Etude et réalisation d'un système de programmation pour la commande numérique des machines-outils

Philippe Gabrini

N° d'ordre

THESE

présentée à

LA FACULTE DES SCIENCES DE L'UNIVERSITE DE GRENOBLE

pour obtenir

LE GRADE DE DOCTEUR INGENIEUR

par

Philippe GABRINI

Ingénieur A. M. Master of Science

Etude et réalisation d'un système de programmation pour la commande numérique des machines-outils

Thèse soutenue le devant la commission d'examen :

MM. J. KUNTZMANN Président

L. BOLLIET
F. GENUYS
R. PERRET Examinateurs
C. SAUVAIRE
R. SIBILLE

L I S T E D E S P R O F E S S E U R S

Doyen Honoraire : Monsieur M. MORET
Doyen : Monsieur E. BONNIER

PROFESSEURS TITULAIRES

MM.	NEEL Louis	Physique Expérimentale
	KRAVTCHENKO Julien	Mécanique Rationnelle
	CHABAUTY Claude	Calcul différentiel et intégral
	BENOIT Jean	Radioélectricité
	CHENE Marcel	Chimie Papetière
	FELICI Noël	Electrostatique
	KUNTZMANN Jean	Mathématiques Appliquées
	BARBIER reynold	Géologie Appliquée
	SANTON Lucien	Mécanique des Fluides
	OZENDA Paul	Botanique
	FALLOT Maurice	Physique Industrielle
	KOSZUL Jean Louis	Mathématiques
	GALVANI Octave	Mathématiques
	MOUSSA André	Chimie Nucléaire
	TRAYNARD Philippe	Chimie Générale
	SOUTIF Michel	Physique Générale
	CRAYA Antoine	Hydrodynamique
	REULOS René	Théorie des Champs
	BESSON Jean	Chimie Minérale
	AYANT Yves	Physique Approfondie
	GALLISSOT François	Mathématiques
Mlle	LUTZ Elisabeth	Mathématiques
MM.	BLANBERT Maurice	Mathématiques
	BOUCHEZ Robert	Physique Nucléaire
	LLIBOUTRY Louis	Géophysique
	MICHEL Robert	Minéralogie et pétrographie
	BONNIER Etienne	Electrochimie et Electrométallurgie
	DESSAUX Georges	Physiologie animale
	PILLET Emile	Physique Industrielle-Electrotechnique
	YOCCOZ Jean	Physique Nucléaire théorique
	DEBELMAS Jacques	Géologie Générale
	BERBER Robert	Mathématiques
	PAUTENET René	Electrotechnique
	MALGRANGE Bernard	Mathématiques Pures
	VAUQUOIS Bernard	Calcul Electronique
	BARJON Robert	Physique Nucléaire
	BARBIER Jean Claude	Physique
	SILBERT Robert	Mécanique des Fluides
	BUYLE-BODIN Maurice	Electronique
	DREYFUS Bernard	Termodynamique

MM.	KLEIN Joseph	Mathématiques
	VAILLANT François	Zoologie et Hydrobiologie
	ARNAUD Paul	Chimie
	SENGEL Philippe	Zoologie
	BARNOUD Fernand	Biosynthèse de la Cellulose
	BRISSONNEAU Pierre	Physique
	GAGNAIRE Didier	Chimie Physique
Mme	KOFLER Lucie	Botanique
MM.	DEGRANGE Charles	Zoologie
	PEBAY-PEROULA Jean Claude	Physique
	RASSAT André	Chimie Systématique
	DUCROS Pierre	Cristallographie Physique
	DODU Jacques	Mécanique Appliquée I.U.T.
	ANGLES D'AURIAC Paul	Mécanique des Fluides
	LACAZE Albert	Thermodynamique
	GASTINEL Noël	Analyse Numérique
	GIRAUD Pierre	Géologie
	PERRET René	Servo-mécanisme
	PAYAN Jean Jacques	Mathématiques Pures

PROFESSEURS SANS CHAIRE

MM.	GIDON Paul	Géologie
Mme	BARBIER Marie-Jeanne	Electrochimie
Mme	SOUTIF Jeanne	Physique
MM.	COHEN Joseph	Electrotechnique
	DEPASSEL R.	Mécanique des Fluides
	GLENAT René	Chimie
	BARRA Jean	Mathématiques Appliquées
	COUMES André	Electronique
	PERRIAUX Jacques	Géologie et Minéralogie
	ROBERT André	Chimie Papetière
	BIARREZ Jean	Mécanique Physique
	BONNET Georges	Electronique
	CAUQUIS Georges	Chimie Générale
	BONNETAIN Lucien	Chimie Minérale
	DEPOMMIER Pierre	Physique Nucléaire-Génie Atomique
	HACQUES Gérard	Calcul Numérique
	POLOUJADOFF Michel	Electrotechnique
Mme	KAHANE Josette	Physique
Mme	BONNIER Jane	Chimie
MM.	VALENTIN Jacques	Physique
	REBECQ Jacques	Biologie
	DEPORTES Charles	Chimie
	SARROT-REYNAULD Jean	Géologie
	BERTRANDIAS Jean Paul	Mathématiques
	AUBERT Guy	Physique

PROFESSEURS ASSOCIES

MM.	RODRIGUES Alexandre	Mathématiques Pures
	MORITA Susumu	Physique Nucléaire
	RADHAKRISHNA	Thermodynamique

MAITRES DE CONFERENCES

MM.	LANCIA Roland	Physique Atomique
Mme	BOUCHE Liane	Mathématiques
MM.	KAHANE André	Physique Générale
	DOLIQUE Jean Michel	Electronique
	BRIERE Georges	Physique
	DESRE Georges	Chimie
	LAJZEHOWICZ Joseph	Physique
	LAURENT Pierre	Mathématiques Appliquées
Mme	BERTRANDIAS Françoise	Mathématiques Pures
MM.	LONGEQUEUE Jean Pierre	Physique
	SOHM Jean Claude	Electrochimie
	ZADWORNY François	Electronique
	DURAND Françis	Chimie Physique
	CARLIER Georges	Biologie Végétale
	PFISTER Jean Claude	Physique
	CHIBON Pierre	Biologie Animale
	IDELMAN Simon	Physiologie animale
	BLOCH Daniel	Electrotechnique I.P.
	MARTIN-BOUYER Michel	Chimie (C.S.U. Chambéry)
	SIBILLE Robert	Construction mécanique (I.U.T.)
	BRUGEL Lucien	Energétique I.U.T.
	BOUVARD Maurice	Hydrologie
	RICHARD Lucien	Botanique
	PELMONT Jean	Physiologie Animale
	BOUSSARD Jean Claude	Mathématiques Appliquées (I.P.G.)
	MOREAU René	Hydraulique I.P.G.
	ARMAND Yves	Chimie I.U.T.
	BOLLIET Louis	Informatique I.U.T.
	KUHN Gérard	Energétique I.U.T.
	PEFFEN René	Chimie I.U.T.
	GERMAIN Jean Pierre	Mécanique
	JOLY Jean René	Mathématiques Pures
Melle	PIERY Yvette	Biologie Animale
	BERNARD Alain	Mathématiques Pures
	MOHSEN Tahsin	Biologie (C.S.U. Chambéry).
	CONTE René	Mesures Physiques I.U.T.
	LE JUNTER Noël	Génie Electrique Electronique I.U.T.
	LE ROY Philippe	Génie Mécanique I.U.T.
	ROMIER Guy	Technique Statistiques Quantitatives I.U.T.
	VIALON Pierre	Géologie
	BENZAKEN Claude	Mathématiques Appliquées
	MAYNARD Roger	Physique
	DUSSAUD René	Mathématiques (C.S.U. Chambéry)
	BELORIZKY Elie	Physique (C.S.U Chambéry)
Mme	LAJZEROWICZ Jeannine	Physique (C.S.U. Chambéry)
M.	JULLIEN Pierre	Mathématiques Pures
Mme	RINAUDO Marguerite	Chimie
MM.	BLIMAN Samuel	E.I.E.
	BEGUIN Claude	Chimie Organique
	NEGRE Robert	I.U.T.

MAITRES DE CONFERENCES ASSOCIES

MM.	YAMADA Osamu	Physique du Solide
	NAGAO Makoto	Mathématiques Appliquées
	MAREZIO Massimo	Physique du Solide
	CHEECKE John	Thermodynamique
	BOUDOURIS Georges	Radioélectricité
	ROZMARIN Georges	Chimie Papetière

Je tiens à remercier

Monsieur le Professeur Jean KUNTZMANN, Directeur du Service de Mathématiques Appliquées, qui a bien voulu me faire l'honneur de présider le Jury de thèse.

Monsieur le Professeur René PERRET, Directeur du Laboratoire d'Automatique de l'Institut Polytechnique, qui a bien voulu s'intéresser à mon travail et faire partie du Jury.

Monsieur le Professeur Robert SIBILLE, Directeur du département de Génie Mécanique de l'I.U.T. de Grenoble, qui a bien voulu s'intéresser à mon travail et faire partie du Jury.

Monsieur le Professeur Louis BOLLIET, qui par ses encouragements et ses conseils m'a permis de mener ce travail à bien.

Monsieur François GENUYS, Directeur du développement du Service Bureau à IBM France, qui a suivi mon travail et dont les conseils m'ont été précieux.

Monsieur Christian SAUVAIRE, chef des Services techniques de l'ADEPA, qui a bien voulu faire partie du Jury et dont j'ai apprécié l'intérêt pour ce travail.

L'ADEPA, le groupe commande numérique de la CII et les établissements RICHIER de Pont de Claix de l'intérêt qu'ils ont porté à ce travail.

Jean LE PALMEC sans qui ce travail n'eût pas été possible.

Tous les membres du Laboratoire et le personnel des services de dactylographie et de reproduction qui ont contribué de près ou de loin à la réalisation de cet ouvrage.

à Anne-Marie

TABLE DES MATIERES

INTRODUCTION

L'utilisation des machines-outils pour l'usinage des pièces mécaniques n'est pas une technique récente. Il a cependant fallu attendre ces dernières années pour voir l'apparition d'un progrès vraiment considérable, la commande numérique. Les machines-outils à commande numérique ont pris une importance grandissante et se sont rapidement répandues. Ces machines étant des machines-outils programmées, on a vu apparaître un grand nombre de systèmes de programmation automatique basés sur les langages de programmation pour commande numérique. La prolifération de ces langages a fait apparaître une tendance à la normalisation. Des groupes de travail de l'organisation internationale de normalisation (I.S.O.) ont été formés pour étudier en détail la normalisation du langage APT et des langages qui lui sont apparentés comme 2CL, EXAPT et IFAPT. Malgré cette tendance on voit toujours apparaître de nouveaux langages destinés à la commande numérique des machines-outils et qui n'ont aucun rapport avec APT. Ainsi l'an dernier à la conférence internationale PROLAMAT (Programming Languages for Machine Tools) ont été présentés plusieurs de ces nouveaux langages. Il est cependant à noter que les langages de ce type qui apparaissent à l'heure actuelle sont très spécialisés et en général destinés à être utilisés sur ordinateurs de taille réduite. Les langages apparentés à APT ne peuvent en effet être traités que sur de gros ordinateurs ; IFAPT, qui parmi les systèmes de la famille APT est certainement le plus récent et le moins encombrant, nécessite un ordinateur ayant 64K octets de mémoire centrale. C'est pourquoi malgré leur manque d'uniformité, de petits langages spécialisés peuvent encore faire leur apparition et peuvent encore être préférés pour des applications particulières, aux langages plus généraux apparentés à APT.

Il y a déjà plusieurs années le Laboratoire de Calcul de la Faculté des Sciences de Lille a mis au point un système de programmation pour machines-outils à commande numérique, il s'agit de MECALGOL. L'originalité de ce système venait de ce qu'il était constitué de procédures

ALGOL et de ce que l'écriture d'un programme de pièce revenait à écrire un programme ALGOL simple. Il était de plus, facilement adaptable à différents types de machines-outils à commande numérique ; il offrait aussi une grande facilité d'extension par addition de nouvelles procédures. MECALGOL fut repris par la compagnie BULL-GENERAL ELECTRIC qui l'intégra au système FORTRAN accessible en télétraitement et le commercialisa sous le nom de MECA. En un sens il est dommage que MECALGOL qui semblait si prometteur lors de sa mise en oeuvre et qui offrait potentiellement de nombreuses possibilités, n'ait pas connu un développement plus important.

L'apparition de petits systèmes apparentés à APT et l'intérêt suscité par de telles réalisations ont montré l'existence d'un besoin réel dans ce comaine pour des petits systèmes de programmation pour la commande numérique des machines-outils. Nous avons donc essayé d'aborder l'étude des systèmes de programmation apparentés à APT sous un angle nouveau.

Le but du travail que nous présentons ici a d'abord été la conception d'une méthode permettant la réalisation d'un système répondant aux critères suivants :

- traitement de programmes de pièce écrits dans un langage de la famille APT
- fonctionnement possible sur petits ordinateurs
- possibilités de modifications et d'extensions faciles permettant de minimiser les travaux de programmation
- transférabilité du système d'un ordinateur à un autre de la manière la plus rapide possible
- structure simple pouvant être facilement comprise afin de faciliter la mise au point et la maintenance.

Cette méthode une fois définie, la seconde partie du travail a été consacrée à la réalisation d'un tel système de programmation en commande numérique.

Nous avons ainsi obtenu une version générale de ce système fonctionnant en mode conversationnel sous les systèmes CP67/CMS. Pour vérifier la possibilité de fonctionnement sur petit ordinateur, cette version générale a été adaptée à un ordinateur IBM 1130 (mémoire centrale : 8K mots de 16 bits, mémoire secondaire : un disque). Le système obtenu n'est encore qu'expérimental et en l'absence d'un perforateur de ruban connecté à l'ordinateur IBM 1130 utilisé, les ordres de commande numérique sont imprimés ou perforés sur cartes ; il faut alors utiliser un programme de transfert cartes-bande pour obtenir la bande perforée destinée au système de commande numérique.

Cette version du système de programmation fonctionnant sur ordinateur IBM 1130 a été alors successivement adaptée à deux types de systèmes de commande numérique. Ces adaptations ont été vérifiées à l'aide d'exemples d'usinages réels et ont donné les résultats attendus ; elles devront cependant être encore testées de façon plus complète. Une nouvelle adaptation de ce système est actuellement en cours, qui est destinée à une utilisation industrielle.

L'idée de MECALGOL consistant à utiliser des sous-programmes de taille réduite nous a permis de "partitionner" notre système le plus possible. Il est ainsi possible d'utiliser séparément les différentes parties du système ce qui peut être intéressant dans le cas où l'on ne dispose que d'un ordinateur avec une petite mémoire centrale (IBM 1130 avec 4K mots de mémoire centrale par exemple). On peut également utiliser la version la plus générale du système fonctionnant en mode conversationnel sous les systèmes CP67/CMS ou sous d'autres systèmes conversationnels, soit telle quelle, soit pour mettre au point diverses extensions du système relatives par exemple aux travaux d'usinage en mode continu ou à l'introduction des données technologiques à la programmation automatique.

CHAPITRE 1

PRINCIPES DE BASE DE LA COMMANDE NUMERIQUE

1.1. GENERALITES.

Dans la mesure où l'on peut considérer le tour du potier comme une des premières machines créées par l'homme, on se rend compte que l'idée de machine-outil n'est pas nécessairement récente. L'expression "machine-outil" contient d'ailleurs en elle-même sa définition ; elle désigne un matériel ayant pour but le déplacement d'un outil par l'intermédiaire d'une transmission motorisée. Le but de l'usinage sera alors de façonner un produit à la forme requise et aux dimensions demandées. Les premières machines-outils mettaient en valeur l'habileté manuelle de l'ouvrier, le travail s'effectuant par approches successives, suivies chacune d'une mesure de la pièce. Ces machines ne devinrent cependant des outils de précision qu'après l'introduction de verniers gradués sur les vis de commande. Il fallut ensuite attendre ces dernières années pour voir l'apparition d'un progrès considérable : la commande numérique.

La commande numérique est un procédé d'automatisation permettant le déplacement automatique des différents organes mobiles d'une machine vers des positions déterminées par leurs coordonnées. L'emploi des techniques de la commande numérique ne se limite d'ailleurs pas aux machines-outils, les premières machines à commande numérique ayant été les métiers à tisser de Jacquard ; le champ d'application de la commande numérique comprend tous les domaines où se posent des problèmes de guidage et de déplacement : mouvements des radars, écartement des rouleaux de laminoirs , etc... En restant dans le domaine des machines-outils, la commande numérique est une forme d'automatisation qui vient compléter celle qui s'applique aux productions de grandes séries qui est moins récente et qui a donné naissance aux machines spéciales, destinées à la réalisation d'un seul type de pièce. Ces machines spéciales manquent de souplesse et sont condamnées le jour où l'on abandonne la fabrication du modèle de pièce qu'elles usinent. Les machines à commande numérique ont au contraire pour caractéristique la souplesse convenant parfaitement à la production de pièces en petites séries ; on a pensé pouvoir allier

cette souplesse à la production en grande série en équipant des machines-transfert de commande numérique pour les rendre adaptables à plusieurs fabrications, mais l'intérêt pratique de telles réalisations reste discutable.

La technique de la commande numérique a été créée pour résoudre des problèmes techniques d'usinage de pièces aux formes complexes : cames de pompes d'injection de moteurs d'avion, dont la fabrication était presque irréalisable par les machines classiques. Les premières études furent entreprises au M.I.T. et permirent de déterminer les conditions à remplir par des machines permettant ces usinages complexes. Le problème consistait à combiner les mouvements d'un outil simultanément selon plusieurs axes de coordonnées, ce qui ne peut être demandé à un ouvrier. Afin d'assurer le guidage précis de l'outil il fallut définir la trajectoire de l'outil par un grand nombre de points par lesquels l'outil passait successivement. Le grand nombre d'informations à manipuler exigeait l'association d'un calculateur électronique à la machine-outil, ceci à une époque où la technique des servomécanismes et celle des calculateurs étaient loin de ce qu'elles sont actuellement.

On équipa d'abord des machines universelles classiques de dispositifs de commande numérique et les résultats obtenus conduirent à la réalisation de machines spécialement conçues pour la commande numérique qui étaient capables d'engendrer des surfaces plus ou moins compliquées. Ce n'est que plus tard que la généralisation de ces techniques fit apparaître des machines plus simples. Les progrès faits dans les servomécanismes, les calculateurs et la programmation ainsi que l'adjonction de changeurs automatiques d'outils et d'axes de mouvement complémentaires ont permis une augmentation considérable des possibilités des machines-outils à commande numérique.

1.2. PRINCIPES DE BASE.

La commande numérique reste très simple dans son principe : toutes les dimensions de la pièce à façonner sont définies par des données numériques (ou cotes). L'automatisme commandé numériquement fonctionne directement à partir de ces données numériques qui constituent le "langage machine".

On peut considérer qu'il existe trois grandes classes de commande numérique des déplacements (48)[1] suivant le contrôle exercé sur l'exécution du mouvement.

La figure 1 représente schématiquement un chariot de machine muni de ses dispositifs auxiliaires habituels : vis d'avance, glissières, etc... et son élément moteur. Nous avons là un exemple de fonctionnement en chaîne ouverte pour le déplacement d'un chariot selon un axe de la machine-outil. Le système d'entraînement S est conçu de telle façon qu'il tourne d'un angle défini pour une impulsion reçue. Le chariot se déplace alors d'une quantité correspondant à l'information reçue i. Cette méthode constitue un procédé très simple pour la construction d'une machine-outil à commande numérique, cependant la précision finale dépend essentiellement du dispositif de commande (par exemple un moteur pas-à-pas ou un système de roue à crémaillères) et on ne l'emploie encore que très peu dans l'état actuel de la technique.

Les deux autres types de commande sont des commandes en boucle fermée qui contrôlent la position du chariot au moyen d'instruments de mesure indépendants du système d'entraînement.

---

1 -Les chiffres entre parenthèses permettent de se référer à la bibliographie.

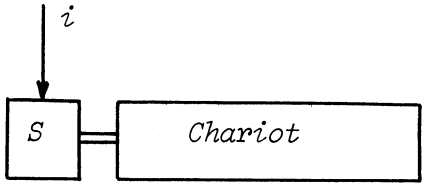


Figure 1 . *Fonctionnement en chaîne ouverte. (48).*

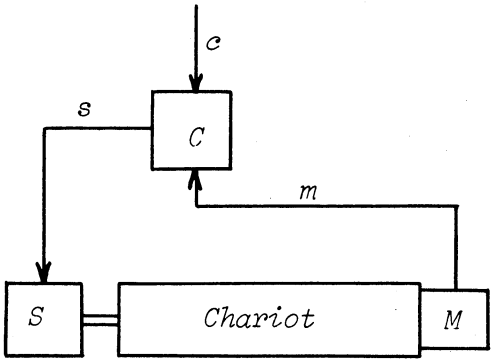


Figure 2 . *Commande par mesure du déplacement du chariot (48).*

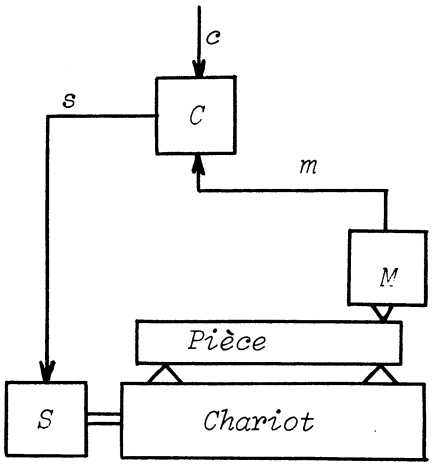


Figure 3 . *Commande par mesure de la pièce usinée (48).*

La figure 2 représente schématiquement un exemple de commande par mesure du déplacement du chariot. On a associé au chariot un dispositif de mesure M qui évalue la position courante du chariot ou encore le déplacement à effectuer pour atteindre la position finale. Cette mesure m est alors comparée à une grandeur de consigne c dans le comparateur C. Ce dernier envoie un signal de réaction ("feedback") s qui commande le système d'entraînement S. Le comparateur C peut être de deux types. Dans un premier type de comparateur ne fournit de signal qu'à la coïncidence de la grandeur de consigne et de la grandeur de mesure ; ce signal peut alors provoquer l'arrêt du système d'entraînement. Dans un second type le comparateur fournit un signal tant que la grandeur de consigne et la grandeur de mesure ne coïncident pas.

La figure 3 représente un exemple de commande par mesure de la pièce usinée P. La méthode précédente de mesure du déplacement du chariot pour l'évaluation des dimensions de la pièce usinée n'est pas obligatoirement la meilleure. Le chariot et la table d'une machine-outil ne sont que des supports où est fixée la pièce et il peut y avoir dilatation ou resserrement au cours du travail de coupe. Pour obtenir une meilleure précision il faut prélever directement les mesures sur la pièce usinée au moyen du capteur du système de mesure M. On transmet alors cette mesure m au comparateur C qui la compare à la valeur de consigne c, et envoie alors un signal s au système d'entraînement S comme dans le cas précédent. Cette méthode se heurte pourtant à des difficultés dues aux conditions de travail (chaleur dégagée, liquide de refroidissement, copeaux, etc...) et aux formes parfois biscornues des pièces usinées. C'est pourquoi elle n'est appliquée dans la pratique qu'aux machines travaillant sur des surfaces cylindriques ou des feuilles planes.

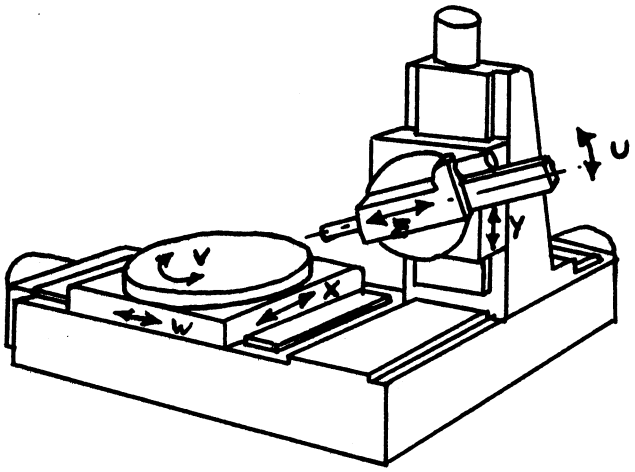
1.3. MODES DE FONCTIONNEMENT DES MACHINES-OUTILS.

Sans aller plus avant nous pouvons déjà considérer la machine-outil comme un système traitant l'information ; une machine-outil classique traitera par exemple l'information fournie par un palpeur et une came tandis qu'une machine-outil à commande numérique traitera l'information des données numériques qui constituent en quelque sorte son lan-

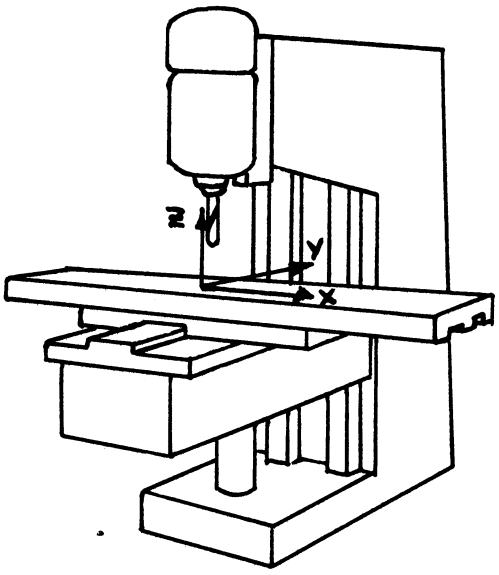
gage machine. Une machine-outil comporte la plupart du temps plusieurs éléments mobiles se déplaçant suivant des axes. Une machine-outil à commande numérique peut ainsi comporter plusieurs axes dont le positionnement est commandé numériquement (5) mais le principe de commande des axes reste simple. (Voir figure 4)

La commande des machines-outils est réalisée au moyen de deux types d'ordres ou d'informations : les informations de commutation et les informations de déplacement. Les informations de commutation agissent directement sur la machine, comme par exemple le déclenchement de la rotation de la broche à une certaine vitesse. Les informations de déplacement n'opèrent sur la machine-outil qu'après un certain nombre d'opérations sur les informations (comme par exemple des opérations de calcul).

En reprenant l'exemple de commande par mesure de parcours du chariot et en le modifiant nous obtenons la figure 5 qui est un exemple du traitement interne de l'information par la machine-outil pour le parcours d'un axe. Un organe d'entrée E reçoit les informations de fonctionnement i et les sépare en grandeur de consigne c et en informations de commutation d. Le comparateur C reçoit de l'organe de mesure M une grandeur de mesure m qu'il compare à la grandeur de consigne c, et envoie un signal s au système d'entraînement du chariot S. Une machine-outil comportant plusieurs axes de coordonnées nous pouvons avoir deux modes de fonctionnement différents de ces machines : les machines sans relation de fonctionnement entre les déplacements selon les différents axes de coordonnées et les machines où il existe une relation de fonctionnement entre les mises en position sur des axes de coordonnées différents.



b) Centre d'usinage à six axes



a) Exemple de machine-outil

Figure 4 . Définition des axes des machines (5).

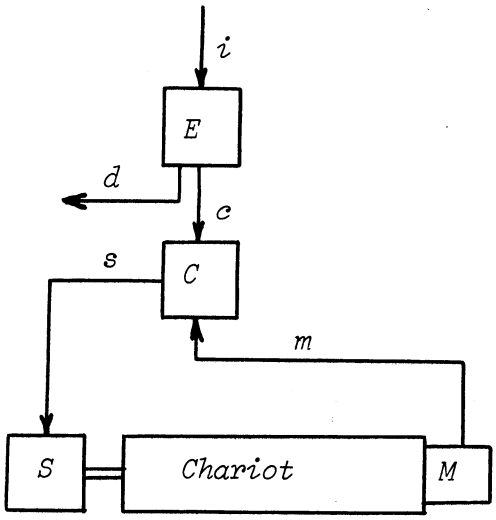


Figure 5 . *Déplacement selon un axe (48).*

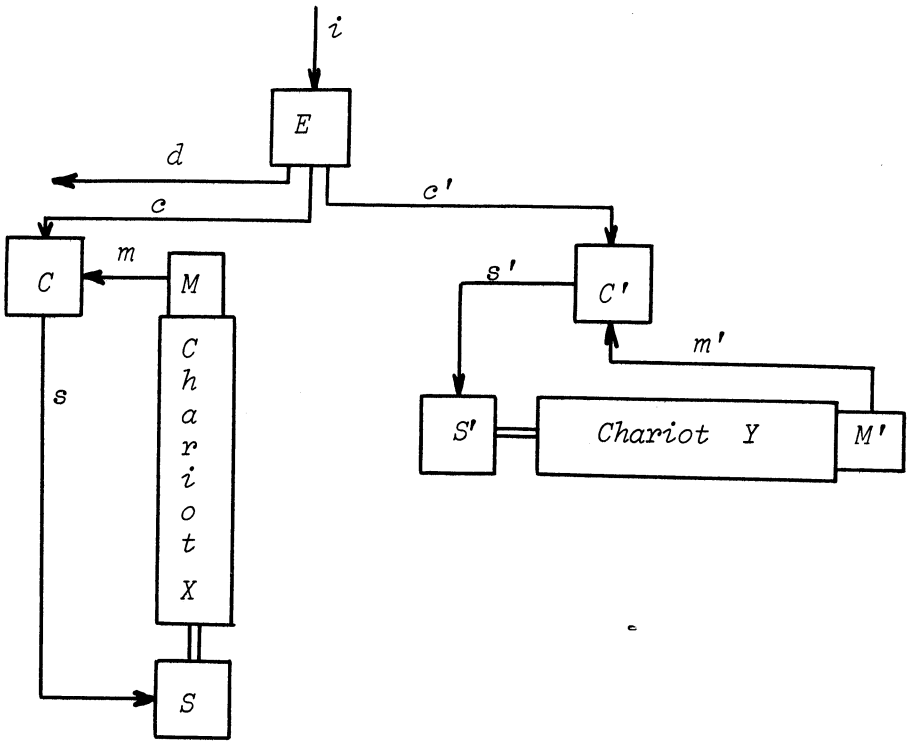


Figure 6 . *Commande suivant deux axes indépendants (48).*

1.3.1. Mise en position sans relation entre les déplacements sur les différents axes de coordonnées.

La figure 6 représente un exemple de commande sur deux axes sans relation de fonctionnement entre les parcours. Nous retrouvons le même principe déjà appliqué sur un seul axe.

Le dispositif d'entrée E reçoit les informations de travail i qu'il décompose en informations de commutation d et en informations de parcours c. Les informations de parcours c sont traitées par chacun des deux dispositifs de déplacement de la même manière et d'une façon indépendante.

Les commandes sans relation de fonctionnement peuvent être encore divisées en deux groupes :

a) La machine réalise des mises en position successives de la pièce par rapport aux outils, ceux-ci ne travaillant pas pendant le déplacement des chariots. L'opération d'usinage n'a lieu qu'une fois la position souhaitée atteinte. C'est le fonctionnement point à point (par exemple le fonctionnement d'une perceuse automatique à coordonnées).

b) En plus du fonctionnement de point à point la machine peut effectuer une opération d'usinage au cours du déplacement d'un chariot. L'outil peut donc travailler lorsque les déplacements se font selon une parallèle à l'un des axes commandés numériquement. On appelle ce fonctionnement de point à point et paraxial (par exemple le fonctionnement des aléseuses, des fraiseuses, des tours).

1.3.2. Mise en position avec relation entre axes de déplacement.

Ce mode de fonctionnement est celui des machines qui possèdent un dispositif d'asservissement entre les déplacements suivant les différents axes commandés numériquement : il permet la réalisation de surfaces plus ou moins complexes. L'outil est conduit selon un trajet dont

la forme est définie à volonté. C'est ce qu'on appelle le contournage.

La figure 7 donne un exemple de commande avec relation de fonctionnement entre les parcours des deux axes commandés numériquement. Le dispositif d'entrée E reçoit les informations de travail i et les sépare en informations de parcours c, et en informations de commutation d qui sont utilisées de la même manière que dans le cas des commandes point à point ou paraxiales. Les informations de parcours c sont fournies à un interpolateur (ou générateur de courbes) I qui doit produire entre deux points repères une forme de courbe déterminée (par exemple un segment de droite ou un arc de cercle), et qui doit également calculer la vitesse d'avance résultante à partir de celle des différents chariots et agir sur les composantes de la vitesse pour obtenir la trajectoire désirée. L'interpolateur constitue un fait un petit calculateur numérique ou analogique qui dirige les déplacements sur les différents axes.

1.4. INSTRUCTIONS DE COMMANDE NUMERIQUE.

Les machines-outils à commande numérique sont conçues pour traiter des informations numériques. Ces dernières doivent décrire pleinement les coordonnées des points à atteindre ou des déplacements à faire ainsi que diverses fonctions auxiliaires à effectuer.

1.4.1. Ordres donnés à la machine.

Les ordres numériques fournis à la machine doivent lui permettre d'effectuer tous les mouvements possibles pour elle. Une machine comporte habituellement plusieurs organes mobiles selon des axes commandés numériquement. Les informations numériques préciseront donc à la machine les coordonnées à atteindre sur les différents axes, et comporteront aussi des ordres auxiliaires comme par exemple le changement automatique d'un outil. On pourra également à l'aide de ces ordres auxiliaires indiquer une suite d'opérations à effectuer ou cycle d'usinage ; un cycle de perçage ferait par exemple les opérations suivantes :

- avance rapide jusqu'à la surface de la pièce
- avance de travail jusqu'au fond du trou
- arrêt de l'avance
- retour de la broche en vitesse rapide

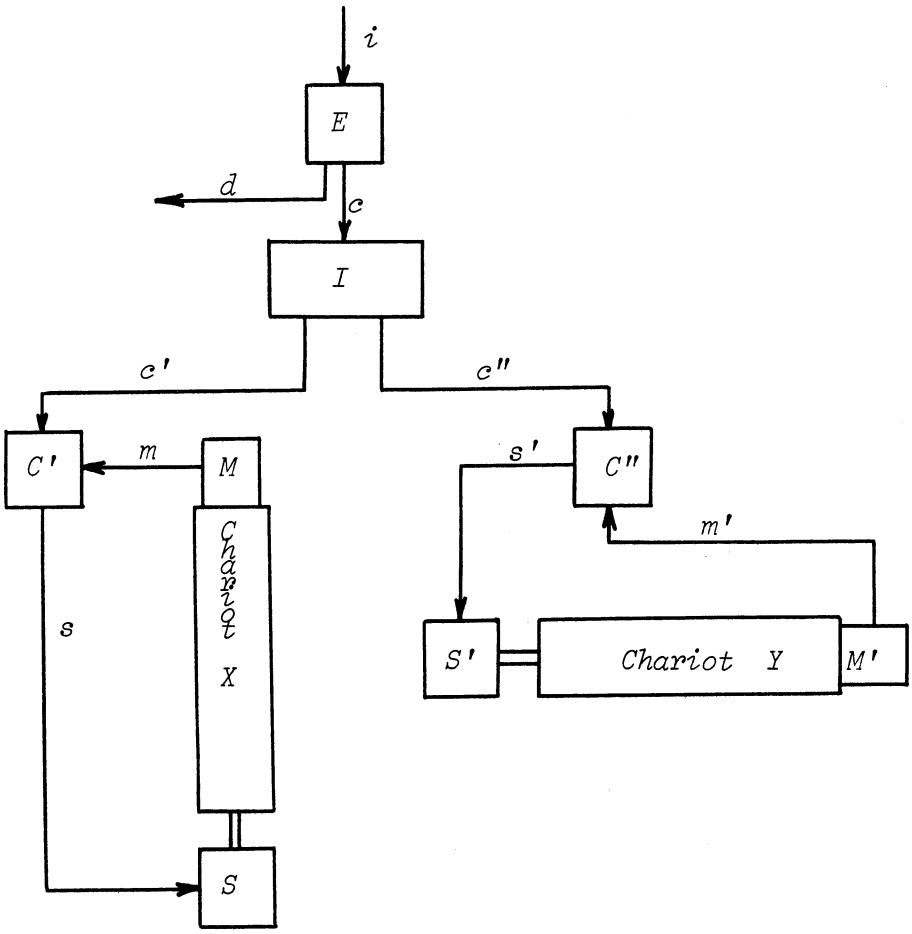


Figure 7. Commande avec relation de fonctionnement entre parcours (48).

Ces ordres numériques une fois élaborés devront être adaptés à la machine utilisée qui peut fonctionner suivant deux modes différents de programmation : la programmation absolue où toutes les informations de position auront une origine commune et la programmation relative où chaque information numérique de position représentera un déplacement par rapport à la position précédente.

Il existe alors deux moyens de transmettre ces informations numériques à la machine. Le premier moyen est entièrement manuel ; il consiste à afficher au pupitre de la machine à l'aide des boutons de commande, les coordonnées désirées. La machine exécute alors l'ordre donné. Elle se comporte en fait comme une machine-outil classique perfectionnée, plus facile et plus agréable à conduire. Le second moyen consiste à utiliser un support d'information comme les bandes ou les cartes perforées. Cette méthode diminue considérablement les temps morts de la machine et permet une préparation des ordres numériques dans de meilleures conditions.

1.4.2. Supports de programmation.

On utilise divers supports d'information pour transmettre les ordres numériques aux machines. Les cartes perforées sont utilisées dans certains cas, surtout pour des systèmes de commande numérique à positionnement continu. Le support d'information le plus utilisé pour les machines à positionnement point à point est la bande perforée. On utilise également les bandes magnétiques pour certains systèmes à positionnement continu.

1.4.2.1. Bandes perforées.

Le principe de la bande perforée ou du ruban perforé est connu depuis longtemps. La bande perforée est souple, pratique, économique et robuste car elle peut être faite de matériaux plastiques ; c'est pourquoi on l'utilise beaucoup à l'heure actuelle. Il existe d'ailleurs un grand nombre de systèmes de perforation et de lecture des bandes perforées. La bande utilisée universellement a un pouce de large et huit canaux.

Il existe cependant plusieurs codes de perforation pour ces bandes. Le code EIA à six bits d'information reste le plus utilisé bien que certains se servent du code ASCII et que l'organisation internationale de normalisation ait élaboré un code normalisé ISO à sept bits d'information qui devrait se répandre (4). Les bandes de papier sont suffisamment résistantes pour permettre un minimum de dix à vingt passages successifs sur le lecteur ; les bandes en matières plastiques ou à renfort d'aluminium sont évidemment beaucoup plus robustes.

Les lecteurs de bandes perforées sont très nombreux mais on peut les classer en deux catégories : les lecteurs électromécaniques très sûrs et les lecteurs photoélectriques aussi sûrs qui demandent moins d'entretien et dont la vitesse de lecture peut être dix fois supérieure à celle des lecteurs électromécaniques. Les lecteurs pratiquent en général une lecture "ligne à ligne" ; comme la programmation d'une seule opération des machines à commande numérique requiert plusieurs "lignes" de bande à 8 canaux, il est nécessaire d'avoir une mémoire pouvant emmagasiner toutes les données d'un "bloc d'information". Il existe cependant des lecteurs qui donnent accès directement à l'ensemble d'un bloc ; ceci évite la nécessité d'une mémoire, la bande pouvant être considérée elle même comme une mémoire avec un lecteur multiligne (50).

1.4.2.2. Bandes magnétiques.

Les bandes magnétiques utilisées en commande numérique sont semblables à celles utilisées avec les ordinateurs. Chaque fois qu'il s'agit de manipuler une grande quantité d'informations on préfère la bande magnétique à la bande perforée. Elle peut conserver des informations sous une forme beaucoup plus dense que la bande perforée. Cependant son emploi est plus coûteux à cause des appareils d'enregistrement et de lecture, et c'est un support plus fragile dans les conditions de travail d'un atelier.

1.5. ORGANISATION GENERALE DE LA COMMANDE NUMERIQUE DES MACHINES.

Les systèmes de commande schématiques que nous avons vus doivent s'insérer dans une structure plus générale qui permette de faire ressortir la manière dont sont traitées les informations depuis les

données définissant la pièce jusqu'à la réalisation de celle-ci. Nous emprunterons ici au livre "Commande numérique des machines-outils" du professeur W. Simon trois schémas types qui décrivent les organisations générales les plus courantes et qui constituent la figure 8.

Les données nécessaires à la réalisation de la pièce peuvent être classées en deux catégories : les données de forme qui décrivent la géométrie de la pièce à usiner et les données technologiques qui décrivent le mode opératoire (vitesses d'avance, de broche, profondeurs et nombre de passes, etc...). La figure 8A donne un exemple de l'organisation générale des commandes point à point et paraxiales pour lesquelles il n'y a pas de relations entre les différents axes de la machine, dont deux seulement, X et Y, ont été représentés. Le traitement interne des données à partir du dispositif d'entrée E a déjà été vu ; nous devons cependant noter que ce dispositif d'entrée peut être soit manuel (par touches), soit automatique (par lecture du support d'information utilisé).

Les figures 8B et 8C représentent deux organigrammes de systèmes de contournage, ils diffèrent principalement l'un de l'autre par la place de l'interpolateur I. Le traitement interne de la figure 8B a déjà été vu plus haut ; le traitement externe des données sera fait d'une manière semblable à la commande point à point et paraxiale : suivant la complexité de la pièce à usiner les calculs seront plus ou moins importants et l'on pourra envisager de se faire aider par un ordinateur. On calculera ainsi un ensemble de points principaux de la pièce à réaliser : l'interpolateur déterminera alors les éléments intermédiaires entre les points repères et calculera les vitesses des différents axes pour obtenir une vitesse d'avance constante le long de la trajectoire. Ce type de système permet en général d'apporter des corrections pour tenir compte du diamètre réel de l'outil utilisé.

La figure 8C représente un système de contournage à interpolateur externe. Le traitement de l'information se fait principalement à l'extérieur de la machine et l'interpolateur externe conduit en général à transmettre au dispositif d'entrée E un grand nombre d'informations

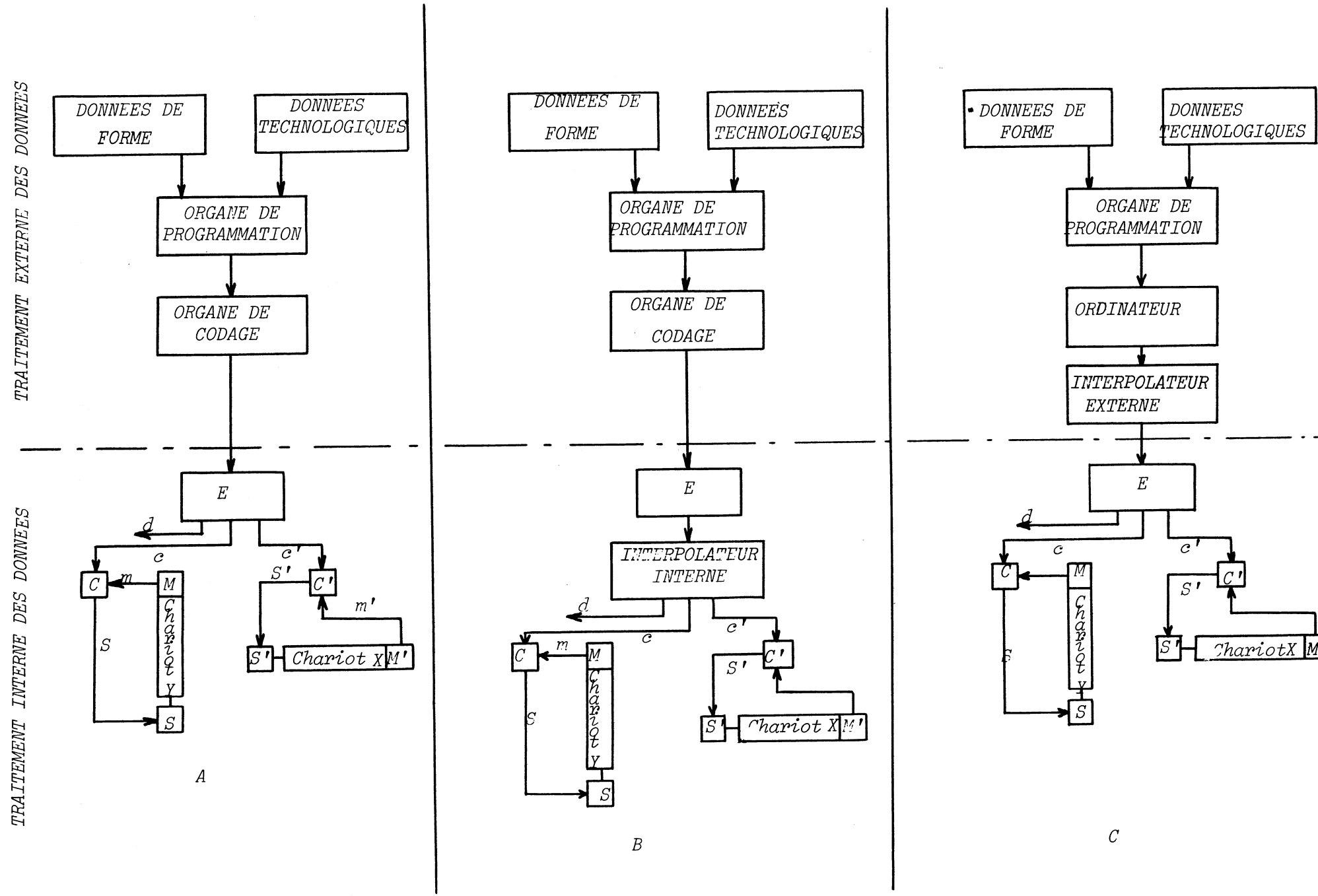


Figure 8 . ORGANISATION GENERALE DE LA COMMANDE NUMERIQUE DES MACHINES (48).

pouvant nécessiter comme support d'information des bandes magnétiques (37). Ces systèmes ne permettent en général d'ailleurs pas d'effectuer des corrections dues au diamètre réel de l'outil.

Le plus utilisé de ces systèmes est actuellement le système schématisé par la figure 8A ; la plupart des systèmes de contournage courants correspondent à la figure 8B.

1.6. PRINCIPAUX SYSTEMES.

Nous avons vu que les machines à commande numérique pouvaient être classées en deux grandes catégories (50) : les machines à mise en position point à point et les machines à mise en position continue ou de contournage.

1.6.1. Mise en position point à point.

1.6.1.1. Systèmes analogiques.

Les ordres de commande de la machine sont des informations numériques ; celles-ci sont transformées à l'entrée en signaux analogiques. Tous les organes de la machine fonctionnent alors de façon analogique. La figure 9 donne une représentation schématique d'un tel système. On a introduit un transformateur numérico-analogique après le dispositif d'entrée E (d représente les informations de commutation). Le flux des informations n'est pas modifié, les organes de mesure, de comparaison et d'entraînement sont alors des systèmes analogiques. Les grandeurs analogiques le plus fréquemment utilisées sont des tensions électriques, des angles de phase entre deux tensions ou des fréquences (36) ; les appareils de mesure peuvent être rotatifs ou linéaires. Les systèmes analogiques permettent des performances remarquables grâce à la linéarité des caractéristiques de la chaîne de commande. Les signaux électriques mis en jeu sont des tensions permanentes pouvant être plus facilement protégées contre des perturbations parasites que les systèmes à trains d'impulsions. Chaque fois qu'il est nécessaire de synchroniser deux mouvements avec une grande précision l'emploi des systèmes analogiques s'impose. Cependant les systèmes analogiques sont assez coûteux.

