applications to edge and range image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 11, pp. 1,115–1,138, Nov. 1991.

[11] G. Taubin, "An improved algorithm for algebraic curve and surface fitting," *Proc. Fourth Int'l Conf. Computer Vision*, pp. 658–665, Berlin, Germany, May 1993.

[12] G. Taubin, F. Cukierman, S. Sullivan, J. Ponce, and D.J. Kriegman, "Parameterized families of polynomials for bounded algebraic curve and surface fitting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 3, pp. 287–303, Mar. 1994.

[13] D. Terzopoulos and K. Fleischer, "Deformable models," *The Visual Computer*, vol. 4, pp. 306–311, 1988.

[14] G.L. Tindle, *The Mathematics of Surfaces II*, pp. 387–394. Oxford, England: Clarendon Press, 1987.

Detail-Preserving Variational Surface Design with Multiresolution Constraints

Ioana Boier-Martin
IBM T. J. Watson Research Center
Hawthorne, New York, USA
ioana@us.ibm.com

Remi Ronfard
INRIA Rhône-Alpes
Montbonnot, France
remi.ronfard@inrialpes.fr

Fausto Bernardini
IBM T. J. Watson Research Center
Hawthorne, New York, USA
fausto@us.ibm.com

Abstract

We present a variational framework for rapid shape prototyping. The modeled shape is represented as a Catmull-Clark multiresolution subdivision surface which is interactively deformed by direct user input. Free-form design goals are formulated as constraints on the shape and the modeling problem is cast into a constrained optimization one. The focus of this paper is on handling multiresolution constraints of different kinds and on preserving surface details throughout the deformation process. Our approach eliminates the need for an explicit decomposition of the input model into frequency bands and the overhead associated with saving and restoring high-frequency detail after global shape fairing. Instead, we define a deformation vector field over the model and we optimize its energy. Surface details are considered as part of the rest shape and are preserved during free-form model editing. We explore approximating the solution of the optimization problem to various degrees to balance trade-offs between interactivity and accuracy of the results.

1. Introduction

A common design paradigm is to allow designers to interactively deform an initial geometric shape to obtain a new one that satisfies certain requirements. The requirements are typically formulated as a set of constraints and the underlying geometric representation is modified to meet these constraints (see Figure 1). In general, the desired result is the one that has the most pleasing or fairest overall shape among all solutions that satisfy the constraints. A commonly used procedure to attain this result is to optimize a fairness measure representing physical parameters of a real object bearing the shape. A standard such measure [26] is the linear combination of the so-called *stretch* and *bend* energies:

$$E = \alpha \int ||I||^2 dS + \beta \int ||II||^2 dS \qquad (1)$$

where I and II are the first and second fundamental forms of the surface S and $||.||$ is a suitably chosen matrix norm.

It is often the case, however, that the input model has high-frequency geometric detail across multiple resolutions which needs to be preserved during global deformations of its shape. Fairing techniques like the one just described tend to smooth out not only the global shape of the object, but the high-frequency detail as well (see Figure 2).

To avoid this problem, multi-band decomposition schemes have been proposed [13], in which a multiresolution modeling operation and the associated fairing step are applied within one frequency band. Subsequently, the higher frequency detail is reconstructed using a displacement map. This implies computing and storing the displacement map prior to editing and restoring it afterwards. We opt for an alternative approach that avoids the computation of the displacement map and the overhead associated with saving and restoring high-frequency information by considering the deformations applied to the initial shape as a vector field defined over the input model. Instead of optimizing the energy of the deformed shape, we optimize the energy of the deformations and we apply the resulting smooth vector field to the original shape to obtain the deformed one. Using this approach, the input model becomes the rest shape to which the optimization converges in the absence of constraints. In the language of elastic body deformations, this is equiva-

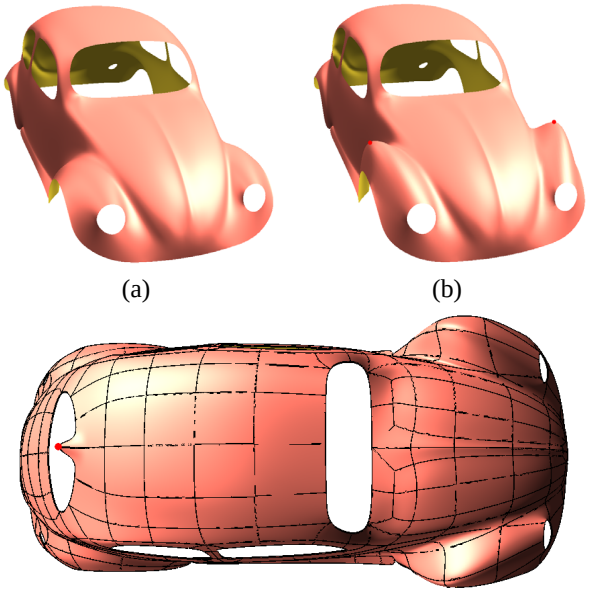


Figure 1. Variational design with multiresolution constraints. (a) Input model. (b) Coarse-scale edits affect the global shape. (c) Fine scale edit with local effect (patch structure of underlying surface representation is shown). Red dots indicate constraints.

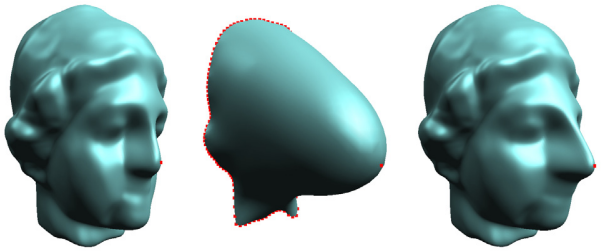


Figure 2. Free-form modeling with constraints. (a) Original model with multiresolution details. A point constraint at the tip of the nose is used to deform the model. (b) The model is deformed using thin-plate energy minimization. Note the smoothing of the original details (boundary constraints are necessary to prevent the collapse of the model). (c) Deformation with detail preservation (no boundary constraints needed in this case).

lent to considering the input shape as the *natural state* of the model [26].

In this paper we describe the design and implementation of a variational modeler that allows interactive editing of complex objects of arbitrary topology. We use Catmull-Clark multiresolution subdivision surfaces as our underlying representation and we take advantage of their hierarchical organization to allow editing with constraints at different resolution levels.

The main contributions of our research are:

1. A variational approach that leads to a smooth shape while preserving multiresolution details.
2. A framework for free-form design with constraints which can be imposed at different levels of resolution.
3. An implementation setup in which a fast approximate solution is computed at interactive rates during the deformation process. Solutions of increasing accuracy may follow upon user request.

The work presented in this paper adds a new technique to a growing set of surface modeling tools based on multiresolution subdivision surfaces that have been proposed in recent years (e.g., boolean operations [1], engraving, embossing, trimming [3], cut-and-paste editing [2]).

The remaining sections of the paper are organized as follows: in section 2 we discuss related work, in section 3 we briefly review the underlying data representation and we outline our approach; modeling with constraints is presented in section 4; results are discussed and illustrated in section 5; section 6 concludes our paper.

2. Related Work

Variational design of surfaces has emerged as a powerful modeling paradigm. It entails finding a surface that satisfies a number of constraints and minimizes a given continuous functional that represents the energy of the surface. Expression (1) gives an example of a commonly used such functional. In practice, discrete approximations of each of its two terms are used.

Following the pioneering work of Duchon [6] and Meinguet [19], many authors (e.g., [5, 12, 32, 11, 27]) approximate surface energy by one or a combination of the following parameterization-dependent expressions:

$$E_{stretch} \approx \int_{\Omega} \left(\frac{\partial S}{\partial u}\right)^2 + \left(\frac{\partial S}{\partial v}\right)^2 dudv \qquad (2)$$

$$E_{bend} \approx \int_{\Omega} \left(\frac{\partial^2 S}{\partial u^2}\right)^2 + 2\left(\frac{\partial^2 S}{\partial u \partial v}\right)^2 + \left(\frac{\partial^2 S}{\partial v^2}\right)^2 dudv$$

where Ω denotes the parametric domain of the surface S. In our approach we also make use of these linearized forms of the stretch and bend energies. We note that other approximations, generally more expensive to compute, have also been proposed [20, 21, 8], as well as alternative energy functionals [16, 28]. Various types of constraints can be imagined. We address here some of the most common categories.

Constrained optimization Variational surfaces are often modeled by specifying a set of constraints and solving for the solution that minimizes the objective functional (1) or an approximation thereof and satisfies the constraints.

Interpolating point constraints prescribe desired positions for points on the surface. This type of constraints are considered by virtually all methods. In our approach we translate point constraints into linear combinations of mesh vertices and we perform energy minimization with linear constraints.

Interpolating normal constraints prescribe desired normal vectors at given points on the surface. A common approach (see, for example, [11]) is to formulate the fact that desired normal N at a point $S(u_0, v_0)$ must be perpendicular to the vectors defined by the partial derivatives, i.e.:

$$N\frac{\partial S}{\partial u}(u_0, v_0) = 0 \text{ and } N\frac{\partial S}{\partial v}(u_0, v_0) = 0$$

These expressions can also be translated into linear constraints on the mesh vertices. Alternatively, some authors [15] enforce normal constraints by freezing all the vertices of a planar mesh face. While more stable, this approach tends to overly constrain the optimization problem.

Interpolating curve constraints prescribe desired positions at points along one or more curves on the surface. Such constraints typically require the constraint curves to be aligned with mesh edges or iso-parameter lines [27]. Using the re-parameterization idea of [3] we are able to avoid this requirement and to allow for constraints to be imposed along arbitrary curves.

Multiresolution We take advantage of our underlying hierarchical surface representation to handle constraints in a multiresolution fashion. An important issue to be addressed is identifying the *region of influence* of a given constraint. As observed in [22], the same constraint can be satisfied at a coarse scale by the global rigid motion of the entire surface or at a fine scale by the motion of an isolated surface point. The designer's intent usually lies somewhere in between. Except for the work of Takahashi [24, 25], this problem has received little attention in existing literature. Takahashi has developed a wavelet-based framework that accommodates linear constraints at multiple resolutions. His framework, however, is limited to topological patches. Surfaces of arbitrary topological type are handled by "gluing" patches and only simple examples are documented. In our work, we use a similar approach in which we propagate constraints from fine to coarse scales. However, we give the user explicit control over the region of influence of each constraint and we rely on subdivision rules to perform constraint restriction in a more straightforward fashion. Arbitrary topology is handled automatically due to the nature of the surface representation we use.

Another relevant question is how to reconcile different energy measures computed at different resolutions. In [24, 25] they are combined into a weighted sum of energy functions at multiple levels. This approach causes undesirable side-effects of constraints at coarse resolutions over the shape at finer levels. A recursive scheme of solving the shape level-by-level is used to avoid the interactions between constraints at different levels. Instead, we have chosen to fix a target resolution level at which the energy is minimized and to use the solutions at coarser resolutions as approximations in a multigrid fashion.