1.6.1.2. Systèmes numériques codés.

Les ordres de commande de la machine sont des informations numériques. La comparaison des ordres et des mesures ainsi que l'élaboration des signaux effectuées par le comparateur C (figure 10), se font dans le domaine numérique. La transformation numérique-analogique du signal s est faite au niveau des organes de commande du moteur. La transformation analogique-numérique produisant la grandeur de mesure m est faite par le capteur de position. Comme les systèmes de commande analogiques, les systèmes de commande numériques à codage de la position présentent l'avantage de travailler sur des signaux de position permanents. C'est un facteur important de sécurité que les systèmes à comptage ne possèdent pas. Ces systèmes à codage ont encore un autre avantage sur les systèmes analogiques : ils permettent de supprimer la transformation numérique-analogique des ordres de déplacement, qui n'a pas en général de solution simple.

Comme pour les systèmes analogiques le gros inconvénient des systèmes à codage reste le prix du capteur de position ; c'est ce qui explique pourquoi peu de machines-outils utilisent ces systèmes. La commande numérique par codage est plus employée par les appareillages spéciaux (pilotage des antennes de radar par exemple).

1.6.1.3. Systèmes numériques à comptage.

Dans ces systèmes le capteur ne donne pas l'image de la position du chariot mais l'image du déplacement du chariot. Il fournira une impulsion chaque fois que le chariot se déplacera d'une valeur donnée. On utilise quatre configurations principales (37). La figure 11a montre un système où un compteur-décompteur additionnant ou soustrayant les impulsions qu'il reçoit donne une valeur de la position courante du chariot de la machine. Cette valeur est comparée à la coordonnée c dans le comparateur C qui fournit un signal s au système de commande du moteur S (d représente les informations de commutation).

La figure 11b reprend l'exemple précédent en le simplifiant. Le sens du déplacement à effectuer est programmé et transmis directement au

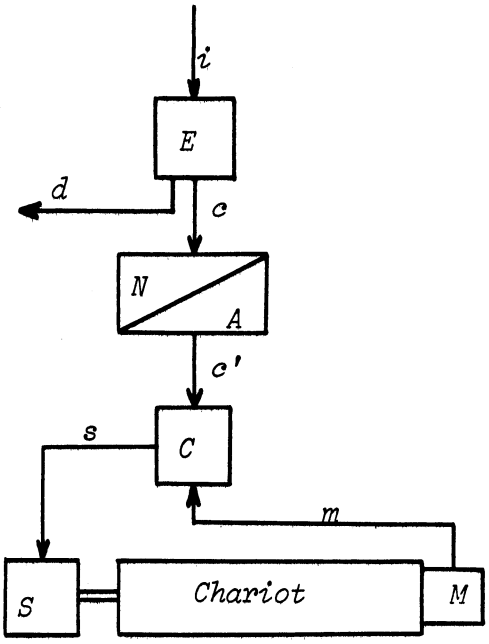


Figure 9 . *Système analogique (48).*

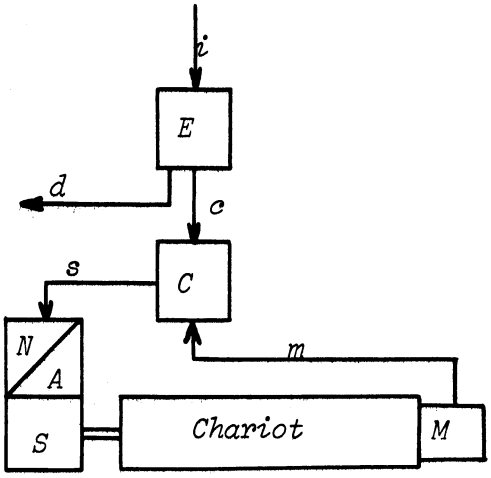
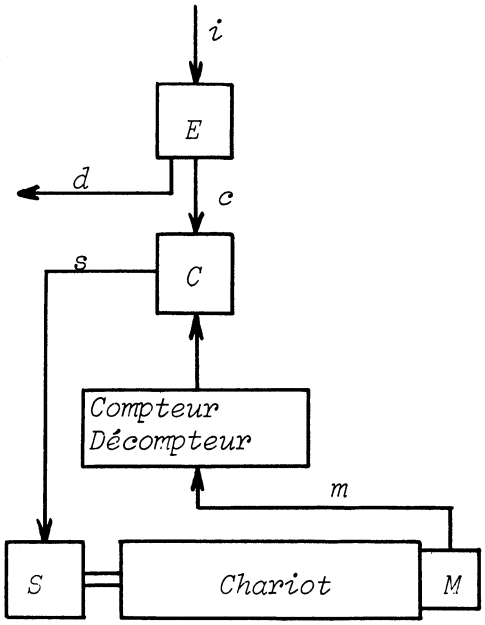
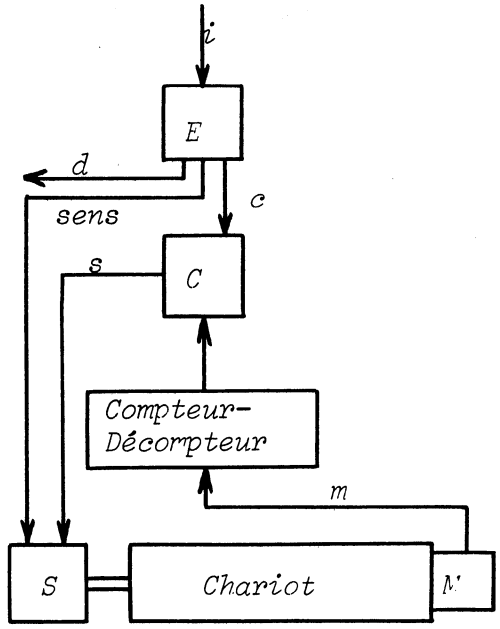


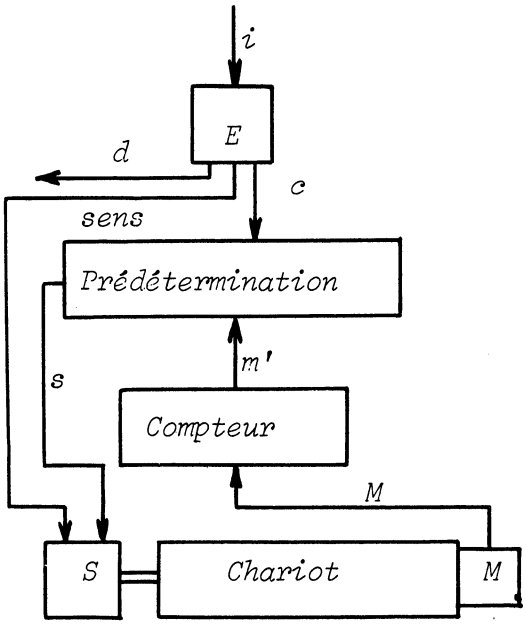
Figure 10 . *Système numérique codé (48).*



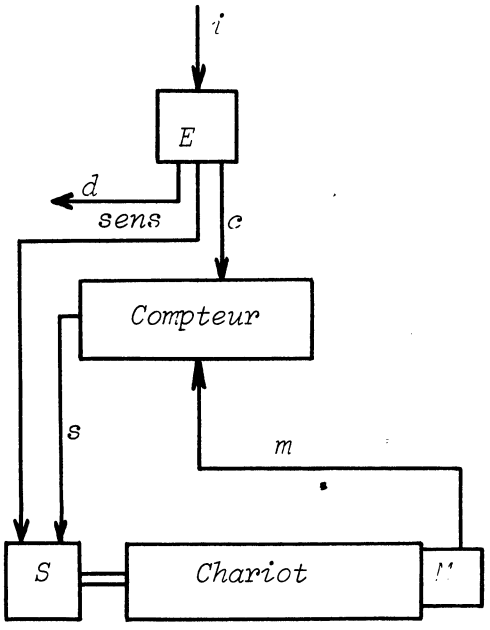
a



b



c



d

Figure 11 . *Systèmes numériques à comptage (37).*

système de commande du moteur. Le comparateur indique alors seulement la coïncidence des valeurs prédéterminées et comptées.

La figure 11c donne un exemple de système où le compteur est remis à zéro au départ et où la grandeur c n'indique plus une coordonnée à atteindre mais un déplacement à effectuer. L'ordre d'arrêt du mouvement ne sera donné que lorsque le compteur aura enregistré le nombre d'impulsions requis par la valeur de prédétermination.

La figure 11d montre un système où le circuit de prédétermination est supprimé et où le compteur fonctionne en soustracteur. On charge le compteur de la valeur du déplacement à effectuer et le dispositif de mesure M envoie des impulsions qui le ramènent progressivement à zéro. L'apparition du zéro détermine l'arrêt du chariot.

La méthode de comptage présente l'inconvénient majeur d'une faible sécurité de fonctionnement ; ces systèmes sont sensibles aux pertes d'impulsions ainsi qu'aux impulsions parasites. Les organes mécaniques et électriques de tels systèmes se révèlent être bien meilleur marché que les organes des systèmes analogiques et des systèmes à codage. L'économie réalisée et le fait d'être parvenu à maîtriser la sensibilité aux parasites expliquent pourquoi ces systèmes sont actuellement les plus répandus.

1.6.1.4. Systèmes mixtes.

Ce sont des systèmes utilisant à la fois des techniques analogiques et des techniques de comptage. L'un des plus connus est le système Mark Century de la General Electric (50). Ces systèmes donnent l'avantage d'une sécurité supérieure à celle des systèmes à comptage ; ils ont un appareillage plus simple que les systèmes analogiques mais demeurent tout de même assez onéreux.

1.6.2. Mise en position continue.

Le domaine d'application de la mise en position continue est actuellement plus limité que celui de la mise en position point à point. Le fraisage est le domaine où l'on a vu apparaître le plus grand nombre

d'applications mais c'est aussi le domaine où les problèmes sont les plus complexes car c'est lui qui implique la coordonation du plus grand nombre de mouvements selon des axes différents. Il existe deux grandes classes de systèmes de mise en position continue : les systèmes à calculateur extérieur et les systèmes à calculateur intégré.

1.6.2.1. Systèmes à calculateur extérieur.

On fournit à la machine toutes les informations de déplacement nécessaires utilisables directement, ce qui représente un volume considérable d'information. Ce sont des systèmes correspondant au cas de la figure 8C. Le système le plus connu est celui de Ferranti en Grande Bretagne, dans lequel les informations sont fournies sur bande magnétique. Les avantages de ces systèmes sont les suivants : la machine, ne comportant pas d'interpolateur, est simplifiée et son prix est réduit ; un calculateur extérieur (aussi appelé "directeur") peut alimenter en programmes une vingtaine de machines ; il est également plus facile de modifier un calculateur que vingt machines.

1.6.2.2. Systèmes à calculateur intégré.

Ce sont des systèmes à interpolateur intégré dont un exemple a été donné par la figure 8B. On ne fournit à la machine que les principaux points de la trajectoire, ce qui représente un volume d'information relativement réduit. Il y a trois types d'interpolation : interpolation linéaire qui calcule les points intermédiaires sur un segment de droite, interplation circulaire calculant les points intermédiaires sur un secteur de cercle et pouvant faire l'interpolation linéaire (qui représente un cas particulier de l'interpolation circulaire), interpolation parabolique de réalisation simple et pouvant également faire l'interpolation linéaire.

Le prix des machines à interpolateur intégré n'est pas beaucoup plus élevé que celui des machines qui n'en comportent pas et ce système permet une autonomie plus poussée des machines.

1.7. ELABORATION DES INSTRUCTIONS DE COMMANDE.

1.7.1. Types de programmation.

Il existe deux méthodes de programmation des machines-outils à commande numérique : la programmation "manuelle" et la programmation "automatique".

La programmation "manuelle" consistera à calculer (manuellement ou à l'aide d'un calculateur de bureau) les coordonnées des différentes positions à atteindre et à perforer manuellement ces coordonnées sur le ruban. Cette méthode introduit des risques d'erreurs au calcul, puis à la perforation de la bande. Elle est encore assez répandue pour les systèmes à positionnement point à point.

La programmation "automatique" nécessite l'emploi d'un calculateur électronique. Elle consiste à décrire à l'aide d'un langage symbolique évolué les formes géométriques à suivre et à laisser le soin au calculateur d'effectuer le calcul des diverses coordonnées et la perforation de la bande. Les risques d'erreurs sont ainsi bien diminués.

Une fois la bande perforée par l'une ou l'autre de ces méthodes, il faut s'assurer qu'elle décrit bien le travail voulu et qu'il ne subsiste pas d'erreurs ; pour ce faire on utilise une table traçante ou on essaye la bande perforée sur la machine sans monter les outils.

1.7.2. Problèmes de détermination de la trajectoire de l'outil.

Les systèmes de commande numérique point à point ne posent pas de difficultés particulières pour la détermination des instructions de commande ; par contre les systèmes de mise en position continue posent quelques problèmes particuliers comme : définition de la trajectoire de l'outil, calcul d'un nombre de points suffisant pour la précision voulue, définition d'une surface complexe.

Le profil théorique d'une pièce correspondra directement au dessin de cette pièce et sera en général constitué d'éléments géométriques

simples (segments de droites, arcs de cercles). A partir de ce profil théorique on détermine un profil pratique constitué en général d'une suite de segments rectilignes ou curvilignes dont la définition est connue de la machine. Si le profil théorique est un arc de cercle et si la machine ne dispose que d'un interpolateur linéaire, le profil pratique sera un polygone approchant du cercle avec une certaine précision, tandis que si la machine possède un interpolateur circulaire, le profil pratique sera l'arc de cercle du profil théorique. Suivant les possibilités de la machine le profil pratique sera donc plus ou moins éloigné du profil théorique.

La trajectoire pratique du centre de l'outil est déduite du profil pratique en reportant sur chaque normale à ce profil, la valeur du rayon de l'outil. Pour obtenir une précision suffisante l'interpolateur calcule les coordonnées d'un nombre de points plusieurs centaines de fois supérieur au nombre de points caractérisant la définition de la trajectoire.

La définition d'une surface se ramène à la définition d'une trajectoire permettant d'engendrer cette surface. Une des méthodes les plus anciennes consiste à décomposer cette surface en une série de profils bidimensionnels qui représentent la section de cette surface par une série de plans parallèles à l'un des plans de référence de la machine. On détermine différents profils en faisant varier la cote de ce plan par rapport au plan de référence, et si les variations de cote du plan mobile sont assez faibles, on engendre un profil pratique voisin du profil théorique. Cette méthode allie la commande continue par mouvements coordonnés selon deux axes et la commande discontinue par points selon le troisième axe ; elle est quelquefois appelée méthode des courbes de niveau. Elle ne permet pourtant pas de traiter tous les cas pratiques qui sont souvent très complexes. Il faut alors employer une méthode de définition mathématique des courbes génératrices de la surface et une loi de déformation des courbes génératrices pour engendrer la surface (15).

1.8. IMPORTANCE DE LA COMMANDE NUMERIQUE.

En reprenant ce qui a été dit sur les machines-outils à commande numérique, nous pouvons schématiser un système de commande numérique pour machine-outil par la figure 12. Et en résumé nous pouvons dire que la commande numérique marque un progrès important dans l'automatisation des machines-outils destinées à l'usinage en petites et moyennes séries. Elle permet de supprimer la pose de taquets, règles porte-taquets, cames et butées sur la machine et donc de réduire les temps de réglage prohibitifs. Les avantages résultant de l'emploi d'une machine à commande numérique sont difficilement chiffrables. Des expériences pratiques ont fait ressortir les avantages suivants :

- amélioration du rapport temps de coupe/temps d'arrêt ; gain important sur les temps morts pendant lesquels la machine ne fait pas de copeaux : élimination des temps d'arrêt nécessaires à l'ouvrier pour vérifier ses cotes, enchaînement des opérations l'une derrière l'autre sans perte de temps.

- diminution des rebuts de pièces.

- économie de place dans l'atelier, une machine à commande numérique remplaçant en général plusieurs machines conventionnelles.

- réduction du besoin d'outillages spéciaux (montages, gabarits).

- organisation améliorée du fait de la connaissance à l'avance du temps d'occupation de la machine.

- diminution des stocks (fabrication des pièces de rechange à la demande).

Il arrive cependant que l'on accorde aux machines-outils à commande numérique une importance plus grande qu'elles n'ont en réalité. Elles sont beaucoup plus pratiques que les machines conventionnelles et on a tendance à reporter sur elles tout le travail qu'elles peuvent faire ; pour amortir ces machines qui représentent un investissement important on a

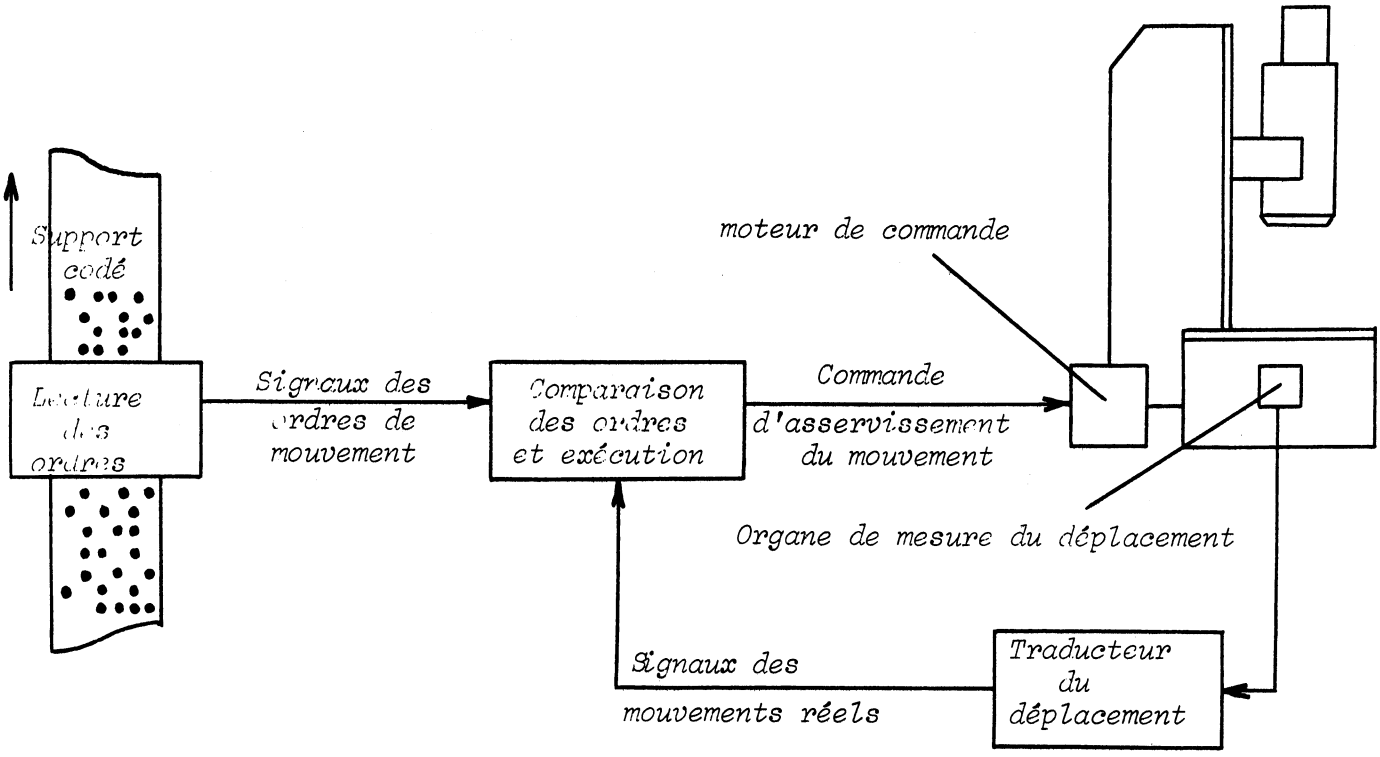


Figure 12 . *Système de commande numérique pour machine-outil (36).*

tendance à les faire travailler en plusieurs équipes.

Pour donner une idée du développement de la commande numérique citons quelques chiffres. En 1967 on évaluait à 13 000 le nombre de machines à commande numérique aux Etats Unis, pour 1 350 en Grande Bretagne, 650 en Allemagne, 460 en France et 300 en Suède. Ces chiffres seraient passés actuellement à 20 000 aux Etats Unis et à 600 en France.

La commande numérique est une technique d'automatisation qui ne peut se concevoir qu'avec du matériel de qualité. La qualité du matériel à automatiser doit être plus grande que celle du matériel conventionnel ; ceci s'applique à la définition des éléments, aux tolérances d'usinage et de montage et à la résistance à l'usure qui détermine le maintien de bonnes caractéristiques dans le temps. La qualité des éléments de l'appareillage de commande doit être étudiée d'abord du point de vue de la sécurité de fonctionnement et seulement ensuite, des performances désirées.

CHAPITRE 2

PROGRAMMATION EN COMMANDE NUMERIQUE

2.1. GENERALITES.

Les différentes phases de la production d'une pièce mécanique comportent toujours en premier lieu la conception de la pièce par le bureau d'études (qui peut d'ailleurs être faite au moyen de systèmes d'aide à la conception par ordinateur existant à l'heure actuelle). La seconde phase sera l'établissement des gammes des procédés d'usinage et des opérations d'usinage par le bureau des méthodes : détermination de la suite des opérations et des modes d'usinage, détermination des opérations d'usinage individuelles, choix des outils, de la profondeur de passe, de la vitesse d'avance et de coupe, etc... La troisième phase sera celle des calculs géométriques : calcul de la trajectoire de l'outil en tenant compte des collisions possibles et des corrections de rayon de l'outil. La dernière phase sera l'usinage proprement dit. L'utilisation d'une machine-outil à commande numérique nécessitera toujours un travail de préparation du bureau des méthodes, qui sera plus ou moins long suivant le système de commande numérique utilisé.

Comme nous l'avons vu, les machines-outils à commande numérique possèdent plusieurs modes de travail. Le mode manuel peut revenir à utiliser la machine comme une machine-outil conventionnelle ou à afficher au pupitre de la machine les diverses informations nécessaires à une opération d'usinage : informations de position, de vitesses, fonctions préparatoires et auxiliaires (par exemple démarrage ou arrêt de la broche porte-outil, changement d'outil, etc...). Le mode de travil automatique utilise une bande perforée contenant toutes les informations nécessaires à l'exécution complète de la pièce. C'est ce dernier mode que nous utiliserons dans ce qui va suivre.

Nous avons également vu que l'élaboration des ordres numériques décrivant les opérations à effectuer pouvait se faire manuellement ou automatiquement. La programmation "manuelle" consiste à calculer (manuellement ou à l'aide d'un calculateur de bureau) les coordonnées des diffé-

rentes positions à atteindre et à perforer manuellement ces coordonnées sur le ruban. La programmation automatique nécessite l'emploi d'un ordinateur et consiste à décrire à l'aide d'un langage symbolique évolué les formes géométriques à suivre et à laisser le soin à l'ordinateur d'effectuer le calcul des diverses coordonnées et la perforation du ruban. La figure 13 schématise les différents modes d'utilisation d'une machine-outil à commande numérique.

2.2. PROGRAMMATION MANUELLE ET FORME DES ORDRES DE COMMANDE NUMERIQUE.

2.2.1. Programmation manuelle

En programmation manuelle on part du dessin de la pièce et du catalogue d'outils d'une machine à commande numérique donnée. On établit alors une gamme d'opérations détaillée permettant de déterminer les vitesses d'avance, les vitesses de coupe et les mouvements successifs des divers outils. On calcule ensuite les coordonnées correspondant à ces mouvements ; ces calculs préliminaires peuvent être faits à l'aide d'un petit calculateur de bureau. On transcrit alors ces coordonnées sur bande perforée par l'intermédiaire d'une machine à perforer. On obtient ainsi un programme sur bande perforée qui permettra d'effectuer les usinages successifs. La moindre erreur de calcul ou de perforation ayant des conséquences sur la pièce finie, plusieurs contrôles et vérifications sont nécessaires. La préparation de la bande sera longue et coûteuse car une pièce peut comporter un grand nombre de points à usiner successivement. Les problèmes relatifs à l'établissement de la bande perforée viennent en général du volume de calcul à effectuer et de la précision de ces calculs. Les calculs sont d'ailleurs relativement simples (transformation de coordonnées, décomposition en mouvements élémentaires, détermination de courbes) et ne présentent pas de difficultés mathématiques.

La commande numérique a pour principal objectif la réduction des délais séparant la conception et la réalisation d'une pièce. Il est rapidement apparu que l'établissement manuel des informations nécessaires à la commande de la machine devait être limité à quelques cas très simples d'usinage point à point. Dans tous les autres cas le nombre des informations

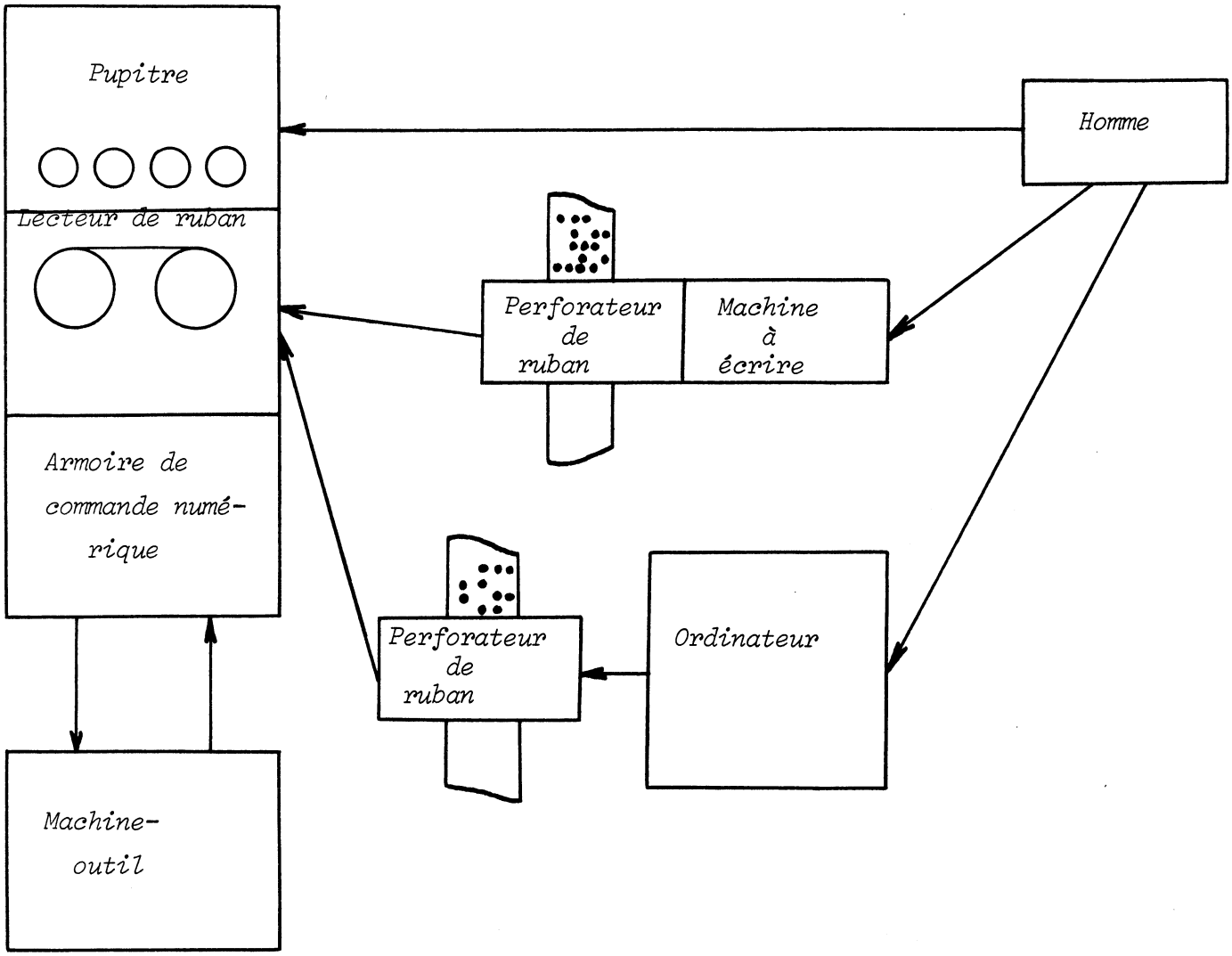


Figure 13 . *Modes d'utilisation des machines à commande numérique.*

à transmettre à la machine est tel que la préparation des données ne peut se concevoir sans un recours à l'ordinateur.

2.2.2. Ordres de commande numérique.

La bande perforée établie par le programmeur est transmise à l'organe de lecture de la machine-outil à commande numérique. Celle-ci exécute les opérations commandées conformément aux indications portées sur la bande. Les instructions de base d'une machine-outil à commande numérique sont habituellement des instructions de déplacement sur les axes de la machine commandés numériquement, des instructions de vitesse de déplacement, des instructions de vitesse de rotation de la broche porte-outil, des instructions d'appel de fonctions câblées (perçage, taraudage, arrosage, etc...) et des instructions de contrôle destinées au lecteur de ruban perforée.

Le programme qui se trouve donc sur bande perforée représente la suite d'opérations à effectuer par la machine à commande numérique. Ce programme est subdivisé en "blocs" d'information correspondant en général aux positionnements successifs ou aux usinages successifs. Chaque bloc correspond donc à une tâche élémentaire de la machine et se trouve placé entre deux instructions de contrôle "fin de bloc". Chaque bloc comporte un certain nombre de "mots" ou instructions de base de la machine. Un bloc peut contenir un nombre fixe ou variable de mots suivant le format utilisé.

Un bloc à format fixe (9) a une longueur constante. Aucun mot ne peut donc être omis même s'il n'y a pas de changement dans les données par rapport au bloc précédent. La signification d'un caractère est donnée par l'emplacement de celui-ci dans le bloc qui ne doit pas contenir de caractères alphabétiques. On trouve généralement dans un bloc les informations suivantes :

- numéro de bloc
- fonction préparatoire
- coordonnée X
- coordonnée Y

- coordonnée Z
- fonction vitesse d'avance
- fonction vitesse de broche
- fonction outil
- fonction auxiliaire

La figure 14 donne un exemple de bande perforée en format fixe.

Un bloc à format variable a une longueur variable (7,8) et ne comporte que les instructions (ou mots) à exécuter effectivement. Il est constitué de la manière suivante : un mot "numéro de bloc", des mots de données et un caractère "fin de bloc".

Chaque mot comprend une lettre adresse (voir figure 15) et des chiffres précédés ou non d'un signe. Les mots, à l'exception du caractère de tabulation, peuvent être omis lorsqu'ils ne sont pas nécessaires dans un bloc d'information particulier. Ceci doit être interprété comme signifiant qu'il n'y a aucun changement dans l'état de la machine par rapport à la fonction représentée par le mot omis. Le caractère "fin de bloc" peut être utilisé après n'importe quel mot complet. Dans un bloc à format variable on trouve en général les mêmes informations que dans un bloc à format fixe.

- le numéro de bloc est constitué par trois chiffres.

- les fonctions préparatoires (6) servent à préparer la machine pour des opérations particulières ou des cycles fixes de travail (par exemple cycles de perçage). Elles sont repérées par l'adresse G suivie d'un nombre codé de deux chiffres (voir figure 16).

- les déplacements sont indiqués par une adresse (par exemple, X, Y ou Z) indiquant l'axe intéressé suivie d'un nombre de n chiffres avec ou sans signe indiquant les coordonnées à atteindre.

ANNEXE E

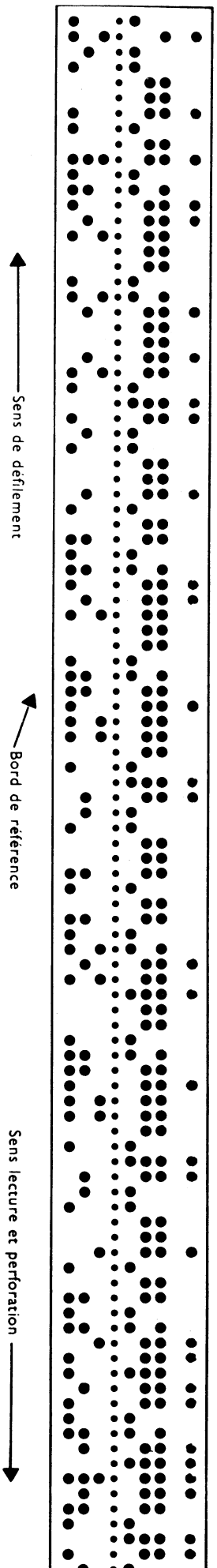
EXEMPLE DE BANDE PERFORÉE A BLOC A FORMAT FIXE

La présente note donne un exemple de bande perforée à bloc à format fixe avec tabulation.

Le jeu de caractères défini est en accord avec la norme NF Z 68-010 « Code pour commande numérique des machines (compatible avec le jeu de caractères codés à 7 éléments) » et sa représentation sur bande est conforme à la norme NF Z 62-110 « Représentation sur bande perforée du jeu de caractères codés à 7 éléments ».

Le texte de la feuille de préparation correspondante est également donné.

En principe, l'astérisque n'est pas utilisé, mais dans l'exemple présent, indique la position du caractère « Fin de bloc » qui ne s'imprime pas.



FEUILLE DE PRÉPARATION

```
%
001 07 +12500  −20025 81*
002 03 +12500  +31550 82*
003 03 −25800  +31550 82*
004 03 +41921  +28723 81*
005
```

Figure 14.

ANNEXE B

CARACTÈRES

B.1 CARACTÈRES POUR ADRESSE

Caractère	Signification
A	Coordonnée angulaire autour de l'axe X.
B	Coordonnée angulaire autour de l'axe Y.
C	Coordonnée angulaire autour de l'axe Z
D	Coordonnée angulaire autour d'un axe spécial ou : troisième vitesse d'avance (I).
E	Coordonnée angulaire autour d'un axe spécial ou : seconde vitesse d'avance (I).
F	Vitesse d'avance.
G	Fonction préparatoire.
H	Disponible de façon permanente (II).
I	Disponible } Ne pas utiliser dans les systèmes de mise en position et d'usinage parallèle aux axes.
J	Disponible }
K	Disponible }
L	A ne pas utiliser.
M	Fonction auxiliaire.
N	Numéro de bloc.
O	A ne pas utiliser.
P	Mouvement tertiaire parallèle à l'axe des X (I).
Q	Mouvement tertiaire parallèle à l'axe des Y (I).
R	Premier déplacement rapide sur l'axe des Z ou : mouvement tertiaire parallèle à l'axe des Z (I).
S	Vitesse de rotation de la broche.
T	Fonction outil.
U	Mouvement secondaire parallèle à l'axe des X (I).
V	Mouvement secondaire parallèle à l'axe des Y (I).
W	Mouvement secondaire parallèle à l'axe des Z (I).
X	Mouvement principal parallèle à l'axe des X.
Y	Mouvement principal parallèle à l'axe des Y.
Z	Mouvement principal parallèle à l'axe des Z.
:	Fonction subdivision de programme (III).

Figure 15.

(I) Lorsque les lettres D, E, P, Q, R, U, V, W ne sont pas utilisées comme indiqué ci-dessus, elles sont disponibles et doivent être utilisées si nécessaire pour des applications spéciales.

(II) Dans le bloc à format fixe, le caractère H est utilisé comme un mot d'adresse de bloc.

(III) Après un mot « fonction subdivision de programme », toute information nécessaire pour commencer ou recommencer l'usinage doit être codée. Le caractère « fonction subdivision de programme » doit être utilisé au lieu de N en tant que caractère adresse pour le mot « numéro de bloc ». Le caractère « fonction subdivision de programme » peut

- les fonctions vitesse d'avance sont repérées par l'adresse F suivie d'un certain nombre de chiffres suivant la machine.

- les fonctions vitesse de rotation sont similaires aux fonctions vitesse d'avance mais sont repérées par l'adresse S.

- les fonctions outil servent pour des machines munies d'un changeur automatique d'outil. Elles ont pour adresse T, qui est suivie d'un nombre dont la longueur dépend de la machine utilisée.

- Les fonctions auxiliaires (6) sont repérées par l'adresse M suivie de deux chiffres. Elles servent à commander différents types de fonctions machine (voir figure 16).

La figure 17 donne un exemple de bande perforée en format variable.

Les spécifications relatives aux bandes perforées que nous venons de citer sont conformes aux recommandations de l'Organisation Internationale de Normalisation (I.S.O), mais cette normalisation est loin d'être adoptée par tous les constructeurs. La plupart des fabricants de commande numérique pour machines-outils utilisent leur propre format surtout en ce qui concerne les différentes fonctions préparatoires G, vitesses d'avance F, vitesses de rotation S et fonctions auxiliaires M. Cependant la plupart des systèmes de programmation en commande numérique comportent une structure de bloc avec mots adressés.

2.3. PROGRAMMATION AUTOMATIQUE

2.3.1. Généralités.

L'établissement manuel des informations nécessaires à la commande des machines doit nécessairement se limiter à l'usinage en mode point à point. Dans tous les autres cas le nombre d'informations numériques à communiquer à la machine est tel que la préparation des données doit

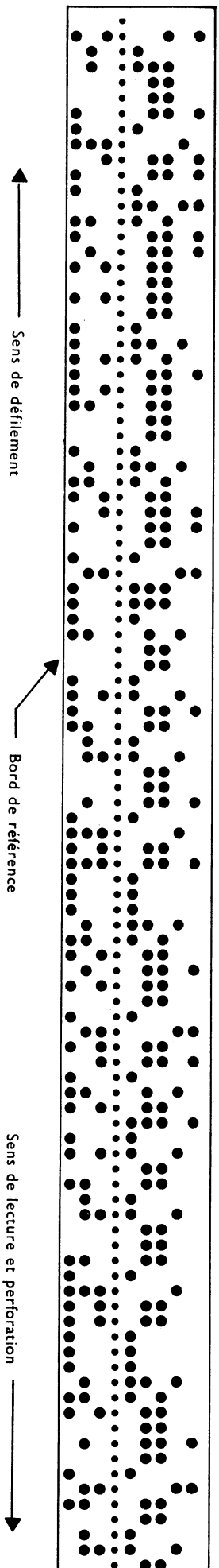
2 Figure 15. **CODAGE DES FONCTIONS PRÉPARATOIRES G**

CODE	SIGNIFICATION
G00	Point-à-point, mise en position.
G01	Interpolation linéaire (moyennes distances).
G02	Interpolation circulaire, sens antitrigonométrique (moyennes distances).
G03	Interpolation circulaire, sens trigonométrique (moyennes distances).
G04	Stationnement temporisé.
G05	Stationnement non temporisé.
G06	Non attribué.
G07	Non attribué.
G08	Accélération.
G09	Décélération.
G10	Interpolation linéaire (grandes distances).
G11	Interpolation linéaire (courtes distances).
G12	Interpolation en trois dimensions.
G13-G16	Choix des axes.
G17	Choix du plan XY.
G18	Choix du plan ZX.
G19	Choix du plan YZ.
G20	Interpolation circulaire, sens antitrigonométrique (grandes distances).
G21	Interpolation circulaire, sens antitrigonométrique (courtes distances).
G22	Mouvements conjugués (positif).
G23	Mouvements conjugués (négatif).
G24	Non attribué.
G25-G29	Non attribué de façon permanente.
G30	Interpolation circulaire, sens trigonométrique (grandes distances).
G31	Interpolation circulaire, sens trigonométrique (courtes distances).
G32	Non attribué.
G33	Filetage à pas constant.
G34	Filetage à pas croissant.
G35	Filetage à pas décroissant.
G36-G39	Réservé pour l'emploi du système de commande.
G40	Annulation de correction d'outil.
G41	Correction d'outil (à gauche).
G42	Correction d'outil (à droite).
G43	Correction d'outil (positive).
G44	Correction d'outil (négative).
G45	Correction d'outil X + /Y +
G46	Correction d'outil X + /Y −
G47	Correction d'outil X − /Y −
G48	Correction d'outil X − /Y +
G49	Correction d'outil X = O/Y +
G50	Correction d'outil X = O/Y −
G51	Correction d'outil X + /Y = O
G52	Correction d'outil X − /Y = O
G53	Annulation décalage d'origine de coordonnées linéaires.
G54	Décalage d'origine (coordonnée X).
G55	Décalage d'origine (coordonnée Y).
G56	Décalage d'origine (coordonnée Z).
G57	Décalage d'origine (coordonnées X et Y).
G58	Décalage d'origine (coordonnées X et Z).
G59	Décalage d'origine (coordonnées Y et Z).
G60	Mise en position précision 1.
G61	Mise en position précision 2.
G62	Mise en position rapide.
G63	Taraudage.
G64	Changement de vitesse d'avance.
G65-G79	Réservé exclusivement à la mise en position.
G80	Annulation du cycle fixe.
G81	Cycle fixe 1.
G82	Cycle fixe 2.
G83	Cycle fixe 3.

CODE	SIGNIFICATION
G84	Cycle fixe 4.
G85	Cycle fixe 5.
G86	Cycle fixe 6.
G87	Cycle fixe 7.
G88	Cycle fixe 8.
G89	Cycle fixe 9.
G90-G99	Non attribué.

3 CODAGE DES FONCTIONS AUXILIAIRES M

CODE	SIGNIFICATION
M00	Arrêt programmé.
M01	Arrêt facultatif.
M02	Fin de programme.
M03	Rotation broche, sens antitrigonométrique.
M04	Rotation broche, sens trigonométrique.
M05	Arrêt de la broche.
M06	Changement d'outil.
M07	Arrosage n° 2 en marche.
M08	Arrosage n° 1 en marche.
M09	Arrêt de l'arrosage.
M10	Bridage.
M11	Débridage.
M12	Démarrage du programme.
M13	Rotation de la broche, sens antitrigonométrique, avec arrosage.
M14	Rotation de la broche, sens trigonométrique, avec arrosage.
M15	Déplacement sens positif.
M16	Déplacement sens négatif.
M17-M18	Non attribué.
M19	Arrêt de la broche, orientation déterminée.
M20-M29	Non attribué de façon permanente.
M30	Fin de bande.
M31	Suspension d'interdiction.
M32-M35	Vitesse de coupe constante.
M36	Gamme de vitesses d'avance n° 1.
M37	Gamme de vitesses d'avance n° 2.
M38	Gamme de vitesses de rotation n° 1.
M39	Gamme de vitesses de rotation n° 2.
M40-M45	Changement de vitesses (s'il existe); sinon, non attribué.
M46-M49	Réservé pour l'emploi du système de commande.
M50	Arrosage n° 3 en marche.
M51	Arrosage n° 4 en marche.
M52-M54	Non attribué.
M55	Décalage d'origine (emploi de la broche 1).
M56	Décalage d'origine (emploi de la broche 2).
M57-M59	Non attribué.
M60	Changement de pièce.
M61	Décalage d'origine (coordonnées linéaires, position de pièce n° 1).
M62	Décalage d'origine (coordonnées linéaires, position de pièce n° 2).
M63-M67	Non attribué.
M68	Bridage de la pièce.
M69	Débridage de la pièce.
M70	Non attribué.
M71	Décalage d'origine (coordonnées angulaires, position de pièce n° 1).
M72	Décalage d'origine (coordonnées angulaires, position de pièce n° 2).
M73-M77	Non attribué.
M78	Bridage de table.
M79	Débridage de table.
M80-M99	Non attribué.



ANNEXE E

EXEMPLE DE BANDE PERFORÉE A BLOC A FORMAT VARIABLE

La présente note donne un exemple de bande perforée à bloc à format variable avec tabulation et avec adresses.

Le jeu de caractères défini est en accord avec la norme NF Z 68-010 « Code pour commande numérique des machines (compatible avec le jeu de caractères codés à 7 éléments) » et sa représentation sur bande est conforme à la norme NF Z 62-110 « Représentation sur bande perforée du jeu de caractères codés à 7 éléments ».

Le texte de la feuille de préparation correspondante est également donné.

En principe. l'astérisque n'est pas utilisé, mais dans l'exemple présent, il indique la position du caractère « fin de bloc » qui ne s'imprime pas.

FEUILLE DE PRÉPARATION

```
%
: 001 G41  X+125050 Y-15300 Z+5410 F99 S00 M13*
N002 G57                    Z+5250 F54 S58 M03*
N003 G55                    Z+5020 F30
N004
```

Figure 17.

s'effectuer à l'aide d'un ordinateur. Le préparateur du bureau des méthodes devra donc établir un programme de pièce au moyen d'un langage symbolique qui permettra de définir la forme de la pièce à l'aide d'éléments géométriques simples et de spécifier la trajectoire de l'outil ou les différentes opérations d'usinage à effectuer. Le système de traitement des programmes de pièces symboliques produit directement une bande perforée comportant les ordres numériques utilisables par la machine. La figure 18 schématise le processus de programmation automatique en commande numérique.

2.3.2. Traitement des langages.

D'une façon générale les instructions symboliques d'un programme de pièce pour commande numérique sont traitées en deux phases dans l'ordinateur. La première phase est indépendante de la machine-outil et du système de commande numérique, c'est le "processeur". La seconde phase est au contraire liée à la machine-outil et au système de commande numérique utilisés, c'est le "post-processeur" aussi appelé programme d'adaptation.

Le "Processeur" est un programme de traitement général qui traduit les instructions du programme de pièce en données numériques caractérisant les positions successives de l'outil. Ce travail de traduction et de calcul est fait sans tenir compte des caractéristiques particulières de la machine-outil à commande numérique à laquelle seront confiées les opérations d'usinage (49). Le processeur décode d'abord les instructions symboliques et en vérifie la syntaxe ; il effectue les diverses opérations arithmétiques et l'analyse des éléments géométriques programmés ; il calcule enfin les points remarquables de la trajectoire de l'outil en tenant compte de sa forme et des tolérances données (approximation des courbes de la trajectoire par des segments de droite, un cercle étant ainsi approché par un polygone dont le nombre de côtés dépend de la tolérance donnée). Le résultat de ce traitement est généralement placé sur bande magnétique (CLTAPE pour "Cutter Location Tape").