Subdivision and variational design Halstead et al. [11, 10] were among the first to propose a variational modeling approach in the context of subdivision surfaces. An alternative re-formulation using wavelets was introduced by Gortler and Cohen [7]. Recent work by Warren, Weimer, Kobbelt, and Schröder [29, 30, 14, 15] emphasizes the relationship between variational methods and subdivision. *Variational subdivision* seeks to define subdivision rules that produce a sequence of shapes that not only converges to a limit shape that follows the initial control shape, but also minimizes the energy functional associated with the limit surface.

An appealing aspect of subdivision hierarchies which we exploit for efficiency is that they naturally accommodate multigrid-type solvers [9]. The basic operations, i.e., *restriction* and *prolongation* can be easily formulated as local masks. The restriction operator maps the data from a fine level to a coarser level. When applying a deformation at the finer level, the main question to be addressed is how project it onto coarser levels. The prolongation operator achieves the inverse mapping, from coarse to fine. We apply the Catmull-Clark subdivision masks for this purpose, in contrast with the approach suggested in [31] in which special-purpose prolongation operators are devised.

3. Multiresolution Variational Design

We reformulate the optimization problem previously introduced in a discrete multiresolution setting. To justify our choices, we begin with an overview of the underlying representation used. Basic concepts related to variational calculus can be found in any standard textbook (e.g., [17]).

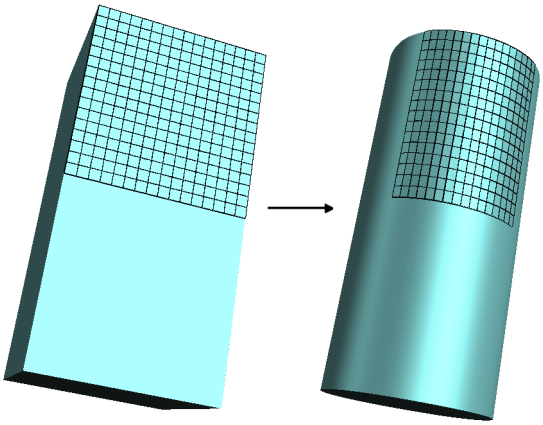


Figure 3. Natural parameterization of a subdivision surface. Each time we apply the subdivision rules to compute the finer control mesh we also apply midpoint subdivision to a copy of the initial control mesh. As we repeatedly subdivide, we get a mapping from a denser and denser subset of the control polyhedron (left) to the control points of a finer and finer control mesh (right). In the limit we get a map from the control polyhedron to the surface.

3.1. Multiresolution Subdivision Surfaces

The representation we use was introduced by several authors in different forms [18, 23, 33]. Subdivision defines a smooth surface recursively as the limit of a sequence of meshes. Each finer mesh is obtained from a coarse mesh by using a set of fixed refinement rules. In the work described in this paper we use the Catmull-Clark rules [4]. Multiresolution subdivision extends the concept of subdivision by allowing *detail vectors* to be introduced at each level. Hence, a finer mesh is computed by adding detail offsets to the subdivided coarse mesh. Given a *semi-regular mesh*, i.e., a mesh with subdivision connectivity, it can be easily converted to a multiresolution surface by defining a smoothing operation to compute a coarse level from a finer level. The details are then computed as differences between levels.

For our purposes, it is important to recognize that a multiresolution subdivision surface can be naturally interpreted as a function on the domain defined by the base mesh (see Figure 3). This interpretation is useful in many circumstances, including dealing with constraints along arbitrary curves as described in section 4.

It is, however, a known fact that the first and second order partial derivatives of the surface with respect to the natural parameterization diverge around extraordinary points. Therefore, for the purpose of evaluating expressions (2) we define a different parameterization as described in section 3.3.

3.2. Problem Formulation

Let $\mathcal{H} = (M^0, M^1, \cdots, M^{L-1})$ denote a multiresolution subdivision hierarchy with L levels such that the control mesh M^l at each level l is obtained from the coarser mesh M^{l-1} by subdividing it and adding detail offsets. Let $P^l = \{P_i^l\}, i = 0, \cdots, N-1$ denote the vertices of M^l. By applying quadrature formulas to discretize the integrals in (2), we obtain a discrete formulation of the energy associated with M^l:

$$E(M^l) = E(P^l) = \alpha \sum_i E_{stretch}(P_i^l) + \beta \sum_i E_{bend}(P_i^l)) \qquad (3)$$

where

$$E_{stretch}(P_i^l) = ||\frac{\partial M_i^l(u,v)}{\partial u}||^2 + ||\frac{\partial M_i^l(u,v)}{\partial v}||^2$$

$$E_{bend}(P_i^l) = ||\frac{\partial^2 M_i^l(u,v)}{\partial u^2}||^2 + 2||\frac{\partial^2 M_i^l(u,v)}{\partial u \partial v}||^2 + ||\frac{\partial^2 M_i^l(u,v)}{\partial v^2}||^2$$

and $M_i^l(u,v)$ denotes the parameterization of M^l around vertex P_i^l.

In the presence of constraints on subsets of the vertices $\{P_i^l\}$, the constrained energy optimization problem for level l becomes:

$$E(P^l) \rightarrow min$$
$$f_k^l(P_{i_1}^l, \cdots, P_{i_k}^l) = C_k^l, k = 1, \cdots, m_l$$

with C_k^l the target value of the k^{th} constraint on level l. In this paper we restrict our attention to cases in which the constraints f_k^l are linear combinations of the vertices $P_{i_j}^l$.

As mentioned in the previous section, instead of combining energies defined at different resolutions, we set as our goal the minimization of the energy of the finest level mesh $E(P^{L-1})$. To efficiently handle the optimization problem at this level, we use solutions from coarser levels as part of a multigrid approach.