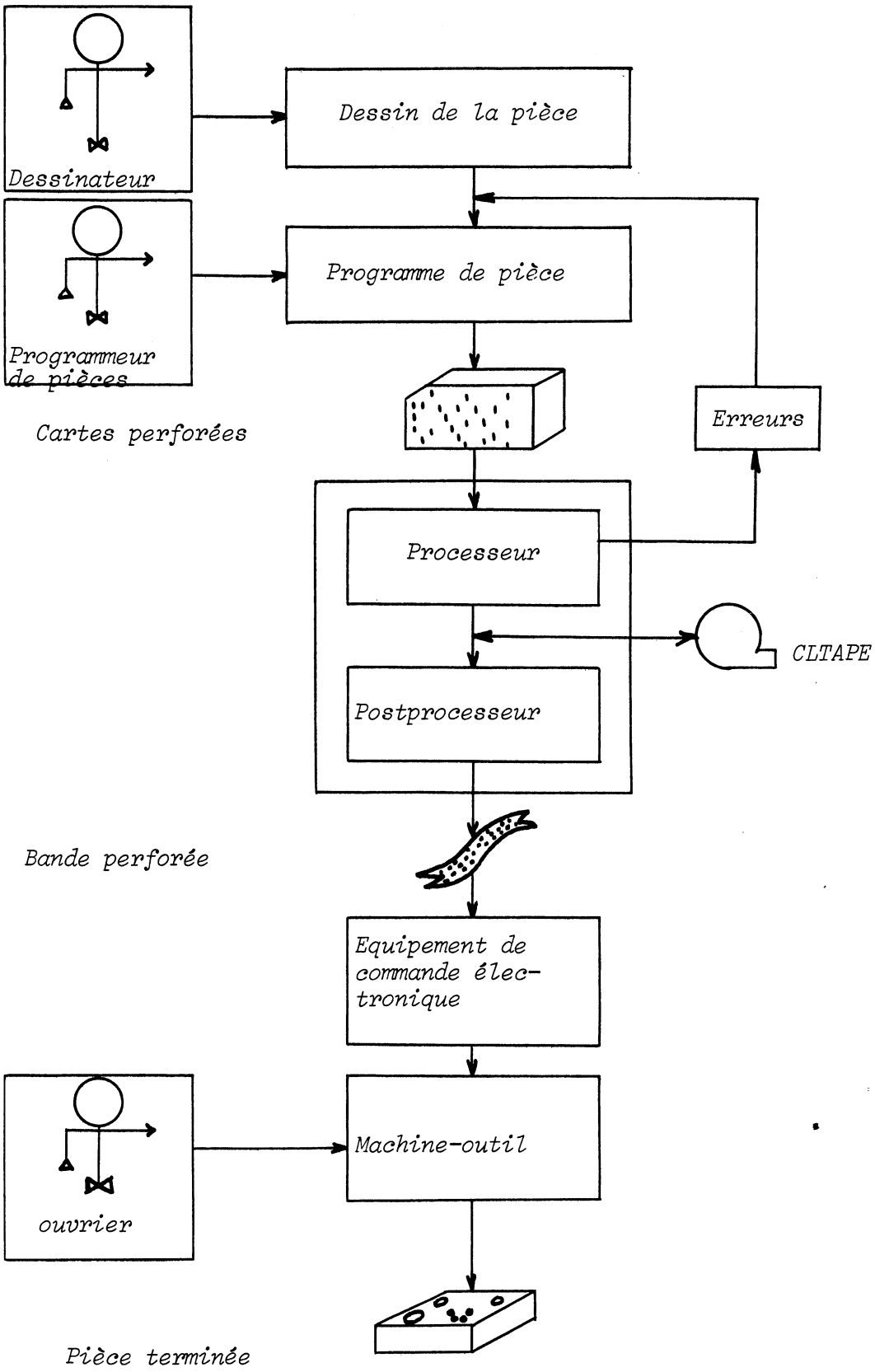


Figure 18 . *Programmation automatique en commande numérique (21).*

L'adaptation des données numériques générales ainsi obtenues aux caractéristiques particulières de la machine-outil à commande numérique qui sera chargée d'exécuter l'usinage, nécessite l'emploi d'un programme de traitement complémentaire appelé postprocesseur ou programme d'adaptation. Cette adaptation tient compte du code utilisé par le système de commande numérique associé à la machine-outil,du format des instructions machine et des limitations de la machine-outil proprement dite (vitesse d'avance, vitesse de rotation de broche, possibilité de freinage avant l'arrêt, rattrapage des jeux, limites de la table, accélérations et décélérations admissibles, etc...) ainsi que de la méthode d'interpolation prévue pour le système de commande numérique utilisé (linéaire, circulaire ou parabolique). Le postprocesseur fournit alors la bande perforée directement utilisable par le système de commande numérique de la machine-outil.

2.3.3. Les différents langages.

Il existe à l'heure actuelle de nombreux langages de programmation pour les systèmes de commande numérique (22). Un petit nombre de ces langages donne une bonne idée des diverses tendances dans ce domaine (44).

2.3.3.1. APT et l'usinage multi-axes.

Une fois le premier système de commande numérique réalisé au M.I.T., il devint apparent que l'usinage continu des surfaces complexes nécessitait une programmation assistée par ordinateur. C'est en 1953 que le M.I.T. commença à étudier et à réaliser pour l'armée de l'air américaine le système de programmation APT (Automatically Programmed Tool). Le programme devint bientôt APT II et déjà à ce stade représentait un effort supérieur à 100 hommes-année. Ce programme eut d'ailleurs une grande influence sur les autres langages de programmation pour commande numérique. La structure du système APT était en avance sur son époque, allant jusqu'à permettre l'exécution du programme après l'élimination des parties erronnées ou permettant le traitement du même programme de pièce par plusieurs postprocesseurs, choses courantes à l'heure actuelle mais qui représentaient alors une véritable innovation.

APT est le seul système de programmation pour commande numérique qui soit universellement répandu (30 % des utilisateurs de commande numérique des Etats Unis l'emploient) et qui puisse traiter les usinages multi-axes. C'est en effet le seul système qui puisse traiter complètement l'usinage continu simultané sur trois axes. C'est enfin le système qui est le mieux documenté et qui utilise actuellement le plus grand nombre d'heures d'ordinateur ; il possède aussi plus de postprocesseurs que tout autre système. Il peut donc passer pour "universel".

Malgré ces avantages le système et le langage APT présentent quelques inconvénients. Le langage est verbeux et peut être redondant en contournage à deux dimensions ; il ne se prête que très mal au travail de point à point comportant des cycles d'usinage comme le taraudage. La forme de ses instructions en fait un langage de mathématiciens plutôt qu'un langage d'ingénieurs ou de techniciens et qui demande tout de même un niveau assez élevé. Il exige du programmeur de pièce des connaissances techniques que le type de personne employé comme tel ne possède pas obligatoirement. La structure interne du système APT (20) lui interdisait de pouvoir être facilement associé à des programmes plus généraux d'automatisation de la conception des pièces mécaniques. Enfin il ne permet pas d'utiliser tous les avantages des systèmes de commande numérique actuels qui peuvent donner de meilleurs états de surface que ceux engendrés à l'aide de vecteurs de coupe constitués par des segments de droite.

Le processeur APT n'a d'ailleurs pas eu des débuts très faciles, ayant d'abord souffert de la lenteur des calculateurs de deuxième génération et la première version largement utilisée, APT III, nécessitant déjà un ordinateur IBM 7090 qui n'a jamais été très répandu, même aux Etats-Unis. Les problèmes de mise au point et d'amélioration du système ont été très aigus jusqu'à la formation du groupe ALRP ("APT Long Range Program") pris en charge par l'IITRI ("Illinois Institute of Technology Research Institute") et qui comptait en 1968 cent vingt et un membres dont vingt et un membres européens. Toutes ces difficultés ont contribué au développement du premier sous-ensemble d'APT pour ordinateurs moyens ADAPT (31) qui n'eut cependant qu'un succès limité (bien que 10% des utilisateurs de commande numérique des Etats-Unis l'emploient), et des langages appa-

rentés à APT actuels : 2CL, EXAPT et IFAPT.

2.3.3.2. PROFILE-DATA et l'usinage en "deux axes et demi".

Les machines-outils à commande numérique ont été développées en Europe parallèlement à ce qui se faisait aux Etats-Unis, sans pourtant utiliser les mêmes méthodes. En Grande Bretagne on s'intéressa surtout au contournage, probablement à cause de l'intérêt porté à l'oxycoupage commandé numériquement des tôles pour les coques de navires. Pour traiter ces problèmes de contournage plusieurs programmes furent créés, mais seul PROFILE-DATA fut adopté en Europe. C'est un système simple et nombre de ses utilisateurs sont persuadés qu'ils peuvent écrire et mettre au point des programmes de pièce plus rapidement et avec du personnel moins qualifié que pour APT. La vitesse de traitement de PROFILE-DATA est bonne surtout parce qu'il n'a pas adopté les concepts des définitions symboliques et les concepts de syntaxe du langage APT. L'usinage ne peut se faire que dans les plans principaux, mais en Europe il y a peu d'usinage qui ne soit de ce type en dehors de l'industrie aéronautique. PROFILE-DATA permet donc d'usiner des surfaces complexes par courbes de niveau, c'est ce qu'on appelle usinage en deux axes et demi. En bref c'est un langage simple à format semi-fixe sans autres possibilités spéciales que les répétitions simples. Il ne possède pas de possibilité de calcul arithmétique comme FORTRAN et ne peut traiter que des points, des cercles, des droites et des paraboles. Il ne peut traiter les programmes destinés aux machines-outils avec tête orientable. Il est cependant très utilisé en Grande Bretagne et en Europe continentale.

2.3.3.3. AUTOSPOT et les centres d'usinage.

Les centres d'usinage capables de travaux de point à point et d'usinage selon des arcs de cercle, avec changement automatique d'outil, ne firent leur apparition qu'après la définition et la réalisation des deux systèmes précédents. La programmation manuelle de ces machines relativement peu complexes n'était tout de même pas très satisfaisante, ce qui conduisit à la définition d'un langage et à la réalisation d'un système plus simple que le système APT ; ce furent ainsi des ingénieurs de fabrication qui conçurent AUTOSPOT pour la fabrication. On peut dire qu'à

l'heure actuelle 85 % de tous les travaux de perçage et de fraisage en commande numérique peuvent être programmés en **AUTOSPOT (30)**. On lui reproche d'abord de n'exister que sur du matériel IBM et ensuite de n'être pas entièrement compatible avec **APT**, tant du point de vue du langage que du point de vue de la forme des données numériques qu'il produit bien que les différences soient peu marquées. Il existe pourtant de nombreux post-processeurs pour ce langage qui est très utilisé (près de 15 % des utilisateurs de commande numérique des Etats-Unis l'emploient).

2.3.3.4. EXAPT et l'introduction des données technologiques.

Beaucoup plus récemment à partir de problèmes de tournage fut créé en Allemagne **AUTOPIT**, un des premiers langages comprenant une partie technologique. En effet les problèmes posés par le tournage ne sont pas des problèmes géométriques, les formes pouvant être décrites par des segments de droite ou des arcs de cercle simples, mais des problèmes technologiques ; il faut en effet calculer pour chaque tour, pour chaque pièce, pour chaque outil et pour chaque métal les vitesses d'avance et de rotation. Le succès obtenu par ce premier essai conduisit à la réalisation par IBM d'**AUTOPOL ("Automatic Programming of Lathes")** permettant d'écrire des programmes de pièce pour tournage plus simples et plus compacts. C'est alors que fut décidée la définition et la réalisation de trois langages et systèmes (46) : **EXAPT 1** pour les travaux de point à point, **EXAPT 2** pour le tournage et **EXAPT 3** pour le contournage selon deux axes. Ces systèmes acceptent un langage d'entrée du type **APT** (mais non compatible avec **APT**) et comprennent un processeur géométrique et un processeur technologique. **EXAPT 1** permet d'accéder à une cartothèque outils, à une cartothèque matières et à des séquences d'usinage prédéterminées ; il sera donc possible de calculer automatiquement les vitesses d'avance et les vitesses de coupe et de déterminer automatiquement l'exécution de cycles d'usinage. **EXAPT 2** engendre une succession d'usinage avec les vitesses d'avance et les vitesses de coupe correctes, à partir des profils de la pièce brute et de la pièce usinée et des données définissant les outils et la technologie de la coupe.

EXAPT marque un grand pas dans la voie de l'automatisation complète de l'usinage des pièces mécaniques. On peut lui reprocher malgré tout d'utiliser un langage qui, comme APT, est trop complexe et d'avoir des processeurs nécessitant de gros ordinateurs. Par ailleurs il faut noter que la technologie de l'atelier varie non seulement de pays à pays mais aussi de compagnie à compagnie ce qui nécessite des définitions technologiques variables dont EXAPT ne tient pas compte à l'heure actuelle. Mais c'est un système récent certainement promis à un bel avenir ; nous le verrons plus en détail au chapitre 6.

2.3.3.5. Tendances actuelles.

Malgré la supériorité de l'APT, le domaine des langages de programmation pour commande numérique présente un exemple opposé à la normalisation ; presque tous les grands producteurs de machines-outils et de commande numérique ont créé leurs propres systèmes de programmation. On dénombre au moins les 36 langages suivants (38) :

ACTION	AUTOPROPS	FMILL	PROFILEDATA
ADAPT	AUTOSPOT	IFAPT	PRONTO
APT	AUTOSURF	INCA	ROMANCE
APTLOFT	BSURF	MECA	SNAP
AUTOMAT	CAMP	MILMAP	SPLIT
AUTOPIT	CINAPT	MINIAPT	SYMPAC
AUTOPOL	CLAM	NUMERISCRIPT	UNIAPT
AUTOPRESS	COCOMAT	PAGET	ZAP
AUTOPROMPT	EXAPT	PMTZ	2CL

La diversité des langages existants a fait apparaître un besoin de normalisation, une entreprise ne pouvant traiter chaque problème dans un langage différent et pouvant difficilement se permettre de former des programmeurs pour plusieurs de ces langages. Ceci n'empêche d'ailleurs pas de nouveaux langages et systèmes de voir le jour.

Cependant l'universalité et la puissance de l'APT restent incontestées et le désignent comme système de référence pour les tenants de la

normalisation. L'organisation internationale de normalisation (I.S.O.) a déjà commencé l'élaboration de recommandations destinées à assurer une certaine uniformité dans le domaine de la programmation automatique en commande numérique ; ceci explique l'importance prise récemment par les dérivés de l'APT comme 2CL (42), EXAPT et IFAPT (2). Nous pouvons donc ébaucher une classification de ces langages en deux grandes catégories, d'un côté ceux qui s'apparentent à APT (en anglais "APT-like"), de l'autre les systèmes indépendants.

Parmi les langages non apparentés à APT le plus utilisé est AUTOSPOT dont nous avons déjà parlé. Les autres langages sont moins connus ou moins répandus, certains étant destinés à une seule marque de machines comme SPLIT de Sundstrand (employé par 7 % des utilisateurs de commande numérique américains) ou PAGET d'Olivetti. Citons encore la conception originale de MECA conçu à la Faculté des Sciences de LILLE. La table 1 donne une vue plus complète des langages les plus importants.

Langage	Origine	Calculateur Mémoire nécessaire	Caractéristiques
AUTOPROPS	IBM PRATT & WHITNEY	IBM 1401	Point à point 2 axes
AUTOSPOT	IBM KEARNEY & TRECKER	IBM 1620+disque 1311 IBM 360/30 + "	Point à point 3 axes
CAMP I & II	WESTINGHOUSE IBM	IBM 7090 IBM 7094	Point à point 3 & 5 axes
MILMAP	ICT MILWAUKEEMATIC		Point à point 3 axes
PRONTO	GE MILWAUKEEMATIC	GE 225	Point à point 3 axes
PROFILE-DATA	FERRANTI	ICT 1900	continu
PAGET	OLIVETTI	GE 425 IBM 7040 IBM 360/40-64K	continu
SPLIT	SUNDSTRAND	IBM 1620 IBM 360/30-64K	Continu 3 ou 5 axes
AUTOPOL	IBM	IBM 360/30-32K IBM 1130-8K	Continu 2 axes Tournage
ROMANCE	IBM	IBM 1130-8K	Continu
AUTOPROMPT	IBM	IBM 7090-8K+8 bandes	Continu 3 axes
AUTOMAP	IBM	IBM 1620	Continu 3 axes
MECA	BULL-GE	BULL-GE	Point à point 3 axes
APT III APT IV	M.I.T. I.I.T.R.I.	UNIVAC 1107 & 1108 IBM 7090 & 7094 GE 625 CDC 3600 IBM 360/40-256K + disque & 6 bandes	Point à point & Contournage 5 axes
ADAPT	IBM	IBM 1620+disque 1311 IBM 360/30-64K+ " GE 425 HONEYWELL 200	Point à point & contournage 5 axes

(suite page suivante)

(suite)

Langage	Origine	Calculateur Mémoire nécessaire	Caractéristiques
2CL	NEL	ICL	Contournage 2 axes 1/2 Données technologiques
EXAPT 1	ALLEMAGNE	ICL IBM	Point à point 3 axes Données technologiques
EXAPT II		UNIVAC 1107 & 1108	Tournage 2 axes
IFAPT P	FRANCE	GAMMA 40 IBM	Point à point
IFAPT C	CII-LCA	CII 90-40 (64K)	Contournage

TABLE 1 (22)

CHAPITRE 3

ETUDE DES SYSTEMES DE PROGRAMMATION ACTUELS ET DU LANGAGE APT

3.1. SYSTEMES ACTUELS.

La conception d'un nouveau système de programmation pour la commande numérique de machines-outils nécessite une étude préalable des systèmes de programmation en commande numérique existants. On peut dire qu'actuellement il a été admis qu'APT était un langage assez puissant et assez général pour constituer en quelque sorte une référence de normalisation. L'organisation internationale de normalisation (I.S.O.) examine maintenant quatre langages candidats à la normalisation ; il s'agit d'APT (Etats-Unis), d'EXAPT (Allemagne), de 2CL (Grande Bretagne) et d'IFAPT (France). Ces trois derniers sont dits langages "type-APT" (ou en anglais "APT-like") et on pourrait en conclure qu'il ne devrait pas être difficile de définir un langage normalisé complet APT comprenant tous les langages "type-APT". Les choses ne sont malheureusement pas aussi simples que cela, car il n'existe aucune méthode rigoureuse de définition de ces langages. Les langages "type-APT" ont cependant été créés à partir d'APT et en forment des sous-ensembles spécialisés présentant parfois des particularités n'existant pas dans APT (comme par exemple l'introduction des données technologiques d'EXAPT).

Afin de simplifier les problèmes posés par la normalisation de ces langages on a classé les sous-ensembles possibles du langage de référence complet en cinq groupes permettant différentes sortes de travaux :

1 . Point à point
2 . Contournage deux dimensions
3 . Contournage deux dimensions et demie (usinage des surfaces par courbes de niveau)
4 . Contournage trois dimensions
5 . Contournage multi-axes

Le langage minimum sera donc un langage permettant l'usinage de point à point.

Les différents systèmes de programmation en commande numérique apparentés à APT ont tous des programmes de traitement de taille importante, ce qui ne leur permet pas d'occuper la mémoire centrale de l'ordinateur sans avoir au préalable été découpés en différentes sections ou phases. Chaque section joue alors un rôle bien défini et les différentes sections occupent la mémoire à tour de rôle.

Le système APT III comprend ainsi quatre sections principales appelées en mémoire à tour de rôle ; chaque section a pour but le traitement de certaines classes d'instructions (instructions de définition géométrique, instructions spéciales), la dernière section comprenant un ensemble de post-processeurs parmi lesquels on choisit le postprocesseur voulu. Ce système a été récemment repensé afin de rendre le traitement des programmes de pièces plus rationnel et de permettre une maintenance du système plus aisée tout en ouvrant le nouveau système APT IV aux développements graphiques mais le découpage en phases ou en sections a été conservé. La figure 19 permet d'établir une comparaison entre différents systèmes actuels et permet de remarquer que l'organisation générale est la même. Ils sont toujours divisés en phases et le postprocesseur constitue la dernière phase du traitement.

Pour des raisons pratiques les processeurs et les postprocesseurs sont écrits en FORTRAN. La complexité des opérations à effectuer rend l'utilisation d'un langage évolué souhaitable car ceci permet une mise au point des programmes plus rapide ; on perd un peu en efficacité par rapport à des programmes directement codés en langage d'assemblage mais il faut se rendre compte qu'une fois qu'un programme de pièce APT est au point on ne l'utilise pratiquement plus, la bande perforée étant reproductible par d'autres moyens.

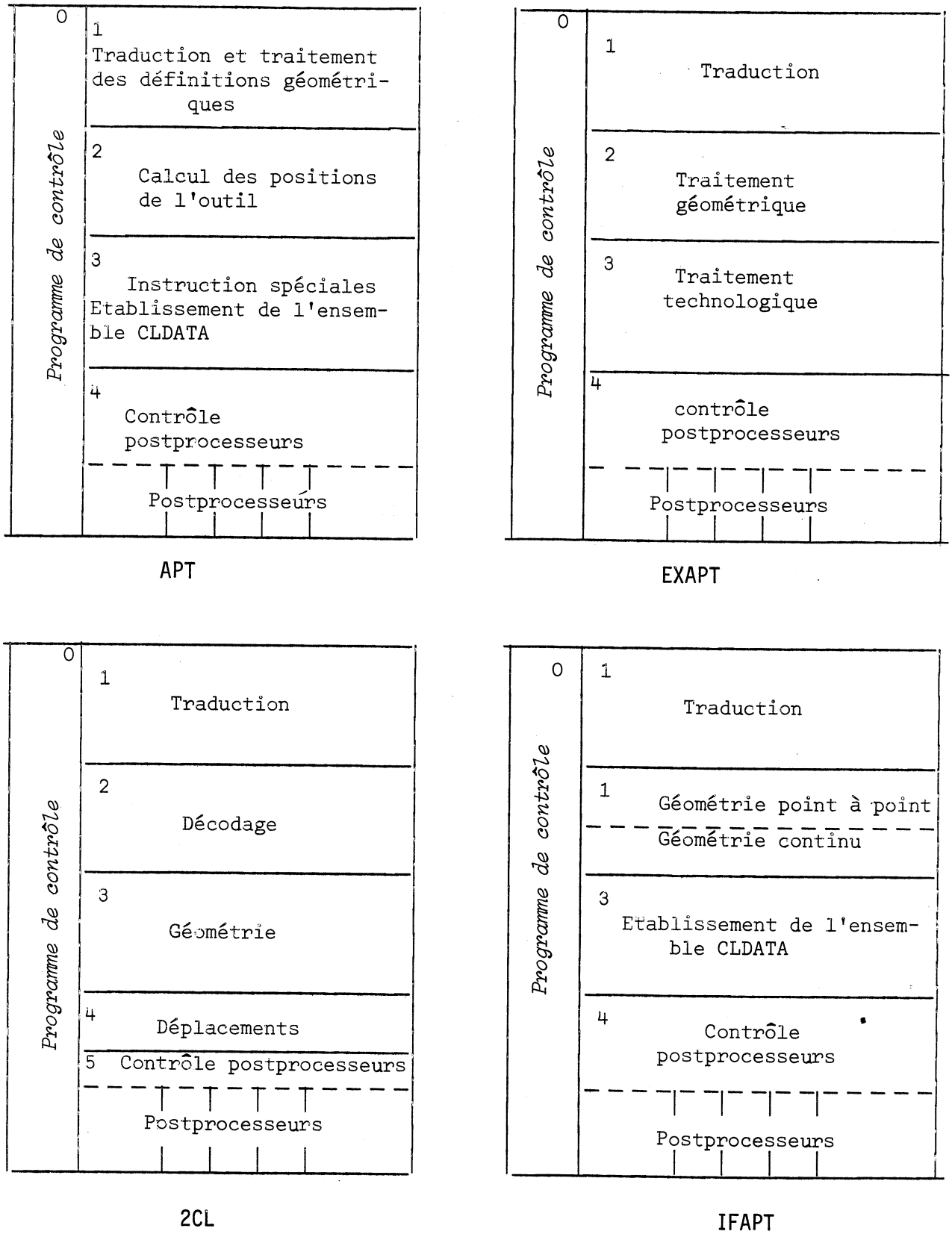


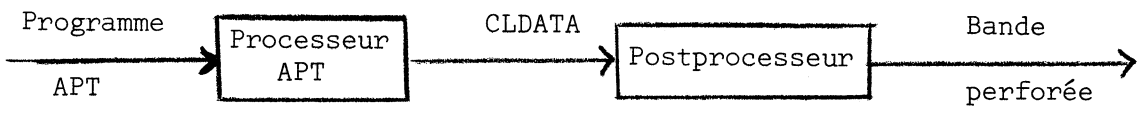
Figure 19 : Systèmes de programmation actuels.

3.2. APT.

3.2.1. Généralités.

Comme nous l'avons déjà dit APT a été le premier langage de programmation spécialement créé pour la commande numérique des machines-outils. Il a eu une influence considérable sur le développement de la programmation automatique en commande numérique, et à cause de cela les principaux langages utilisés en commande numérique ont en général des structures assez proches les unes des autres. APT est le langage de programmation le plus complet dans le domaine de la commande numérique. On lui reproche pourtant d'être verbeux, de n'utiliser qu'un petit nombre de constructions de base ce qui l'oblige à avoir des formes lexicales très strictes et un très grand nombre de mots réservés. Le langage APT peut laisser à désirer mais à le mérite d'exister, ce qu'on a déjà pu dire au sujet du langage FORTRAN. C'est d'ailleurs un langage de la "génération" de FORTRAN, c'est à dire un des premiers langages de programmation évolués.

Malgré quelques essais de formalisation du langage APT(17), sa définition demeure assez descriptive (16). Le langage APT a pour objet la description d'opérations d'usinage et est pratiquement indépendant de toute machine-outil particulière. Comme nous venons de le voir un programme APT doit d'abord être traité par un processeur APT qui traduit le programme de pièce en un langage intermédiaire indépendant de la machine, appelé CLDATA ("Cutter Location Data" pour données de positionnement de l'outil) ou aussi CLTAPE. Le langage intermédiaire CLDATA doit alors être adapté à une machine-outil donnée et converti en une bande perforée par un postprocesseur. La bande ainsi obtenue est transmise à la machine-outil donnée qui l'interprête en exécutant les usinages prévus. Ce processus peut être schématisé de la manière suivante :



L'organisation internationale de normalisation (I.S.O.) a reconnu que la normalisation d'un langage n'était pas possible sans une méthode rigoureuse de définition du langage considéré. Le dictionnaire APT et les manuels de programmation n'ont pas une rigueur suffisante pour servir de norme. Le langage APT comprend de très nombreuses règles syntaxiques mettant en jeu des variables métalinguistiques. Pour ces raisons fut créé un méta langage numérique pour la description d'APT (51) ; il utilise une forme analogue à la notation de Backus. Par exemple la variable métalinguistique 52 qui représente une "spécification de point" est décrite de la façon suivante :

52	POINT	SPECIFICATION
	IS (1)	42
	OR (2)	(POINT/452)
	OR (3)	(42 = POINT/452)

où 42 est un identificateur qui est apparu préalablement à gauche du signe d'affectation "=" dans une instruction d'affectation du type de celle apparaissant dans l'alternative (3) ; 452 est l'une des listes de paramètres de point définies par la règle 452 donnant toutes les manières possibles de définir un point. Cette manière de définir le langage APT aboutit quand même à un manuel assez important et peu lisible (32). On trouvera en annexe une description détaillée du langage APT.

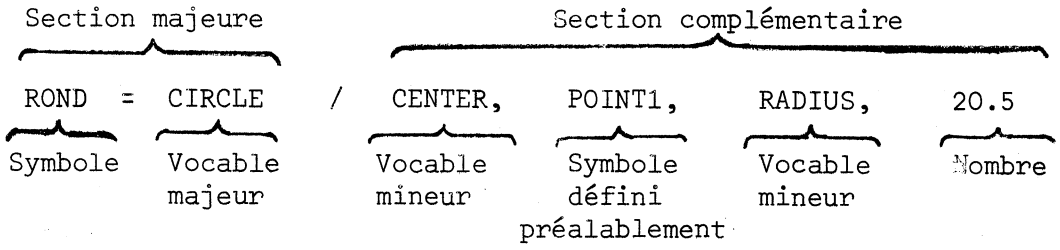
3.2.2. Description du langage APT.

On utilise le langage APT pour décrire des surfaces et pour mouvoir un outil dans l'espace le long de l'intersection de deux de ces surfaces ; ces surfaces représentent celles de la pièce usinée, il est donc facile de les définir directement à partir du dessin de la pièce. Le langage APT comporte des instructions dont la forme se rapproche souvent de la langue anglaise. Le programme de pièce est une suite logique d'instructions définissant les surfaces géométriques de la pièce à usiner, les déplacements de l'outil le long de ces surfaces et diverses fonctions de la machine-outil (11).

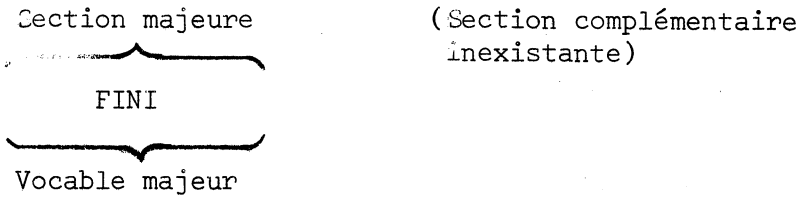
3.2.2.1. Structure du langage.

Les signes autorisés dans le langage APT sont les 26 lettres de l'alphabet majuscule, les dix chiffres décimaux et certains signes spéciaux (point, virgule, barre oblique, astérisque, parenthèses droites et gauches, espacement, signes plus, moins et dollar). Un programme de pièce est formé d'une suite d'instructions, elles-mêmes formées d'éléments du langage. Les éléments du langage sont des mots, des nombres ou des caractères spéciaux ; les mots sont formés avec des lettres ou bien avec des lettres et des chiffres, et comportent au plus six caractères, le premier étant une lettre.

Les mots du langage peuvent être de deux types : d'un côté les vocables ou mots du dictionnaire APT (environ 300), de l'autre les symboles créés par le programmeur et obligatoirement différents des vocables (28). Une instruction APT se compose d'une section majeure et éventuellement d'une section complémentaire, toutes deux séparées par une barre oblique (/). Les éléments formant la section complémentaire sont séparés par des virgules. La section majeure est composée d'un vocable majeur soit seul, soit précédé d'un nom symbolique et du signe égal.



L'exemple ci-dessus est une instruction APT définissant un cercle appelé ROND, par son centre POINT1 (point défini au préalable) et son rayon 20,5. L'instruction ci-dessous est une des instructions APT les plus simples :



Certaines instructions APT peuvent comporter une étiquette numérique ou alphanumérique en tête de l'instruction et séparée du reste de l'instruction par une parenthèse droite comme par exemple :

```
FIN3)      LOOPND
```

où FIN3 est l'étiquette de l'instruction de fin de boucle LOOPND.

Les symboles servent à définir des données géométriques ou arithmétiques aussi bien que des étiquettes. Parmi les signes spéciaux citons le signe dollar qui indique que l'instruction se poursuit sur la ligne ou la carte suivante et le signe double dollar qui indique la fin d'une instruction et permet l'insertion de commentaires. Les parenthèses sont utilisées de la même manière que dans le langage FORTRAN et servent aussi à délimiter les définitions imbriquées.

```
ROND = CIRCLE/CENTER, (POINT1=POINT/3,2,0), RADIUS, 20.5
```

L'exemple ci-dessus reprend l'exemple vu précédemment en utilisant une définition imbriquée ; POINT1 n'a plus à être défini ailleurs dans le programme.

3.2.2.2. Instructions APT.

D'une façon générale il existe des instructions arithmétiques, des instructions spéciales, des instructions de définitions géométriques, des instructions de déplacements géométriques et des instructions d'usinage.

a . Les instructions arithmétiques sont des instructions d'affectation semblables aux instructions d'affectation du langage FORTRAN, et qui peuvent également faire appel à des fonctions (sinus, cosinus, arctangente, racine carrée).

Exemple : `ALFA = 3.14 * (COS(BETA) + 1.414/B)`

b . Les instructions spéciales sont relativement peu nombreuses. L'instruction **PARTNO** sert à identifier le programme de pièce ; l'instruction **MACHIN** définit la machine et le postprocesseur utilisés. Si le programmeur désire effectuer des boucles ou des sauts à l'intérieur de son programme, il doit délimiter la zone du programme dans laquelle les boucles et les sauts seront effectués à l'aide des instructions **LOOPST** et **LOOPND** ; ces règles ne paraissent plus très utiles mais certains programmes de traitement les exigent et elles ont été conservées. Le programmeur peut définir des sous-programmes grâce à l'instruction **MACRO** comme par exemple :

PERCE = MACRO/X,Y,Z=12

qui définit un sous-programme appelé PERCE qui possède trois paramètres d'appel X, Y et Z ; on donne à Z la valeur initiale 12. Ce sous-programme pourra être appelé à l'aide de l'instruction **CALL** comme par exemple :

CALL/PERCE, Y=3, X=10

qui fera exécuter le sous-programme, 10, 3 et 12 étant respectivement les valeurs des paramètres X, Y et Z, On pourra également avoir :

CALL/PERCE, Z=10.5, X=0, Y=4

qui fera exécuter le sous-programme, 0, -4 et 10.5 étant respectivement les valeurs des paramètres X, Y et Z.

c . Les instructions de définitions géométriques permettent de définir des formes tridimensionnelles et d'effectuer des transformations géométriques. Il existe une grande variété de définitions : dix définitions de points, 9 définitions de vecteurs, 8 définitions de plans, 13 définitions de droites,dix définitions de cercles, etc... On peut également définir des coniques, des surfaces de révolution, des surfaces réglées, des réseaux de points, des sphères et des quadriques.

A titre d'exemple un cercle peut être défini par :

1 . Les coordonnées du centre et le rayon

2 . Le centre et une droite à laquelle il est tangent

3 . Le centre et un point de la circonférence

4 . Trois points de la circonférence

5 . Le centre et un cercle tangent

6 . Deux droites se coupant et tangentes au cercle et le rayon

7 . Une droite à laquelle il est tangent, un point par lequel il passe et le rayon.

8 . Le rayon, une droite à laquelle il est tangent et un cercle auquel il est tangent.

9 . Le rayon et deux cercles auxquels il est tangent.

10. Le rayon, une droite à laquelle il est tangent et un cyclindre tabulé auquel est tangent.

Certaines de ces définitions conduisent à plusieurs résultats, il faut alors utiliser des modificateurs comme XLARGE, ZSMALL, OUT pour résoudre les ambiguïtés.

C2 = CIRCLE / YLARGE, D1, XSMALL, IN, C1, RADIUS, 25

L'exemple ci-dessus correspond à la définition de cercle n° 8 ; le cercle C2 est définit comme étant tangent à la droite D1 du côté des Y positifs (YLARGE), comme étant tangent intérieurement (IN) au cercle C1 du côté des X inférieurs (XSMALL) et comme ayant un rayon de 25 (Figure 20).

Parmi les instructions de définitions retenons l'instruction PATERN permettant de définir des réseaux de points se trouvant sur des droites, des cercles ou définis aléatoirement, et également d'associer plusieurs réseaux entre eux pour créer un nouveau réseau. Cette instruction trouvera son application en usinage de point à point.

d . Instructions de déplacements. En positionnement point à point la plupart des mouvements peuvent être réalisés par l'instruction GOTO en spécifiant un point à atteindre ou un réseau de points àdécrire, ou par l'instruction GODLTA spécifiant un mouvement incrémental (Δx, Δy et Δz) par rapport à la position précédente. L'exécution de de l'ordre GOTO est une mise en position qui, en mode point à point, ne correspond pas à un usinage , l'usinage étant fait une fois la po-

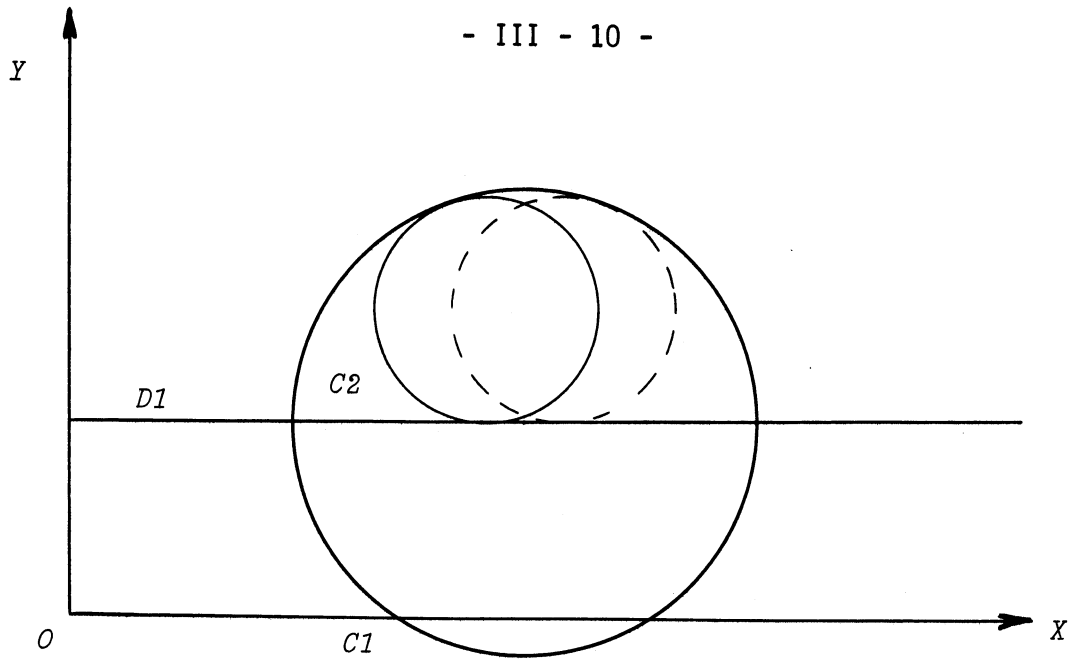


Figure 20. Définition d'un cercle

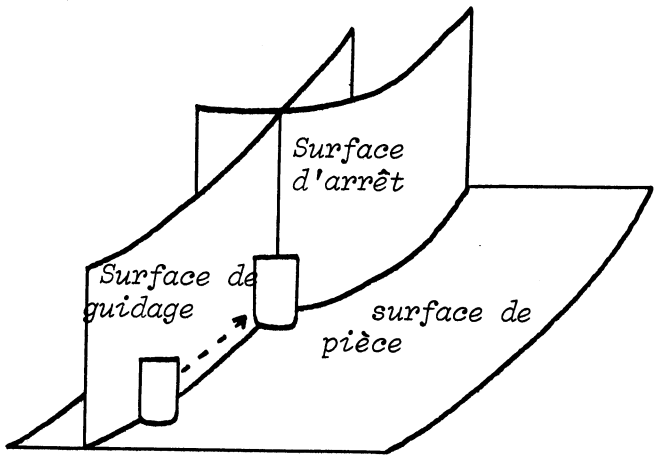


Figure 21 . Surfaces repères des déplacements géométriques.

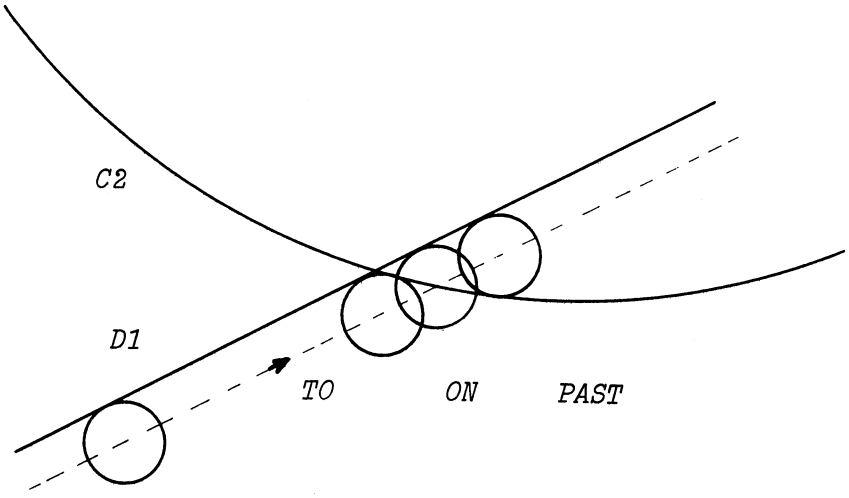


Figure 22 . Ordres de mouvement

sition atteinte.

Toutefois en mode continu l'usinage a lieu tout au long du déplacement de l'outil ; les instructions de déplacements géométriques se font par rapport à trois surfaces repères, la surface de pièce, la surface de guidage et la surface d'arrêt (Figure 21). On utilise alors les ordres **GO, GOLFT, GORGT, GOFWD, GOBACK, GOUP et GODOWN.**

GORGT / D1, TO, C2

L'exemple ci-dessus fait suivre à l'outil la surface D1 en se déplaçant vers la droite (GORGT pour "go right") par rapport au mouvement précédent, et ceci jusqu'à ce qu'il soit tangent extérieurement à la surface C2 (Figure 22). Les modificateurs TO, ON, PAST indiquent la position finale de l'outil par rapport à la surface d'arrêt ; outil tangent à la surface d'arrêt sans l'avoir traversée, outil sur la surface d'arrêt, outil tangent à la surface d'arrêt après l'avoir traversée.

e . Les instructions d'usinage définissent les conditions d'usinage valables pour les instructions de déplacement qui suivent ; elles permettent ainsi de définir les tolérances intérieures et extérieures qui seront utilisées dans l'approximation des courbes par des segments de droites (voir figure 23). Les instructions d'usinage correspondant aux "mots postprocesseur" ne sont pas traitées par le processeur mais seulement par le postprocesseur ; elles ont trait aux différentes fonctions de la machine-outil utilisée. On peut ainsi définir la mise en route de l'arrosage de refroidissement (COOLNT/ON), la vitesse d'avance en mm/minute (FEDRAT/250), la vitesse de rotation de la broche en tours par minute dans un sens donné (SPINDL/1500, CLW) ainsi que des cycles d'usinage.

3.2.2.3. Exemple de programme de piece.

Nous allons donner maintenant un exemple (11) qui nous permettra de voir un programme complet. Le programme de pièce qui suit correspond à l'usinage de la pièce de la figure 24.

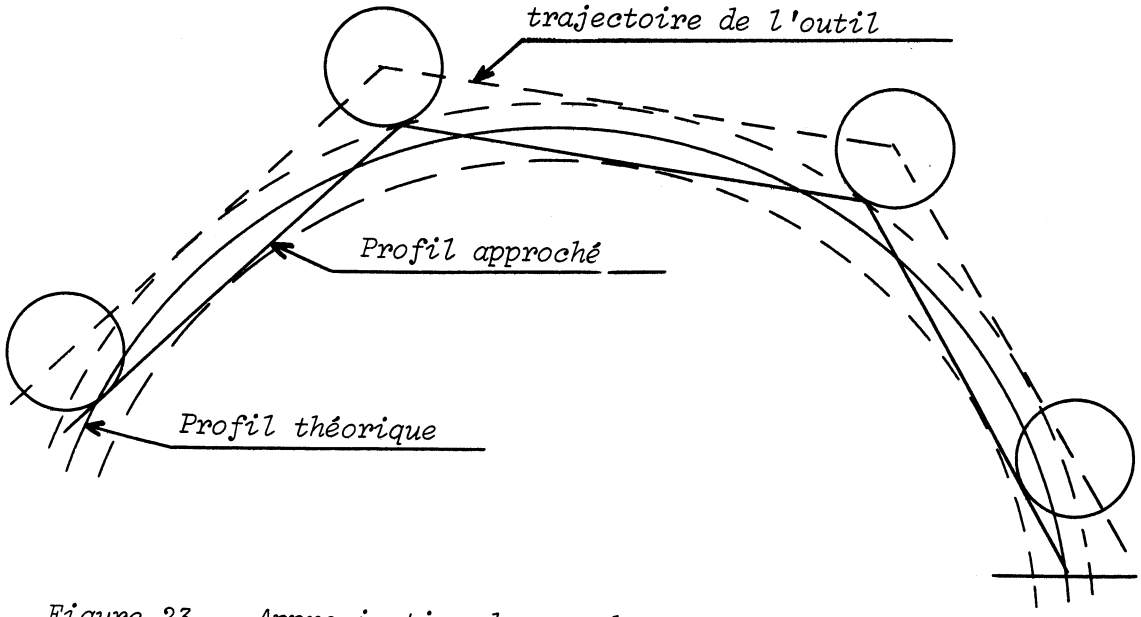


Figure 23 . *Approximation des courbes.*

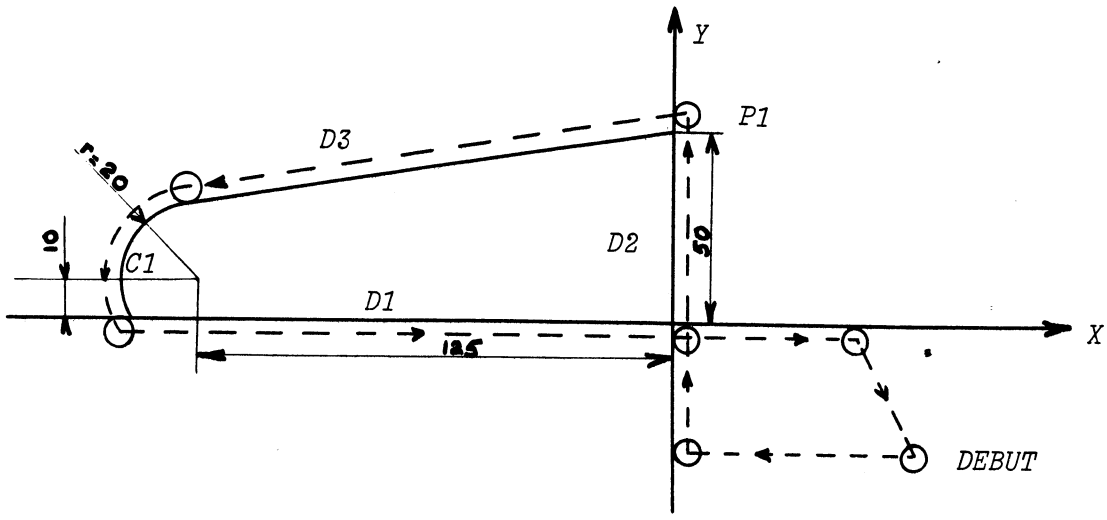


Figure 24 . *Exemple d'usinage APT (11).*

```
PARTNO PIECE D'ESSAI                          (1)
       MACHIN/UNIV,1                          (2)
       TRANS/-600,-700,0                      (3)
       INTOL/0.5                              (4)
       OUTTOL/0.5                             (5)
       MCHTOL/0.5                             (6)
REMARK DEFINITIONS GEOMETRIQUES               (7)
  C1 = CIRCLE/-125,10,0,20                    (8)
  D1 = LINE/0,0,0,-10,0,0                     (9)
  D2 = LINE/0,0,0,0,10,0                      (10)
  P1 = POINT/0,50,0                           (11)
  D3 = LINE/P1,RIGHT,TANTO,C1                 (12)
 PL1 = PLANE/0,0,1,0                          (13)
DEBUT = POINT/65,-35,8                        (14)
       CUTTER/60                              (15)
REMARK DEPLACEMENTS DE L'OUTIL                (16)
       FROM/DEBUT                             (17)
       SPINDL/900,CLW                         (18)
       FEDRAT/500                             (19)
       GO/TO,D2,TO,PL1                        (20)
       COOLNT/FLOOD                           (21)
       GORGT/D2,PAST,D3                       (22)
       GOLFT/D3,TANTO,C1                      (23)
       GOFWD/C1,PAST,D1                       (24)
       GOLFT/D1,PAST,D2                       (25)
       GOTO/50,0,50                           (26)
       COOLNT/OFF                             (27)
       FEDRAT/5000                            (28)
       GOTO/DEBUT                             (29)
       FINI                                   (30)
```

Examinons maintenant le programme ligne par ligne.