3.3. Local Parameterization

In section 3.1 we pointed out that the natural parameterization induced by subdivision over the base control mesh is not suitable for evaluating partial derivatives everywhere on the surface. While feasible, finding a global parameterization that satisfies certain smoothness requirements is not a simple task. Fortunately, for the purpose of the work presented here, local parameterizations that allow us to approximate first and second order derivatives with divided differences are sufficient. We have opted for using local quadratic

polynomial interpolants as done in [15]. We briefly review this approach next.

To compute divided differences in the vicinity of a vertex P_0^l, a quadratic interpolating polynomial is fitted in least-squares sense to the local geometry defined by P_0^l and its immediate neighbors (see Figure 4):

$$Q(u,v) = Q + uQ_u + vQ_v + \frac{1}{2}u^2Q_{uu} + uvQ_{uv} + \frac{1}{2}v^2FQ_{vv}$$

To solve this problem we need at least six interpolation conditions which we formulate by assigning local parameter values (u_i, v_i) to P_0^l and its one-ring neighbors (we consider both edge and face neighbors). Since our underlying representation is a quadrilateral mesh, each vertex (interior or on the boundary) has at least five neighbors, with the exception of boundary vertices of valence one when only three direct neighbors exist. In such cases, we compute a least norm solution. Following [15], we assign coordinates $(0,0)$ to P_0^l and

$$(u_i, v_i) = ||P_i^l - P_0^l|| \left(\cos\left(\sum_{j \in R(0)} \alpha_j \right), \sin\left(\sum_{j \in R(0)} \alpha_j \right) \right)$$

to its neighbors, where $R(0)$ denotes the set of indices of vertices in the one-ring of P_0^l and

$$\alpha_j = \frac{2\pi \angle(P_j^l P_0^l P_{j+1}^l)}{\sum_{j \in R(0)} \angle(P_j^l P_0^l P_{j+1}^l)}.$$

The least-squares solution obtained by solving the interpolation problem yields values for the partial derivatives:

$$[Q_u, Q_v, Q_{uu}, Q_{uv}, Q_{vv}]^T = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T[\cdots, P_i^l - P_0^l, \cdots]^T$$

where

$$\mathbf{\Phi} = \begin{pmatrix} \vdots & \vdots & \vdots & \vdots & \vdots \\ u_i & v_i & \frac{1}{2}u_i^2 & u_iv_i & \frac{1}{2}v_i^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

In the remainder of the text, we denote by $\mathbf{D^{i,l}} = \left[D_{m,k}^{i,l}\right]_{m=\overline{1,5},k\in R(i)} = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T$ the coefficients of the divided difference operators corresponding to vertex P_i^l.

3.4. Discrete Energy Formulation

By replacing the partial derivatives in the energy expression (3) with divided differences computed from the local parameterization, we obtain our discrete energy formulation:

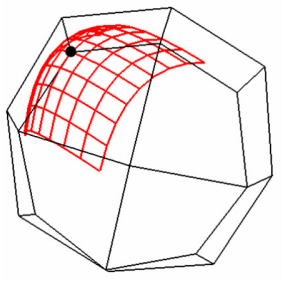


Figure 4. Local quadratic interpolant used to approximate first and second order derivatives.

$$E(P) = \sum_i \sum_{j,k \in R(i)} E_{ijk}(P_j - P_i)^T(P_k - P_i)$$

where the level index l is omitted to simplify the notation, and the coefficients E_{ijk} are defined as follows:

$$E_{ijk} = \alpha E_{ijk}^{stretch} + \beta E_{ijk}^{bend}$$
$$E_{ijk}^{stretch} = D_{1,j}^i D_{1,k}^i + D_{2,j}^i D_{2,k}^i,$$
$$E_{ijk}^{bend} = D_{3,j}^i D_{3,k}^i + 2D_{4,j}^i D_{4,k}^i + D_{5,j}^i D_{5,k}^i$$

In matrix form, this is equivalent to

$$E(P) = \frac{1}{2}P^T\mathbf{H}P,$$

where $\mathbf{H}$ is an $N \times N$ matrix. The minimum of $E(P)$ is found by setting all partial derivatives with respect to P_i to zero and solving the corresponding system:

$$\frac{\partial E(P)}{\partial P_i} = 0, i = 0, \cdots, N-1$$

or equivalently:

$$\nabla E(P) = \mathbf{H}P = 0 \qquad (4)$$

Since the functional $E(P)$ is quadratic in every vertex, the system (4) is linear. In the next section we describe a strategy for solving it taking into account constraints.

4. Detail-Preserving Modeling with Constraints

4.1. Shape Deformation

We regard the deformations applied to a given mesh M as vector offsets with respect to the original vertex positions:

$$\Delta P_i = P_i^{deformed} - P_i^{original}$$

Instead of minimizing the energy of the deformed mesh $E(P^{deformed})$, we would like to minimize solely its change in stretching and bending with respect to the initial shape.

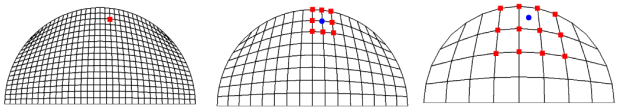


Figure 5. Constraint propagation from a fine level (left) to coarser ones. Constrained vertices are shown as (red) squares. The target value of the constraint is marked with a (blue) circle.

Thus, our constrained optimization problem at resolution level l becomes:

$$E(\Delta P^l) \rightarrow min$$
$$f_k^l(\Delta P_{i_1}^l, \cdots, \Delta P_{i_k}^l) = C_k^l, k = 1, \cdots, m_l$$

Before we present our solution to this problem, we describe the types of constraints we consider and how they are propagated and enforced at different resolutions.

4.2. Constraints

We model shape by prescribing points, normals, and curves that should be interpolated by the surface. The energy optimization model discussed in the previous section defines the behavior of the shape in the regions without constraints so that the designer does not need to directly specify the surface in these regions.

Region of influence In our multiresolution setting, constraints can be imposed at any level. However, since it is typically the actual surface that needs to interpolate the constraints and not the mesh at some intermediate level, we first project all constraints to the finest level (or, alternatively, the limit surface), and then we propagate them through the multiresolution hierarchy to coarser levels.

We define the *region of influence* of a constraint as the portion of the surface affected by the constraint. We control the region of influence by the coarse level to which a constraint is propagated. Hence, a constraint propagated to a relatively fine level in the hierarchy will have only *local* impact (see Figure 1 (c)), whereas a constraint propagated to a very coarse level will have an effect on the *global* shape of the model (see Figure 1 (b)).

Point constraints Figure 5 shows how positional constraints are generated. In general, if C_i^{l+1} is the prescribed deformation at a mesh vertex P_i^{l+1} on level $l+1$, then a linear constraint is generated at level l using the Catmull-Clark subdivision masks. This choice allows us to apply the Catmull-Clark subdivision rules to the deformation vectors to obtain a fast approximate solution during interactive manipulation. This solution is subsequently improved by using a solver as discussed later in this section. With the notations in Figure 6, the expressions for the linear constraints generated depending on the position of the constrained vertex in the mesh are given in Table 1.

$(1-\beta-\gamma)\Delta P_i^l + \frac{\beta}{k}\sum_{2j\in R(i)} \Delta P_{2j}^l + \frac{\gamma}{k}\sum_{2j+1\in R(i)} \Delta P_{2j+1}^l = C_i^{l+1}$
$\frac{3}{4}\Delta P_i^l + \frac{1}{8}(\Delta P_0^l + \Delta P_1^l) = C_i^{l+1}$
$\frac{1}{4}(\Delta P_0^l + \Delta P_1^l + \Delta P_2^l + \Delta P_3^l) = C_i^{l+1}$
$\frac{3}{16}(\Delta P_0^l + \Delta P_3^l) + \frac{1}{8}(\Delta P_1^l + \Delta P_2^l + \Delta P_4^l + \Delta P_5^l) = C_i^{l+1}$
$\frac{1}{2}(\Delta P_0^l + \Delta P_1^l) = C_i^{l+1}$

Table 1. Linear constraints are created at coarse levels using the Catmull-Clark rules. From top to bottom, the rules are for interior even vertex, boundary even vertex, inteior odd face vertex, interior odd edge vertex, and boundary odd vertex, respectively. Vertex indexing corresponds to Figure 6 (k denotes vertex valence, $\beta = \frac{3}{2k}$, and $\gamma = \frac{1}{4k}$).

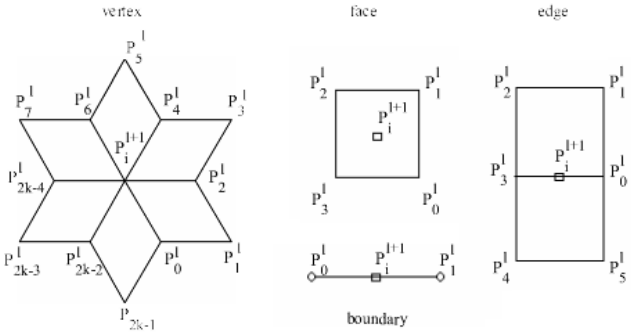


Figure 6. Linear constraints are generated on coarse levels using the Catmull-Clark masks accoring to the expression listed in Table 1.

Normal constraints Normal constraints are imposed by constraining two tangent vectors at a point to be perpendicular to the normal at that point. We use the vectors Q_u and Q_v estimated from the quadratic interpolant in place of the two tangents:

$$Q_u(P_i^l) \cdot N(P_i^l) = 0 \text{ and } Q_v(P_i^l) \cdot N(P_i^l) = 0$$

Using the notations from section 3.3, we see that these two conditions translate into linear equations involving the point P_i^l and its immediate neighbors:

$$\sum_{j \in R(i)} D^{i,l}_{k,j}(P^l_j - P^l_i)^T N(P^l_i) = 0,\ k = 1,2$$

The equivalent conditions on the deformations become:

$$\sum_{j \in R(i)} D^{i,l}_{k,j}(\Delta P^l_j - \Delta P^l_i)^T N(P^l_i) =$$
$$- \sum_{j \in R(i)} D^{i,l}_{k,j}(P^{l,original}_j - P^{l,original}_i)^T N(P^l_i),\ k = 1,2$$

Curve constraints Besides interpolation of prescribed points and normals, it may be useful to manipulate the surface by modifying the shape of embedded lower-dimensional entities. We restrict our attention to arbitrary curves lying on the surface. The most common example is modifying an open surface by rigidly manipulating its boundary curves (see Figures 9 (a) and (b)). In this case, we can sample the curves with boundary vertices and create collections of point constraints at these vertices. However, in the general case, a curve lying on the surface is not aligned with the underlying mesh edges, so a direct sampling with vertices is not possible. To allow for constraints along such curves, we re-parameterize the surface as in [3] to align it with the curve. The details of the re-parameterization are appended for convenience in the Appendix at the end of the paper. After re-parameterization, the original curve is approximated by a piecewise linear one that passes through mesh vertices. Point constraints are placed at vertices along the approximating curve and propagated through the multiresolution hierarchy as previously described.