(1) donne le titre du programme

(2) définit le système de commande numérique

(3) situe l'origine des coordonnées de la machine par rapport aux coordonnées de la pièce.

(4) et (5) définissent les tolérances intérieures et extérieures

(6) définit la tolérance machine aux changements de direction de l'outil

(7) et (16) sont des commentaires sans influence sur le programme.

(8) définit le cercle C1 par les coordonnées de son centre et son rayon.

(9) définit la droite D1 par les coordonnées de deux points.

(10) définit la droite D2 par les coordonnées de deux points.

(11) définit le point P1 par ses coordonnées

(12) définit la droite D3 partant de P1 et tangente à droite au cercle C1

(13) définit le plan XOY par les coefficients de son équation.

(14) définit la position initiale de l'outil

(15) définit le diamètre de l'outil

(17) indique le point de départ

(18) vitesse de broche 900t/mn dans le sens des aiguilles d'une montre (CLW pour "clockwise")

(19) vitesse d'avance 500mm/mn

(20) avance de l'outil jusqu'à ce qu'il soit tangent à D2 et à PL1

(21) indique le mode d'arrosage et le déclenche

(22) usine la première face de la pièce

(23) usine la seconde face de la pièce

(24) usine le bout rond de la pièce

(25) usine la troisième face de la pièce

(26) dégage l'outil

(27) arrête l'arrosage

(28) déclenche la vitesse rapide

(29) retourne au point de départ

(30) indique la fin du programme

3.2.3. Conclusion.

Dans le domaine de la commande numérique, APT est le langage de programmation le plus complet qui existe ; le système APT est le seul système de programmation pour commande numérique qui soit universellement répandu et qui permette de traiter les usinages multi-axes.

Le langage APT est un langage relativement simple qui n'emploie qu'un petit nombre de constructions de base et qui, à cause de cela, a des formes lexicales très strictes et un grand nombre de mots réservés. Le nombre de constructions syntaxiques fixes est élevé ce qui gêne souvent le programmeur de pièce qui doit avoir en tête un grand nombre d'instructions à forme fixe (ceci est surtout vrai pour les instructions de définitions géométriques et les instructions de déplacement de l'outil) ; cela est aussi un inconvénient pour les programmes de traitement du langage. Le fait de ne pouvoir réutiliser les identificateurs peut être gênant pour les petits systèmes qui ne peuvent avoir de tables de symboles importantes. Certaines règles ne sont plus très utiles mais sont conservées pour des raisons de compatibilité. Ce sont par exemple les instructions **LOOPST et LOOPND** qui délimitent une zone de programme dans laquelle des boucles et des sauts seront effectués. Les premiers processeurs APT traitaient en effet des boucles comme des macro-instructions, conservant une copie de la partie de programme située entre **LOOPST et LOOPND** qu'ils traitaient chaque fois qu'une boucle était faite. L'instruction **MACRO** permet de définir de véritables macro-instructions ; on conserve une copie des instructions formant la **MACRO** et on traite cette copie chaque fois que l'on rencontre un appel à cette **MACRO**. APT est un langage qui a été défini à une époque où peu de langages symboliques existaient. Il a évolué avec le temps mais cette évolution a été surtout faite d'additions au langage ; il n'y a pas vraiment eu de révision de ce langage et on a conservé les formes existant depuis sa première définition.

Le principal défaut d'APT vient surtout du fait qu'en contournage deux dimensions le langage devient redondant et qu'il ne se prête que très mal au travail de point à point comportant des cycles d'usinage : c'est un langage essentiellement destiné aux usinages complexes. C'est à cause de cela qu'on a vu apparaître des langages tels **2CL, EXAPT et IFAPT** qui se limitent à certains travaux moins complexes et qui forment des langages plus adaptés à leur tâche, l'universalité d'APT n'étant valable que pour les problèmes d'usinage vraiment complexes.

3.3. PROGRAMMES D'ADAPTATION OU POSTPROCESSEURS.

Comme nous l'avons vu les données provenant du processeur doivent être adaptées aux caractéristiques particulières de la machine-outil utilisée pour l'usinage, au moyen d'un programme complémentaire appelé postprocesseur.

Le premier postprocesseur a été mis au point en 1961 chez Boeing à Seattle (U.S.A.), il était écrit en langage d'assemblage pour ordinateurs IBM 7090/7094. Actuellement les postprocesseurs sont écrits en **FORTRAN** et il en existe pour tous les gros ordinateurs pouvant disposer d'un processeur **APT**. Malgré la simplicité d'ensemble des programmes d'adaptation leur coût est généralement élevé ; c'est ainsi qu'un postprocesseur point à point coûtera environ 10 000 francs tandis qu'un postprocesseur continu 3 axes coûtera 25 000 francs (ce qui correspond à plusieurs centaines d'heures d'ingénieur) et un postprocesseur continu 6 axes coûtera 50 000 francs. On ne peut donc pas négliger les postprocesseurs dans l'évaluation d'un système de commande numérique.

3.3.1. Postprocesseurs point à point.

Les processeurs **APT** produisent un ensemble de données appelé **CLDATA** ou **CLTAPE**. Cet ensemble est constitué d'enregistrements de types différents : mots postprocesseur, définitions de surfaces, définitions de mouvements, fin de programme. Ces enregistrements sont de longueur variable et ont la forme générale suivante (3,33) :

Longueur de l'enregistrement	Mot de contrôle	Numéro de séquence	Classe	Sous-classe	Donnée 1	Donnée 2	.....

Les programmes d'adaptation traiteront les données suivant la classe et la sous-classe de l'enregistrement. La figure 25 donne la structure générale d'un postprocesseur.

Les postprocesseurs sont essentiellement constitués d'un grand nombre de sous-programmes spécialisés. Dans un but de généralisation (41,1) on divise les postprocesseurs en deux parties. Une première partie, dite

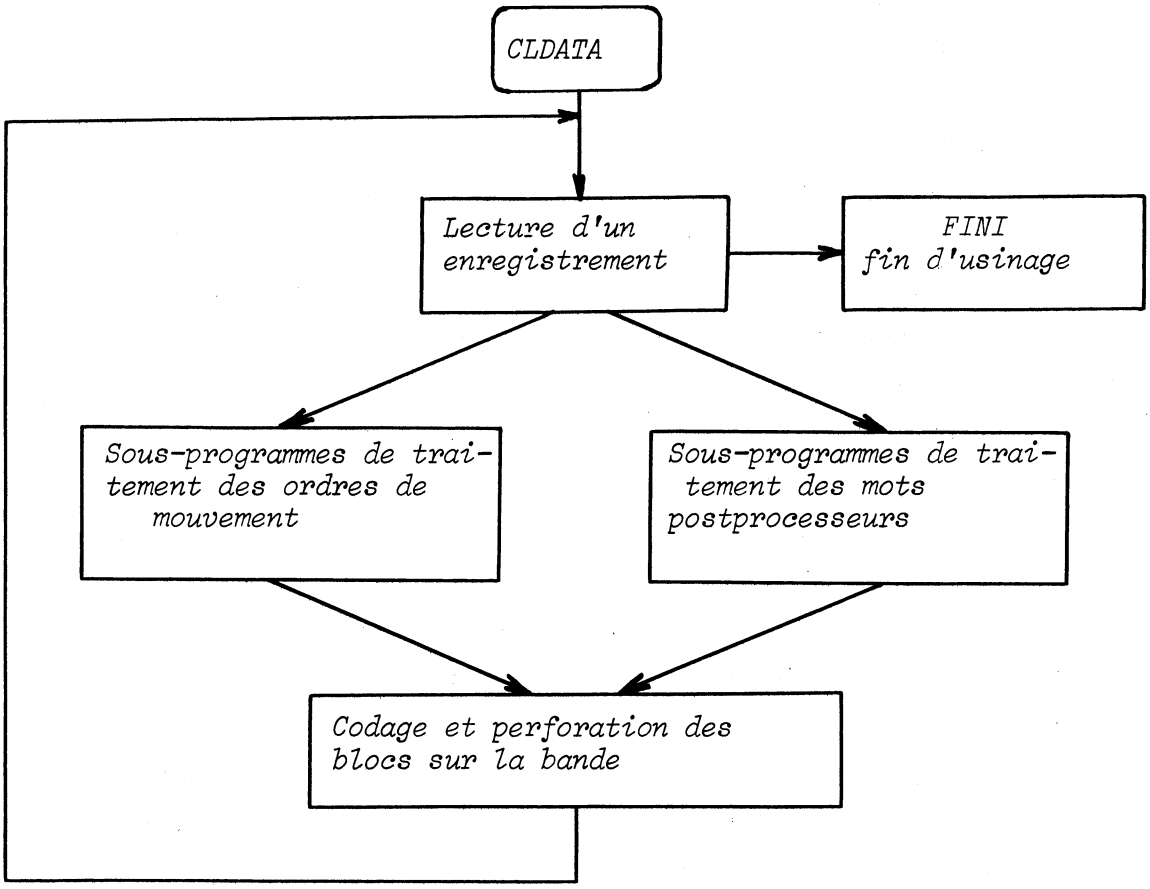


Figure 25 . Structure générale d'un postprocesseur

indépendante, regroupe les sous-programmes généraux traitant les ordres de déplacement et la géométrie ; une seconde partie, dite dépendante, comprend les sous-programmes particuliers et dépendant du type du système de commande numérique utilisé. Cette organisation des postprocesseurs permet une adaptation rapide et facile d'un postprocesseur à une machine-outil donnée, la seconde partie ne constituant généralement que le quart du postprocesseur.

Les sous-programmes de traitement des mots postprocesseur seront ceux qui engendreront les fonctions préparatoires et auxiliaires G et M que nous avons vues au chapitre 2 (voir figure 16), et comprendront aussi des sous-programmes traitant les cycles d'usinage. Chaque machine à commande numérique peut posséder des cycles particuliers. On définira dans le programme de pièce le cycle d'usinage que l'on désire effectuer au moyen des diverses instructions CYCLE ; ces instructions ont une forme donnée, mais suivant le système de commande numérique employé, le traitement de ces instructions pourra être différent, les cycles correspondant à un usinage donné pouvant différer d'un système à l'autre. Dans le cas d'un cycle de perçage et suivant la machine utilisée il sera par exemple possible de faire les choses suivantes :

1 . indiquer le cycle de perçage, la profondeur, la vitesse d'avance, la vitesse et le sens de rotation et le déclenchement de l'arrosage

2 . exécuter le cycle aux différents points dont les coordonnées sont spécifiées.

Tandis que pour un changement d'outil on pourra :

1 . annuler les cycles d'usinage précédents, remonter la broche en vitesse rapide et arrêter l'arrosage

2 . se déplacer en vitesse rapide jusqu'au point de changement d'outil ; changer l'outil

3 . revenir en vitesse rapide au point de travail

4 . descendre la broche en vitesse rapide à la cote du plan de travail ; annuler le cycle de déplacement rapide.

L'instruction FINI du programme de pièce indique la fin de l'usinage, dans ce cas on pourra :

1 . annuler les cycles d'usinage précédents, remonter la broche en vitesse rapide, arrêter l'arrosage

2 . aller en vitesse rapide jusqu'à l'origine des coordonnées, rembobiner le ruban et annuler les cycles de déplacement rapide.

Comme ces quelques exemples le montrent, les opérations à effectuer dans la plupart des cas sont relativement simples. Les programmes d'adaptation point à point sont également simples mais doivent comprendre un grand nombre de sous-programmes spécialisés ce qui explique leur coût relativement élevé.

2.3.2. Postprocesseurs en mode continu.

La structure des postprocesseurs pour le travail en mode continu correspond à celle de la figure 25 ; ils comprennent un ensemble de sous-programmes de traitement des ordres qui leur sont spécifiquement destinés et un ensemble de sous-programmes de traitement des ordres de déplacement. C'est dans ces derniers sous-programmes et dans ceux qui gèrent la perforation de la bande que l'on notera des différences par rapport aux postprocesseurs point à point.

Les sous-programmes de mouvement ont à résoudre les problèmes de l'interpolation. Nous en donnerons un aperçu en prenant un exemple (figure 26) : soit à usiner un arc de cercle entre deux segments de droite à l'aide d'un système de commande numérique Bunker-Ramo. Les données CLDATA comprendront :

1 . un ordre de mouvement jusqu'à X_1, Y_1, Z_1

2 . un enregistrement de définition du cercle C (X_c, Y_c, Z_c, et R_c)

3 . plusieurs ordres de mouvement indiquant des points intermédiaires (X', Y', Z', X", Y", Z") et aboutissant à X_2, Y_2, Z_2.

Les interpolateurs circulaires sont en général limités à un ou deux quadrants du cercle. Dans le cas pris en exemple, où l'interpolateur est limité à un quart de cercle, il faudra engendrer trois blocs de données correspondant aux trois quadrants mis en jeu par l'usinage de l'arc de cercle. A partir du rayon du cercle R_c et du premier point X_1, Y_1, Z_1 on calcule le rayon du cercle de la trajectoire de l'outil (décalée par rapport au cercle C) ; on calcule également le sens de rotation à l'aide des trois premiers points se trouvant dans les ordres de mouvements qui suivent la définition du cercle. On définit alors X, Y, i et j qui sont nécessaires à l'interpolateur Bunker-Ramo utilisé ; X et Y dépendent du nombre de quadrants auquel est limité l'interpolateur, i et j dépendent de la position de l'outil par rapport au cercle. La figure 26 donne un exemple ou X est identique à i ; si l'interpolateur utilisé était limité à deux quadrants du cercle, X serait différent de i. L'équipe APT recommande d'effectuer le premier mouvement de X_1, Y_1, Z_1 à X', Y', Z' en interpolation linéaire pour obtenir une décélération acceptable. L'interpolation n'est donc pas un problème complexe mais elle met en jeu des algorithmes qui peuvent varier avec le système de commande numérique utilisé.

Les problèmes les plus délicats que l'on rencontre au cours de l'écriture des postprocesseurs sont en fait des problèmes pratiques d'ajustement des vitesses et des déplacements élémentaires ainsi que des problèmes d'élaboration des blocs de la bande perforée d'une manière compatible avec les caractéristiques du système utilisé. L'élaboration du ruban perforé doit répondre à plusieurs impératifs qui ne sont pas toujours compatibles et qui sont les suivants :

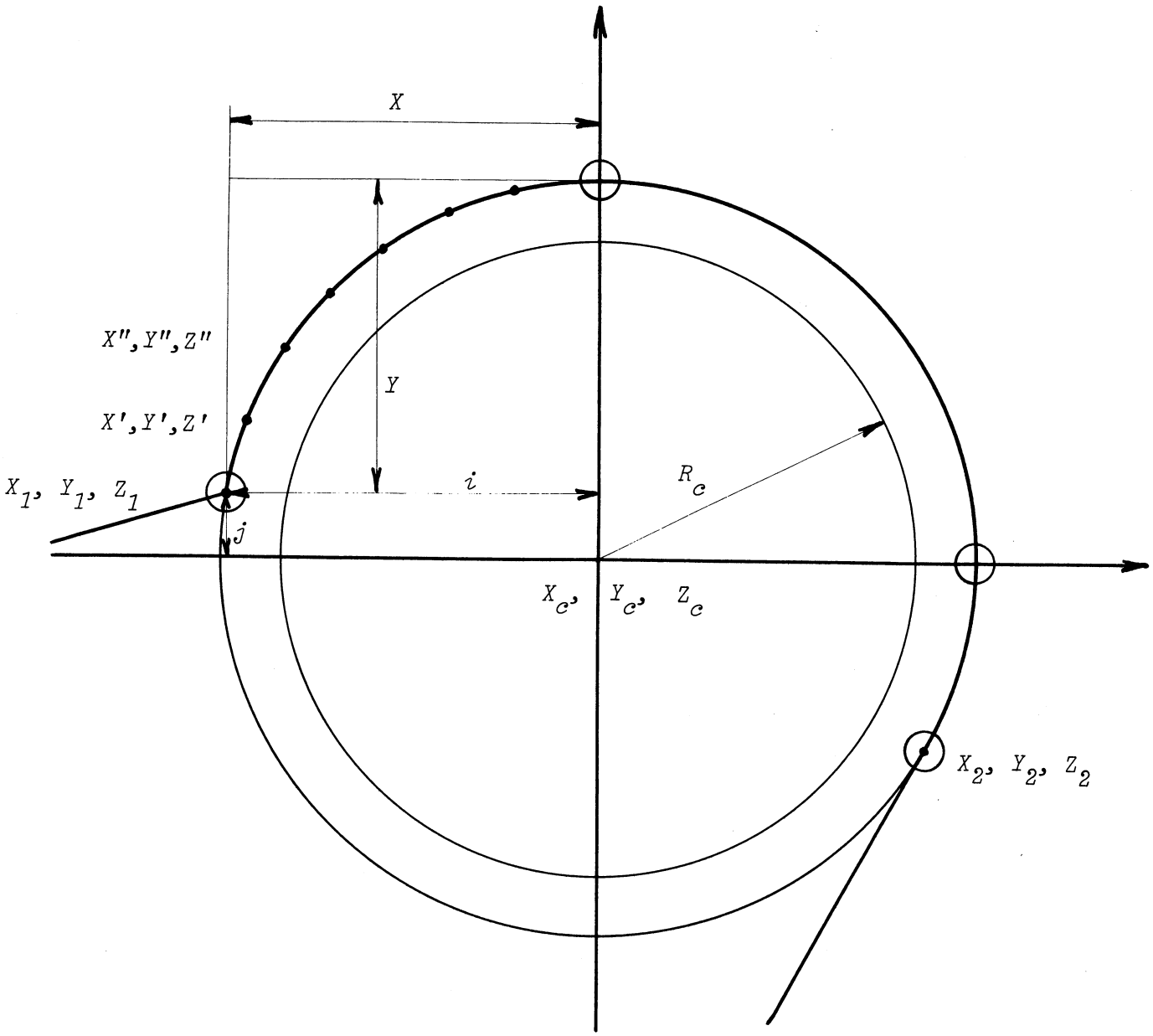


Figure 26 . *Exemple d'interpolation*

1 . Le ruban perforé doit être le plus court possible, les machines à commande numérique n'acceptant à cause de l'encombrement qu'une longueur maximum de 180 mètres.

2 . La vitesse de lecture du ruban est de 300 caractères par seconde.

3 . Chaque axe de la machine qui est commandé numériquement possède une vitesse de déplacement maximum.

4 . Lorsqu'on passe d'un bloc d'information à l'autre sur le ruban, la vitesse ne peut varier que d'un maximum de 250 mm/minute.

5 . Le déplacement relatif d'un bloc à l'autre ne peut excéder 9,999mm.

Un exemple nous permettra de voir comment s'appliquent ces critères (figure 27) : soit à usiner deux angles dans le plan de deux axes, X et Y, commandés numériquement et asservis. La vitesse maximum selon l'axe X est de 1500 mm par minute et celle selon l'axe Y est de 1000 mm par minute. L'usinage des angles B et C ne pourra se faire correctement qu'en annulant la vitesse selon l'axe Y aux points B et C.

L'usinage de l'angle B nous impose de décélérer avant B et d'accélérer après B selon l'axe Y. Le critère n° 4 nous impose une différence des vitesses inférieure ou égale à 250 mm/minute. Nous devrons donc avoir V_1 égale à + 125 mm/minute et V_2, dans le bloc suivant, égale à -125 mm/minute selon l'axe Y.

Supposons que l'on usine le segment BC, que l'on doive usiner en C un angle avec rebroussement du parcours selon l'axe Y, et que l'usinage BC soit fait à l'aide de 5 blocs d'information. La valeur de la vitesse selon Y sera de 125 mm/minute pour le premier bloc, la contrainte n° 4 nous impose alors des vitesses maxima de 375 mm/minute pour le second bloc,

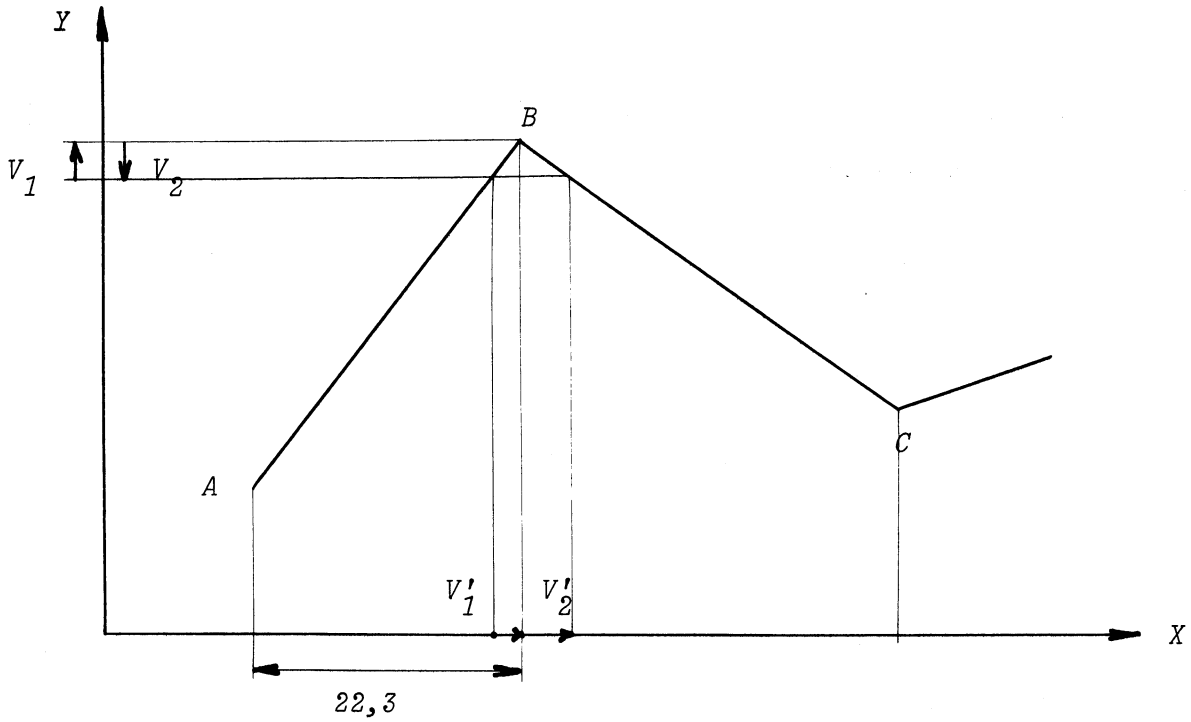


Figure 27 . *Exemple d'usinage avec angles.*

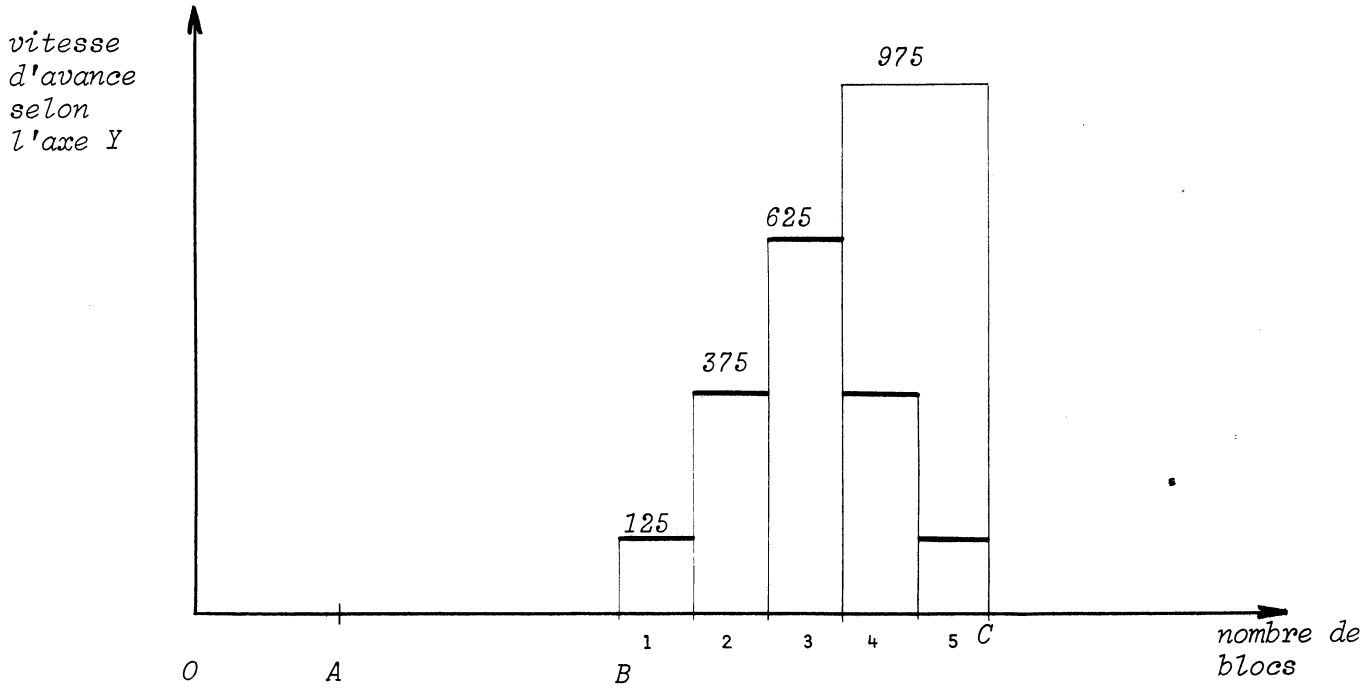


Figure 28 . *Détermination des vitesses.*

625 mm/minute pour le troisième bloc, et ainsi de suite comme le montre la figure 28. On augmente donc de bloc en bloc la vitesse du maximum autorisé jusqu'à ce qu'on atteigne la vitesse maximum permise sur l'axe considéré (ici 1000 mm/minute). Dans ces conditions on atteint le point C avec la vitesse 975 mm/minute ; si on doit usiner un angle en C, on risque de dépasser largement C à cause de la vitesse avant de pouvoir reprendre un chemin correct. Il faut donc revenir en arrière à partir de C en calculant les vitesses de chaque bloc comme on l'a fait à partir de B jusqu'à ce qu'on obtienne la courbe de vitesse finale.

Pour l'usinage du segment AB, qui a une longueur de 22,3 mm selon l'axe Y, le critère n° 5 nous impose soit de découper le segment en plusieurs blocs, soit de n'utiliser qu'un seul bloc avec un facteur de répétition de 10 ou de 100. Ces facteurs de répétition permettent de répéter l'exécution d'un bloc 10 ou 100 fois sans avoir à l'écrire 10 ou 100 fois sur la bande et permettent donc une économie sur la longueur de bande utilisée (critère n° 1).

Certains systèmes de commande numérique n'acceptent que des vitesses de deux chiffres. Si dans notre exemple nous optons pour un facteur de répétition de 10, chaque bloc aura un déplacement de 2,23 mm selon Y. La vitesse d'avance selon Y sera donc (à 10 % près) : F = 900/2,23=400, ce qui est trop élevé et nous oblige à choisir la solution du découpage du segment AB en trois blocs ce qui donne une vitesse d'avance de 40 mm/minute. Il faut bien entendu répéter ces calculs pour l'axe X.

Les sytèmes de commande numérique travaillent à l'aide de deux "buffers", un "buffer" actif pour l'exécution du bloc n et un "buffer" tampon pour la lecture du bloc n+1. Le temps de lecture du bloc tampon n+1 doit être inférieur au temps d'exécution du bloc n actif ; il ne peut en effet y avoir de temps mort pour la machine à commande numérique qui, en l'absence de l'ordre suivant, continue le travail précédent et risque de détériorer la pièce et de subir des à-coups. Il faut donc s'assurer que le temps d'exécution du bloc n est supérieur au temps de lecture du bloc n+1. Si tel n'est pas le cas, il faut augmenter le temps d'exécu-

tion du bloc n, en réduisant la vitesse par exemple. Si, ceci fait, la variation de vitesse entre le bloc n et le bloc n-1 est supérieure à 250 mm/minute il faut modifier le bloc n-1, et ainsi de suite. Ceci oblige à conserver dans une table les 40 blocs précédant le bloc courant pour correction, solution recommandée par l'équipe APT.

Tous ces problèmes sont des problèmes essentiellement pratiques pour lesquels n'existe aucune méthode générale. Chaque machine présente d'ailleurs des caractéristiques particulières qui rendent la généralisation des postprocesseurs difficile. Cependant il arrive que l'on retrouve les mêmes problèmes avec des machines différentes ce qui peut quand même permettre une certaine généralisation (1). Les machines-outils et les systèmes de commande numérique évoluent constamment et il n'est pas impossible que soient remises en question l'organisation générale et la structure des postprocesseurs futurs.

CHAPITRE 4

REALISATION DU SYSTEME

4.1. DESCRIPTION DU NOUVEAU SYSTEME

4.1.1. Conception.

A partir d'APT ont été réalisés des systèmes de programmation en commande numérique dont les langages sont des sous-ensembles spécialisés d'APT. Les sous-ensembles choisis permettent de travailler dans un système de coordonnées à deux axes en point à point (perçage) ou en continu (tournage), ou à deux axes et demi (exécution des surfaces par courbes de niveau) tandis qu'APT permet un travail dans un système de coordonnées tridimensionnel avec une machine ayant jusqu'à cinq axes commandés numériquement. Il est admis maintenant que le langage APT est trop riche pour la majorité des applications et qu'il est par conséquent possible de n'en utiliser que des sous-ensembles spécialisés. Malgré les simplifications dues aux sous-ensembles choisis, l'introduction des données technologiques d'EXAPT représente une extension du système qui n'a pas permis de réduire la taille du processeur utilisé. On recherche pourtant actuellement une simplification permettant de réduire la taille du programme de traitement et d'utiliser des ordinateurs moins importants (35, 52).

Suivant la tendance actuelle de la normalisation des langages type-APT, notre système est apparenté à APT pour le langage d'écriture des programmes de pièces, mais il est beaucoup moins général que les systèmes de la "famille APT" (21). Notre but est en effet un système pouvant fonctionner sur petits ordinateurs et qui soit facilement adaptable à des problèmes particuliers(10).

Nous avons essayé d'autre part de rendre ce système aussi modulaire que possible ; pour ce faire nous sommes partis d'un état minimal du système. Le travail d'usinage en commande numérique le plus simple étant le travail de point à point, le noyau du système permettra un travail de point à point minimal. Il sera donc possible d'y inclure des instructions

et des ordres nouveaux permettant un travail d'usinage en continu.

C'est pourquoi nous avons repris une idée introduite par MECALGOL et retenu comme solution un système simplifié à base de sous-programmes courts et indépendants les uns des autres(40,53)Ce système permettra d'introduire facilement des modifications et sa simplicité permettra une maintenance aisée. Les programmes d'adaptation ou postprocesseurs posant souvent de nombreux problèmes, le système permettra l'utilisation ou la non utilisation de ces programmes d'adaptation.

Nous avons décidé d'écrire le système en FORTRAN basique (12,13,26) afin de rester aussi indépendant de l'ordinateur que possible. Il existe en effet des compilateurs pouvant traiter des programmes écrits en FORTRAN basique sur la plupart des petits ordinateurs, et on peut ainsi passer d'un ordinateur à l'autre sans avoir à beaucoup modifier la structure des programmes. L'utilisation du FORTRAN permet d'autre part une mise au point des programmes beaucoup plus rapide et également l'obtention d'un système plus accessible à des programmeurs non spécialistes des systèmes.

4.1.2. Structure du système.

La figure 29 schématise l'organisation du système tel qu'il a été réalisé. On peut diviser le système en deux phases principales.

a . La phase 1 accepte en entrée le programme de pièce symbolique qui est formé d'instructions APT et, en option spéciale, d'instruction FORTRAN. Le programme de pièce est traité par un préprocesseur dirigé par des tables qui, après avoir vérifié sa syntaxe, le traduit en un programme FORTRAN intermédiaire constitué essentiellement d'appels de sous-programmes et de fonctions.

b . La phase 2 reprend ce programme intermédiaire et le transmet à un compilateur FORTRAN classique. On exécute alors le programme compilé grâce à une bibliothèque de sous-programmes de

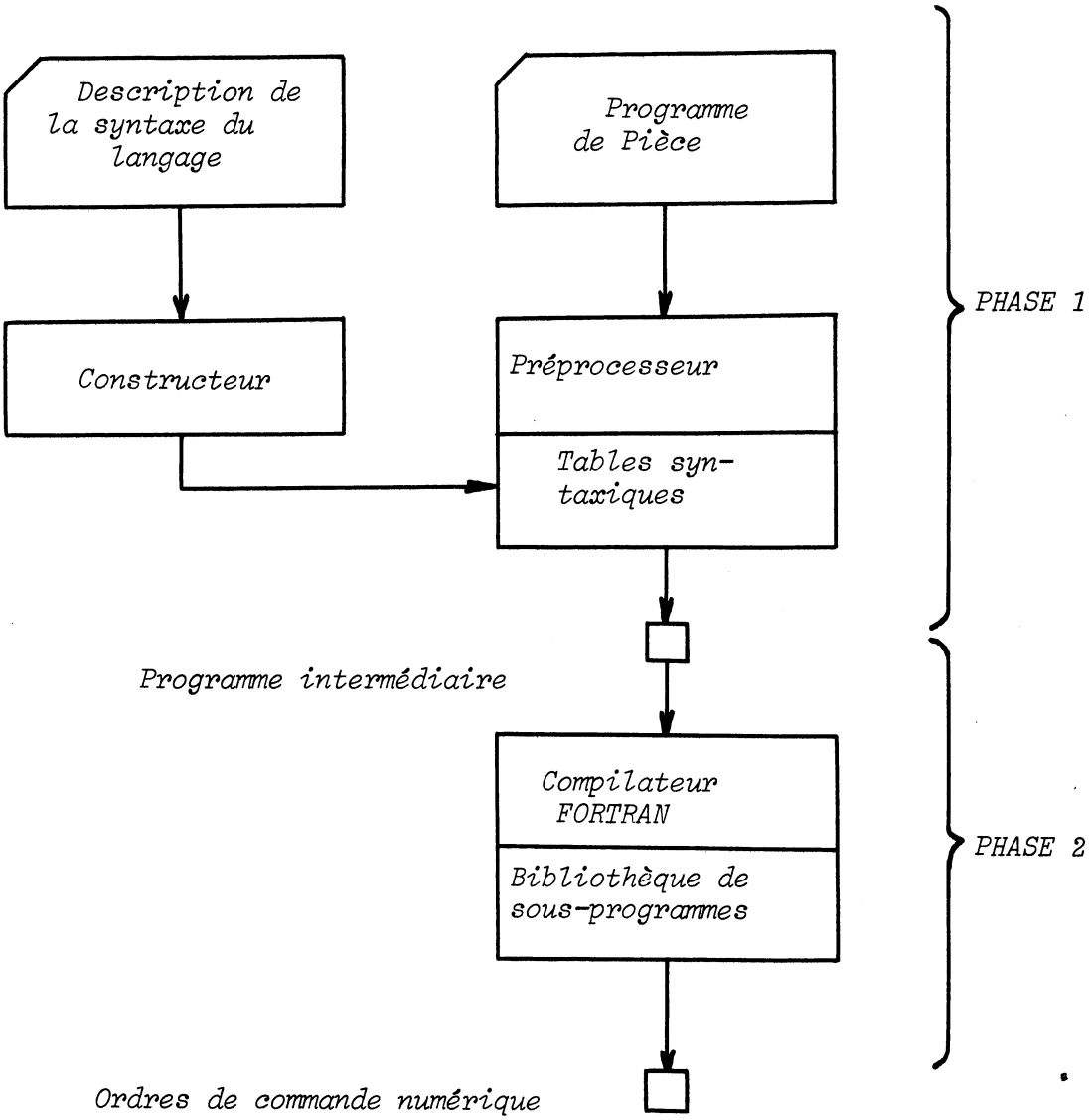


Figure 29 . *Structure du système*

définitions géométriques et d'ordres d'usinage. Le résultat de cette exécution est une suite d'ordres numériques dans un format donné correspondant soit au système de commande numérique d'une machine-outil donnée, soit à l'ensemble CLDATA des systèmes APT classiques.

Le passage de la phase 1 à la phase 2 dépend essentiellement de l'ordinateur utilisé.

Une troisième partie du système, le "constructeur", n'intervient pas directement dans le processus de traduction allant du programme de pièce aux ordres de commande numérique. Le préprocesseur est dirigé par des tables décrivant la syntaxe du langage d'écriture des programmes de pièce. Le but du constructeur est de construire ces tables syntaxiques et également de les modifier en ajoutant ou en supprimant des instructions au langage d'entrée. Ceci permet de modifier le système et d'y faire des extensions utiles.

4.1.3. Réalisation.

L'écriture et la mise au point du système de programmation pour la commande numérique des machines-outils ont été faites au moyen des systèmes CP67/CMS (14) qui sont utilisés à Grenoble sur l'ordinateur IBM 360/67. Ce système a été ensuite adapté à un ordinateur IBM 1130 mais dans ce chapitre nous nous en tiendrons au système général fonctionnant en mode conversationnel sous CP67/CMS. Le système a été entièrement écrit en FORTRAN basique (23,24).

La liaison entre les deux phases du système dépend essentiellement du calculateur utilisé. Il faudra donc étudier le problème pour chaque ordinateur utilisé. Notre but principal étant de produire un système assez général pour être adaptable à plusieurs ordinateurs de petite taille, nous n'étudierons pas cette liaison en détail ici.

4.2. PREPROCESSEUR.

Le préprocesseur traite le programme de pièce et le traduit en un programme intermédiaire.

La méthode que nous avons retenue pour l'analyse des instructions d'un programme de pièce tient compte de la forme particulière des instructions APT. Nous avons vu au chapitre 3 qu'une instruction APT non arithmétique se compose d'une section majeure et d'une section complémentaire, toutes deux séparées par une barre oblique. La section majeure comporte toujours un mot majeur, la section complémentaire est formée d'une série de mots mineurs ou de symboles.

La plupart des instructions APT sont à "forme fixe" : leur section complémentaire a effectivement une longueur et une composition fixées, les éléments de cette section complémentaire devant être spécifiés dans un ordre donné. L'instruction de définition géométrique suivante :

C1 = CIRCLE / CENTER, P1, RADIUS, 12

qui permet de définir un cercle C1 par son centre P1 et son rayon 12 est un exemple de ce type d'instruction à "forme fixe". La section majeure comporte le mot majeur CIRCLE et la section complémentaire comporte quatre éléments qui doivent tous être spécifiés dans cet ordre (mot mineur CENTER, symbole définissant un point, mot mineur RADIUS, valeur numérique). Les instructions à "forme fixe" comportent la plupart des instructions de définitions géométriques et des instructions d'usinage.

Un petit nombre d'instructions APT sont à "forme variable" : leur section complémentaire a une longueur et une composition variables. C'est le cas de l'instruction suivante :

RESAU2=PATERN/ARC,C1,30,CCLW,INCR,20,10,3,AT,15,30,10

qui définit un réseau de points sur un arc du cercle C1, le premier point se trouvant sur C1 à 30° par rapport à l'axe OX dans le sens inverse des aiguilles d'une montre (CCLW), les autres points étant définis à partir de lui au moyen d'une liste d'incréments (20°, 10°, 15°, 15°, 15°, 30°, 10°). Dans de telles instructions ni la longueur ni la forme de la liste d'incréments ne sont fixes. Les instructions à "forme variable" comprennent un petit nombre d'instructions de définitions géométriques et d'instructions d'usinage destinées au travail en mode point à point et qui ont trait aux réseaux de points.

Initialement et si l'on met à part les instructions d'affectation arithmétiques et les instructions de définition et d'appel de Macros, le langage APT ne comportait que des instructions à "forme fixe" ; les extensions du langage initial ont apporté de nouvelles instructions dont certaines présentent une "forme variable".

4.2.1. Langage intermédiaire.

Le programme intermédiaire est essentiellement composé d'appels de sous-programmes. Le langage utilisé pour l'écriture de ce programme intermédiaire, ou langage intermédiaire, est le langage FORTRAN basique. L'adoption du langage FORTRAN de base comme langage intermédiaire permet d'une part d'obtenir un système indépendant de l'ordinateur utilisé ; elle permet d'autre part au préprocesseur de conserver les instructions d'affectation arithmétiques APT du programme de pièce telles quelles et de ne pas avoir à les traiter, car elles ont une forme identique à celle des instructions FORTRAN du même type. L'emploi du FORTRAN basique permet en outre d'utiliser toute la puissance du FORTRAN si nécessaire.

Le langage intermédiaire pourrait également être un langage simplifié destiné à être interprété. Dans le cas de l'utilisation du FORTRAN basique comme langage intermédiaire, le préprocesseur doit engendrer des chaînes de caractères correspondant aux instructions du programme intermédiaire. Dans le cas d'un langage destiné a être interprété ce problème sera simplifié ; on pourra engendrer uniquement un numéro qui sera alors interprété au moyen d'un sous-programme se trouvant en bibliothèque. Dans ce cas le système dépendra de l'ordinateur utilisé et ne sera donc pas transférable aussi facilement d'un calculateur à un autre ; le préprocesseur devra également traiter les instructions d'affectation arithmétiques du programme de pièce.

4.2.2. Fonctionnement du préprocesseur.

La structure du préprocesseur est fonction du langage de programmation type-APT utilisé pour écrire les programmes de pièce. La figure 30 donne un organigramme général du préprocesseur. Après la lecture de chaque instruction du programme de pièce, on repère et on empile les différentes unités syntaxiques comme le montre l'exemple de la figure 31 ; au fur et à mesure de l'empilage des unités syntaxiques on reconnait et on traite les imbrications.

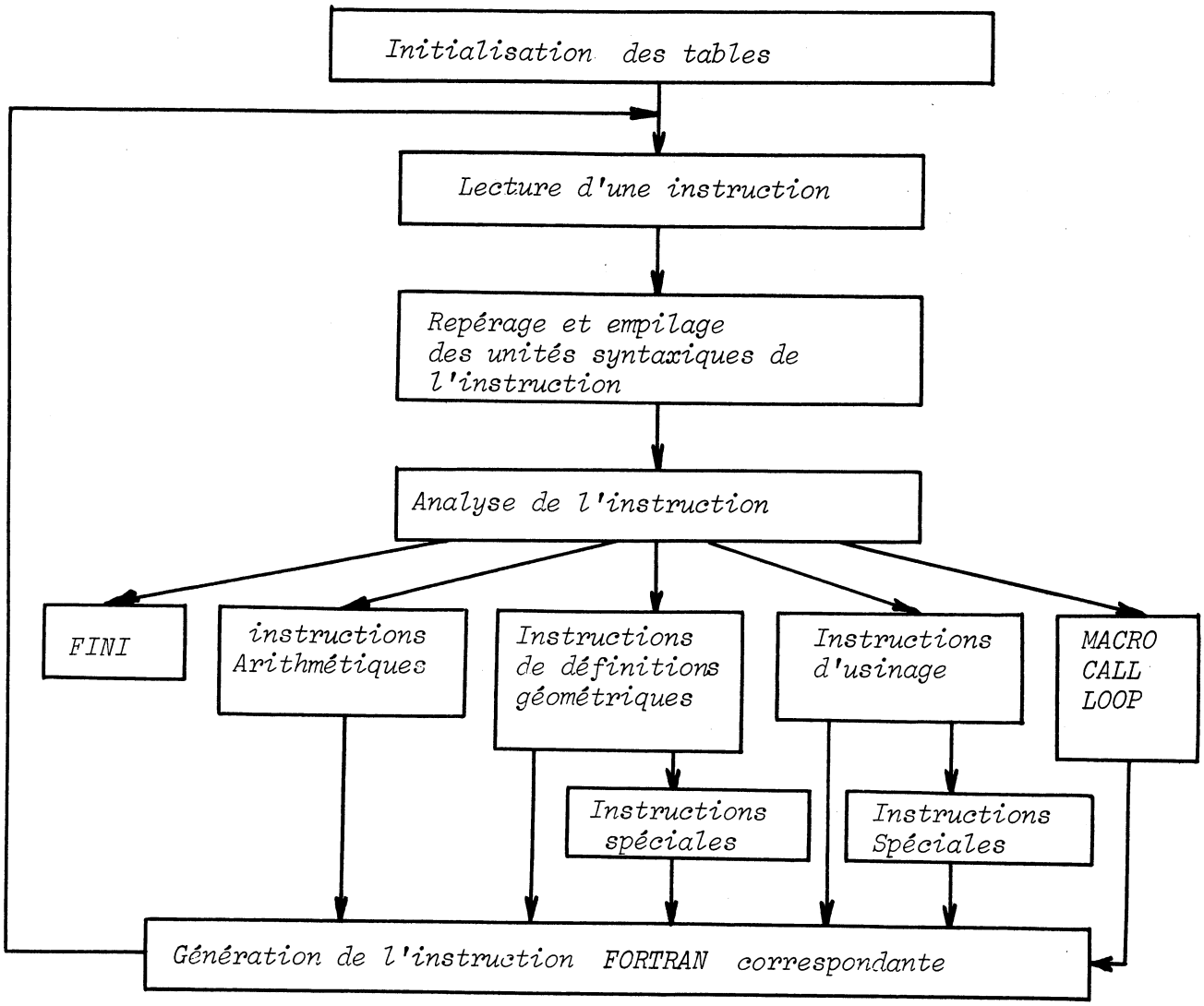
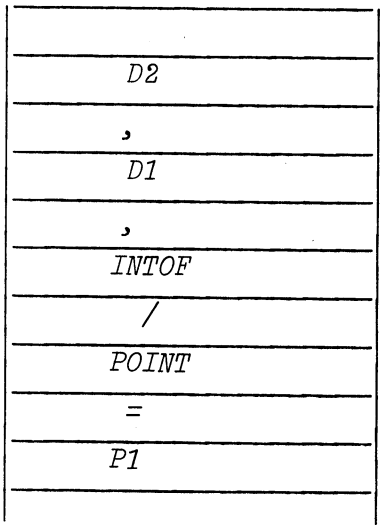


Figure 30 Organigramme général du préprocesseur.