4.3. Constrained Optimization

The simplest approach to solving our constrained minimization problem is to fix the deformation vectors at the constrained vertices to certain values and to solve an unconstrained energy minimization problem for the remaining ones. To determine the fixed deformations for the constrained vertices, we solve the linear system of constraints in the least-squares sense. Using matrix notation, we can write the system as:

$$\mathbf{A}\Delta P^l_c = C^l \qquad (5)$$

where ΔP^l_c denotes the vector of deformations corresponding to the constrained vertices at level l. The least-squares solution is given by:

$$\Delta P^l_c = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T C^l$$

We update the deformations associated with the unconstrained vertices iteratively, using a Gauss-Seidel approach ($\Delta P^{l,n}_i$ denotes the deformation at iteration n and $\{H_{ij}\}$ are entries of the matrix $\mathbf{H}$):

$$\Delta P^{l,n+1}_i = \frac{1}{H_{ii}}(-\sum_{j<i} H_{ij}\Delta P^{l,n+1}_j - \sum_{j>i} H_{ij}\Delta P^{l,n}_j)$$

We have also experimented with successive over-relaxation to accelerate convergence:

$$\Delta P^{l,n+1}_i =$$
$$(1-\omega)\Delta P^{l,n}_i + \frac{\omega}{H_{ii}}(-\sum_{j<i} H_{ij}\Delta P^{l,n+1}_j - \sum_{j>i} H_{ij}\Delta P^{l,n}_j)$$

where ω is a relaxation parameter between 0 and 2. It has been our experience that choosing a suitable value for ω results in considerable speedup. The choice, however, is typically problem dependent.

5. Results and Implementation

Figures 7, 8, and 9 illustrate sample results obtained with our system. Figure 9 top shows the smooth deformation of a planar surface using constraints along the boundary. In particular, the vertices on the inner boundary are rigidly rotated as indicated by the yellow arcball and the surface follows naturally. Figures 9 bottom and 7 illustrate deformations of organic shapes rich in high-frequency detail. Note the preservation of this detail on the modified shapes, on both the cow and the vase models. Figure 8 illustrates different accuracy approximations of the solution to the optimization problem. Combining constraint propagation through the multiresolution hierarchy and applying Catmull-Clark rules as a smoothing operation yields a good initial approximation of the thin plate energy minimization (see Figure 8 (b)) and allows the user to assess the modified shape at interactive rates. An improved solution later computed using a multigrid approach is shown in (c). The shape at intermediate levels and the multiresolution constraints are shown in Figure 8 bottom.

We make several observations pertaining to our implementation.

Point vs. normal constraints In the case of point constraints (including those generated from curve constraints), the system (5) represents, in fact, a collection of three linear systems, one for each of the spatial dimensions of the deformation vectors. Unfortunately, our formulation of normal constraints violates this independence of spatial dimensions which is attractive from a computational point of view. If normal constraints are present in the optimization problem, we set up a single coupled system for all dimensions, and we solve the larger system. Each point constraint is represented by three equations in this system (one for each of the x, y, and z components of a deformation). Each normal constraint translates into two equations as described in section 4.2. The resulting system is larger, and hence, more expensive to compute, however, it allows for more flexible design options.

Multigrid optimization Since the emphasis in our prototype is on interactivity, we explore approximating the solution of the optimization problem to various degrees to

balance interactivity and accuracy. During constraint manipulation, we use Catmull-Clark smoothing which yields a good approximation of the solution very fast (see Figure 8). At the end of a design step (e.g., upon mouse release), a more accurate solution is obtained through energy minimization. Finally, per user request, a fully converged solution will also be provided (typically at non-interactive rates).

To accelerate convergence of the iterative energy minimization process, we use a multigrid approach to compute the solution in a hierarchical fashion. We use a simplified coarse grid correction method consisting of two main steps. *Relaxation* iteratively minimizes the energy at level l using Gauss-Seidel iterations as previously described. *Prolongation* is used to propagate the solution computed at level l to the next finer level. We use Catmull-Clark subdivision as our prolongation operator. Note that we are not currently using restriction from fine to coarse as full multigrid implementations typically do.

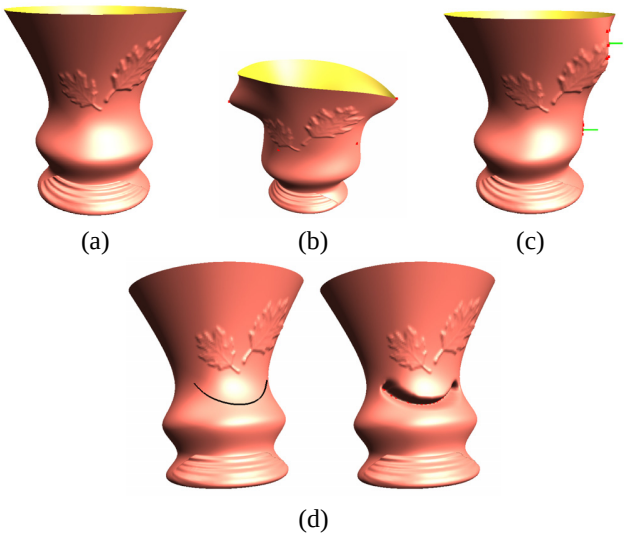


Figure 7. Modeling with detail preservation (a) Original model. (b) Deformation under point constraints (marked with red points). (c) Normal constraints (marked with green lines; red points indicate constrained region). (d) Constraints imposed along an arbitrary curve on the surface (left) are used to rigidly deform the surface in the vicinity of the curve (right).

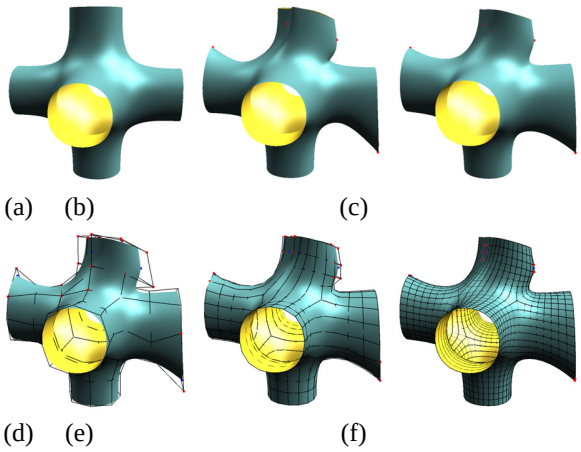


Figure 8. Top: approximating energy minimization. (a) Input model. (b) Fast solution obtained using Catmull-Clark smoothing. (c) Improved solution obtained via multigrid energy minimization. Bottom: (d)-(f) the optimized control mesh and the constraints at levels 2, 3, and 4 during multigrid.

6. Conclusions and Future Work

In this paper we described a variational modeling approach as implemented in a system we have developed. Models are represented by multiresolution subdivision surfaces. During interactive deformations, their shapes are recomputed via energy optimization with constraints. Point, normal, and curve constraints are considered at multiple resolution levels. To preserve multiresolution details, we optimize only the energy of the deformations, instead of the total surface energy.

In our work we have adapted existing variational methods to multiresolution subdivision surfaces and in doing so, we have shown the advantages of using this representation for interactive free-form design. Some of the remaining issues to be solved include the derivation of better stopping criteria for the iterative solver (we are currently using a fixed number of iterations), a full-fledged multigrid implementation including restriction of deformations to avoid recomputing surface regions already optimized, as well as alternative solvers.

Appendix: Re-Parameterization for Placing Constraints Along Arbitrary Curves

We briefly review the re-parameterization algorithm of [3] which we use for allowing constraints to be imposed along arbitrary curves on a surface.

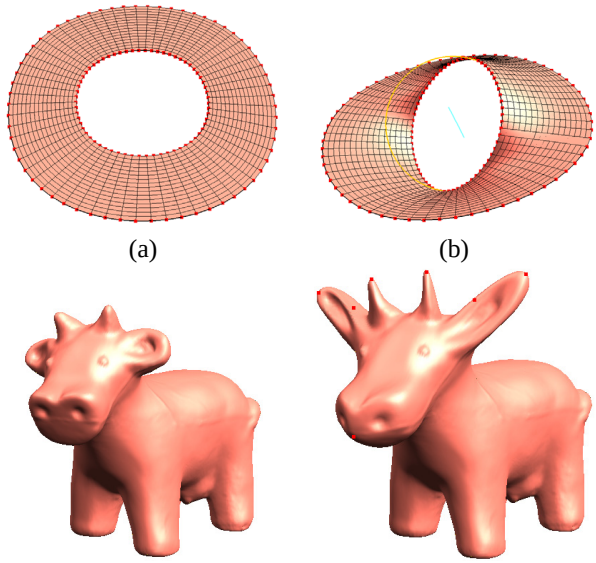


Figure 9. Top: curve constraints along boundary. (a) Input model. (b) Surface after rotating the inner boundary curve and energy optimization (rotation arc and axis are superimposed). Bottom: interactive editing. (c) Input model with details at multiple resolutions. (d) The model in (c) after variational editing. Red points indicate constraints.

As mentioned in section 2, we view the input surface as a parametric surface over the domain defined by the base control polyhedron. The main idea is to re-parameterize the surface to align the parameterization with a given curve or set of curves.

Let X denote the parameter domain of the surface defined by its base mesh and let c denote an input curve defined on X, $c : [0, 1] \rightarrow X$. In general, c traverses the domain X at arbitrary positions. We want to re-parameterize the domain X such that c passes through the vertices of X. Therefore, we compute a one-to-one mapping $\Pi : X \rightarrow X$ which maps vertices of X to curve points: $\Pi(v_i) = c(t_i)$, for some vertices $\{v_0, v_1, \ldots\}$ and curve parameters $\{t_0, t_1, \ldots\}$. The mapping Π is built to satisfy the following *approximation property* (AP):

(AP): *the piecewise linear curve* $[v_0, v_1, \ldots]$ *has the same topology as* c *and either follows along mesh edges or crosses mesh faces diagonally.*

The re-parameterization algorithm alternates between *snapping* and *refinement* steps. The snapping step moves mesh vertices onto the curve if they are sufficiently close. In the refinement step we simply subdivide the parameterization linearly. The algorithm terminates if the sequence of vertices $\{v_0, v_1, \cdots\}$ along c satisfies the approximation property (AP). Property (AP) is guaranteed to be satisfied after a finite number of steps for piecewise-linear curves c.

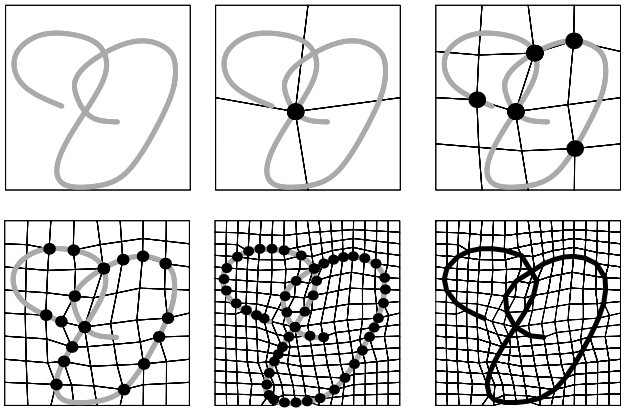


Figure 10. Re-parameterization matching a feature curve. The quad is recursively split and vertices are snapped to the curve. After four subdivision steps, the curve is approximated by a sequence of mesh vertices. Constraints may be imposed at these vertices.