P1=POINT/INTOF,D1,D2



Pile

Figure 31. Exemple d'empilage des unités syntaxiques d'une instruction

Pour analyser une instruction on reconnait d'abord le mot majeur qui permet de déterminer le type de l'instruction.

a) La plupart des instructions d'un programme de pièce sont des instructions de définitions géométriques ou des instructions d'usinage à "forme fixe". Pour ces instructions le processus de reconnaissance et de traduction est automatisé grâce à l'utilisation d'un codage simple des règles syntaxiques. En effet pour les instructions à "forme fixe", à chaque mot majeur du dictionnaire APT sont associées une ou plusieurs règles syntaxiques correspondant à toutes les sections complémentaires pouvant suivre ce mot majeur ; ces différentes sections complémentaires sont codées et placées dans une table syntaxique.

L'analyse d'une instruction à "forme fixe" aboutit à la création d'un masque numérique codé qui reproduit strictement la structure de l'instruction. On recherche alors dans la table syntaxique les règles syntaxiques ayant trait au mot majeur de l'instruction traitée et on les compare globalement une à une au masque numérique codé. Si aucune des règles ne correspond au masque, l'instruction est erronée, un message d'erreur est imprimé et l'instruction est ignorée. Au contraire lorsqu'une règle coïncide exactement avec le masque, la reconnaissance est terminée avec succès. On engendre alors un appel de sous-programme ou de fonction FORTRAN correspondant à l'instruction APT analysée. Les noms de ces sous-programmes ou de ces fonctions sont obtenus en combinant les 4 premiers caractères du mot majeur de l'instruction APT traitée au numéro de la règle syntaxique utilisée. Ainsi l'instruction APT suivante :

```
C1 = CIRCLE / CENTER, P1, RADIUS, 12
```

sera traduite en :

```
C1 = CIRC2 (P1, 12)
```

où CIRC2 est la combinaison du mot majeur CIRCLE et du numéro de la définition de cercle utilisée, P1 et 12 sont les paramètres principaux de l'instruction. Cette instruction aurait également pu être traduite en :

```
CALL CIRC2 (C1, P1, 12)
```

La première solution a une forme plus proche de celle de l'instruction APT et c'est celle que nous avons retenue.

De même l'instruction APT suivante :

SPINDL / 1500, CLW

sera traduite en :

CALL SPIN1 (2, 1500)

où SPIN1 est la combinaison du mot majeur SPINDL et du numéro de la règle syntaxique utilisée, 2 est le sous-code numérique correspondant à CLW ("clockwise") et 1500 est le paramètre principal.

b) Les instructions à "forme variable", beaucoup moins nombreuses, doivent subir un traitement particulier, la méthode de comparaison globale des sections complémentaires ne pouvant leur être appliquée. Les instructions de définitions géométriques et d'usinage à "forme variable" ayant trait aux réseaux de points sont traitées par des sous-programmes spéciaux. Les problèmes posés par ces instructions sont d'une part la reconnaissance de sections complémentaires de forme variable et d'autre part la génération d'appels de sous-programmes. Le premier problème est facilement résolu, la forme des diverses sections complémentaires pouvant être représentée au moyen d'arborescences ; pour résoudre le second problème nous avons été amenés à engendrer pour chaque instruction de ce type un ou plusieurs appels à un même sous-programme, chaque appel comportant un nombre fixe de paramètres. Une instruction de ce type peut donc être traduite par un ou plusieurs appels de sous-programmes FORTRAN.

Le traitement des instructions de définition et d'appel de Macros est fait au moyen de sous-programmes spéciaux qui nécessitent l'utilisation de tables permettant d'associer aux différents noms de Macros les noms des paramètres qui leur sont relatifs et les valeurs normales de ces paramètres (fixées dans la définition de la Macro). Les macros elles-mêmes sont traduites en sous-programmes FORTRAN qui sont conservés pendant le traitement du programme de pièce sur un fichier auxiliaire, et qui sont placés à la suite du programme intermédiaire FORTRAN une fois le traitement terminé.

Les instructions LOOPST et LOOPND qui délimitent les parties du programme à l'intérieur desquelles sont faits les boucles et les sauts, sont traitées par un sous-programme spécial. Ces instructions ne nous sont pas utiles - car l'utilisation de FORTRAN comme langage intermédiaire nous donne des possibilités de sauts et de boucles plus souples ; nous les avons conservées par souci de compatibilité avec les systèmes type-APT existants.

Les instructions de test et de saut IF et JUMPTO sont traitées par des sous-programmes particuliers ayant pour tâche le remplacement des étiquettes APT qui peuvent être alphanumériques par des étiquettes numériques FORTRAN. Il existe à cet effet des tables de correspondance des étiquettes ; à la première apparition d'une étiquette APT dans le programme de pièce, soit comme étiquette de l'instruction traitée, soit comme paramètre d'une instruction IF ou JUMPTO, on lui associe une étiquette équivalente FORTRAN ; l'équivalence des deux étiquettes est conservée tout au long du programme de pièce.

c) Les instructions d'affectation n'appartenant pas aux deux groupes précédents sont vérifiées partiellement avant d'être acceptées comme instructions d'affectation arithmétiques. Elles sont alors conservées telles qelles puisqu'elles ont déjà la forme des instructions FORTRAN.

d) L'instruction FINI indique la fin du programme de pièce et provoque la génération d'un appel au sous-programme de fin d'usinage, suivi de l'ordre FORTRAN END.

Notons enfin que ce système accepte des instructions FORTRAN aussi bien que des instructions type-APT ; toutes les instructions comportant un astérisque en première colonne sont prises comme instructions FORTRAN et sont transmises telles quelles au programme intermédiaire.

Le système général fonctionnant sous les systèmes CP67/CMS permet l'utilisation des définitions imbriquées. Au cours du repérage et de l'empilage des unités syntaxiques de chaque instruction du programme de pièce la rencontre d'une parenthèse droite provoque le traitement de la définition imbriquée. S'il s'agit réellement d'une instruction imbriquée, et non d'une

variable indicée, on la traite, on engendre l'instruction FORTRAN correspondante et on remplace l'instruction imbriquée dans la pile par le symbole correspondant. Cette méthode permet de désimbriquer systématiquement les instructions et d'aboutir en fin de balayage de chaque instruction du programme de pièce, à une instruction APT simple sans imbrication.

Soit l'instruction APT suivante, définissant un point P1 comme l'intersection d'une droite et d'un cercle, eux-mêmes définis à l'intérieur de cette instruction :

```
P1=POINT/XLARGE,INTOF,(LINE/0,0,1,1),$
(C1=CIRCLE/CENTER,(POINT/3,2),RADIUS,10)
```

La longueur de l'instruction ne lui permettant pas de tenir sur une seule carte, elle a été coupée en deux, le signe "$" de la première ligne indique que l'instruction se poursuit sur la ligne suivante. La définition imbriquée de la droite (LINE/0,0,1,1) n'attribue aucun nom symbolique à cette droite ; elle ne sera donc pas connue à l'extérieur de l'instruction de définition du point P1. Par contre la définition imbriquée du cercle lui attribue le nom symbolique C1 ; le cercle C1 pourra donc être utilisé dans d'autres parties du programme. L'instruction globale est traduite par la suite d'instructions suivante :

```
Z990 = LINE0 ( 0 , 0 , 1 , 1 )
Z991 = POIN0 ( 3 , 2 )
C1 = CIRC2 ( Z991 , 10 )
P1 = POIN3 ( 1 , Z990 , C1 )
```

Les définitions imbriquées qui ne possèdent pas de nom symbolique définissent des figures géométriques temporaires dont l'existence est subordonnée à l'exécution de l'instruction qui les englobe. On leur attribue des noms symboliques temporaires (Z990 et Z991). Après exécution de l'instruction de définition du point P1 les figures Z990 et Z991 n'existent plus, les symboles Z990 et Z991 sont libérés et peuvent être utilisés pour d'autres définitions imbriquées.

La méthode d'analyse choisie pour les instructions à "forme fixe" introduit une automatisation qui permet d'augmenter la rapidité du traitement et de diminuer la taille et le nombre des programmes de traitement. Cette

méthode présente l'inconvénient de ne pas permettre la description de toutes les règles syntaxiques du langage puisqu'elle ne peut s'appliquer qu'aux instructions à "forme fixe". Une structure de liste plus complète aurait permis de décrire toutes les règles syntaxiques du langage, à "forme fixe" comme à "forme variable", mais ne parait pas souhaitable dans le contexte d'utilisation.

La version du préprocesseur fonctionnant sous CP67/CMS permet de traiter la plupart des instructions du langage APT sans les restrictions des versions destinées à l'ordinateur IBM 1130. Cette version générale n'a été utilisée jusqu'à présent que pour la mise au point et l'évaluation de nouveaux sous-programmes.

4.2.3. Table syntaxique.

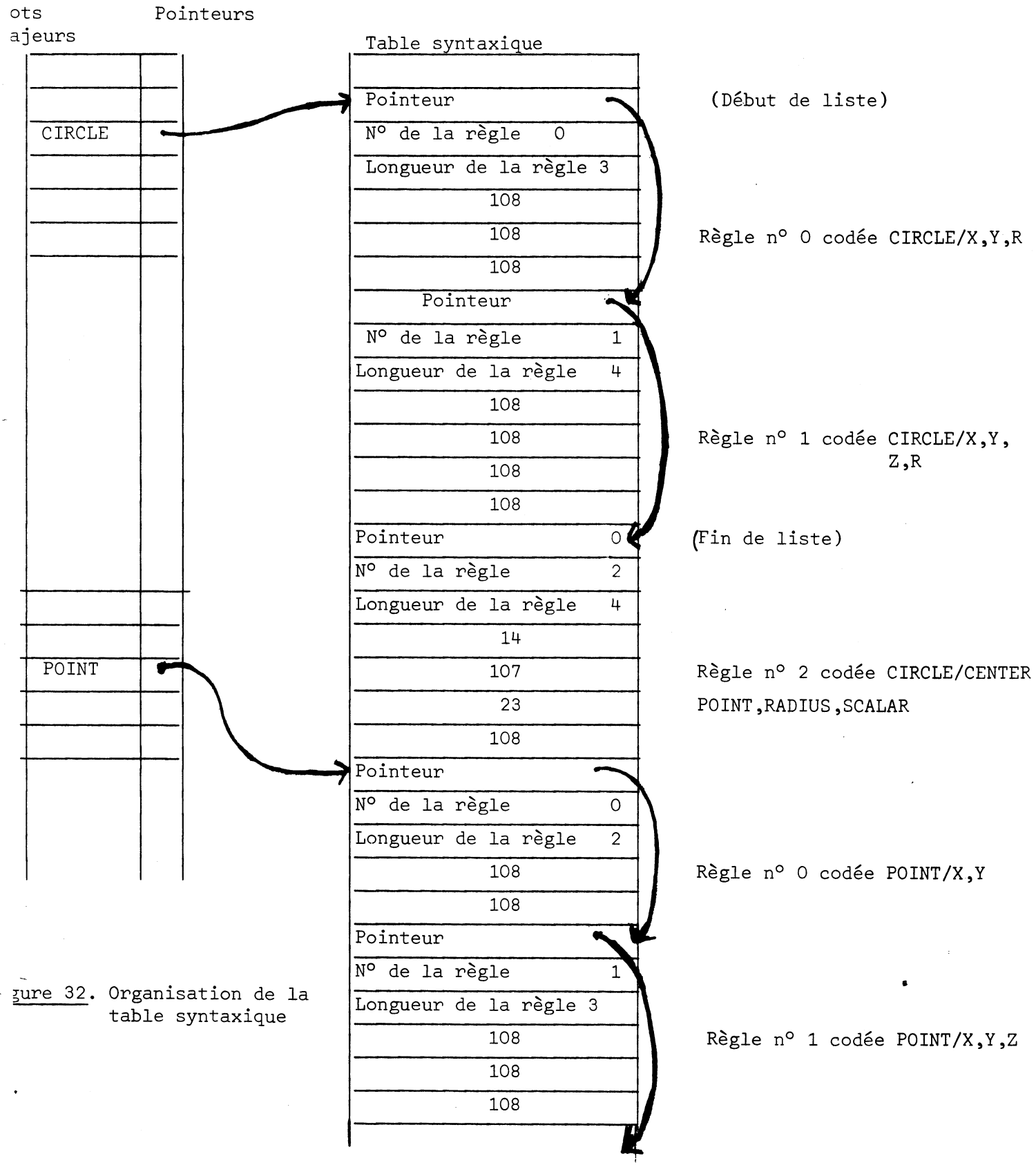
La table syntaxique groupe les règles syntaxiques codées numériquement et correspondant aux instructions à "forme fixe". Ainsi la définition de cercle suivante (par le centre et le rayon) :

CIRCLE/CENTER, POINT, RADIUS, SCALAR

correspondra à la forme codée suivante : (14-107-23-108) qui représente la forme de la section complémentaire et par conséquent une règle syntaxique relative au mot majeur CIRCLE (voir Figure 32) ; 14 est le code numérique du mot mineur CENTER, 107 est le code représentant un symbole correspondant à une définition de point, 23 est le code numérique du mot mineur RADIUS et 108 est le code correspondant soit à une variable arithmétique soit à un nombre.

Certains mots mineurs jouent un rôle particulier, ce sont par exemple les modificateurs utilisés pour lever les ambiguïtés de certaines définitions géométriques ; on peut les grouper suivant le rôle qu'ils jouent au point de vue syntaxique. Tous les membres d'un même groupe possèderont le même code numérique, mais chaque membre du groupe possèdera un sous-code particulier qui sera utilisé comme indicateur. Ainsi XLARGE, XSMALL, YLARGE et YSMALL qui jouent le même rôle dans la syntaxe feront partie du même groupe et possèderont le même code numérique, on leur associera un sous-code permettant de les différencier dans la suite du traitement.

La table syntaxique est organisée de telle manière que les règles correspondant au même mot majeur (comme par exemple les définitions de cercles) soient



gure 32. Organisation de la table syntaxique

chaînées entre elles. Ceci permet de modifier facilement la table syntaxique en lui ajoutant de nouvelles règles ou en supprimant des règles existantes. La figure 32 décrit l'organisation de la table syntaxique telle qu'elle est utilisée par le préprocesseur. Chaque mot majeur possède un pointeur permettant de retrouver la première règle syntaxique qui lui est relative. Le chaînage interne de la table syntaxique permet alors de retrouver successivement toutes les règles syntaxiques relatives au même mot majeur. Le chaînage interne de la table syntaxique est un chaînage simple, chaque règle possèdant un pointeur qui indique la règle suivante relative au même mot majeur. On peut ainsi supprimer une ou plusieurs règles existantes relatives à un mot majeur sans affecter les autres règles relatives à ce même mot majeur ou bien rajouter de nouvelles règles.

4.2.4. Exemple.

Prenons maintenant un exemple simple qui nous permettra d'illustrer l'utilisation du préprocesseur. La figure 33 donne la disposition de huit trous à percer dans une pièce mécanique. Les huit trous doivent d'abord être centrés ; ils doivent être ensuite percés à un diamètre de 10 mm. La figure 34 donne le programme de pièce utilisé pour résoudre ce problème. Un réseau PAT1 définit les trois premiers trous à percer ; un second réseau PAT2 définit les cinq trous suivants situés sur un arc de cercle. Le réseau PAT3 regroupe les réseaux PAT1 et PAT2 précédemment définis et comprend les huits points à percer. On donne ensuite deux séries d'ordres d'usinage définissant les outils, les cycles de perçage et les trous à percer. Le préprocesseur accepte ce programme de pièce, le traite et produit le programme intermédiaire FORTRAN de la figure 37.

4.3. BIBLIOTHEQUE DE SOUS-PROGRAMMES.

L'exécution des sous-programmes de la bibliothèque permet de créer un ensemble de figures géométriques et d'engendrer une suite d'ordres numériques. Chaque figure géométrique élémentaire (point, cercle, droite, etc...) est définie dans un système de coordonnées absolues. Comme tous les sous-programmes de la bibliothèque doivent avoir accès à l'ensemble des figures géométriques, celles-ci sont placées dans une zone commune. Les données numériques définissant chaque figure géométrique sont ainsi conservées dans un tableau, au sens FORTRAN du terme, commun à tous les sous-programmes. On repère alors chaque figure géométrique par sa position dans le tableau commun. Les sous-programmes doivent avoir accès à l'ensemble des figures géométriques et également à un certain nombre de renseignements auxiliaires comme le numéro de l'outil, les données

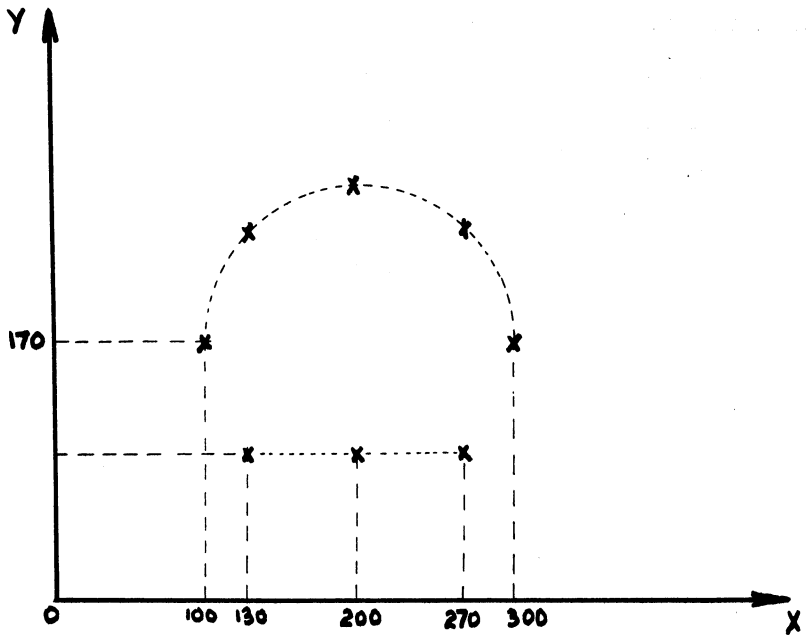


Figure 33 . *Exemple de perçage d'une pièce*

```
PARTNO EXEMPLE DE RESEAU
P1=POINT/130.0,100.0
P2=POINT/270.0,100.0
C1=CIRCLE/200.0,170.0,100.0
PAT1=PATERN/LINEAR,P1,P2,3
PAT2=PATERN/ARC,C1,0.0,180.0,CCLW,5
PAT3=PATERN/RANDOM,PAT1,PAT2
TOOLNO/10,215.0
CYCLE/DRILL,15.0,FE.0,MPM,1.0
GOTO/PAT3
TOOLNO/15,235.0
CYCLE/DRILL,15.0,135.0,MPM,1.0
GOTO/PAT3
FINI
```

Figure 34 . *Programme de pièce exemple.*

relatives au cycle d'usinage en cours,
la position courante de l'outil, etc... Pour des raisons de simplicité ces renseignements auxiliaires ont été groupés dans une partie du tableau commun. C'est pourquoi le tableau commun à tous les sous-programmes est divisé en deux parties. La première partie contient les renseignements auxiliaires ; la seconde partie, qui est aussi la plus importante, est utilisée pour la définition des figures géométriques.

La partie du tableau commun utilisée pour la définition des figures géométriques est divisée en "cellules". Chaque cellule est formée de cinq mots et contient en général les données numériques relatives à une figure géométrique élémentaire. Chaque cellule comporte un mot de contrôle et quatre mots de données numériques. Le mot de contrôle de la cellule est divisé en deux parties ; la partie haute comprend un nombre correspondant au type de la figure géométrique occupant la cellule (point, cercle, droite, vecteur, etc...) ; la partie basse est un pointeur qui sera utilisé lorsqu'on voudra former une liste de cellules. Dans le cas de figures isolées le pointeur indique la cellule elle-même ; une cellule peut donc être considérée comme une liste ne comprenant qu'un seul élément. La cellule est repérée par la position de son mot de contrôle dans le tableau commun. La figure 35 donne un exemple de cellule définissant un cercle, le mot de contrôle indique le type de la figure (cercle = 42) et indique que c'est une figure isolée car le pointeur indique la cellule elle-même (le mot de contrôle est le $51^{ème}$ élément du tableau commun) ; les quatre mots suivants contiennent les indications nécessaires à la définition du cercle.

Certaines figures géométriques élémentaires peuvent être associées pour former de nouvelles figures. On pourra ainsi associer plusieurs points et former un réseau de points, ou encore grouper plusieurs réseaux pour former un nouveau réseau. Dans ce cas la partie pointeur du mot de contrôle d'une cellule donnée est utilisée pour indiquer à quelle autre cellule cette cellule est reliée. Ainsi chaque cellule ne peut être reliée qu'à une seule cellule, mais celle-ci peut également être reliée à une autre cellule, et ainsi de suite. La structure de liste utilisée est donc très simple ; elle n'est utilisée que pour les réseaux de points qui ne nécessi-

		Type	Pointeur
Mot de contrôle	T(51)	42	51
Mots de données	T(52)	X centre	
	T(53)	Y centre	
	T(54)	Z centre	
	T(55)	Rayon	

Figure 35. Exemple de cellule géométrique

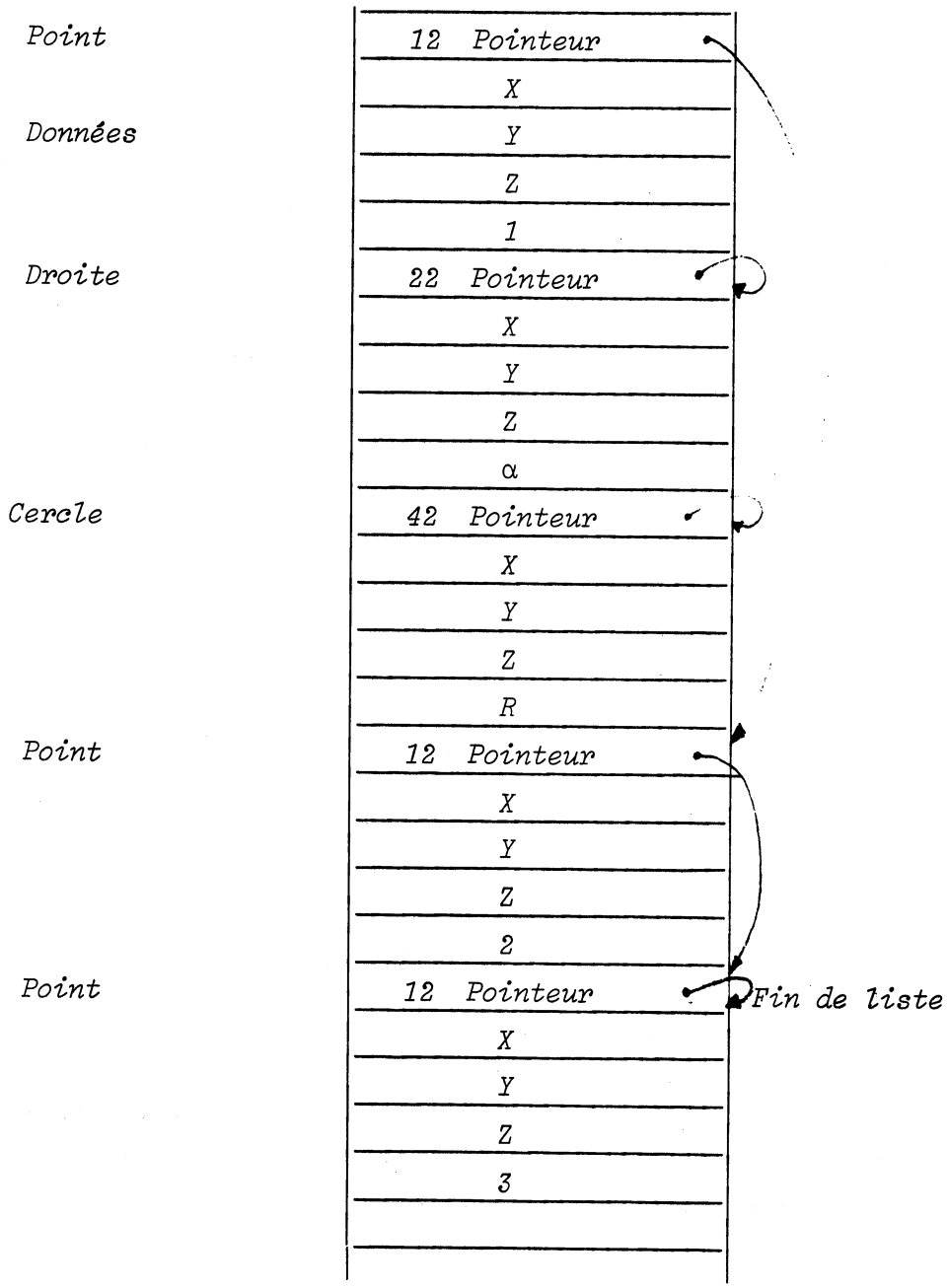


Figure 36 . Tableau des figures géométriques.

tent pas une structure plus complexe. Cette organisation du tableau commun contribue à la simplicité du système que nous avons recherchée. La figure 36 donne en exemple une partie du tableau commun comprenant un réseau de trois points (type 12) et deux autres figures géométriques indépendantes (droite et cercle de types respectifs 22 et 42).

La bibliothèque de sous-programmes comprend des sous-programmes de gestion, des fonctions de définitions géométriques et des sous-programmes d'usinage.

Les sous-programmes de gestion comprennent des sous-programmes généraux (pour l'initialisation du tableau commun, l'impression de ce tableau commun, l'impression des messages d'erreur, etc...), des sous-programmes simples de traitement de listes (pour la recherche de l'élément suivant un élément donné, la recherche du dernier élément d'une liste, le rattachement d'un nouvel élément à la liste, etc...) et des sous-programmes de gestion du tableau des figures géométriques (pour la création d'une cellule, etc...).

Les définitions géométriques sont faites au moyen de fonctions qui calculent les différentes coordonnées d'une figure dans un système de coordonnées absolues et qui rangent les résultats dans les cellules du tableau commun des figures géométriques.

Les sous-programmes d'usinage sont liés au système de commande numérique utilisé. Ils utilisent les données numériques du tableau des figures géométriques pour produire les ordres numériques destinés à un système de commande numérique donné, dans la mesure où l'on ne produit pas l'ensemble CLDATA.

4.3.1. Exemple.

Afin de montrer le fonctionnement de la deuxième phase du système et des sous-programmes de la bibliothèque, nous prendrons le programme intermédiaire de la figure 37 issu du programme de pièce de la figure 34. L'exécution de ce programme intermédiaire à l'aide de la bibliothèque de

```
      COMMON F(1CCC)
      CALL  DEELT
C  EXEMPLE DE RESEAU
      P1= POINC ( 130.0 , 100.0 )
      P2= POINC ( 270.0 , 100.0 )
      C1= CIRCO ( 200.0 , 170.0 , 100.0 )
      PAT1= PATEC ( P1, P2, 3 )
      PAT2= PATE2 ( 1 , C1, C.0 , 180.0 , 5 )
      PAT3= PATE5 ( 5 , PAT1  , PAT2  )
      CALL  TOOLO ( 10, 215.C )
      CALL  CYCLO ( 3 , 1 , 15.0, 88.0, 1.0 )
      CALL  CCTC1 ( PAT3)
      CALL  TOOLO ( 15, 235.0 )
      CALL  CYCLO ( 3 , 1 , 15.0, 135.0 , 1.0 )
      CALL  CCTC1 ( PAT3)
      CALL  FIN1
      END
```

Figure 37. Programme intermédiaire exemple

```
N  1     X13000     Y 10000   W15   T10
N  2     X20000
N  3     X27000
N  4     X30000     Y 17000
N  5     X27071N    Y 24071
N  6     X20000N    Y 17000
N  7     X12929N    Y 24071N
N  8     X10000N    Y 17000N
N  9     X13000     Y 10000N  W15   T15
N 10     X20000
N 11     X27000
N 12     X30000     Y 17000
N 13     X27071N    Y 24071
N 14     X20000N    Y 17000
N 15     X12929N    Y 24071N
N 16     X10000N    Y 17000N
N999     X00000N    Y00000N
```

Figure 38 . Ordres numériques exemple

sous-programmes produit les résultats de la figure 38 ; ce sont des ordres numériques destinés à une perceuse-aléseuse GSP modèle P 10. La lettre N indique le numéro du bloc sur la bande perforée. Les coordonnées sont repérées par les lettres X et Y et suiviesde la lettre N si elles sont inférieures aux coordonnées du bloc précédent. La lettre W indique un numéro de came de profondeur pour le perçage des trous et la lettre T indique le numéro des outils à utiliser et provoque l'arrêt de la machine pour les changements d'outils.

4.4. CONSTRUCTEUR.

Le constructeur est un programme FORTRAN basique de petite taille qui est utilisé pour construire et modifier les tables syntaxiques du préprocesseur. Le constructeur utilise une table de mots majeurs, une table de mots mineurs et des codes qui leur sont associés, et une description de la syntaxe type-APT du langage d'écriture des programmes de pièce. La description de la syntaxe est faite d'une manière simple et accessible au programmeur, le constructeur la met sous la forme codée qui sera utilisée par le préprocesseur. La table 2 donne la liste des mots majeurs utilisés par la version point à point du préprocesseur.

AUXFUN	CALL	CIRCLE	CLPRNT	COOLNT	COPY	CUTTER
CYCLE	DELAY	ELLIPS	END	FEDRAT	FINI	FROM
GCONIC	GODLTA	GOTO	IF	INDEX	JUMPTO	LINE
LOOPND	LOOPST	MACHIN	MACRO	MATRIX	OPSTOP	ORIGIN
PATERN	POINT	PREFUN	RAPID	REFSYS	REWIND	SPINDL
STOP	TERMAC	TOOLNO	TRACUT	VECTOR	ZSURF	

Table 2

La table 3 donne la liste des mots mineurs utilisés par la version point à point du préprocesseur.

ARC	AT	ATANGL	AVOID	BORE	CCLW	CENTER
CLW	DEEP	DRILL	FACE	FLOOD	INCR	INTOF
INVERS	LEFT	LENGTH	LINEAR	MILL	MIST	MODIFY
MPM	MPT	NOMORE	OFF	OMIT	ON	PARLEL
PERPTO	RADIUS	RANDOM	RETAIN	RIGHT	RTHETA	SAME
TANTO	TAP	TAPKUL	TRANSL	XLARGE	XSMALL	XYPLAN
XYROT	YLARGE	YSMALL	ZIGZAG			

Table 3

Une règle syntaxique du langage sera décrite simplement en spécifiant le mot majeur la caractérisant suivi d'une série de mots mineurs ou de symboles géométriques ou arithmétiques. La position des mots CIRCLE, ELLIPS, GCONIC, LINE MATRIX, PATERN, PLANE, POINT et VECTOR dans une règle indique sans ambiguïté s'ils doivent être considérés comme mots majeurs ou symboles géométriques, seul le premier mot d'une règle pouvant être un mot majeur. Le mot SCALAR ne fait pas partie des mots majeurs et représente soit un nombre, soit une variable ayant une valeur numérique.

Afin de mieux montrer la façon dont est décrite la syntaxe prenons pour exemple les différentes définitions d'un cercle, d'un point, d'une droite, d'un conique générale, d'une ellipse et d'un vecteur pour la version point à point du préprocesseur. La figure 39 donne ainsi les descriptions syntaxiques correspondant à trois définitions de cercle, huit définitions de point, cinq définitions de droite, une définition de conique générale, une définition d'ellipse et quatre définitions de vecteur. Les mots majeurs CIRCLE, POINT, LINE, GCONIC, ELLIPS et VECTOR sont spécifiés au début de la première règle relative aux définitions de cercles, de points, de droites, de coniques, d'ellipses et de vecteurs ; ils n'ont pas besoin d'être répétés pour les règles suivantes, le signe dollar placé en tête d'une règle indiquant que la règle suivante a trait au même mot majeur.

Les règles syntaxiques étant définies comme le montre la figure 39, le constructeur lit chaque règle, cherche dans les différentes tables et met la règle sous une forme numérique codée. Il la place alors sous cette forme dans la table syntaxique (voir figure 32).

Le rôle principal du constructeur est la création et la modification des tables utilisées par le préprocesseur. Sa raison d'être est l'expérimentation avec le système : essais du système, extensions du système pour les travaux d'usinage en mode continu, introduction des données technologiques, etc...

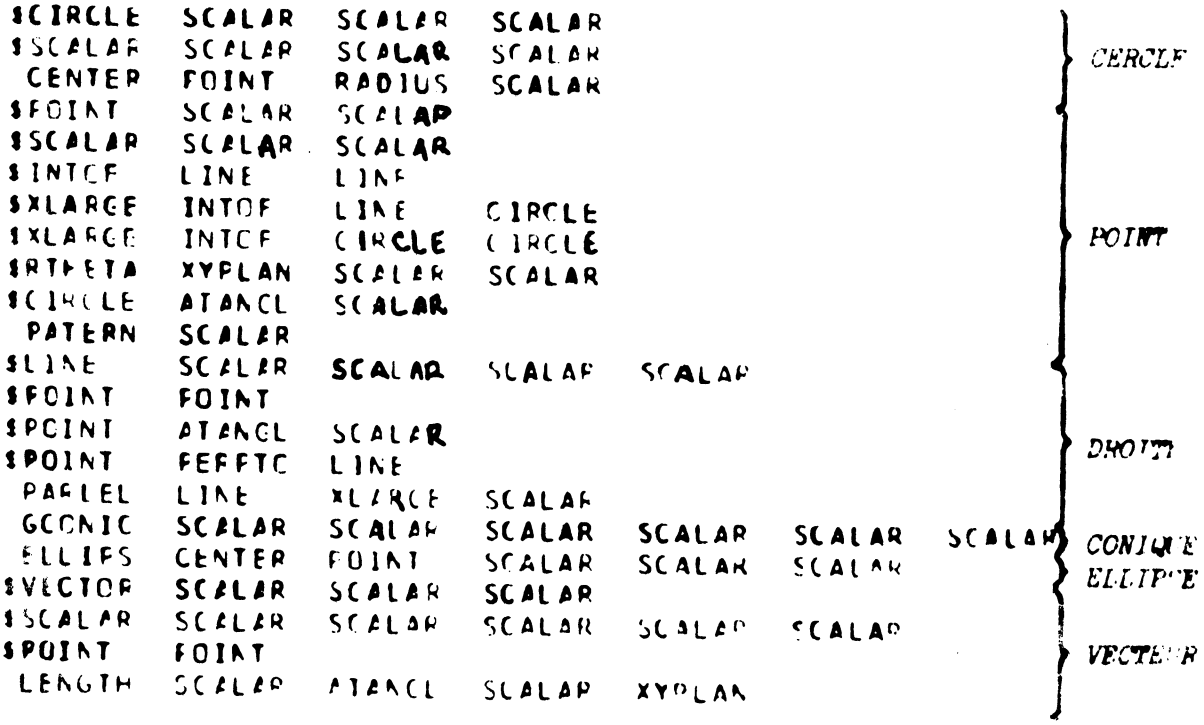


Figure 39 - Exemple de description syntaxique

CHAPITRE 5

ADAPTATION ET EXTENSION DU SYSTEME

5.1. ADAPTATION DU SYSTEME A L'ORDINATEUR IBM 1130.

A partir du système général fonctionnant sous CP67/CMS que nous venons de voir, un système plus petit fut mis au point et adapté à un ordinateur IBM 1130 (mémoire centrale : 8K mots de 16 bits, mémoire secondaire : un disque) sans aucun problème, l'utilisation du FORTRAN basique permettant de n'avoir pas à modifier les programmes.

La liaison entre les deux phases du système dépend de l'ordinateur utilisé. Actuellement pour le système fonctionnant sur l'ordinateur IBM 1130 la phase 1 produit le programme intermédiaire sous forme de cartes perforées ; celles-ci sont alors transmises à la phase 2. Cette liaison devra être automatisée pour des systèmes utilisés de façon intensive. En ce qui concerne l'ordinateur IBM 1130 (27) il faudrait écrire quelques sous-programmes de contrôle et modifier plusieurs parties du compilateur FORTRAN pour aboutir à une liaison automatique.

5.1.1. Préprocesseur.

La partie du système qui est la plus importante au point de vue taille est le préprocesseur ; avec le moniteur et les divers sous-programmes dont ils ont besoin, il nécessite un minimum de 7K mots sur l'ordinateur IBM 1130 (2,5K mots pour le préprocesseur et sa zone d'overlays, 2K mots pour les diverses tables, 0,5K mots pour le moniteur et 2K mots pour les sous-programmes d'entrée-sortie du système). Il existe deux versions du préprocesseur auxquelles correspondent deux types de systèmes. La première version permet un travail de point à point complet et sans contraintes. La seconde version comporte des extensions permettant un travail expérimental en mode continu.

Le préprocesseur a été entièrement écrit en FORTRAN basique. Ceci nous a permis de l'adapter facilement à l'ordinateur IBM 1130. Cette adaptation a été faite principalement à l'aide de recouvrements ou "overlays", afin de ne charger les différents sous-programmes en mémoire centrale qu'au moment de leur appel. Bien que les versions actuelles du

préprocesseur donnent entière satisfaction, il faudra normalement réécrire en langage d'assemblage tous les sous-programmes d'entrée-sortie afin de gagner en efficacité et en espace mémoire.

5.1.2. Bibliothèque de sous-programmes.

La version actuelle de la bibliothèque pour l'ordinateur IBM 1130 permet le travail de point à point et produit des ordres numériques directement utilisables par une machine donnée.

La bibliothèque comporte une quinzaine de sous-programmes de gestion, une trentaine de fonctions de définitions géométriques et une quarantaine de sous-programmes d'usinage. Tous les sous-programmes sont petits ce qui rend l'utilisation de la bibliothèque très simple et n'a pas nécessité l'emploi de la technique des recouvrements ("overlays").

Le système est constitué de plusieurs parties indépendantes que l'on peut utiliser séparément. Sur les ordinateurs très petits (par exemple IBM 1130 avec une mémoire centrale de 4K mots) il peut être intéressant d'utiliser la bibliothèque de sous-programmes seule, car elle ne nécessite qu'une taille de mémoire centrale bien inférieure à celle requise par le préprocesseur, et elle donne au programmeur de pièce les mêmes possibilités que le système complet. Dans ce cas on écrira directement le programme intermédiaire FORTRAN ; ceci demande une bonne connaissance de la bibliothèque de sous-programmes, mais ne présente pas de difficultés particulières, le programme intermédiaire correspondant pratiquement instruction pour instruction au programme de pièce APT.

5.1.3. Limitations.

Les versions du préprocesseur adaptées à l'ordinateur IBM 1130 ne permettent ni les définitions imbriquées, ni l'utilisation de synonymes (instructions APT SYN), ni les variables dimensionnées (instruction APT RESERV). Ces limites ne sont pas de réels désavantages pour un système de commande numérique travaillant de point à point ; elles peuvent devenir un inconvénient pour les travaux en mode continu. Comme le système permet d'utiliser les instructions FORTRAN, il est possible d'utiliser l'ins-

truction DIMENSION du langage FORTRAN à la place de l'instruction RESERV du langage APT, mais cela implique que le programmeur de pièce connaisse FORTRAN, ce qui est rarement le cas. Comme nous l'avons vu les définitions imbriquées ne posent pas réellement de problèmes ; mais dans un système qui utilise la technique des recouvrements, le traitement des imbrications, qui met en jeu de nouveaux sous-programmes, allongera le temps de traitement du programme de pièce. Nous n'avons pas trouvé nécessaire de traiter les imbrications dans une version point à point ou dans une version pour le travail en mode continu fonctionnant sur petits ordinateurs. La version du préprocesseur pour le travail de point à point sur l'ordinateur IBM 1130 occupe 2500 mots tandis que les diverses tables occupent 2000 mots. La version minimale pour le travail en mode continu est un peu plus importante mais la différence est peu sensible.

L'emploi du langage FORTRAN pour le programme intermédiaire a amené deux restrictions dans l'utilisation du système.

a . Les données numériques ou les variables utilisées dans les définitions géométriques devront correspondre au type FORTRAN "réel" ; les coordonnées devront donc comporter un point décimal alors que le langage APT accepte indifféremment les entiers et les réels. On écrira donc :

P1 = POINT / 10.0, 3.0

au lieu de P1 = POINT / 10,3

b . Les noms symboliques donnés aux figures géométriques doivent être de type réel ; ces noms ne devront donc pas commencer par les lettres I, J, K, L, M ou N. Ces restrictions sont dues au fait que les types des paramètres utilisés dans l'appel des fonctions et des sous-programmes FORTRAN de la bibliothèque doivent correspondre aux types des paramètres utilisés dans la définition de ces fonctions et de ces sous-programmes. La première restriction n'est pas réellement gênante car la plupart

des cotes d'une pièce possèdent une partie décimale. La seconde restriction sera moins facile à assimiler par un programmeur de pièce ne connaissant pas FORTRAN mais est quand même fort simple.

L'utilisation de petits ordinateurs limite aussi la taille des diverses tables du système. Le système actuel permet l'utilisation de cinquante étiquettes, la définition de dix MACROS et la définition de cent noms symboliques différents ; le fait qu'APT ne permette pas la réutilisation des identificateurs géométriques peut poser un problème pour les programmes de pièce importants ; ce problème pourra être facilement résolu dans notre système, la réutilisation de ces identificateurs y étant possible sans difficultés.

5.2. ADAPTATIONS DU SYSTEME A DIVERS TYPES DE COMMANDE NUMERIQUE.

Dans le chapitre précédent nous avons donné une description générale du système de programmation pour la commande numérique des machines-outils. Pour pouvoir être utilisé le système doit cependant être adapté aux différents systèmes de commande numérique que l'on désire employer. La version du préprocesseur utilisée normalement sur l'ordinateur IBM 1130 est destinée au travail de point à point. Le préprocesseur est indépendant du système de commande numérique utilisé et ne sera donc pas modifié lorsqu'on passera d'un système de commande numérique à un autre. Par contre une partie de la bibliothèque de sous-programmes dépend du système de commande numérique utilisé : ce sont les sous-programmes d'usinage aux différents systèmes de commande numérique utilisés.

Les adaptations des sous-programmes d'usinage présentées ci-dessous ont été réalisées à titre d'exemple et ont été testées à l'aide d'exemples réels ; elles devront encore être testées de façon plus complète. Les résultats sont imprimés et perforés sur cartes et non sur bande perforée en l'absence d'un perforateur de bande connecté à l'ordinateur IBM 1130 utilisé.

5.2.1. Adaptation à une machine GSP

La machine GSP utilisée est une perceuse aléseuse modèle P10 conçue pour le travail de point à point. C'est une machine-outil à commande numérique très simple destinée au perçage et à l'alésage. Cette machine de dispose que d'une broche porte-outil et l'outil est changé par l'opérateur chaque fois que cela est nécessaire. Cette machine ne possède pas de cycle fixes de perçage ; elle n'a ni fonctions préparatoires ni fonctions auxiliaires. La profondeur du trou à percer ou la descente de l'outil sont in -

diquées par une série de cames de profondeur groupées sur un barillet.

La programmation de la bande perforée se fait selon un format variable (8) analogue à celui de la figure 17. Chaque bloc d'information doit comporter un numéro de séquence de trois chiffres précédé de l'adresse "N" ; le bloc peut ensuite comporter les coordonnées selon les axes X et Y de la machine respectivement précédées des adresses "X" et "Y". Ces coordonnées peuvent être omises, auquel cas la machine utilise l'ancienne valeur des coordonnées omises. Lorsque les coordonnées du bloc courant sont inférieures aux coordonnées du bloc précédent, elles doivent être suivies de la lettre "N" ; on cherche ainsi à optimiser les mouvements de la machine, cette dernière ne se mettant pas directement en position mais allant jusqu'en fin de course avant de revenir en arrière si la lettre N est omise. Après les coordonnées en X et en Y, le bloc peut contenir l'adresse "W" suivi d'un nombre de deux chiffres indiquant la piste du barillet ou la came de profondeur choisies. Le bloc pourra enfin contenir l'adresse "T" suivie d'un nombre de deux chiffres indiquant un changement d'outil et le numéro de l'outil à monter. Un bloc d'information complet aura donc l'aspect suivant :

N056 X39188N Y100068 W20 T12

Les déplacements selon l'axe X sont indiqués par 5 chiffres tandis que les déplacements selon l'axe Y sont indiqués par 6 chiffres ; les dimensions sont données en centièmes de millimètres. Le fonctionnement de la machine et les ordres numériques à lui fournir sont donc très simples. Cela explique pourquoi ce type de machine est encore très utilisé en programmation manuelle. Les sous-programmes d'usinage du système de programmation ont été écrits et mis au point pour produire les ordres de commande numérique sous la forme qui vient d'être décrite.

Afin de bien illustrer l'adaptation du système à cette machine, nous allons prendre un exemple d'usinage réel qui nous a été fourni par les établissements RICHIER de Pont de Claix. Il s'agit du perçage d'un carter assez complexe représenté schématiquement sur la figure 40 qui reproduit également le résultat des calculs des coordonnées des différents

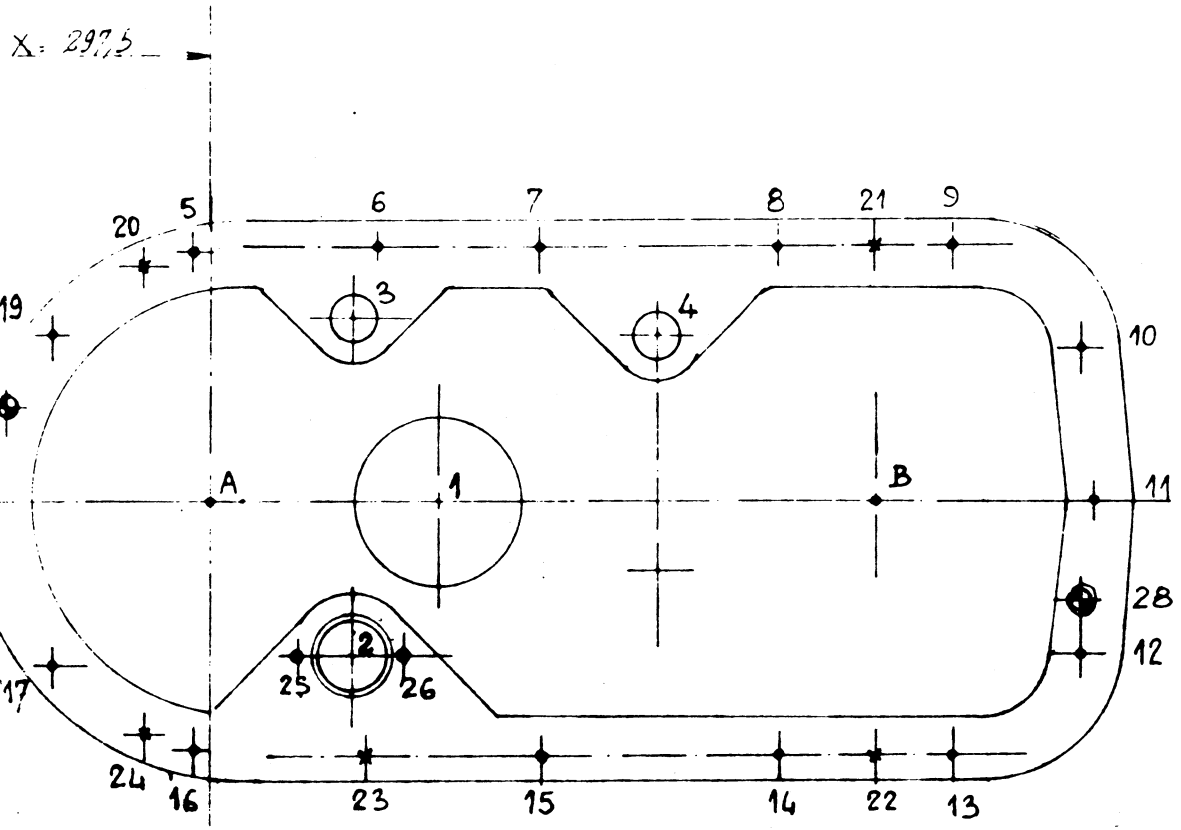
Ste RICHIER
Division Nordest

Service Methodes Pont de Claix

PHASE - 1

P. 4450323 Rep 2

ETABLI PAR: BG DATE: 19-2-69



A	297,5	900	
B	739,98	900	
1	449,52	900	ligne ⌀ 110 H7
2	391,89	1000,70	ligne ⌀ 52 H7 ⌀ 45 H7
3	394,10	780	ligne ⌀ 29 H7
4	594,72	791	ligne ⌀ 29 H7
5	287,50	734,20	•
6	410,50	731,50	
7	515,50	731,50	
8	675,50	731,50	
9	792,50	731,50	
10	877,50	800	⌀ 15
11	886	900	
12	877,50	1000	
13	792,50	1068,50	
14	675,50	1068,50	
15	515,50	1068,50	
16	287,50	1065,80	
17	192,50	1013	
18	149	900	
19	192,50	787	
20	252,50	744,6	
21	740,50	731,50	
22	740,50	1068,50	M. 14 x 2
23	402,50	1068,50	
24	252,50	1055,40	
25	356,89	1000,70	M 8 x 1,25
26	426,89	1000,70	
27	160,50	840	2 ⌀ 14 H7
28	877,50	965	
29			

FIGURE 40 - PERÇAGE D'UN CARTER.

Sté RICHIER — Service Méthodes
DIVISION NORDEST — Pont-de-Claix

GAMME

POUR P10 Phase 1

PIECE : CARTER
N° 4450323 REPERE : 02 IND :
OPE : 03 VISA : Gaillard Feuille 1/2

	N° D'OUTIL	GENRE USIN.	TOURS MN	AVANCE TOUR	PHASE	TYPE D'USINAGE
						Positionnement, et bridage, en A et B de l'outillage 7017411937 réglage fin au pupitre montage de la pièce, centrée en A et B avec cimblots coniques broches dans canons Ø30, Vérification avec broche Ø20 par rapport aux faces intérieures, Verrouillage de la partie en fonte à faire - bridage.
2	T-01	BP	900	0,125		Pointage N° 27-28-26-2-25-3-4.
4	T-02	BP	450	0,18		Perçage Ø12 N° 21-22-23-24-20-27-28
6	T-03	BP	900	0,125		Perçage Ø6,75 N° 26-25
8	T-04	BP	320	0,18		Perçage Ø15 N° 16-17-18-19-5-6-7-8-9-10-11-12-13-14-15
	T-04	CM	320	main		Fraisurage : N° 26-25
	T-04	BP	320	0,25		Fraisurage : N° 24-20-21-22-23
2	T-05	BT	160	0,25		taraudage M16 N° 23-24-20-21-22
	T-06	CM	224	main		taraudage M8 N° 26-25
0	T-07	CM	112	main		Lamer en tirant N° 17-14.
	T-08	BP	224	0,25		Perçage Ø28 N° 3-4
6	T-09	BP	112	0,50		Alésage 29^{H7} N° 4-3
8	T-10	BP	450	0,125		Rectrissage N° 28-27
0	T-11	BP	224	0,50		Alésage $\emptyset 14^{J7}$ N° 27-28

FIGURE 41 GAMME D'USINAGE

Stᵉ RICHIER — Service Methodes
DIVISION NORDEST — Pont-de-Claix

GAMME phase 1
POUR P.10

PIECE : CARTER
N° 4450323 REPERE : 02 IND :
OPE : 03 VISA : Gaillard Feuille 2/2

E	N° D'OUTIL	GENRE USIN.	TOURS MN	AVANCE TOUR	N° PHASE	TYPE D'USINAGE	
00	T.12	CM	112	0,25		Perçage	Ø50 N° 2. pointage bossage intérieur avec foret Ø50
0	T.13	CM	112	0,25		Perçage	Ø43 N° 2 bossage intérieur
0	T.14	CM	160	0,25		Redressage	Ø51,5 N° 2 (Semelle) faire une fasse à 0
0	T.15	CM	112	0,25		Redressage	Ø44,5 N° 2 (Bossage intérieur)
00	T.16	CM	56	0,50		Alésage	$Ø45^{H7}$ N° 2 (Bossage intérieur)
00	T.17	CM	56	0,50		Alésage	$Ø52^{H7}$ N° 2 (Semelle)
00	T.18	CM	40	main		Lamage	Ø70 en fournant du bossage intérieur fous cote $74,5^{+0,2}_{+0}$ N° 2
00	T.19	CM	40	main		Lamage	$Ø54^{+1}_{-0}$ en tirant du bossage intérieur (prof. lamage 1 mm) N° 2
0	T.20	CM	40	main		Lamage	Ø90 en tirant de la semelle fous cote $61,5 \pm 0,2$ N° 2
0	T.99	CM.			999	ARRET	Démontage

FIGURE 41. GAMME D'USINAGE (FIN).

Sté RICHIER
Division Nordest

Service Méthodes Pont de Claix

P.10 phase 1

P. 4450323 rep 02

ETABLI PAR Gaillard DATE 23-12-69

01	Foret à centrer
02	Foret Ø 12 (affutage centre-pointe)
03	Foret Ø 6,75
04	Foret Ø 15 (affutage centre-pointe)
05	Taraud M-14
06	Taraud M-8
07	Porte fraise Ø15 pour lamer en tirant 1 fraise Ø30
08	Foret Ø 28
09	Foret alésoir Ø 29 H7
10	Fraise 2 tailles Ø 13,5 queue cône morse N° 2
11	Alésoir Ø 14 J7
12	Foret Ø 50 court
13	Foret Ø 43 court
14	P.O 701740824 + cale épaisseur 4,4 + cartouche 438-9-1320
15	Foret à fond plat Ø44,5 court
16	Foret alésoir Ø 45 H7
17	Alésoir DAVID BROWN pour Ø 52 H7 avec lèvres carburées
18	P.O. 701741323
19	P.O 701741324.
20	P.O 701741323

Outillage 701741137

1 Tampon Ø 52 H7
1 Tampon Ø 45 H7
1 Tampon Ø 14 J7
1 Tampon Ø 29 H7
1 - taraudeuse

- 9 - V

FIGURE 42. OUTILLAGE.

trous à percer faits en vue d'une programmation manuelle. La figure 41 montre la gamme d'usinage établie par le bureau des méthodes ; on y voit les différentes opérations à effectuer ainsi que l'ordre dans lequel les effectuer, les outils et les pistes du barillet à employer. La figure 42 décrit plus en détail l'outillage utilisé. La figure 43 détaille le programme de pièce établi à partir du dessin de la pièce et de la gamme d'usinage. Ce programme est assez long mais demeure simple. La quatrième ligne du programme effectue un changement d'axes qui amène l'origine au point A de la figure 40. On définit alors des points, des cercles et des droites permettant de définir les points où le perçage devra être fait.

Remarquons ici que nous avons du définir chaque point séparément à cause de la manière dont le dessin de la pièce était coté. Ainsi on aurait pu concevoir que les points P5, P19, P18, P17, P16 soient définis directement comme un réseau de points, étant donné qu'ils se trouvent sur le même arc de cercle. La méthode de cotation du dessin de la pièce ne l'a pas permis. Les établissements RICHIER utilisent la programmation manuelle pour leurs machines GSP et le bureau d'études cote parfois les pièces à partir d'une origine pour faciliter le travail du bureau des méthodes. Notre exemple s'en trouve vraisemblablement un peu faussé mais nous avons voulu conserver un exemple d'usinage réel ; le programme de pièce est long et aurait pu être réduit de moitié en utilisant une cotation normale et en revoyant la gamme d'usinage.

Les points ayant été définis sont regroupés en différents réseaux afin de simplifier les ordres d'usinage. La seconde partie du programme de pièce est formée d'ordres d'usinage, les outils sont définis grâce à l'instruction TOOLNO et on utilise l'instruction CYCLE pour indiquer la came de profondeur utilisée bien que la machine GSP ne possède pas de cycles à proprement parler.

Le préprocesseur traite ce programme de pièce et produit le programme intermédiaire de la figure 44. On exécute alors ce programme intermédiaire à l'aide de la bibliothèque de sous-programmes et on obtient les ordres numériques de la figure 45 destinés à la perceuse-aléseuse. On aboutit ainsi au même résultat qu'en programmation manuelle dans un temps beaucoup

```
PARTNO CARTER FICHIER
REMARK TRANSLATION DU SYSTEME D'AXES
T=MATRIX/TRANSL,297.5,500.0
REFSYS/T
X=168.5
O1=POINT/20.0,0.0
C1=CIRCLE/CENTER,O1,RADIUS,X
P27=POINT/-137.0,-60.0
P28=POINT/580.0,65.0
C2=CIRCLE/0.0,0.0,138.0
C3=CIRCLE/152.3,0.3,116.0
P2=POINT/YLARGE,INTOF,C2,C3
C4=CIRCLE/0.0,0.0,154.0
P=POINT/0.0,-120.0
D1=LINE/P,ATANGL,0.0
P3=POINT/XLARGE,INTOF,D1,C4
C5=CIRCLE/152.3,0.0,152.0
PP=POINT/0.0,45.0
D2=LINE/PP,ATANGL,0.0
O2=POINT/XLARGE,INTOF,D2,C5
D3=LINE/O2,ATANGL,90.0
C6=CIRCLE/CENTER,O2,RADIUS,154.0
P4=POINT/YSMALL,INTOF,D3,C6
C7=CIRCLE/CENTER,P2,RADIUS,35.0
P25=POINT/59.79,100.70
P26=POINT/129.39,100.70
P21=POINT/443.0,-X
P22=POINT/443.0,X
P23=POINT/105.0,X
O5=POINT/-10.0,0.0
D5=LINE/O5,ATANGL,90.0
P16=POINT/YLARGE,INTOF,D5,C1
P5=POINT/YSMALL,INTOF,D5,C1
O6=POINT/-105.0,0.0
D6=LINE/O6,ATANGL,90.0
P17=POINT/YLARGE,INTOF,D6,C1
P19=POINT/YSMALL,INTOF,D6,C1
P18=POINT/-148.5,0.0
P6=POINT/113.0,-X
P7=POINT/218.0,-X
P8=POINT/378.0,-X
P9=POINT/495.0,-X
P10=POINT/580.0,-100.0
P11=POINT/588.5,0.0
P12=POINT/580.0,100.0
P13=POINT/495.0,X
P14=POINT/378.0,X
P15=POINT/218.0,X
O3=POINT/-45.0,0.0
D4=LINE/O3,ATANGL,90.0
P24=POINT/YLARGE,INTOF,D4,C1
P20=POINT/YSMALL,INTOF,D4,C1
PAT1=PATERN/RANDOM,P27,P28
PAT2=PATERN/RANDOM,P3,P4
PAT3=PATERN/RANDOM,P21,P22
```

Figure 43 . *Programme de pièce (début)*

```
PAT4=PATERN/RANDOM,P24,P20
PAT5=PATERN/RANDOM,P26,P25
PAT6=PATERN/RANDOM,P16,P17,P18,P19,P5
PAT7=PATERN/RANDOM,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
REMARK POINTAGE S=500 T/MN, F=0.125
TOOLNO/1,200.0
CYCLE/DRILL,2.0,0.125,MFM,0.0
GOTO/PAT1
GOTO/P26
GOTO/P2
GOTO/P25
GOTO/PAT2
REMARK PERCAGE S=450 T/MN, F=0.180
TOOLNO/2,200.0
CYCLE/DRILL,4.0,0.180,MFM,0.0
GOTO/PAT3
GOTO/P23
GOTO/PAT4
GOTO/PAT1
REMARK PERCAGE S=500 T/MN, D=0.75, F=0.125
TOOLNO/3,200.0
CYCLE/DRILL,6.0,0.125,MFM,0.0
GOTO/PAT5
REMARK PERCAGE D=15, S=320 T/MN, F=0.180
TOOLNO/4,200.0
CYCLE/DRILL,8.0,0.180,MFM,0.0
GOTO/PAT6
GOTO/PAT7
REMARK FRAISURAGE S=320 T/MN, F=MANUELLE
CYCLE/DRILL,0.0,0.0,MFM,0.0
GOTO/PAT5
REMARK FRAISURAGE S=320 T/MN, F=0.25
CYCLE/DRILL,10.0,0.25,MFM,0.0
GOTO/PAT4
GOTO/PAT3
GOTO/P23
REMARK TARAUDAGE M14 S=160 T/MN, F=0.25
TOOLNO/5,200.0
CYCLE/TAP,12.0,0.25,MFM,0.0
GOTO/P23
GOTO/PAT4
GOTO/PAT3
REMARK TARAUDAGE M8 S=224 T/MN,F=MANUELLE
TOOLNO/6,200.0
CYCLE/TAP,0.0,0.0,MFM,0.0
GOTO/PAT5
REMARK LAMAGE EN TIRANT S=112 T/MN, F=MANUELLE
TOOLNO/7,200.0
CYCLE/DRILL,0.0,0.0,MFM,0.0
GOTO/P17
GOTO/P19
REMARK PERCAGE 28 S=224 T/MN, F=0.25
TOOLNO/8,200.0
CYCLE/DRILL,14.0,0.25,MFM,0.0
GOTO/PAT2
```

Figure 43 . Programme de pièce (suite)

```
REMARK ALESAGE 2EP7 S=112 T/MN, F=0.50
TOOLNO/9,200.0
CYCLE/BORE,16.0,0.50,MPM,0.0
GOTO/PAT2,INVERS
REMARK REGRESSAGE S=450 T/MN, F=0.125
TOOLNO/10,200.0
CYCLE/DRILL,18.0,0.125,MPM,0.0
GOTO/PAT1,INVERS
REMARK ALESAGE 14J7 S=224 T/MN, F=0.50
TOOLNO/11,200.0
CYCLE/BORE,20.0,0.50,MPM,0.0
GOTO/PAT1
REMARK ALESAGE DERNIER POINT
TOOLNO/12,200.0
CYCLE/BORE,0.0,0.50,MPM,0.0
GOTO/P2
FINI
```

Figure 43. Programme de pièce (Fin)

```
      COMMON F(1000)
      CALL  DEBLT
C  CARTER RICHIER
C  TRANSLATION DU SYSTEME D'AXES
      T = MATRO ( 297.5 , 500.0 )
      CALL  REFSO ( T )
      X = 166.5
      01= POINO ( 20.0, 0.0 )
      C1= CIRC2 ( 01, X )
      P27 = POINO ( - 137.0 , - 60.0)
      P28 = POINO ( 580.0 , 65.0)
      C2= CIRCO ( 0.0 , 0.0 , 138.0 )
      C3= CIRCO ( 152.0 , 0.0 , 116.0 )
      P2= POIN4 ( 3 , C2, C3)
      C4= CIRCO ( 0.0 , 0.0 , 154.0 )
      P = POINO ( 0.0 , - 120.0 )
      D1= LINE2 ( P , 0.0 )
      P3= POIN3 ( 1 , D1, C4)
      C5= CIRCO ( 152.0 , 0.0 , 152.0 )
      PP= POINO ( 0.0 , 45.0)
      D2= LINE2 ( PP, 0.0 )
      02= POIN3 ( 1 , D2, C5)
      D3= LINE2 ( 02, 90.0)
      C6= CIRC2 ( 02, 154.0 )
      P4= POIN3 ( 4 , D3, C6)
      C7= CIRC2 ( P2, 35.0)
      P25 = POINO ( 59.79 , 100.70)
      P26 = POINO ( 129.39, 100.70)
      P21 = POINO ( 443.0 , - X )
      P22 = POINO ( 443.0 , X )
      P23 = POINO ( 105.0 , X )
      05= POINO ( - 10.0, 0.0 )
      D5= LINE2 ( 05, 90.0)
      P16 = POIN3 ( 3 , D5, C1)
      P5= POIN3 ( 4 , D5, C1)
      06= POINO ( - 105.0 , 0.0 )
      D6= LINE2 ( 06, 90.0)
      P17 = POIN3 ( 3 , D6, C1)
      P19 = POIN3 ( 4 , D6, C1)
      P18 = POINO ( - 148.5 , 0.0 )
      P6= POINO ( 113.0 , - X )
      P7= POINO ( 218.0 , - X )
      P8= POINO ( 378.0 , - X )
      P9= POINO ( 495.0 , - X )
      P10 = POINO ( 580.0 , - 100.0 )
      P11 = POINO ( 566.5 , 0.0 )
      P12 = POINO ( 580.0 , 100.0 )
      P13 = POINO ( 495.0 , X )
      P14 = POINO ( 378.0 , X )
      P15 = POINO ( 218.0 , X )
      03= POINO ( - 45.0, 0.0 )
      D4= LINE2 ( 03, 90.0)
      P24 = POIN3 ( 3 , D4, C1)
      P20 = POIN3 ( 4 , D4, C1)
```

Figure 44 . Programme intermédiaire (Début)

```
      PAT1= PATE5 ( 9 , P27   , P28   )
      PAT2= PATE5 ( 9 , P3    , P4    )
      PAT3= PATE5 ( 9 , P21   , P22   )
      PAT4= PATE5 ( 9 , P24   , P20   )
      PAT5= PATE5 ( 9 , P26   , P25   )
      PAT6= PATE5 ( 9 , P16   , P17   )
      PAT6= PATE5 ( 0 , P18   , P19   )
      PAT6= PATE5 ( 1 , P5    , 0.0   )
      PAT7= PATE5 ( 9 , P6    , P7    )
      PAT7= PATE5 ( 0 , P8    , P9    )
      PAT7= PATE5 ( 0 , P10   , P11   )
      PAT7= PATE5 ( 0 , P12   , P13   )
      PAT7= PATE5 ( 1 , P14   , P15   )
C  POINTAGE S=900 T/MN, F=0.125
      CALL  TOOL0 ( 1 , 200.0 )
      CALL  CYCL0 ( 3 , 1 , 2.0 , 0.125 , 0.0 )
      CALL  GOTO1 ( PAT1)
      CALL  GOTO2 ( P26 )
      CALL  GOTO2 ( P2)
      CALL  GOTO2 ( P25 )
      CALL  GOTO1 ( PAT2)
C  PERCAGE S=450 T/MN, F=0.180
      CALL  TOOL0 ( 2 , 200.0 )
      CALL  CYCL0 ( 3 , 1 , 4.0 , 0.180 , 0.0 )
      CALL  GOTO1 ( PAT3)
      CALL  GOTO2 ( P23 )
      CALL  GOTO1 ( PAT4)
      CALL  GOTO1 ( PAT1)
C  PERCAGE S=900 T/MN, D=0.75, F=0.125
      CALL  TOOL0 ( 3 , 200.0 )
      CALL  CYCL0 ( 3 , 1 , 6.0 , 0.125 , 0.0 )
      CALL  GOTO1 ( PAT5)
C  PERCAGE D=15, S=320 T/MN, F=0.180
      CALL  TOOL0 ( 4 , 200.0 )
      CALL  CYCL0 ( 3 , 1 , 8.0 , 0.180 , 0.0 )
      CALL  GOTO1 ( PAT6)
      CALL  GOTO1 ( PAT7)
C  FRAISURAGE S=320 T/MN, F=MANUELLE
      CALL  CYCL0 ( 3 , 1 , 0.0 , 0.0 , 0.0 )
      CALL  GOTO1 ( PAT5)
C  FRAISURAGE S=320 T/MN, F=0.25
      CALL  CYCL0 ( 3 , 1 , 10.0, 0.25, 0.0 )
      CALL  GOTO1 ( PAT4)
      CALL  GOTO1 ( PAT3)
      CALL  GOTO2 ( P23 )
C  TARAUDAGE M14 S=160 T/MN, F=0.25
      CALL  TOOL0 ( 5 , 200.0 )
      CALL  CYCL0 ( 5 , 1 , 12.0, 0.25, 0.0 )
      CALL  GOTO2 ( P23 )
      CALL  GOTO1 ( PAT4)
      CALL  GOTO1 ( PAT3)
C  TARAUDAGE M8 S=224 T/MN,F=MANUELLE
      CALL  TOOL0 ( 6 , 200.0 )
      CALL  CYCL0 ( 5 , 1 , 0.0 , 0.0 , 0.0 )
```

Figure 44 . Programme intermédiaire (suite)

```
      CALL  GOTO1 ( PAT5)
C  LAMAGE EN TIRANT S=112 T/MN, F=MANUELLE
      CALL  TOOLO ( 7 , 200.0 )
      CALL  CYCLO ( 3 , 1 , 0.0 , 0.0 , 0.0 )
      CALL  GOTO2 ( P17 )
      CALL  GOTO2 ( P15 )
C  PERCAGE 28 S=224 T/MN, F=0.25
      CALL  TOOLO ( 8 , 200.0 )
      CALL  CYCLO ( 3 , 1 , 14.0, 0.25, 0.0 )
      CALL  GOTO1 ( PAT2)
C  ALESAGE 25H7 S=112 T/MN, F=0.50
      CALL  TOOLO ( 9 , 200.0 )
      CALL  CYCLO ( 1 , 1 , 16.0, 0.50, 0.0 )
      CALL  GOTO3 ( PAT2)
C  RECRESSAGE S=450 T/MN, F=0.125
      CALL  TOOLO ( 10, 200.0 )
      CALL  CYCLO ( 3 , 1 , 18.0, 0.125 , 0.0 )
      CALL  GOTO3 ( PAT1)
C  ALESAGE 14J7 S=224 T/MN, F=0.50
      CALL  TOOLO ( 11, 200.0 )
      CALL  CYCLO ( 1 , 1 , 20.0, 0.50, 0.0 )
      CALL  GOTO1 ( PAT1)
C  ALESAGE  DERNIER POINT
      CALL  TOOLO ( 12, 200.0 )
      CALL  CYCLO ( 1 , 1 , 0.0 , 0.50, 0.0 )
      CALL  GOTO2 ( P2)
      CALL  FINI
      END
```

Figure 44 . Programme intermédiaire (Fin)

```
N   1       X1EC50      Y E4000    W 2    T 1
N   2       XE775C      Y 9E500
N   3       X42689N     Y1CC070
N   4       X3518EN
N   5       X35729N
N   6       X39402      Y 78000N
N   7       X5946S      Y 79100
N   8       X74C5C      Y 73150N   W 4    T 2
N   9                   Y1C6E50
N  10       X40250N
N  11       X25259N     Y1C5546N
N  12                   Y 74454N
N  13       X1EC50N     Y E4000
N  14       XE775C      Y 9E50C
N  15       X4268SN     Y1C007C    W E    T 3
N  16       X35729N
N  17       X2E750N     Y1C6581    W 8    T 4
N  18       X19250N     Y1C129SN
N  19       X14900N     Y 900C0N
N  20       X1925C      Y 7E701N
N  21       X2E750      Y 73419N
N  22       X41C50      Y 73150N
N  23       X51550
N  24       XE7550
N  25       X79250
N  26       XE7750      Y 800C0
N  27       XEEEC0      Y 90000
N  28       XE775CN     Y100CC0
N  29       X79250N     Y1C6E5C
N  30       XE7550N
N  31       X51550N
N  32       X426E5N     Y1CC070N   W C    T 4
N  33       X35729N
N  34       X25250N     Y105546    W10    T 4
N  35                   Y 74454N
N  36       X74C5C      Y 73150N
N  37                   Y1C6E5C
N  38       X4C250N
N  39       X40250      Y1C6E5C    W12    T 5
N  40       X25250N     Y1C5546N
N  41                   Y 74454N
N  42       X74C50      Y 7315CN
N  43                   Y1C6E5C
N  44       X426E5N     Y1CC07CN   W 0    T 6
N  45       X35729N
N  46       X19250N     Y1C129S    W 0    T 7
N  47                   Y 7E701N
N  48       X39402      Y 78000N   W14    T 8
N  49       X5946S      Y 79100
N  50       X55469      Y 79100    W1E    T 9
N  51       X39402N     Y 78000N
N  52       XE7750      Y 9E50C    W1E    T10
N  53       X1605CN     Y E40CCN
N  54       X1EC50      Y E4000    W20    T11
N  55       XE7750      Y 9E500
N  56       X35188N     Y1CC068    W 0    T12
N999        XCCCCCN     YCCC000N
```

Figure 45. Ordres numériques

plus court. Toutefois la complexité de la pièce et la méthode de cotation du dessin employée ne donnent pas un exemple de programmation automatique excellent ; les exemples factices sont faciles à établir mais nous avons voulu donner un exemple réel.

5.2.2. Adaptation à une machine Olivetti-Auctor CNZ[1].

Cette machine à commande numérique est destinée au travail de point à point ; elle est plus complexe que la machine que nous venons de voir et comporte des cycles d'usinage ainsi que des fonctions préparatoires et auxiliaires. La programmation de la bande perforée se fait également selon un format variable (8) ; chaque bloc d'information peut comporter jusqu'à 8 informations différentes, chacune précédée d'une adresse.

Un bloc complet devra ainsi comporter un numéro de séquence de trois chiffres précédés de l'adresse "N" ; ce numéro pourra être suivi d'une fonction préparatoire de deux chiffres précédés de l'adresse "G" (voir figure 16), puis de trois coordonnées respectivement précédées des adresses "X", "Y", et "Z". Le bloc pourra également comporter une vitesse d'avance de cinq chiffres précédés de l'adresse "F", puis une vitesse de rotation de la broche de deux chiffres précédés de l'adresse "S" et enfin une fonction auxiliaire de deux chiffres précédés de l'adresse "M" (voir figure 16). Un bloc d'information complet aura donc l'aspect suivant :

N056G80X00185000Y00100000Z-0130000F00600S08M03

Les sous-programmes d'usinage de la bibliothèque du système de programmation ont été modifiés ou réécrits pour produire les ordres de commande numérique dans le format ci-dessus.

Nous prendrons pour exemple le perçage d'une plaque avec utilisation des cycles de perçage (adaptation et exemple établis d'après les Recueils de Conférences des Stages de Commande Numérique organisés par l'INSA de Lyon). La figure 46 représente la disposition des trous à percer sur la plaque. La figure 47 donne le programme de pièce correspondant au perçage désiré. Les trous doivent être centrés puis percés ; certains d'entre eux doivent être taraudés. On définit d'abord les deux ensembles de trois trous en

1. Adaptation réalisée en partie par M. NGUYEN CONG KAHN au cours d'un stage de deux mois.

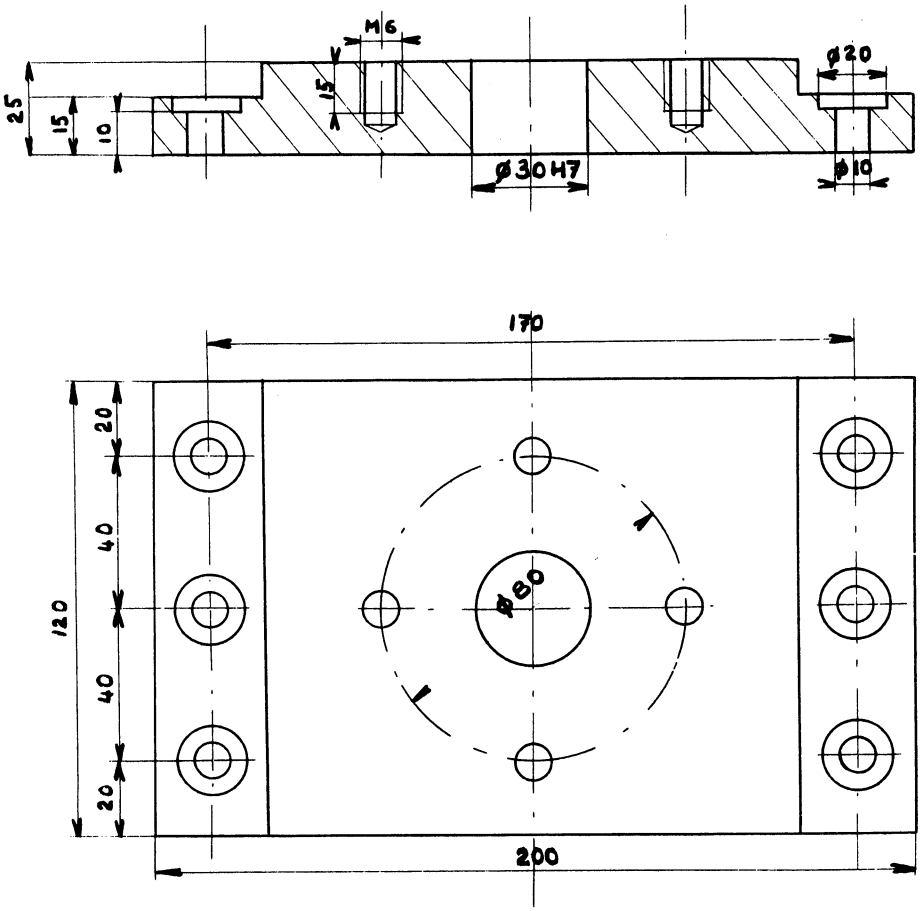


Figure 46 . *Perçage d'une plaque.*

```
PARTNO PERCAGE D'UNE PLAQUE
REMARK DEFINITIONS GEOMETRIQUES
O=POINT/100.0,60.0
P1=POINT/15.0,20.0
P10=POINT/185.0,20.0
V1=VECTOR/0.0,40.0,0.0
PAT1=PATERN/LINEAR,P1,V1,3
PAT2=PATERN/LINEAR,P10,V1,3
C=CIRCLE/CENTER,O,RADIUS,40.0
PAT3=PATERN/ARC,C,90.0,CCLW,INCR,3,AT,50.0
REMARK ORDRES DE POINTAGE
TOOLNO/1,215.0
SPINDL/850,CLW
CYCLE/DRILL,-106.0,EE.0,MPM,-55.0
GOTO/PAT1
CYCLE/DRILL,-56.0,EE.0,MPM,-85.0
GOTO/PAT3
CYCLE/DRILL,-106.0,EB.0,MPM,-55.0
GOTO/PAT2,INVERS
REMARK ORDRES DE PERCAGE
TOOLNO/2,235.0
SPINDL/600,CLW
CYCLE/DRILL,-100.0,135.0,MPM,-100.0
GOTO/PAT2
GOTO/PAT1,INVERS
REMARK ORDRES DE PERCAGE D=8.5
TOOLNO/3,200.0
CYCLE/DRILL,-135.0,135.0,MPM,-100.0
GOTO/PAT3
GOTO/0
REMARK ORDRES DE TARAUDAGE
TOOLNO/5,165.0
SPINDL/270,CLW
CYCLE/TAP,-153.0,350.0,MPM,-135.0
GOTO/PAT3
REMARK ORDRES DE PERCAGE D=25.5
TOOLNO/4,260.0
SPINDL/120,CLW
CYCLE/DRILL,-80.0,48.0,MPM,-40.0
GOTO/0
REMARK ORDRES D'ALESAGE
TOOLNO/6,292.0
SPINDL/120,CLW
CYCLE/BORE,-45.0,60.0,MPM,-8.0
GOTO/0
REMARK ORDRES DE FRAISURAGE
TOOLNO/7,180.0
SPINDL/550,CLW
CYCLE/DRILL,-140.0,50.0,MPM,-130.0
GOTO/PAT2,INVERS
GOTO/PAT1
FINI
```

Figure 47 . Programme de pièce

```
      COMMON F(1000)
      CALL  DEBUT
C  PERCAGE D'UNE PLAQUE
C  DEFINITIONS GEOMETRIQUES
      O = POINO ( 100.0 , 60.0)
      P1= POINO ( 15.0, 20.0)
      P10 = POINO ( 185.0 , 20.0)
      V1= VECTO ( 0.0 , 40.0, 0.0 )
      PAT1= PATE1 ( P1, V1, 3 )
      PAT2= PATE1 ( P10 , V1, 3 )
      C = CIRC2 ( O , 40.0)
      PAT3= PATE6 ( 9 , 1 , C , 90.0, 90.0  , 90.0  )
      PAT3= PATE6 ( 1 , 1 , C , 90.0, 90.0  , 0 . 0 )
C  ORDRES DE POINTAGE
      CALL  TOOLO ( 1 , 215.0 )
      CALL  SPIN1 ( 2 , 850 )
      CALL  CYCLO ( 3 , 1 , - 106.0 , 88.0, - 95.0)
      CALL  GOTO1 ( PAT1)
      CALL  CYCLO ( 3 , 1 , - 96.0, 88.0, - 85.0)
      CALL  GOTO1 ( PAT3)
      CALL  CYCLO ( 3 , 1 , - 106.0 , 88.0, - 95.0)
      CALL  GOTO3 ( PAT2)
C  ORDRES DE PERCAGE
      CALL  TOOLO ( 2 , 235.0 )
      CALL  SPIN1 ( 2 , 600 )
      CALL  CYCLO ( 3 , 1 , - 100.0 , 135.0 , - 75.0)
      CALL  GOTO1 ( PAT2)
      CALL  GOTO3 ( PAT1)
C  ORDRES DE PERCAGE D=8.5
      CALL  TOOLO ( 3 , 200.0 )
      CALL  CYCLO ( 3 , 1 , - 135.0 , 135.0 , - 100.0 )
      CALL  GOTO1 ( PAT3)
      CALL  GOTO2 ( 0 )
C  ORDRES DE TARAUDAGE
      CALL  TOOLO ( 5 , 165.0 )
      CALL  SPIN1 ( 2 , 270 )
      CALL  CYCLO ( 5 , 1 , - 153.0 , 350.0 , - 135.0 )
      CALL  GOTO1 ( PAT3)
C  ORDRES DE PERCAGE D=29.5
      CALL  TOOLO ( 4 , 260.0 )
      CALL  SPIN1 ( 2 , 120 )
      CALL  CYCLO ( 3 , 1 , - 60.0, 48.0, - 40.0)
      CALL  GOTO2 ( 0 )
C  ORDRES D'ALESAGE
      CALL  TOOLO ( 6 , 252.0 )
      CALL  SPIN1 ( 2 , 120 )
      CALL  CYCLO ( 1 , 1 , - 45.0, 60.0, - 8.0 )
      CALL  GOTO2 ( 0 )
C  ORDRES DE FRAISURAGE
      CALL  TOOLO ( 7 , 180.0 )
      CALL  SPIN1 ( 2 , 550 )
      CALL  CYCLO ( 3 , 1 , - 140.0 , 50.0, - 130.0 )
      CALL  GOTO3 ( PAT2)
      CALL  GOTO1 ( PAT1)
      CALL  FINI
      END
```

```
N  1                                                    M06   OUTIL NO  1 L=215
N  2G80                    RCCCCCCCOO
N  3    X   15CCCY    2CCCOR  -95CCC
N  4G81                    Z -106000F  880S10M03
N  5              Y   60CCO
N  6              Y  1CCCCO
N  7G80                    RCOOOCOOO
N  8    X  1CCCCC          R  -85CCC
N  9G81                    Z  -96000
N 10    X   6CCCCY   6CCCO
N 11    X  1CCCCCY   2CCCO
N 12    X  14CCCCY   60CCO
N 13G80                    RCCCCCCCC
N 14    X  185CCCY  10C000R  -95000
N 15G81                    Z -1C6CCO
N 16              Y   60000
N 17              Y   2CCCO
N 18                                                   M06   OUTIL NO  2 L=235
N 19G80                    RCCCCCOCO
N 20                       R  -75CCO
N 21G81                    Z -1CC000F 1350S 9M03
N 22              Y   60CCO
N 23              Y  1CCCOO
N 24    X   15CCC
N 25              Y   60CCO
N 26              Y   2CCOO
N 27                                                   M06   OUTIL NO  3 L=200
N 28G80                    RCOCCOCOO
N 29    X  1CCCCCY  1CCCCOR -1CCCC
N 30G81                    Z -135000
N 31    X   6CCCCY   60CCO
N 32    X  1CCCCCY   20000
N 33    X  14CCCOY   6CCCO
N 34    X  1CCCCC
N 35                                                   M06   OUTIL NO  5 L=165
N 36G80                    RCCCCCCOO
N 37              Y  1CCCCOR -1350CC
N 38G84                    Z -153000F 3500S 6M03
N 39    X   6CCCCY   6CCCO
N 40    X  1CCOOOY   2CCCO
N 41    X  14CCCOY   6CCOO
N 42                                                   M06   OUTIL NO  4 L=260
N 43G80                    RCOOOOOOO
N 44    X  1CCCCC          R  -40CCO
N 45G81                    Z  -8CCCCF  480S 3M03
N 46                                                   M06   OUTIL NO .6 L=292
N 47G80                    RCCOOCCCO
N 48                       R   -8CCO
N 49G81                    Z  -45000F  600S 3M03
N 50                                                   M06   OUTIL NO  7 L=180
N 51G80                    RCCCCCCCC
N 52    X  185CCCY  1C0000R -130CCO
N 53G81                    Z -14CCCOF  500S 8M03
N 54              Y   6CCCO
N 55              Y   2CCCO
N 56    X   15CCC
N 57              Y   60CCO
N 58              Y  1CCCOO
N 59                       RCCCCCCCC           M05
N 60G80X0CCCCCCCYCCCCCCCO                      M30
```

ligne puis les quatre trous situés sur le cercle. On donne ensuite les ordres de pointage puis les différents ordres de perçage et de taraudage. On utilise l'instruction CYCLE pour définir les cycles de perçage (profondeur du trou, distance d'avance rapide et vitesse d'avance). Le préprocesseur traite ce programme de pièce et produit le programme intermédiaire de la figure 48. Ce dernier est exécuté à l'aide de la bibliothèque de sous-programmes relative à la machine Olivetti-Auctor et produit les ordres numériques de la figure 49. Les coordonnées ne sont pas répétées lorsqu'elles ne changent pas d'un bloc à l'autre.

5.2.3. Adaptation au format de sortie CLDATA.

Le système de programmation pour la commande numérique des machines-outils que nous avons mis au point produit des ordres numériques directement utilisables par les machines. Le postprocesseur ou programme d'adaptation a donc été supprimé et remplacé par les sous-programmes de la bibliothèque. Notre but est en effet l'utilisation du système sur petits ordinateurs ; il arrive que les postprocesseurs soient des programmes importants ne pouvant pas tous être utilisés sur des ordinateurs de petite taille. L'utilisation d'un postprocesseur ajoutera une troisième phase au système mais afin de rester compatible avec les systèmes type-APT existants, nous mettons au point une version de la bibliothèque permettant le codage des informations dans un format acceptable par les postprocesseurs, identique au format CLDATA du système IFAPT ; ce format diffère du format CLDATA des systèmes APT fonctionnant sur ordinateurs IBM par la longueur des trois premiers mots de l'enregistrement (pour IFAPT mots doubles, pour APT mots simples). L'ensemble CLDATA, aussi appelé CLTAPE, contient toutes les informations codées destinées au postprocesseur. Ces informations sont transmises dans un format qui varie suivant le type d'enregistrement (3). D'une façon générale un enregistrement a le format suivant :

Numéro de séquence	Type de l'enregistrement	Mot majeur codé ou identificateur	Mots mineurs codés, variables ou nombres ou informations alphanumériques
Mot 1	Mot 2	Mot 3	Mot 4 et mots suivants

L'organisation internationale de normalisation (ISO) a déjà préparé un certain nombre de recommandations visant à normaliser le format de sortie CLDATA (33). Les différents types d'enregistrement sont les suivants :

a . Enregistrement type 2000 concernant tous les mots postprocesseur. Le mot 3 de l'enregistrement contient le mot postprocesseur codé. Le mot 4 et les mots suivants contiennent la section complémentaire codée de l'instruction destinée au postprocesseur.

b . Enregistrement type 3000 concernant les définitions de surfaces. Il comprend la position de l'outil par rapport à la surface (TO, PAST, ON ou TANTO), le type de la surface, le nom symbolique de la surface et les paramètres de définition de la surface sous forme canonique (par exemple X, Y, Z et R pour un cercle).

c . Enregistrement type 5000 concernant les définitions de mouvements. Il comprend le mot majeur codé (FROM, GODLTAA ou GOTO), le nom symbolique du point, du vecteur ou de la surface et les coordonnées à atteindre (X, Y et Z). Cet enregistrement est utilisé pour transmettre les points intermédiaires de la trajectoire calculés par le processeur.

d . Enregistrement type 14000 correspondant au mot FINI et indiquant la fin du programme.

Il existe aussi des enregistrements suites.

Les différences principales de cette adaptation du système par rapport aux adaptations directes à des systèmes de commande numérique particuliers viennent du traitement des "mots postprocesseur". En effet parmi les instructions APT, il existe un certain nombre d'instructions destinées au postprocesseur et qui lui sont transmises telles quelles. La table 4 donne la liste des mots postprocesseur relatifs à la version point à point du système.

AUXFUN	COOLNT	COPY	CYCLE	DELAY	END	FEDRAT
INDEX	INSERT	MACHIN	OPSTOP	ORIGIN	PARTNO	PPRINT
PREFUN	RAPID	REWIND	SPINDL	STOP	TOOLNO	TRACUT

Table 4

Les sous-programmes correspondant aux mots postprocesseur sont donc considérablement simplifiés dans cette version puisqu'il suffit de transcoder l'instruction sans faire de véritable traitement. Quant aux autres sous-programmes d'usinage il possèdent la même forme que dans les autres versions, le seul changement se trouvant dans le format de sortie des informations.

5.3. EXTENSION DU SYSTEME POUR LES TRAVAUX DE CONTOURNAGE.

Nous avons présenté dans les chapitres précédents, la conception et la réalisation d'un système de programmation pour la commande numérique des machines-outils pouvant fonctionner sur petits ordinateurs. La réalisation décrite jusqu'ici a montré qu'un tel système était réalisable pour les travaux d'usinage de point à point (perçage, alésage et taraudage). Il nous a paru intéressant d'étudier dans quelle mesure il était possible d'étendre le système existant aux travaux de contournage dans le cadre de l'utilisation de petits ordinateurs (mémoire centrale de 16K octets). L'utilisation de petits ordinateurs limite obligatoirement la taille du système et il ne saurait être question de transposer toutes les possibilités offertes par les systèmes APT pour les travaux de contournage, la technique de programmation employée n'étant pas comparable à celles des programmes plus généraux comme IFAPT. Le système peut être étendu sans difficultés pour permettre des travaux d'usinage en mode paraxial, mais il est également possible d'aller un peu plus loin et d'envisager quelques travaux de contournage. Nous nous en tiendrons aux travaux de contournage par courbes de niveau ou usinage en deux axes et demi, et nous nous limiterons à quelques instructions permettant cependant de réaliser la plupart des usinages courants.

5.3.1. Ordres de mouvement.

Parmi les instructions à ajouter à la version du système pour les travaux en mode point à point, une première série d'instructions APT (CUTTER, INTOL, OUTTOL et TOLER) permettra de définir les dimensions et les formes des outils et les tolérances utilisées. L'usinage des surfaces selon deux axes et demi se fait par une série d'usinages plans. L'instruction ZSURF permettra de définir à cet effet un plan parallèle à l'un des plans de référence de la machine et de définir ainsi les différents niveaux où s'effectuera l'usinage. La partie géométrique du système point à point offre déjà des définitions de courbes planes que nous pourrons utiliser : droites, cercles et coniques ; il n'est cependant pas possible dans cette version point à point de définir des courbes planes quelconques. Il pourra s'avérer nécessaire d'ajouter à ces instructions de définitions géométriques, l'instruction TABCYL (pour "Tabulated Cylinder"), qui, en APT, permet de définir des surfaces quelconques à partir d'un ensemble de points de ces surfaces.

Pour les ordres de mouvement proprement dits, nous avons retenu certaines formes des instructions de mouvement **APT : GO, GOBACK, GOFWD, GOLFT, GORGT.** La définition générale des instructions GO retenues peut s'écrire :

$$\mathrm{GO/}\left[\left\{\begin{matrix}\mathrm{TO}\\ \mathrm{ON}\\ \mathrm{PAST}\end{matrix}\right\},\right]\mathrm{SURF1}\left[,\left[\left\{\begin{matrix}\mathrm{TO}\\ \mathrm{ON}\\ \mathrm{PAST}\end{matrix}\right\},\right]\mathrm{SURF2}\left[,\left[\left\{\begin{matrix}\mathrm{TO}\\ \mathrm{ON}\\ \mathrm{PAST}\end{matrix}\right\},\right]\mathrm{SURF3}\right]\right]$$

Les crochets indiquent les éléments de l'instruction qui peuvent être omis ; les accolades indiquent un choix à faire parmi les éléments qu'elles entourent. De la même manière la définition générale des instructions GOBACK, GOFWD, GOLFT et GORGT retenues peut s'écrire :

$$\left[\left\{\begin{matrix}\mathrm{TLON}\\ \mathrm{TLLFT}\\ \mathrm{TLRGT}\end{matrix}\right\},\right]\left\{\begin{matrix}\mathrm{GOBACK}\\ \mathrm{GOFWD}\\ \mathrm{GOLFT}\\ \mathrm{GORGT}\end{matrix}\right\}\mathrm{/\ SURF1}\left[,\left[\left\{\begin{matrix}\mathrm{TO}\\ \mathrm{ON}\\ \mathrm{PAST}\end{matrix}\right\},\right]\mathrm{SURF2}\right]$$

Dans l'instruction GO une surface est obligatoire, c'est la surface de guidage SURF1 ; les autres surfaces sont facultatives, SURF2 est la surface d'arrêt et SURF3 la surface de pièce (voir figure 21). Dans les autres instructions de mouvement une seule surface est obligatoire, la surface de guidage SURF1, la surface d'arrêt SURF2 étant facultative. Dans le cas ou seule la surface de guidage est mentionnée dans une instruction, il est de règle dans les systèmes APT, de prendre comme surface d'arrêt la surface de guidage de l'instruction suivante. Ainsi les deux instructions :

GO / S1
GOLFT / S2

permettront d'interpréter la première instruction comme étant équivalente à l'instruction :

GO / S1, S2

En l'absence des modificateurs TO, ON et PAST, qui sont facultatifs, le système choisit par défaut le modificateur TO. Par conséquent l'instruction ci-dessus est équivalente à l'instruction :

GO / TO, S1, TO, S2

De par sa structure notre système traite les instructions une par une et ne permet par conséquent pas l'examen de l'instruction suivante pour la détermination de la surface d'arrêt de l'instruction en cours. Pour cette raison nous n'accepterons que les instructions de mouvement spécifiant à la fois la surface de guidage et la surface d'arrêt. La troisième forme de l'instruction GO spécifie la surface de pièce ce qui n'est vraiment valable que pour les usinages selon trois dimensions ; nous ne l'utiliserons pas pour l'usinage par courbes de niveau. Nous ne conserverons donc que les formes suivantes des instructions de mouvement :

$$GO/\left[\left\{\begin{matrix}TO\\ON\\PAST\end{matrix}\right\},\right] SURF1, \left[\left\{\begin{matrix}TO\\ON\\PAST\end{matrix}\right\},\right] SURF2$$

$$\left[\left\{\begin{matrix}\text{TLON}\\\text{TLLFT}\\\text{TLRGT}\end{matrix}\right\},\right]\left\{\begin{matrix}\text{GOBACK}\\\text{GOFWD}\\\text{GOLFT}\\\text{GORGT}\end{matrix}\right\} \text{ / SURF1,} \left[\left\{\begin{matrix}\text{TO}\\\text{ON}\\\text{PAST}\end{matrix}\right\}, \right] \text{ SURF2}$$

Pour pouvoir traiter ces ordres le préprocesseur n'a besoin que de trouver les règles syntaxiques correspondantes dans la table syntaxique. Cependant lorsqu'une position d'outil est indiquée (TLON, TLLFT ou TLRGT) la forme de l'instruction devient particulière. On la transforme alors pour qu'elle prenne la forme générale des instructions APT, en réduisant la section majeure au seul mot majeur. Ainsi l'instruction :

TLLFT,GOLFT/S1,PAST,S2

sera considérée par le préprocesseur comme étant l'instruction APT fictive :

GOLFT/TLLFT,S1,PAST,S2

et sera ensuite traitée comme les autres instructions et, comme elles, traduite en appel de sous-programme.