References

[1] H. Biermann, D. Kristjansson, and D. Zorin. Approximate boolean operations on free-form solids. In *Proceedings of SIGGRAPH 01*, pages 185–194. ACM SIGGRAPH, 2001.

[2] H. Biermann, I. Martin, F. Bernardini, and D. Zorin. Cut-and-paste editing of multiresolution surfaces. In *Proceedings of SIGGRAPH 02*, page to appear. ACM SIGGRAPH, 2002.

[3] H. Biermann, I. Martin, D. Zorin, and F. Bernardini. Sharp features on multiresolution subdivision surfaces. *Graphical Models*, 64(2):61–77, 2002.

[4] E. Catmull and J. Clark. Recursively generated B-spline surfaces on arbitrary topological meshes. 10(6):350–355, 1978.

[5] G. Celniker and D. Gossard. Deformable curve and surface finite-elements for freeform shape design. In *Proceedings of SIGGRAPH 91*, pages 257–265. ACM SIGGRAPH / Addison Wesley, 1991.

[6] J. Duchon. *Splines minimizing rotation-invariant semi-norms in Sobolev spaces*. Springer–Verlag, 1977.

[7] S. J. Gortler and M. F. Cohen. Hierarchical and variational geometric modeling with wavelets. In *Proceedings Symposium on Interactive 3D Graphics*. Siggraph, May 1995.

[8] G. Greiner. Variational design and fairing of spline surfaces. *Computer Graphics Forum*, 13:143–154, 1994.

[9] W. Hackbusch. *Multi-Grid Methods and Applications*. Springer–Verlag, 1985.

[10] M. Halstead. *Efficient Techniques for Surface Design Using Constrained Optimization*. PhD thesis, UC Berkeley, 1996.

[11] M. Halstead, M. Kass, and T. DeRose. Efficient, fair interpolation using Catmull-Clark surfaces. In *Computer Graphics Proceedings*, Annual Conference Series, pages 35–44. ACM Siggraph, 1993.

[12] M. Kallay. *Constrained optimization in surface design*. Springer–Verlag, 1993.

[13] L. Kobbelt, T. Bareuther, and H.-P. Seidel. Multiresolution shape deformations for meshes with dynamic vertex connectivity. In *Proceedings of Eurographics 00*, pages 249–260, 2000.

[14] L. Kobbelt and P. Schröder. A multiresolution framework for variational subdivision. *ACM Transactions on Graphics*, 17(4):209–237, October 1998. ISSN 0730-0301.

[15] L. P. Kobbelt. Discrete fairing and variational subdivision for freeform surface design. *The Visual Computer*, 16(3-4):142–150, 2000. ISSN 0178-2789.

[16] S. Kuriyama and K. Tachibana. Polyhedra surface modeling with a diffusion system. In *Proceedings of Eurographics 97*, pages 39–46, 1997.

[17] C. Lanczos. *The Variational Principles of Mechanics*. Dover Publications, 1986.

[18] M. Lounsbery, T. DeRose, and J. Warren. Multiresolution analysis for surfaces of arbitrary topological type. *Transactions on Graphics*, 16(1):34–73, January 1997.

[19] J. Meinguet. Multivariate interpolation at arbitrary points made simple. *Journal of Applied Mathematics and Physics*, 30:292–304, 1979.

[20] H. Moreton. *Minimum Curvature Variation Curves, Networks, and Surfaces for Fair Free-Form Shape Design*. PhD thesis, UC Berkeley, 1993.

[21] H. Moreton and C. Sequin. Functional optimization for fair surface design. In *Proceedings of SIGGRAPH 92*, pages 167–176. ACM SIGGRAPH, 1992.

[22] L. F. Palazzi and D. L. Forsey. A multilevel approach to surface response in dynamically deformable models. In *Computer Animation Proceedings*, pages 21–30. Springer-Verlag, 1994.

[23] K. Pulli and M. Lounsbery. Hierarchical editing and rendering of subdivision surfaces. Technical Report UW-CSE-97-04-07, Dept. of CS&E, University of Washington, Seattle, WA, 1997.

[24] S. Takahashi. Variational design of curves and surfaces using multiresolution constraints. *The Visual Computer*, 14(5):208–227, 1998.

[25] S. Takahashi. Multiresolution constraints for designing subdivision surfaces via local smoothing. In *Proceedings of Pacific Graphics 99*, pages 168–178. IEEE, 1999.

[26] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically deformable models. In *Proceedings of SIGGRAPH 87*, pages 205–214. ACM SIGGRAPH / Addison Wesley, 1987.

[27] R. C. Veltkamp and W. Wesselink. Variational modeling of triangular bezier surfaces.

[28] O. Volpin, M. Bercovier, and T. Matskewich. A comparison of invarian energies for free-form surface construction. *The Visual Computer*, 15(4):199–210, 1999.

[29] J. Warren. Draft: Subdivision schemes for variational splines.

[30] H. Weimer. *Subdivision Schemes for Physical Problems*. PhD thesis, Rice University, 2000.

[31] H. Weimer and J. Warren. Subdivision schemes for thin plate splines. In *Proceedings of Eurographics 98*, pages 303–313, 1998.

[32] W. Welch and A. Witkin. Variational surface modeling. In E. E. Catmull, editor, *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26, pages 157–166, July 1992.

[33] D. Zorin, P. Schröder, and W. Sweldens. Interactive multiresolution mesh editing. *Proceedings of SIGGRAPH 97*, pages 259–268, August 1997. ISBN 0-89791-896-7. Held in Los Angeles, California.