5.3.2. Problèmes géométriques.

L'instruction TABCYL permet de définir des courbes planes quelconques au moyen d'un ensemble de points de ces courbes. Cette définition pose donc un problème de lissage de courbe pour la résolution duquel nous avons retenu une méthode relativement simple. Nous illustrerons cette méthode par l'exemple de la figure 50. Prenons d'abord les trois premiers points P_1, P_2 et P_3 de la courbe C que nous voulons définir. Déterminons ensuite le cercle C' passant par ces trois points. Par le premier des trois points, P_1, définissons un système d'axes auxiliaire SP_1T, l'axe P_1T passant par le second des points, P_2, et faisant un angle α avec l'axe OX. La courbe C sera alors approchée par une cubique d'équation :

$$t = as^3 + bs^2 + cs + d$$

dans le nouveau système d'axes. Comme la courbe C passe par la nouvelle origine des coordonnées, P_1, le coefficient d est nul. Le coefficient c est la pente de la tangente à l'origine que l'on prend égale à la pente de la tangente au cercle C' au point P_1. On détermine ensuite les coefficients a et b par la pente de la tangente au cercle C' au point P_2. On a donc

déterminé pour le point P_1 : X_1, Y_1, Z_1, a, b, c et α. On considère alors le cercle passant par les trois points suivants P_2, P_3 et P_4 et on détermine de la même manière X_2,Y_2,Z_2,a',b',c' et α'. Pour chaque point de la courbe C on détermine ainsi X, Y, Z, a, b, c et α. A partir de ces données on peut appliquer plusieurs méthodes permettant d'obtenir une courbe finale répondant à un critère préalablement fixé : continuité de la pente de la tangente, pentes de tangentes connues en certains points, continuité de la courbure. Dans ce dernier cas les coefficients a,b et c sont corrigés par une méthode itérative jusqu'à ce que la variation du rayon de courbure de part et d'autre des points de base de la courbe soit inférieure à une valeur donnée ; la courbe recherchée est une courbe spline. Ces différentes méthodes mettent en jeu des sous-programmes de calculs géométriques de plusieurs centaines d'instructions FORTRAN, c'est pourquoi nous n'avons finalement pas retenu l'instruction TABCYL. L'expérience montrera si cette instruction est réellement nécessaire à un système permettant un travail de contournage réduit.

L'approximation des courbes par des segments de droites en fonction de la tolérance permise, constitue une seconde catégorie de problèmes géométriques. L'exemple le plus souvent rencontré est celui du cercle. Comme le montre la figure 51a les tolérances internes et externes (définies au moyen des instructions INTOL et OUTTOL) permettent de définir deux cercles concentriques au cercle définissant la trajectoire. Le point de départ réel P est défini par le mouvement précédent. On prend alors la tangente au cercle concentrique intérieur au niveau du point de départ P ; on obtient alors le point S intersection de cette tangente et du cercle concentrique extérieur. On obtient ainsi la valeur de l'angle POS qui permet de définir le découpage à effectuer sur le cercle. En utilisant une valeur approchée de α, on obtient le polygone approchant le cercle de la trajectoire théorique. C'est ce découpage qui est fait par le processeur et les points intermédiaires ainsi déterminés sont placés dans l'ensemble CLDATA.

Les problèmes géométriques posés par l'usinage en mode continu sont nombreux ; les intersections de courbes posent les problèmes les plus complexes, mais les problèmes les plus délicats sont ceux relatifs à la détermination

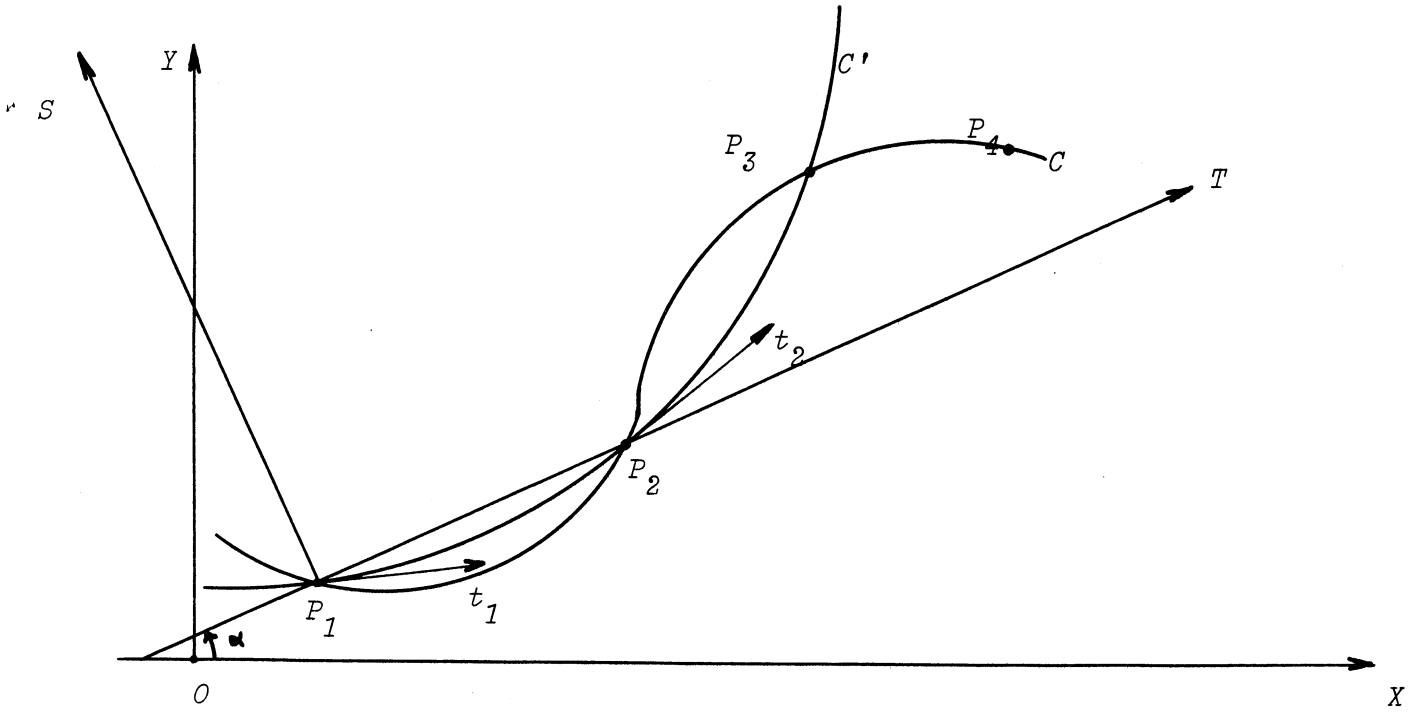


Figure 50 . *Exemple de définition TABCYL.*

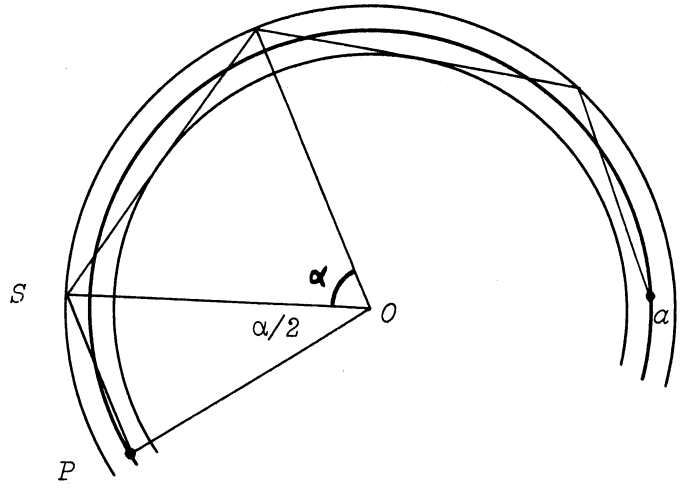


Figure 51a. Approximation d'un cercle par un polygone

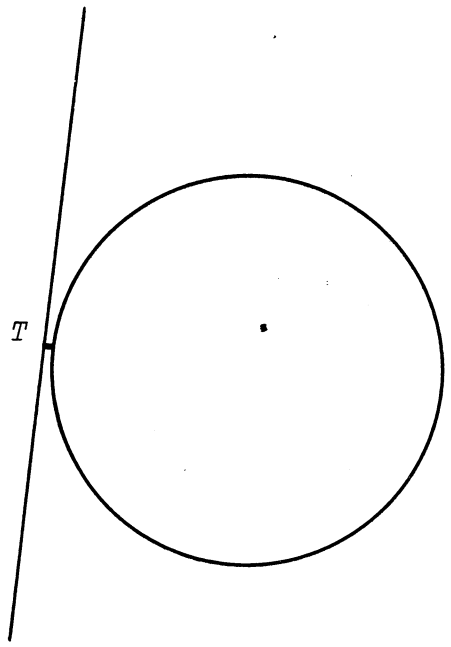


Figure 51b. Point de tangence mal défini.

des points de tangence. Une droite peut avoir été définie comme étant tangente à un cercle ; il se peut alors qu'à la suite d'erreurs dues à la précision des calculs effectués, la trajectoire de l'outil comporte une discontinuité au point de tangence théorique de la droite et du cercle comme le montre la figure 51b. Ceci est surtout vrai dans le cas des droites dont la pente est importante ce qui affecte beaucoup la précision des calculs. C'est pourquoi nous avons été amenés à conserver l'indication de la tangence de deux courbes afin de pouvoir éviter de tels problèmes.

5.3.3. Problèmes posés par la réalisation.

Les ordres de mouvement relatifs au contournage sont traités par des sous-programmes spécifiques faisant appel à un ensemble de sous-programmes de la bibliothèque. Cet ensemble comprend :

a . des sous-programmes déterminant la position d'un point par rapport à une courbe (à droite, à gauche, au dessus, en dessous, à l'extérieur, à l'intérieur ou sur la courbe).

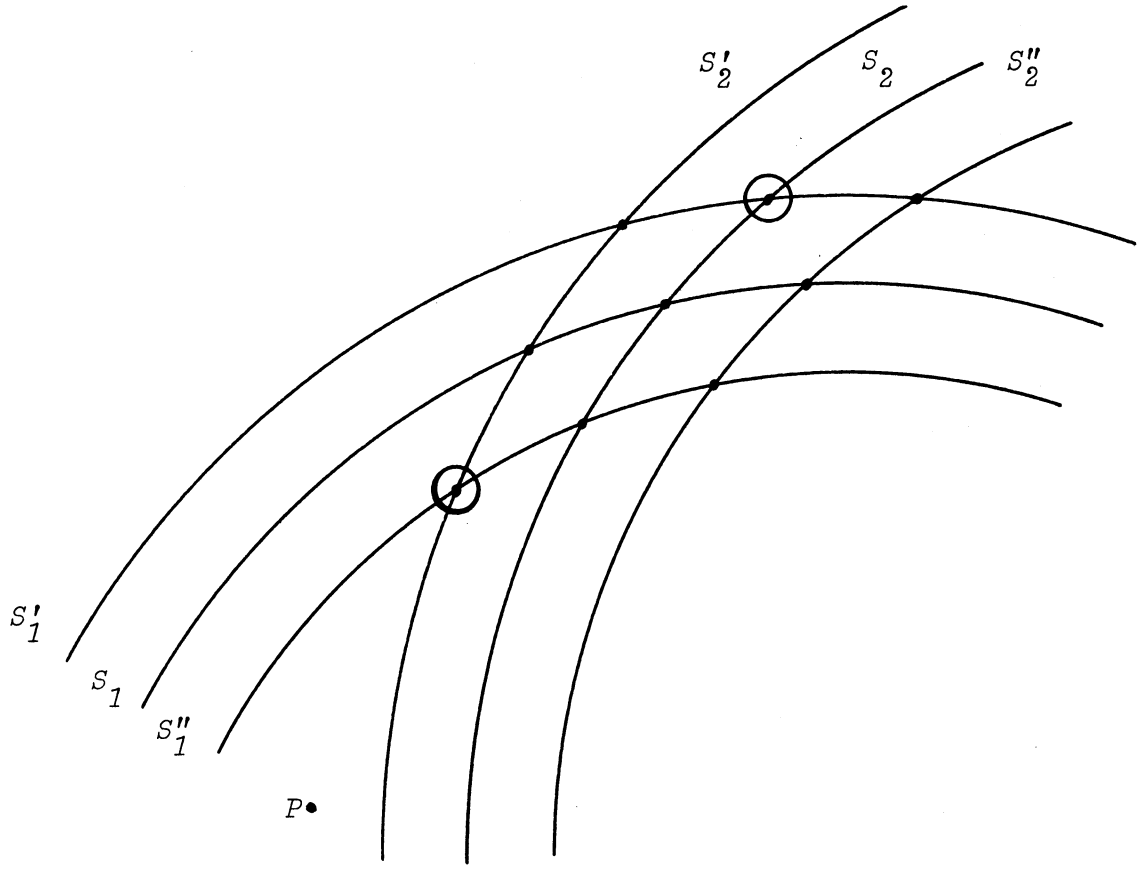
b . des sous-programmes définissant des courbes décalées par rapport aux courbes utilisées dans les instructions, en fonction du diamètre de l'outil. A partir de ces courbes on pourra déterminer la position finale de l'outil parmi les différentes positions possibles suivant les modificateurs TO, ON et PAST d'une instruction de mouvement. Prenons l'exemple de la figure 52a ; en partant du point P avec l'instruction :

GO / TO, S1, TO, S2

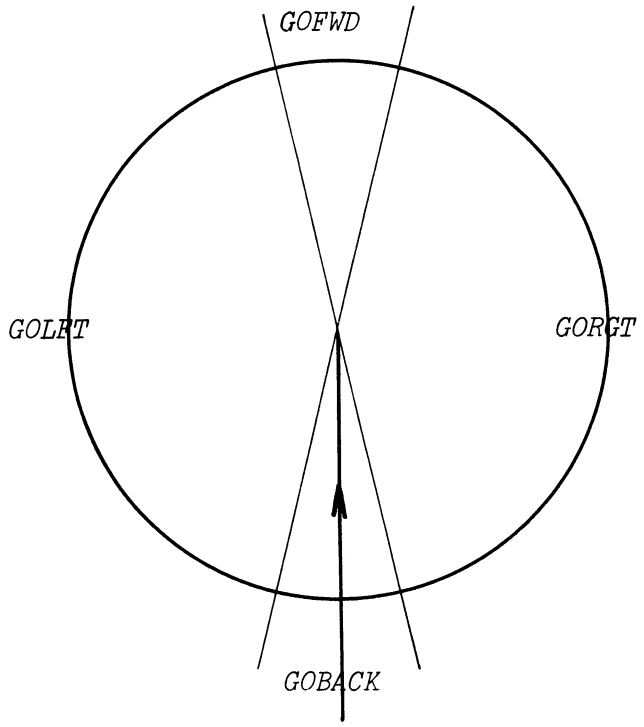
la position finale sera donnée par l'intersection des courbes décalées S_1'' et S_2', tandis que l'instruction :

GO / PAST, S1, ON, S2

donnera comme position finale l'intersection des courbes décalées S_1' et S_2.



a . Position finale.



b . Direction des mouvements;

Figure 52 .

c . des sous-programmes déterminant les intersections de deux courbes (droites, cercles, coniques).

d . des sous-programmes permettant de déterminer la position finale du mouvement en fonction des positions relatives de l'outil et des courbes, ceci pour les instructions GOBACK, GOFWD, GOLFT et GORGT. (Voir figure 52b).

e . des sous-programmes permettant d'approcher les courbes par un contour polygonal en fonction des tolérances données.

Le nombre de sous-programmes composant cet ensemble est important et nous nous sommes d'abord limités aux droites et aux cercles, ce qui simplifie un peu les problèmes mais n'en supprime aucun. Nous avons ainsi établi un ensemble de sous-programmes suffisant pour traiter les ordres GO, GOBACK, GOFWD, GOLFT et GORGT relatifs aux droites et aux cercles. La taille des sous-programmes obtenus est en général supérieure à celle des sous-programmes de la bibliothèque relatifs à l'usinage de point à point. Alors que les sous-programmes relatifs aux travaux en mode point à point sont indépendants, les sous-programmes relatifs aux ordres de contournage font appel aux nombreux autres sous-programmes mentionnés ci-dessus. Les sous-programmes de traitement des ordres GO qui ne comportent qu'une quarantaine d'instructions FORTRAN font ainsi appel à quatre de ces sous-programmes dont le plus important comporte plus d'une centaine d'instructions FORTRAN. Les sous-programmes de traitement des ordres GOBACK, GOFWD, GOLFT et GORGT font appel à six sous-programmes dont le plus important comporte environ 150 instructions FORTRAN.

L'utilisation des recouvrements a cependant permis de traiter ces problèmes sur l'ordinateur IBM 1130. Bien que l'ensemble des sous-programmes continus soit loin d'être complet nous pouvons déjà affirmer qu'il est possible de traiter certains problèmes de contournage à l'aide du système réalisé pour l'ordinateur IBM 1130.

5.3.4. Problèmes de précision .

Il reste cependant à s'assurer sur de nombreux exemples réels que le problème de la précision des calculs ne pose pas de difficultés. En ce qui concerne l'ordinateur IBM 1130 les calculs sont faits en FORTRAN avec une précision de six chiffres significatifs, ce qui est très suffisant pour les tra-

vaux d'usinage de point à point, les calculs mis en jeu étant relativement simples et ne donnant pas lieu à une accumulation d'erreurs. Les travaux de contournage mettent en jeu des calculs beaucoup plus nombreux et cette précision est insuffisante. La DOUBLE PRECISION n'existant pas en FORTRAN basique, les problèmes posés par la précision ne pourront donc pas être résolus d'une façon générale ; il faudra résoudre ces problèmes spécifiquement pour chaque ordinateur utilisé. Ainsi l'ordinateur IBM 1130 offre une précision étendue (27) qui permet de faire des calculs avec neuf chiffres significatifs, ce qui est suffisant dans la majorité des cas. Un système fonctionnant sur petits ordinateurs permet donc de résoudre les problèmes posés par les travaux de contournage plan. Cependant les problèmes posés par la précision peuvent rendre les systèmes de programmation pour travaux de contournage en commande numérique dépendants des petits ordinateurs utilisés.

5.4. CONCLUSIONS PRATIQUES.

Le système est actuellement entièrement écrit en FORTRAN basique, ce qui le rend aussi indépendant de l'ordinateur utilisé que possible. L'ensemble du système comporte approximativement 2500 cartes FORTRAN (1100 cartes pour le préprocesseur et 1400 cartes pour la bibliothèque de sous-programmes), alors qu'un système beaucoup plus complexe comme le système IFAPT comprend près de 15 000 cartes et que la version la plus récente du système APT comprend environ 70 000 cartes FORTRAN. Le travail de conception et de réalisation du système pour les travaux de point à point a occupé une personne pendant un an et demi alors que l'on peut admettre que la somme des efforts consacrés au système APT de sa conception à sa version actuelle est proche de 400 hommes/année. On ne peut cependant donner de signification à une telle comparaison, les systèmes apparentés à APT ayant considérablement profité des travaux de l'équipe APT ; grâce à cela l'effort de programmation du système IFAPT n'a été que de 8 hommes/année. Mais on ne peut vraiment comparer le système que nous avons réalisé au système IFAPT, beaucoup plus complexe et encore moins aux systèmes APT encore bien plus complexes. Le seul lien réel qui existe entre ces systèmes est la syntaxe du langage d'écriture des programmes de pièce.

L'encombrement du système fonctionnant sur l'ordinateur IBM 1130 pourrait encore être réduit en augmentant le nombre des sous-programmes qui le constituent, mais ce serait au détriment du temps de traitement des programmes de pièce. Le système actuel est entièrement écrit en FORTRAN y compris

les sous-programmes d'entrée-sortie ; ceci ne nous permet pas de faire des mesures des performances du système, qui soient significatives. Il serait bon en effet de réécrire les sous-programmes relatifs aux entrées-sorties pour chaque ordinateur utilisé ; ceci sera impératif pour les systèmes destinés à être utilisés de façon intensive.

Actuellement le système peut être utilisé sur ordinateur IBM 1130 et en mode conversationnel sous les systèmes CP67/CMS. On peut utiliser la bibliothèque de sous-programmes sans passer par le préprocesseur ce qui permet d'utiliser le système sur un ordinateur ayant une mémoire centrale trop petite pour le préprocesseur. Le faible encombrement relatif du système peut rendre intéressante son utilisation en mode conversationnel sur un gros ordinateur. Cette dernière solution est celle que nous employons pour la mise au point et les extensions du système ; elle permet de faire facilement des essais pour les extensions du système aux travaux d'usinage en mode continu ou pour l'introduction des données technologiques.

La possibilité d'utiliser des instructions FORTRAN dans les programmes de pièce est une chose nouvelle, dont il est difficile de mesurer tous les avantages. En particulier l'utilisation d'instructions FORTRAN étend au minimum la puissance de calcul du langage APT ; cependant les programmeurs de pièce ont rarement un niveau leur permettant de maîtriser avec aisance plusieurs langages de programmation.

CHAPITRE 6

PERSPECTIVES OFFERTES PAR LES DEVELOPPEMENTS DE LA COMMANDE NUMERIQUE

La commande numérique est une technique récente et ses développements sont encore nombreux. Tous ceux-ci ont pour but une automatisation plus ou moins poussée de l'usinage des pièces mécaniques.

6.1. INTEGRATION DES DONNEES TECHNOLOGIQUES A LA PROGRAMMATION AUTOMATIQUE.

Tous les langages de programmation pour commande numérique permettent de décrire la trajectoire de l'outil, mais parmi ces langages seul EXAPT permet de définir avec précision les conditions de l'usinage. Ces informations technologiques sont habituellement déterminées par le bureau des méthodes, mais la réalisation d'EXAPT a montré qu'il était possible d'intégrer cette phase du travail dans la phase de programmation en utilisant des "données technologiques intégrées".

Les systèmes EXAPT (46) ont été développés par les universités de Berlin et d'Aix-la-Chapelle ; ce sont des sous-ensembles étendus de l'APT. EXAPT1 est destiné aux opérations de perçage et aux travaux de point à point ; en plus des définitions géométriques habituelles de l'APT, ce système permet de définir des opérations d'usinage comme centrage, perçage, chanfreinage, alésage, taraudage, et surfaçage. Le système peut déterminer automatiquement la vitesse de coupe et la vitesse d'avance, il peut aussi choisir les outils ainsi que des suites d'opérations d'usinage aboutissant à l'état final désiré. EXAPT1 est actuellement utilisé en Europe.

EXAPT2 est destiné aux opérations de tournage ; il permet de définir l'état initial de la pièce et son état final. Le programmeur doit définir l'outil et les différents segments composant l'usinage ; à partir de cela le système détermine le nombre de passes à exécuter et les vitesses d'avance et de coupe de chaque passe. La mise au point d'EXAPT2 n'est pas encore entièrement terminée.

EXAPT3 est destiné aux opérations de fraisage ; c'est un surensemble d'EXAPT1 comportant en plus les instructions permettant l'usinage des surfaces par courbes de niveau (travail selon deux axes et demi). Ce système qui est encore dans sa phase de conception permettra les travaux de contournage et la détermination automatique des vitesses de coupe et d'avance, la détermination automatique du nombre de passes, la sélection automatique des outils et la détermination des cycles de travail. Sa réalisation pose encore de nombreux problèmes.

L'automatisation des systèmes EXAPT est basée sur deux fichiers "technologiques". Le premier fichier est un "fichier outil" comprenant les descriptions d'une série d'outils utilisables par la machine à commande numérique considérée. Chaque outil est repéré par un nombre permettant de connaître sa forme ; on lui associe un ensemble de données précisant ses différentes dimensions et un ensemble de données définissant les conditions de son utilisation. Le second fichier est un "fichier matière" contenant les informations technologiques relatives à l'usinage d'un matériau donné. C'est à partir de ces données que le système pourra choisir le lubrifiant et le type de l'outil et pourra déterminer les vitesses de coupe et d'avance en fonction du diamètre usiné.

Le système EXAPT1 offre également la possibilité de déterminer automatiquement des suites d'usinages pour certains travaux (centrage, perçage, chanfreinage, alésage, taraudage et surfaçage) ; pour chacun de ces travaux le système pourra déterminer la succession des opérations, les différents outils utilisés et les conditions de coupe (45). La figure 53 donne un exemple de détermination des suites d'usinages pour un alésage. Une première définition technologique (PART) définit le matériau utilisé pour la fabrication de la pièce et spécifie son état initial : non usiné (UNMACH) et surface rugueuse (ROUGH). La seconde instruction définit une opération d'usinage, c'est un alésage (REAM) à un diamètre de 40 mm, une profondeur de 35 mm avec chanfrein (BEVEL) et un positionnement précis (TOLPO). A partir de ces deux instructions le système suit l'organigramme de la figure 53 et examine l'état initial de la pièce ; si la pièce comporte un fourreau (CORED) on passe

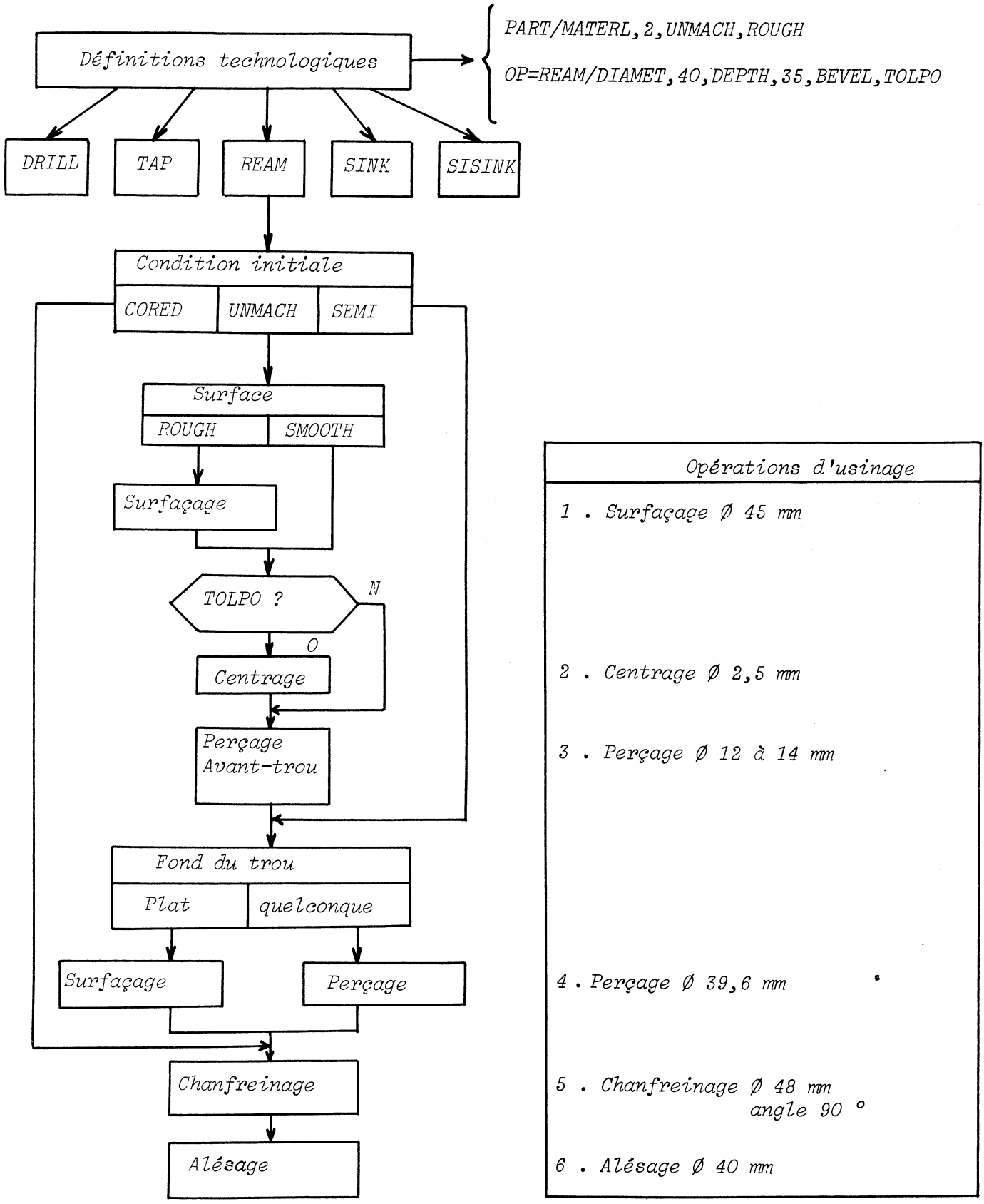


Figure 53 . Détermination de suites d'usinages (45).

tout de suite au chanfreinage et à l'alésage. Si la pièce a déjà subi un usinage préparatoire (SEMI) on saute les premières opérations. Dans notre exemple la pièce est non usinée et rugueuse, on commence par faire un surfaçage, le modificateur TOLPO requiert ensuite un centrage ; ceci fait, on perce un avant-trou, le fond du trou ayant une forme quelconque ; la quatrième opération sera un perçage simple ; on exécutera ensuite le chanfreinage et l'alésage. Une seule instruction permet ainsi de définir un cycle qui comprend ici 6 opérations d'usinage différentes utilisant des outils différents.

On peut diviser un programme EXAPT1 en quatre grandes parties : une partie de spécifications générales, une partie de définitions géométriques semblables à celles d'un programme APT, une partie de définitions technologiques et une partie d'ordres d'usinage. Afin de donner une meilleure idée des possibilités offertes par EXAPT1 nous prendrons un exemple simple correspondant à l'usinage de la pièce de la figure 54. Afin d'effectuer cet usinage on utilise le programme de pièce suivant (34) :

```
PARTNO   EXEMPLE DE PIECE                              (1)
         MACHIN/BOWE3                                  (2)
         TRANS/200,100,0                               (3)
REMARK   DEFINITIONS GEOMETRIQUES                      (4)
         ZSURF/12                                      (5)
   PM =  POINT/0,0,12                                  (6)
   C1 =  CIRCLE/CENTER,PM,RADIUS,55                    (7)
  PAT =  PATERN/ARC,C1,45,CCLW                         (8)
REMARK   DEFINITIONS TECHNOLOGIQUES                    (9)
         PART/MATERL,12                                (10)
         CLDIST/0.8                                    (11)
PERC 1 = DRILL/SO,DIAMET,5,DEPTH,12,TOOL,315           (12)
FILET  = TAP/SO,DIAMET,6,DEPTH,12,TOOL,416             (13)
PERC2  = DRILL/DIAMET,11,DEPTH,12                      (14)
LAMAGE = SINK/SO,DIAMET,17.5,DEPTH,8.3                 (15)
ALESE  = REAM/DIAMET,20,DEPTH,12                       (16)
REMARK   ORDRES D'USINAGE                              (17)
         COOLNT/ON                                     (18)
         FROM/-100,0,100                               (19)
         WORK/PH,PERC1,FILET                           (20)
```

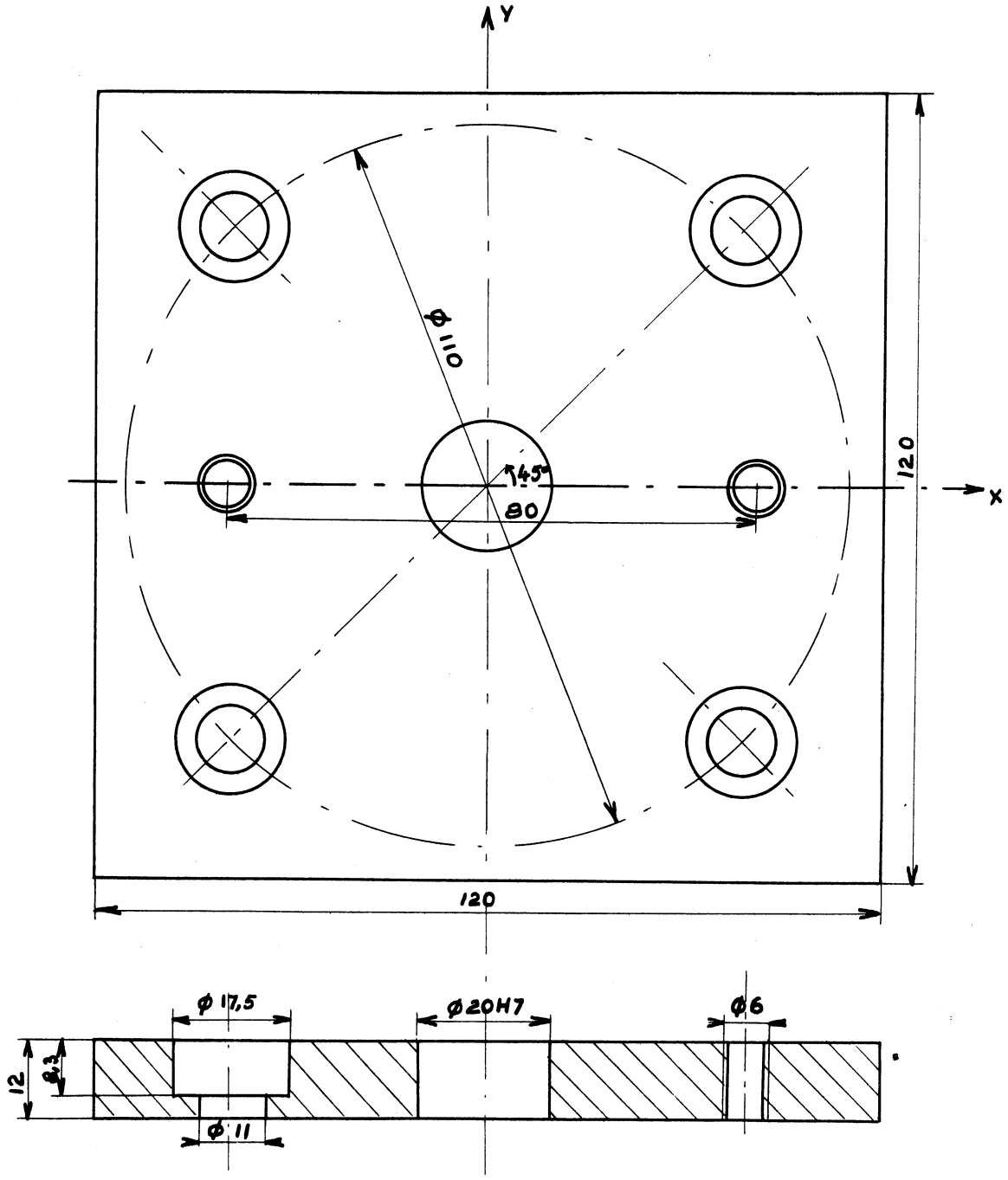


Figure 54 . *Perçage d'une plaque en EXAPT1 (34).*

```
GOTO/-40,0,12                    (21)
GOTO/40,0,12                     (22)
WORK/ALESE                       (23)
GOTO/PM                          (24)
WORK/PERC2,LAMAGE                (25)
GOTO/PAT                         (26)
FINI                             (27)
```

Examinons maintenant ce programme instruction par instruction.

(1) Identification du programme.

(2) Nomme la machine-outil et le postprocesseur utilisés.

(3) Situe l'origine des coordonnées de la machine par rapport aux coordonnées de la pièce.

(4), (9) et (17) sont des commentaires sans effet sur le programme.

(5) Spécifie la surface supérieure de la pièce.

(6) Définit le point central de la pièce.

(7) Définit le cercle C1 de diamètre 110.

(8) Définit le réseau de quatre trous sur C1, espacés à 90°.

(10) Fixe le matériau utilisé pour la fabrication.

(11) Fixe la distance de sécurité à 0,8 mm au dessus de la pièce, comme limite de l'avance rapide.

(12) Définit une opération de perçage en une seule passe (SO pour "single operation") avec l'outil numéro 315.

(13) Définit une opération de taraudage en une seule passe (SO) avec l'outil numéro 416.

(14) Définit une opération de perçage à un diamètre de 11 mm, l'outil sera choisi par le système qui déterminera les différentes passes à effectuer.

(15) Définit une opération de lamage en une seule passe (SO).

(16) Définit une opération d'alésage à un diamètre de 20 mm. Les outils seront choisis par le système et les différentes passes seront déterminées automatiquement.

(18) Déclenche l'arrosage.

(19) Donne la position initiale de l'outil.

(20) Appel des opérations de perçage PERC1 et de taraudage FILET qui seront exécutées successivement sur chaque trou (PH pour "per hole") : on perce puis on taraude chaque trou avant de passer au suivant.

(21) et (22) définissent les positions où s'appliquent les opérations définies en (20).

(23) Appel de l'opération d'alésage ALESE.

(24) Définit la position où se feront les opérations définies en (23).

(25) Appel des opérations de perçage PERC2 et de lamage LAMAGE qui, en l'absence du modificateur PH seront exécutées séparément : on percera d'abord tous les trous, on effectuera ensuite tous les lamages.

(26) Définit les positions où se feront les opérations définies en (25).

(27) Fin de l'usinage.

La structure d'un programme EXAPT2 sera différente de celle d'un programme EXAPT1. Un programme EXAPT2 comprend des spécifications générales, une description de la pièce brute, une description de la pièce usinée dans laquelle les contours sont définis au moyen de segments correspondant aux différents usinages à effectuer, une partie technologique définissant différentes opérations d'usinage et les outils utilisés, et enfin une partie décrivant la suite des usinages à effectuer suivant les segments précédemment définis.

L'automatisation des deux systèmes EXAPT1 et EXAPT2 n'est pas complète mais elle constitue un énorme progrès par rapport à la programmation automatique telle qu'elle est conçue dans les systèmes APT. Bien que les langages EXAPT1 et EXAPT2 soient des sous-ensembles (du point de vue de la géométrie seulement) du langage APT, leurs programmes de traitement nécessitent des ordinateurs importants, l'introduction des données technologiques ayant engendré de nouveaux programmes de traitement. Antérieurement

à EXAPT les langages de programmation étaient surtout consacrés aux définitions géométriques permettant de définir la trajectoire de l'outil ; dans les systèmes EXAPT au contraire la géométrie ne constitue qu'une partie auxiliaire et nécessaire des programmes de pièce, ceux-ci sont maintenant résolument orientés vers l'usinage. L'introduction des données technologiques a marqué un grand pas en avant dans la voie de l'automatisation complète de l'usinage des pièces mécaniques ; tous les systèmes de programmation actuels en tiennent d'ailleurs compte.

La structure du système que nous avons réalisé nous permettra d'introduire facilement de nouvelles instructions du langage ; le traitement de ces instructions sera assuré par de nouveaux sous-programmes. Il est donc possible d'ajouter sans peine à notre système une partie de traitements technologiques. Cependant l'introduction des données technologiques requiert l'existence d'un fichier outil et d'un fichier matière. La taille de ces fichiers n'est pas nécessairement compatible avec les petits ordinateurs que nous voulons utiliser. D'un autre côté le système peut être utile pour faire facilement des essais d'introduction de données technologiques, en les faisant naturellement sur un ordinateur de taille convenable.

6.2. COMMANDE ADAPTATIVE

L'avantage principal des machines-outils à commande numérique est de réduire les temps morts des machines. On peut encore augmenter la productivité des machines à commande numérique s'il est possible de mesurer les différents paramètres de coupe du métal au cours de l'usinage et de corriger cet usinage en fonction des valeurs mesurées. Ce processus est appelé commande adaptative. La commande adaptative permettra d'usiner aux vitesses d'avance maximum, quitte à ralentir si le métal de la pièce comporte des points durs, et de prendre en compte l'usure de l'outil. La commande adaptative permet aussi d'optimiser à l'exécution les usinages que l'on n'aurait pas pris la peine de déterminer et de programmer dans un système conventionnel. Si l'on vient à usiner une partie de pièce déjà évidée, la commande adaptative permettra de passer en vitesse rapide alors qu'un système conventionnel usinerait lentement le vide de la partie creuse traversée. Pour faire tout cela la commande adaptative utilise

une série de capteurs et de palpeurs. Les résultats produits par les systèmes expérimentaux de commande adaptative sont très encourageants et permettent des gains de temps importants ; mais la technique des palpeurs en est encore à un stade trop peu avancé pour pouvoir être utilisée d'une façon sûre sans coût excessif.

6.3. COMMANDE NUMERIQUE DIRECTE.

Le domaine de la commande numérique est en perpétuelle évolution et la commande numérique directe qui vient de faire son apparition est certainement un progrès aussi considérable que l'avènement de la commande numérique elle-même.

Les progrès récents de la technologie des ordinateurs (tant "hardware" que "software") et en particulier le télétraitement, l'apparition de petits ordinateurs peu coûteux, le temps partagé et la multiprogrammation ont permis des améliorations importantes dans le domaine de la commande numérique des machines-outils et de la gestion de la fabrication mécanique. La commande numérique directe est une des conséquences de ces progrès ; la commande numérique directe consiste à remplacer plusieurs dispositifs de commande numérique conventionnels par un petit ordinateur qui gère ainsi le travail de plusieurs machines-outils.

L'emploi d'une commande numérique directe est parfois imposé par le travail à effectuer. Tel fut le cas de rectifieuses trois axes commandées numériquement et usinant des ogives (43) ; l'usinage d'une ogive demanderait une bande perforée de plusieurs kilomètres de long, alors que l'utilisation d'un petit ordinateur pour commander directement deux rectifieuses permet d'utiliser une bande magnétique où se trouvent les données numériques relatives à 13 types d'ogive. L'utilisation de la commande numérique directe dans des cas plus généraux prend cependant une importance considérable et des constructeurs de petits ordinateurs aussi bien que des constructeurs de systèmes de commande numérique ont commencé l'étude de nouveaux systèmes de commande numérique directe (19,39).

La figure 55 décrit l'organisation générale d'un système de commande numérique directe. Ce système possède trois niveaux. Le niveau supérieur peut être appelé niveau de gestion (ou également niveau APT) et comprend un gros ordinateur pouvant traiter des programmes de pièce APT. Le niveau intermédiaire peut être appelé niveau des données ; il comprend un petit ordinateur et une mémoire de masse dans laquelle sont rangées les données numériques produites par un système APT. Le niveau inférieur peut être appelé niveau de fabrication ; il comprend un ensemble de sous-systèmes de commande numérique directe. La composition des sous-systèmes n'est pas uniforme à l'heure actuelle et en fait chaque constructeur présente une structure de sous-système particulière. D'une façon générale un sous-système de commande numérique directe comprend un système de commande relié au niveau supérieur, un panneau de contrôle et la machine-outil proprement dite.

Le nombre de sous-systèmes reliés au petit ordinateur du niveau intermédiaire est variable suivant les cas et la taille de l'ordinateur ; on pense en général qu'il faut un minimum de trois sous-systèmes par petit ordinateur pour que la commande numérique directe soit plus économique que la commande numérique conventionnelle. La différence fondamentale entre les différents systèmes de commande numérique directe vient surtout de la conception des systèmes de commande. Un système de commande numérique conventionnel dirige une seule machine-outil et a pour rôle la lecture du ruban perforé, l'interpolation, le contrôle des servomécanismes de la machine-outil, le décodage et l'exécution des fonctions auxiliaires et la communication avec l'opérateur au moyen du panneau de contrôle. Certains veulent employer ces systèmes de commande conventionnels auxquels serait retiré le dispositif de lecture du ruban perforé. D'autres préfèrent remplacer le système de commande conventionnel, essentiellement "hardware", par un petit ordinateur programmé et obtenir ainsi un système de commande "software" (39). D'autres enfin conçoivent des systèmes de commande entièrement nouveaux, le petit ordinateur du niveau intermédiaire faisant l'interpolation complète (25) ou ne faisant qu'une interpolation grossière à partir de laquelle le système de commande fait une interpolation fine (18). L'incertitude qui règne quant au choix du système de commande montre à quel point le domaine est récent et montre qu'il faudra encore attendre quelque temps pour avoir des résultats permettant de mieux juger les dif-

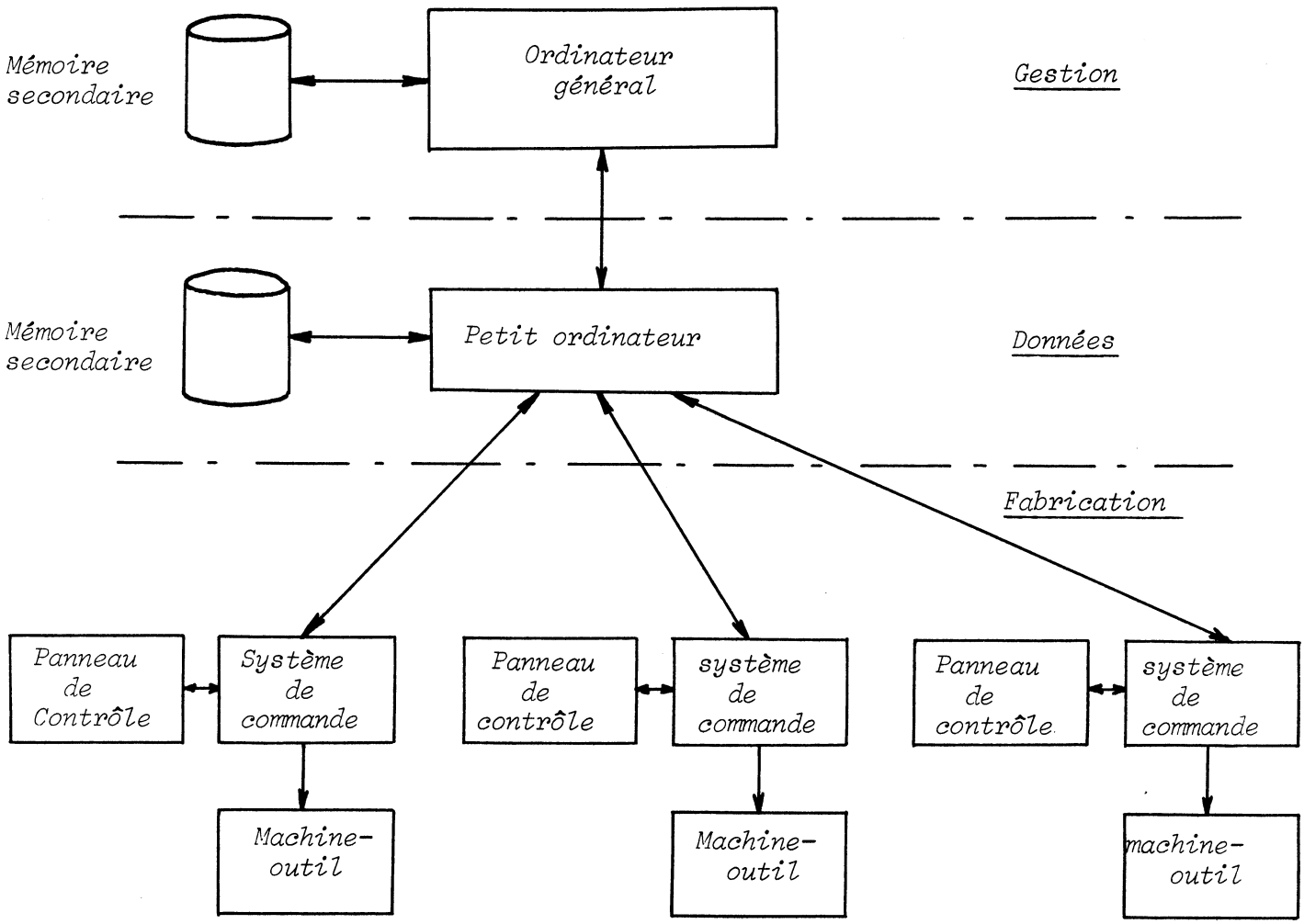


Figure 55 . Système de commande numérique directe.

férents types de commande numérique directe.

Les réalisations exploitées actuellement sont encore peu nombreuses (18, 25, 43) ; nous prendrons en exemple une des réalisations les plus avancées. Il s'agit d'un système mis au point par la Mc Donnel-Douglas Corporation au cours des deux dernières années (25). La figure 56 décrit la structure du système. Le traitement des programmes de pièce est fait par un ordinateur IBM 360/65 qui fournit un ensemble de données numériques à l'ordinateur intermédiaire IBM 1800 DACS. Ce dernier contrôle n machines-outils différentes et joue également le rôle d'interpolateur. Le système de commande attaché à chaque machine-outil comporte un panneau de contrôle et exécute les fonctions suivantes :

1 . acquisition des données
2 . contrôle des servomécanismes
3 . décodage et exécution des fonctions auxiliaires
4 . accumulation et transmission de données statistiques (pour la gestion de l'atelier)
5 . communication avec l'opérateur
6 . arrêt de sécurité en cas de panne

Le système tel qu'il est conçu utilise 28K mots de la mémoire centrale de 32K mots de l'ordinateur IBM 1800 DACS, et peut contrôler au maximum dix machines-outils.

L'utilisation d'un tel système a permis de tirer un certain nombre de conclusions ; la réduction de l'espace occupé dans l'atelier est négligeable et les circuits électroniques n'ont pas été réduits autant que prévu, ils ont été surtout modernisés et simplifiés. Parmi les avantages apportés par ce système on cite l'élimination des rubans perforés et de leur lecteur, la réduction des pannes dues aux circuits électroniques, la réduction des rebuts, la gestion améliorée, la réduction du nombre de pièces de rechange électroniques, la réduction de l'entretien et la possibilité de repartir automatiquement après un arrêt. D'autres essais permettront de juger de la validité de ces conclusions.

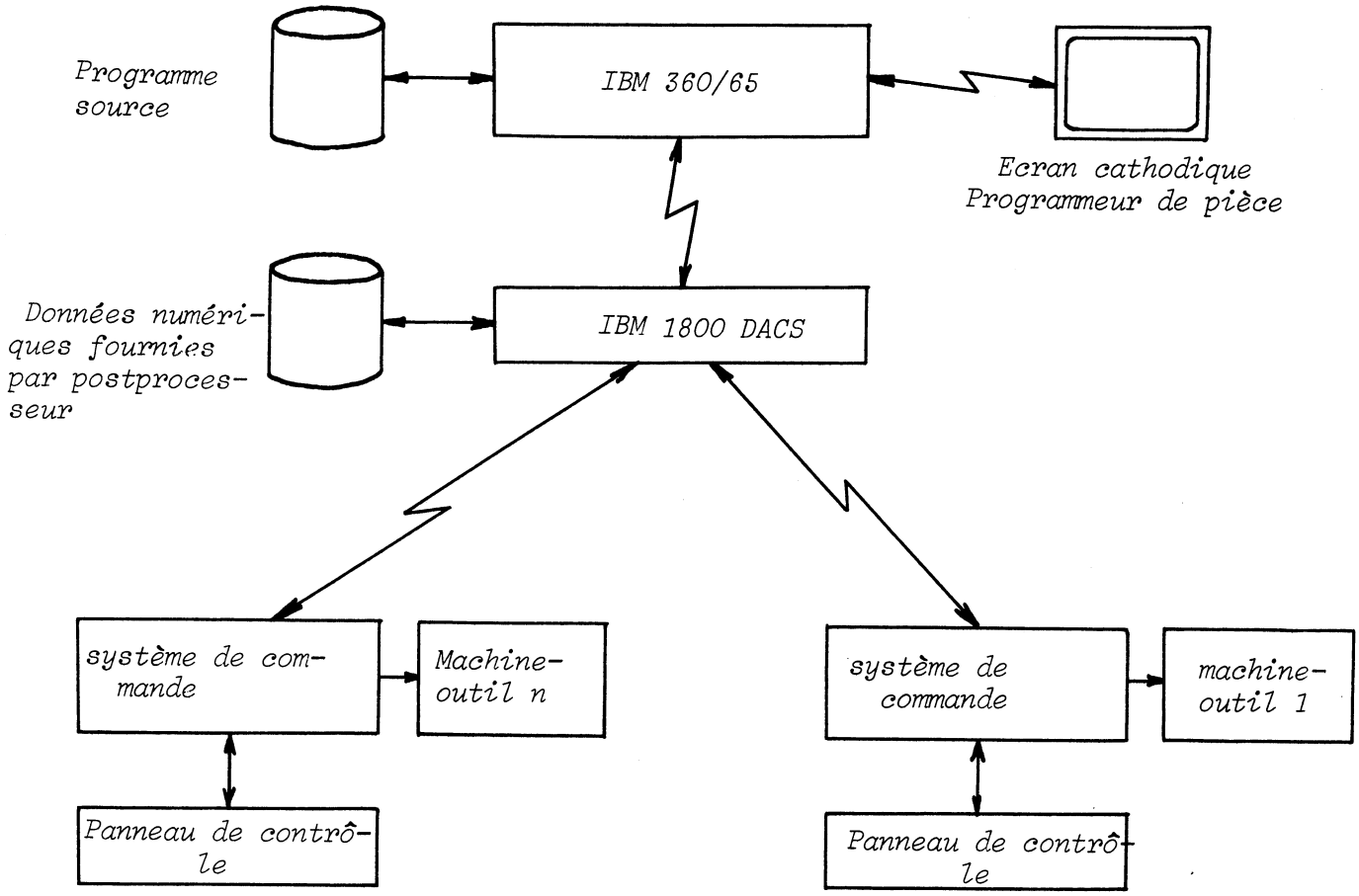


Figure 56 . *Système de commande numérique directe Mc Donnel-Douglas (25).*

Le concept de commande numérique directe est facilement adaptable à des réalisations comprenant la commande adaptative, l'inspection en cours de travail ou la gestion intégrée d'un atelier de commande numérique. On peut considérer la commande numérique directe comme constituant une partie, petite mais importante, d'une structure plus générale de conception assistée par ordinateur et de fabrication assistée par ordinateur (CAD/CAM pour "computer aided design/computer aided manufacturing") qui comprendrait aussi un système de gestion (MIS) donnant des rapports continus et précis sur le déroulement des opérations de fabrication. La commande numérique directe est un pas en avant dans la voie de l'automatisation complète de l'usine.

6.4. GESTION INTEGREE DES ATELIERS DE COMMANDE NUMERIQUE.

Les machines-outils à commande numérique représentent une proportion considérable de l'ensemble de toutes les machines-outils en service actuellement. Le nombre d'ateliers de machines à commande numérique est important et la prolifération de ces ateliers dans les grosses compagnies américaines a apporté une multitude de problèmes. Parmi ceux-ci se trouvent les problèmes relatifs à la gestion de l'atelier de machines à commande numérique ; le coût de fonctionnement plus élevé des machines à commande numérique fait que les délais de production doivent être évités. Une synchronisation précise est essentielle afin que les bandes correctes, les outils, les montages et les machines à commande numérique soient prêts en même temps pour réduire au minimum les délais de production. C'est pourquoi on a vu apparaître des systèmes de contrôle pour les ateliers de commande numérique, fonctionnant sur petits ordinateurs (43) ; ces systèmes permettent une meilleure synchronisation des machines de l'atelier et permettent ainsi l'optimisation de la production de cet atelier. Il existe actuellement un important développement de la gestion intégrée des ateliers de commande numérique au moyen de petits ordinateurs.

6.5. AIDES GRAPHIQUES ET COMMANDE NUMERIQUE.

La commande numérique des machines-outils est une application toute trouvée des techniques graphiques. La plus simple des applications est l'utilisation, maintenant courante, d'une table traçante pour dessiner la trajectoire de l'outil et vérifier ainsi la validité du programme de pièce sans risquer d'endommager la machine-outil. Le programmeur de pièce doit travailler à partir du

dessin de la pièce et il est logique de penser que son travail sera amélioré s'il voit directement sur un écran cathodique le résultat de son programme. L'importance des applications graphiques en commande numérique a été reconnue très tôt mais les efforts faits ont été souvent indépendants et non coordonnés. La General Motors a été la première compagnie à utiliser un système graphique pour la commande numérique, d'abord avec l'écran cathodique DAC 1 et l'ordinateur IBM 7094, puis avec plusieurs unités 2250 connectées à un ordinateur IBM 360/67 (44). Plusieurs autres compagnies américaines ont reconnu la complexité des problèmes mis en jeu par le contournage 3 axes et se sont groupées pour créer un système graphique pour la commande numérique connu sous le nom de "Numerical Control Graphics" ou NCG (29).

Ce système permet à un programmeur de pièce d'utiliser un terminal graphique et à partir de là de créer des figures géométriques visibles sur l'écran cathodique en utilisant un photostyle, les clés de fonction et le clavier du terminal pour construire, modifier, supprimer, déplacer et faire varier la taille du profil de pièce. Ce système permet d'observer l'usinage automatique des évidements et de définir les usinages pas à pas ; il crée lui-même un ensemble CLDATA compatible avec les ensembles CLDATA des systèmes APT. En fait ce système n'est qu'un système APT amélioré. Le programmeur n'a plus à connaître rigoureusement la syntaxe du langage APT, l'appel de chaque ordre d'usinage faisant apparaître sur l'écran les données à préciser. Le système permet alors au programmeur de pièce de voir la pièce telle qu'il l'a définie et les différents usinagesqu'il a prévus.Il peut alors déterminer dans certains cas les collisions possibles.

Un tel système permet de réduire le temps de passage du dessin de la pièce à la bande perforée et permet aussi aux ingénieurs du bureau d'études de se rendre compte des difficultés d'usinage des pièces qu'ils créent. Un système de ce type permet difficilement de montrer l'usinage réel, l'écran cathodique étant utilisé comme une table traçante pour représenter des plans de la pièce à usiner. Un système plus complet devrait pouvoir montrer les évolutions de l'outil dans l'espace et déterminer automatiquement les collisions ce qui pose encore de nombreux problèmes.

Les techniques numériques de relevés de cotes sur tables à dessiner spéciales qui existent actuellement, permettent de passer simplement du dessin de la pièce à la bande perforée la représentant, et sont parfois des solutions plus simples. L'utilisation des écrans cathodiques est actuellement limitée par le coût élevé des terminaux graphiques évolués et par la complexité mise en jeu par la représentation de figures dans un espace à trois dimensions.

CONCLUSION

Notre travail avait pour but la réalisation d'un système de programmation pour la commande numérique des machines-outils, simple, facilement modifiable et susceptible de fonctionner sur petits ordinateurs. La réalisation effective d'un système de programmation pour travaux d'usinage de point à point et fonctionnant sur un ordinateur IBM 1130 (mémoire centrale : 8K mots, mémoire secondaire : un disque) pris comme exemple, nous a permis de montrer que l'idée de départ, consistant à utiliser de petits sous-programmes en guise de modules, était valable et applicable.

L'étude des problèmes posés par l'usinage en mode continu a montré qu'il était possible d'utiliser le système réalisé pour des travaux de contournage limités cependant à certains cas d'usinage de surfaces par courbes de niveau. La structure du système et la facilité avec laquelle on peut le modifier le rendent compatible avec les systèmes existants et permettent d'envisager des extensions dans plusieurs domaines.

L'utilisation d'un ordinateur de petite taille impose des limites au système mais un petit système spécialisé est souvent plus intéressant pour des utilisateurs disposant d'un petit ordinateur, qu'un système très général, souvent beaucoup plus lourd à manier. L'adaptation du système à d'autres petits ordinateurs d'un type voisin de celui qui a été utilisé (comme par exemple l'ordinateur CII 10020) ne pose pas de problèmes.

Bien que ce système ne soit certainement pas le système idéal, il possède certains avantages par rapport aux systèmes récents destinés aux petits ordinateurs (10, 35, 52) qui n'en possèdent pas la souplesse. En effet la simplicité et la clarté du système sont bien réelles et les modifications sont très faciles à faire ; le système s'est également révélé être un outil commode pour les expérimentations. Dans un avenir proche la version point à point de ce système doit être mise à la disposition d'utilisateurs de commande numérique et les extensions du système pour les travaux de contournage seront développées.

ANNEXE

DESCRIPTION DU LANGAGE APT

Cette description ne prétend pas décrire le langage APT d'une façon complète ; elle a été établie à partir des références 16, 28 et 32 auxquelles on se reportera pour plus de détails.

Un programme de pièce APT est formé d'une suite d'instructions. La plupart des instructions APT sont divisées en deux sections, séparées par une barre oblique (/). A gauche de la barre oblique se trouve un mot majeur ; la section complémentaire se trouve à droite de la barre oblique lorsqu'elle existe. Par exemple l'instruction :

D1 = LINE / P1, P2

définit la droite D1 passant par les points P1 et P2. Les imbrications sont permises comme par exemple dans l'instruction :

GOTO / (POINT/INTOF,D1,D2)

qui déplace l'outil vers un point qui est l'intersection de deux droites D1 et D2 précédemment définies.

La notation utilisée dans la description qui suit est identique au métalangage utilisé dans le Rapport révisé sur le langage algorithmique ALGOL 60 édité par Peter Naur. Les symboles de base du métalangage sont les suivants :

::= connexion métalinguistique signifiant "est défini être".
| connexion métalinguistique signifiant "ou".
< > crochets entourant les variables métalinguistiques.

1.0. SYMBOLES DE BASE. IDENTIFICATEURS. NOMBRES. CHAINES.

1.0.1. Sémantique.

Le langage APT est composé d'un certain nombre de symboles de base.

1.0.2. Syntaxe.

<symbole de base> ::= <lettre>|<chiffre>|<délimiteur>

1.1. LETTRES.

1.1.1. Sémantique.

On utilise l'alphabet majuscule romain pour former les identificateurs et les chaînes.

1.1.2. Syntaxe.

<lettre> ::= A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z

1.2. CHIFFRES.

1.2.1. Sémantique.

On utilise les chiffres décimaux pour former les identificateurs, les nombres et les chaînes.

1.2.2. Syntaxe.

<chiffre> ::= 0|1|2|3|4|5|6|7|8| 9

1.3. DELIMITEURS.

1.3.1. Sémantique.

Les délimiteurs sont des combinaisons d'un ou de plusieurs symboles de base qui ont un sens donné dans le langage et constituent eux-mêmes des symboles de base.

1.3.2. Syntaxe

<délimiteur> ::= <opérateur>|<séparateur>|<mot réservé>

<séparateur> ::= ,|(|)

<opérateur> ::= +|-|*|/

<mot réservé> ::= <mot du dictionnaire APT>

1.3.3. Exemples.

GOLFT

+

(

-

)

COOLNT

CUTTER

1.4. IDENTIFICATEURS.

1.4.1. Sémantique.

Les identificateurs sont des chaînes comprenant de 1 à 6 lettres ou chiffres. A part les étiquettes qui peuvent ne comporter que des chiffres, un identificateur valable doit comprendre au moins une lettre qui se trouvera en tête de l'identificateur.

1.4.2. Syntaxe.

<identificateur> ::= <lettre>|<chiffre><identificateur>|
<identificateur><chiffre>|<identificateur><lettre>

1.4.3. Exemples.

ABLE

123456

X74

1.5. NOMBRES.

1.5.1. Sémantique.

Un nombre est composé de 1 à 12 caractères, y compris le point décimal. Sans point décimal un nombre est pris comme un entier. Le seul type de nombre utilisé en APT est le nombre décimal flottant en représentation interne.

1.5.2. Syntaxe.

<nombre> ::= <nombre sans signe> |+<nombre sans signe>| -<nombre sans signe>
<nombre sans signe> := <entier sans signe>|<partie décimale>|
<entier sans signe><partie décimale>
<partie décimale> ::=.<entier sans signe>
<entier sans signe> ::= <chiffre>|<entier sans signe><chiffre>

1.5.3. Exemples.

0

0.0

0.1234

+7362

7360.7360

1.6. CHAINES.

1.6.1. Sémantique.

Les chaînes sont des suites de symboles de base, de moins de 66 caractères.

1.6.2. Syntaxe.

<chaîne> ::= <toute séquence de symboles de base>|<vide>

<vide> ::=

1.6.3. Exemples.

2966XQLF**)0(

CECI EST UNE CHAINE

1.7. COMMENTAIRES.

1.7.1. Sémantique.

Les commentaires sont des parties de texte introduites dans un programme APT pour améliorer la présentation. Ils n'ont pas d'autre signification.

1.7.2. Syntaxe.

<commentaire> ::= REMARK <chaîne>

1.7.3. Exemple.

REMARK CECI EST UN COMMENTAIRE

2.0. EXPRESSIONS.

2.0.1. Sémantique.

Une expression est une règle pour définir un élément géométrique ou pour calculer une valeur numérique.

2.0.2. Syntaxe.

<expression> ::= <expression arithmétique>|<expression géométrique>

2.1. VARIABLES.

2.1.1. Sémantique.

Une variable est un identificateur servant de nom à un élément géométrique ou à une valeur numérique définie par une expression. Elle peut être utilisée à la place de l'expression à chaque utilisation de cette expression.

2.1.2. Syntaxe.

<variable> ::= <variable simple>|<variable indicée>

<variable indicée> ::= <identificateur de tableau> (<expression en indice>)

<identificateur de tableau> ::= <identificateur>

<expression en indice> ::= <expression arithmétique>

<variable simple> ::= <identificateur>

<variable arithmétique> ::=<variable affectée à une expression arithmétique>

<variable géométrique> ::=<variable affectée à une expression géométrique>

2.1.3. Exemples.

A

A(3)

2.2. INDICATEURS DE FONCTION.

2.2.1. Sémantique.

Les indicateurs de fonction définissent des valeurs arithmétiques simples suivant des algorithmes spéciaux du système APT.

2.2.2. Syntaxe.

<indicateur de fonction> ::= <identificateur de fonction>
(<liste de paramètres de fonction>)

<liste de paramètres de fonction> ::= <expression>|<variable>|
<liste de paramètres de fonction>,<liste de paramètres de fonction>

<identificateur de fonction> ::= DOTF|SQRTF|SINF|COSF|EXPF|LOGF|ABSF|
LNTHF|ATANF

2.2.3. Exemples.

SINF (A+B)

LNTHF (VECTOR/P1,P2)

2.3. EXPRESSIONS ARITHMETIQUES.

2.3.1. Sémantique.

Une expression arithmétique est une règle pour calculer une valeur numérique.

2.3.2. Syntaxe.

<expression arithmétique> ::= <terme>|<opérateur additif><terme>|
<expression arithmétique><opérateur additif><terme>

<terme> ::= <facteur>|<terme><opérateur multiplicatif><facteur>

<facteur> ::= <primaire arithmétique>|<facteur> ** <primaire arithmétique>

<primaire arithmétique> ::=<nombre sans signe>|<variable arithmétique>
<indicateur de fonction>|(<expression arithmétique>)

<opérateur multiplicatif> ::= * |/

<opérateur additif> ::= +|-

2.3.2. Exemples.

A + COSF(A+B+6.02/4)JKL

A(7) + COSF(Q(6+I+X))

2.4. EXPRESSIONS GEOMETRIQUES.

2.4.1. Sémantique.

Une expression géométrique est une règle pour définir un élément géométrique comme un cercle, une droite, etc. L'élément est spécifié au moyen d'une forme géométrique. Pour chaque forme géométrique il y a de 1 à 14 différentes méthodes de définition. Le groupement spécifique d'expressions, de variables et de modificateurs (un sous-ensemble de mots réservés) en une liste de paramètresdétermine quelle est la définition à utiliser.

2.4.2. Syntaxe.

<expression géométrique> ::= <forme géométrique>/<liste de paramètres>
<forme géométrique> ::= POINT|PLANE|CIRCLE|LINE|CYLNDR|ELLIPS|
HYPERB|CONE|GCONIC|LCONIC|SPHERE|QADRIC|POLCON|TABCYL|MATRIX|
VECTOR
<liste de paramètres> ::= (<expression>)|<nombre>|<variable>|
<liste de paramètre>,<modificateur>|<modificateur>,
<liste de paramètres>|<liste de paramètres>,<liste de paramètres>
<modificateur> ::= <mot réservé>

2.4.3. Exemples.

POINT/3.5,7,9 définit un point à (3.5,7,9)
CIRCLE/CENTER,PT,RADIUS,7 définit un cercle dont le centre est situé aux coordonnées associées à la variable PT et de rayon 7.
CIRCLE/CENTER,(POINT/3.5,7,9),RADIUS, 7 définit un cercle centré en (3.5,7,9) et de rayon 7.
CIRCLE/PT1, (POINT/3.5,7,9), PT4 définit un cercle passant par les coordonnées de PT1, (3.5,7,9) et PT4.

3.0. INSTRUCTIONS.

3.0.1. Sémantique.

Les instructions sont les unités de base du langage APT.

3.0.2. Syntaxe.

<instruction> ::= <étiquette>)<instruction non étiquetée>|
<instruction non étiquetée>
<instruction non étiquetée>::= <instruction d'affectation>|
<instruction de positionnement de l'outil>|
<instruction de contrôle>|
<instruction de contrôle du postprocesseur>|
<instruction d'appel de procédure>|
<instruction de contrôle d'entrée-sortie>
<boucle> ::= LOOPST <liste d'instructions> LOOPND
<liste d'instruction> ::= <instruction>|<liste d'instruction><instruction>

3.1. INSTRUCTIONS D'AFFECTATION.

3.1.1. Sémantique.

Les instructions d'affectation servent à affecter des variables aux valeurs d'expressions arithmétiques et aux définitions d'expressions géométriques. Lorsqu'une variable est affectée à une expression géométrique on ne peut la réaffecter. Les variables affectées aux expressions arithmétiques peuvent être réaffectées.

3.1.2. Syntaxe.

<instruction d'affectation> ::= <variable> =
<expression arithmétique> | <variable> = <expression géométrique>

3.1.3. Exemples.

```
A = A + 1
B = CIRCLE/(POINT/0, 0, 0), PTA, PTB
```

3.2. INSTRUCTIONS DE POSITIONNEMENT DE L'OUTIL.

3.2.1. Sémantique.

Les machines-outils à commande numérique sont commandées en se servant d'un point de contrôle sur l'outil. Pour commander le positionnement de ce point de contrôle (en général le centre du bout de l'outil) on dispose de deux catégories de commandes.

La première, ou commande de positionnement explicite, spécifie une nouvelle position à l'aide de coordonnées absolues ou d'incréments par rapport à la position présente. La seconde catégorie permet un mouvement continu de l'outil le long d'une courbe dans l'espace. Comme l'outil, qui est une surface de révolution, coupe la matière à sa périphérie, il est commode de considérer que l'outil doit rester tangent à deux surfaces appelées surface de pièce et surface de guidage. On spécifie la position de l'outil par rapport à ces surfaces à l'aide des instructions de mouvements continus. Ceci met nécessairement en jeu une direction du mouvement basée sur des instructions ou des déclarations antérieures. Un tel mouvement continu est terminé en spécifiant également une condition de tangence à une troisième surface appelée surface d'arrêt.

Lorsque cette condition de tangence est remplie, le mouvement est terminé.

3.2.2. Syntaxe.

<instruction de positionnement de l'outil> ::=
<instruction de positionnement explicite>|
<instruction de mouvement continu initiale>|
<instruction de mouvement continu intermédiaire>|
<instruction de mouvement continu terminale>

3.3. INSTRUCTIONS DE POSITIONNEMENT EXPLICITE.

3.3.1. Sémantique.

Lorsqu'on connait la position à atteindre par l'outil, on peut le positionner directement sans utiliser les surfaces de contrôle en utilisant les instructions de positionnement explicite.

GOTO positionne l'outil aux coordonnées spécifiées.
GODLTA positionne l'outil en le déplaçant d'un incrément donné à partir de sa position courante.
FROM indique la position initiale de l'outil.

3.3.2. Syntaxe.

<instruction de positionnement explicite> ::=
GOTO/(<expression géométrique>)|
GOTO/<variable géométrique>|GOTO/<paramètre arithmétique>,
<paramètre arithmétique>,<paramètre arithmétique>|
GODLTA/<paramètre arithmétique>,
<paramètre arithmétique>,<paramètre arithmétique>|
FROM/(<expression géométrique>)|FROM/<variable géométrique>|
FROM/<paramètre arithmétique>,<paramètre arithmétique>,
<paramètre arithmétique>
<paramètre arithmétique> ::= (<expression arithmétique>)|
<nombre>|<variable arithmétique>

3.3.3. Exemples.

```
GOTO/3.5, 4.7, .0037
GOTO/(POINT/2.1, 1.2, 0)
GODLTA/0, .333, .6
FROM/1, 2, 3
FROM/PT4
```

3.4. INSTRUCTIONS DE MOUVEMENT CONTINU INITIALES.

3.4.1. Sémantique.

Une instruction spéciale de ce type est nécessaire avant chaque groupe d'instructions de mouvement continu intermédiaires pour positionner l'outil par rapport aux surfaces de pièce et de guidage avec les tolérances données. Cette instruction permet aussi de diriger l'outil vers une zone particulière des surfaces de contrôle.

3.4.2. Syntaxe.

```
<instruction de mouvement continu initiale> ::=
        GO/<expression géométrique adéquate>|
        GO/<expression géométrique adéquate>,
       <expression géométrique adéquate>|
        GO/<expression géométrique adéquate>,
       <expression géométrique adéquate>,
       <expression géométrique adéquate>|OFFSET/
       <expression géométrique adéquate>
<expression géométrique adéquate> ::=
       <indicateur de tangence de l'outil>,
      (<expression géométrique>)|<indicateur de tangence de l'outil>,
       <variable géométrique>| (<expression géométrique>)|
       <variable géométrique>
<indicateur de tangence de l'outil> ::= TO|ON|PAST|TANTO
```

3.4.3. Exemples.

GO/C3

GO/TO, C3

GO/PAST, C2, ON, S7

GO/A, B, C

GO/ON, PLN5, PAST, LIN2, TO, CYL9

3.5. INSTRUCTIONS DE MOUVEMENT CONTINU INTERMEDIAIRES.

3.5.1. Sémantique.

Le but principal du langage APT est de déplacer un outil donné dans une direction spécifiéetout en le maintenant tangent à deux surfaces géométriques. Chaque instruction dépend de l'instruction précédente pour l'établissement de la direction du mouvement et de l'instruction suivante pour la détermination de la surface d'arrêt qui sera la surface de guidage de cette instruction suivante.

3.5.2. Syntaxe.

<instruction de mouvement continu intermédiaire>:: =
<ordre de mouvement continu>/<surface de guidage>
<ordre de mouvement continu> ::= GOLFT|GORGT|GOFWD|GOBACK|GOUP|GODOWN
<surface de guidage> ::= (<expression géométrique>)|
<variable géométrique>

3.5.3. Exemples.

GOLFT/LN2

GOFWD/CIRL

GORGT/(LINE/P1,P2)

3.6. INSTRUCTIONS DE MOUVEMENT CONTINU TERMINALES.

3.6.1. Sémantique.

L'instruction de mouvement continu terminale n'a pas besoin d'instruction suivante pour déterminer la surface d'arrêt. Elle doit donc contenir une référence à une surface d'arrêt explicite et à des conditions de tangence finales. Dans certains cas il peut arriver que la configuration géométrique soit

telle que l'outil satisfasse le critère de tangence à la surface d'arrêt en plus d'un point. Dans ce cas le mouvement se termine lorsque le critère est satisfait pour la première fois. Il existe la possibilité de terminer le mouvement lorsque le critère aura été satisfait pour la n-ième fois.

3.6.2. Syntaxe.

<instruction de mouvement continu terminale> ::=
<instruction de mouvement continu intermédiaire>,
<indicateur de tangence de l'outil>,<surface d'arrêt>|
<instruction de mouvement continu intermédiaire>,
<surface d'arrêt>|
<instruction de mouvement continu intermédiaire>,
<indicateur de tangence de l'outil>,
<numéro d'intersection>, INTOF, <surface d'arrêt>
<surface d'arrêt> ::= (<expression géométrique>)|<variable géométrique>
<numéro d'intersection> ::= <entier sans signe>

3.6.3. Exemples.

GOLFT/LN2, TO, CIR6
GORGT/(LINE/P3,P6),CIRK
GOBACK/LX4361, ON, 4, INTOF,POL6

3.7. INSTRUCTIONS DE CONTROLE.

3.7.1. Sémantique.

Une instruction de contrôle interrompt l'exécution séquentielle normale des instructions.

3.7.2. Syntaxe.

<instruction de contrôle> ::= <instruction de transfert arithmétique>|
<instruction de transfert géométrique>|<instruction terminale>

3.8. INSTRUCTIONS DE TRANSFERT ARITHMETIQUE.

3.8.1. Sémantique.

Une instruction de transfert arithmétique et les instructions associées aux étiquettes mises en jeu dans l'instruction de transfert doivent se trouver dans une boucle ou dans une procédure.

3.8.2. Syntaxe.

<instruction de transfert arithmétique> ::=
JUMPTO/<étiquette>|IF (<expression arithmétique>)
<étiquette>,<étiquette>,<étiquette>

3.8.3. Exemples.

JUMPTO/123XQ
IF(A-X) ST1, ST2, ST3

3.9. INSTRUCTIONS DE TRANSFERT GEOMETRIQUE.

3.9.1. Sémantique.

Une instruction de transfert géométrique interrompt également l'exécution séquentielle d'un programme APT. Elle n'a pas besoin de se trouver dans une boucle ou dans une procédure. Si elle apparait dans une boucle ou dans une procédure elle ne peut mentionner que des instructions apparaissant plus tard dans le programme. Une instruction de mouvement à choix de surface multiple choisit l'une des deux instructions suivantes possibles en fonction de quelle tangence est réalisée en premier.

3.9.2. Syntaxe.

<instruction de transfert géométrique> ::= TRANTO/<étiquette>
<instruction de mouvement à surfaces d'arrêt multiples>
<instruction de mouvement à surfaces d'arrêt multiples> ::=
<instruction de mouvement continu finale>,<étiquette>,
<surface d'arrêt>,<étiquette>|
<instruction de mouvement continu finale>,<étiquette>,
<indicateur de tangence de l'outil>,<surface d'arrêt>,
<étiquette>

3.9.3. Exemple.

```
        TLRGT
        GOFWD/CIRC
        GOFWD/L1,TO,L2,ID1,TO,L3,ID2
   ID1) TLLFT,GOLFT/L2,PAST,L4
        TRANTO/ID3
   ID2) TLLFT,GOLFT/L3,TO,L2
        GOLFT/L2,PAST, L4
   ID3) -----------------
```

3.10. INSTRUCTION TERMINALE.

3.10.1. Sémantique.

Cette instruction est la dernière du programme et à pour effet de terminer le programme.

3.10.2. Syntaxe.

<instruction terminale> ::= FINI

3.11. INSTRUCTIONS DE CONTROLE DU POSTPROCESSEUR.

3.11.1. Sémantique.

Dans le système APT un postprocesseur est un programme qui transforme les informations générales de coordonnées et les fonctions de contrôle et les adapte à un système de commande numérique donné. Les ordres de contrôle du postprocesseur déclenchent des fonctions spéciales de ce postprocesseur ; iis sont différenciés de la chaîne de coordonnées produites par les ordres de mouvement. Les instructions de contrôle du postprocesseur contrôlent des fonctions internes comme la rotation de la broche, le déclenchement et l'arrêt de l'arrosage de refroidissement, etc.

3.11.2. Syntaxe.

<instruction de contrôle du postprocesseur> ::=
<mot de contrôle du postprocesseur>|<mot de contrôle du postproces
<liste de paramètres>|<mot de contrôle du postprocesseur><chaîne>|
<instruction de positionnement de l'outil>,
<indicateur de vitesse d'avance>

<mot de contrôle du postprocesseur> ::= <mot réservé>

<indicateur de vitesse d'avance> ::= (<expression arithmétique>)|
<variable arithmétique>*<nombre>

3.11.3. Exemples.

COOLNT/ON

SPINDL/ON

PPRINTTEXTE ARBITRAIRE

END

3.12. INSTRUCTIONS D'APPEL DE PROCEDURE.

3.12.1. Sémantique.

Une instruction d'appel de procédure est remplacée par le corps de la procédure. Les paramètres formels sont remplacés dans le corps de la procédure par les paramètres normaux spécifiés dans la déclaration de la procédure sauf lorsque l'appel de procédure comporte des paramètres qui sont alors utilisés.

3.12.2. Syntaxe.

<instruction d'appel de procédure> ::= CALL/
<identificateur de procédure>|CALL/
<identificateur de procédure>,
<liste de paramètres de procédure>

<liste de paramètres de procédure> ::= <paramètre formel> =
<paramètre effectif>|<liste de paramètres de procédure>,
<paramètre formel> = <paramètre effectif>

<paramètre effectif >::= <variable>|<nombre>*<mot réservé>|<étiquette>

<paramètre formel> ::= <identificateur>

3.12.3. Exemples.

```
CALL/MACR,A=GORGT,B=CIRCLE,C=QLX,D=7.632
CALL/MXR
```

3.13. INSTRUCTIONS DE CONTROLE D'ENTREE-SORTIE.

3.12.1. Sémantique.

Ces instructions permettent l'entrée et la sortie de procédures, d'éléments géométriques et de valeurs numériques.

3.12.2. Syntaxe.

```
<instruction de contrôle d'entrée-sortie> ::= <mot de contrôle d'entrée-sorti
    <indicateur de format>,<liste entrée-sortie>|TITLES<chaîne>
<mot de contrôle d'entrée-sortie> ::= READ|PUNCH|PRINT
<liste entrée-sortie> ::= ALL|<liste de variables>
<liste de variables> ::=<variable>|<liste de variables>, <variable>
<indicateur de format> ::= 0|1|2|3
```

3.12.3. Exemples.

```
PRINT/3, ALL
PUNCH/1,X,HQC,J
READ/1, A, B, X
TITLES SIN COS TAN
```

4.0. DECLARATIONS.

4.0.1. Sémantique.

Les déclarations définissent des propriétés des éléments géométriques et des quantités arithmétiques utilisés dans le programme. Elles peuvent aussi affecter le système dans lequel certaines instructions sont exécutées.

4.0.2. Syntaxe.

<déclaration> ::= <déclaration de tableau>|
<déclaration de transformation des coordonnées>|
<déclaration de surface Z>|<déclaration de procédure>|
<déclaration d'équivalence de vocabulaire>|
<déclaration de calcul du décalage de l'outil>

4.1. DECLARATIONS DE TABLEAU.

4.1.1. Sémantique.

Une déclaration de tableau spécifie qu'un ou plusieurs identificateurs représentent des tableaux linéaires de quantités ou d'éléments géométriques.

4.1.2. Syntaxe.

<déclaration de tableau> ::= RESERV/<liste de tableau>
<liste de tableau> ::= <segment de tableau>|<liste de tableau> ,
<segment de tableau>
<segment de tableau> ::= <identificateur de tableau>, (<expression arithmétique>)
<identificateur de tableau>,<variable arithmétique>

4.1.3. Exemple.

RESERV/A, 12, B, 26, X, C

4.2. DECLARATIONS DE TRANSFORMATION DES COORDONNEES.

4.2.1. Sémantique.

Cette déclaration créé un système spécial dans lequel on peut définir des variables géométriques. Lorsqu'on réfère à une variable ainsi définie, à partir du système spécial, ses coordonnées sont transformées comme il est

spécifié dans la déclaration de transformation de coordonnées. Le mot NOMORE fait revenir au système normal.

4.2.2. Syntaxe.

<déclaration de transformation des coordonnées> ::=
REFSYS/<matrice de transformation>|REFSYS/NOMORE
<matrice de transformation> ::= <variable géométrique>

4.2.3. Exemples.

REFSYS/M
REFSYS/NOMORE

4.3. DECLARATION DE SURFACE Z.

4.3.1. Sémantique.

Un point est l'un des éléments géométriques qui peuvent être définis dans un programme APT. Au moment de la définition et à moins qu'on ne le définisse explicitement, on suppose que le point se trouve dans le plan contenant les axes X et Y. Cette déclaration permet d'écarter cette hypothèse et déclare que les points se trouveront à partir de là sur le plan spécifié.

4.3.2. Syntaxe.

<déclaration de surface Z> ::= ZSURF/
(<expression géométrique>)|ZSURF/<variable géométrique>

4.3.3. Exemple.

ZSURF/PL2
ZSURF/(PLANE/PNT1, PNT2, PNT3)

4.4. DECLARATION DE PROCEDURE.

4.4.1. Sémantique.

Une déclaration de procédure définit la procédure associée à un identificateur de procédure. Lorsqu'une procédure est appelée par une instruction d'appel de procédure, les identificateurs se trouvant dans le corps de la procédure et qui sont déclarés paramètres formels dans la tête de procédure

sont remplacés par les noms des paramètres effectifs correspondants. A la fin de ce processus, tout paramètre formel non remplacé par un paramètre effectif, est remplacé par le nom normal correspondant au paramètre formel de la tête de procédure.

4.4.2. Syntaxe.

<déclaration de procédure> ::= <tête de procédure><corps de procédure>
<fin de procédure>
<tête de procédure> ::= <identificateur de procédure> = MACRO/
<liste de paramètres formels>
<liste de paramètres formels> ::= <vide>|<paramètre formel>|
<paramètre formel> = <nom normal>|<liste de paramètres formels>,
<liste de paramètres formels>
<nom normal> ::= <nombre>|<mot réservé>|<étiquette>
<corps de procédure> ::= <instruction>|<corps de procédure><instruction>
<fin de procédure> ::= TERMAC
<identificateur de procédure> ::= <identificateur>

4.4.3. Exemple.

```
MAC = MACRO/J = TLLFT, K, Z, H = 0
      FROM/0, 0, 0
      GOTO/1, 1, 1
      GO/SURF1
      J, GOLFT/XURF1
      GORGT/SURF2, K, Z, H
      TERMAC
```

4.5. DECLARATIONS D'EQUIVALENCE DE VOCABULAIRE.

4.5.1. Sémantique.

Cette déclaration permet de rendre un identificateur arbitraire équivalent à un mot réservé APT.

4.5.2. Syntaxe.

<déclaration d'équivalence de vocabulaire> ::= SYN/<liste d'équivalence>
<liste d'équivalences> ::= <identificateur>,<mot réservé>|
<liste d'équivalences>,<identificateur>,<mot réservé>

4.5.3. Exemple.

SYN/GT, GOTO, TT, TANTO

4.6. DECLARATIONS DE CALCUL DU DECALAGE DE L'OUTIL.

4.6.1. Sémantique.

Ces instructions permettent de traiter les instructions de contrôle du décalage de l'outil.

4.6.2. Syntaxe.

<déclaration de calcul du décalage de l'outil> ::=
<déclaration de direction>|<déclaration des paramètres de calcul>|
<déclaration de la position de l'outil>

4.7. DECLARATIONS DE DIRECTION.

4.7.1. Sémantique.

Cette déclaration établit ou rétablit la direction du mouvement de l'outil. On l'utilise principalement pour résoudre les ambiguïtés des commandes de mouvement continu initiales, comme lorsqu'on veut mettre l'outil en position par rapport à un cercle et que la position initiale de l'outil est en fait le centre du cercle. Deux types de déclarations permettent de spécifier la direction suivant un vecteur ou vers un point.

4.7.2. Syntaxe.

<déclaration de direction> ::= INDIRP/<indicateur de direction>|INDIRV/
<indicateur de direction>
<indicateur de direction> ::= (<expression géométrique>)|
<variable géométrique>|<paramètre arithmétique>, <paramètre arithmétique>,
<paramètre arithmétique>

4.7.3. Exemples.

INDIRP/(POINT/1, 1, 1)
INDIRP/1, 1, 1

```
INDIRV/7, 6, 3
INDIRV/(VECTOR/P1, P2)
INDIRV/VECT1
```

4.8. DECLARATIONS DES PARAMETRES DE CALCUL.

4.8.1. Sémantique.

Ces déclarations spécifient les paramètres et les constantes arithmétiques et logiques pour les calculs internes réduisant les instructions de mouvement continu en une suite de coordonnées du centre de l'outil.

4.8.2. Syntaxe.

<déclaration des paramètres de calcul> ::=
<spécification des tolérances>|<spécification de l'outil>|
<contrôle des constantes de calcul>

4.9. SPECIFICATIONS DES TOLERANCES.

4.9.1. Sémantique.

Tous les ordres de mouvement continu sont réduits à une suite de mouvements en ligne droite approchant la trajectoire idéale avec une tolérance donnée. La tolérance interne (INTOL) fait usiner en dessous des cotes et la tolérance externe (OUTTOL) usine une pièce avec des cotes supérieures aux cotes du dessin.

4.9.2. Syntaxe.

<spécification des tolérances> ::= TOLER/<paramètre arithmétique>|
INTOL/<paramètre arithmétique>| OUTTOL/<paramètre arithmétique>

4.9.3. Exemples.

```
TOLER/.01
INTOL/(A/4.621)
OUTTOL/Q
```

4.10. SPECIFICATIONS DE L'OUTIL.

4.10.1. Sémantique.

On définit l'outil, une surface de révolution, par les différentes dimensions et angles de son profil.

4.10.2. Syntaxe.

<spécification de l'outil> ::= CUTTER/<d>|CUTTER/<d>,<r>|
CUTTER/<d>,<r>,<e>,<f>,<α>,<β>,<h>

<d> ::= paramètre arithmétique

<r> ::= paramètre arithmétique

<e> ::= paramètre arithmétique

<f> ::= paramètre arithmétique

<α> ::= paramètre arithmétique

<β> ::= paramètre arithmétique

<h> ::= paramètre arithmétique

4.10.3. Exemples.

CUTTER/1.02

CUTTER/6, 4.3, 7.2, 9, C, (C+0.7), A

4.11. CONTROLE DES CONSTANTES DE CALCUL.

4.11.1. Sémantique.

Deux grandes catégories de déclarations contrôlent les constantes arithmétiques et logiques utilisées dans les calculs du décalage de l'outil. CUT et DNTCUT provoque ou empêche respectivement la transmission au postprocesseur des coordonnées résultant des instructions de positionnement de l'outil.

MULTAX fait transmettre au postprocesseur les cosinus directeurs de l'axe de l'outil.

TLAXIS spécifie les cosinus directeurs de l'axe de l'outil. Le mode NORMPS spécifie que l'axe de l'outil doit être en tout point normal à la sur-

face de pièce.

MAXDP spécifie la longueur du pas maximum permis dans les calculs du décalage de l'outil.

NUMPTS spécifie le nombre maximum de points qu'un ordre de mouvement continu peut produire.

THICK ajoute une surépaisseur uniforme aux surfaces de contrôle de l'outil.

PSIS spécifie une nouvelle surface de pièce.

4.11.2. Syntaxe.

<contrôle des constantes de calcul> :=
<contrôle logique des calculs>|
<contrôle des constantes arithmétiques de calcul>
<contrôle logique des calculs> ::= CUT|DNTCUT|MULTAX|3DCALC|2DCALC|NDTEST|
NOPS
<contrôle des constantes arithmétiques de calcul> ::=
TLAXIS/<paramètre arithmétique>,<paramètre arithmétique>,
<paramètre arithmétique>| TLAXIS/(<expression géométrique>)|TLAXIS/
<variable géométrique>|TLAXIS/NORMPS|MAXDP/
<paramètre arithmétique>|NUMPTS/<paramètre arithmétique>|
THICK/<paramètre arithmétique>,<paramètre arithmétique>,
<paramètre arithmétique>|PSIS/(<expression géométrique>)| PSIS/
<variable géométrique>

4.11.3. Exemples.

```
CUT
DNTCUT
TLAXIS/(VECTOR/PT1,PT2)
MAXDP/10
THICK/0, 0, 0.2
```

4.12. DECLARATIONS DE POSITION DE L'OUTIL.

4.12.1. Sémantique.

On utilise la déclaration de position de l'outil à la place d'un indicateur de tangence dans les instructions de mouvement continu intermédiaires. On applique les règles suivantes :

Position de l'outil	Mouvement	Indicateur de tangence
TLLFT	GOLFT	TO
TLLFT	GORGT	PAST
TLRGT	GOLFT	PAST
TLRGT	GORGT	TO
TLON	GOLFT	ON
TLON	GORGT	ON
tous	GOFWD	TANTO
tous	GOBACK	TANTO
tous	GOUP	erreur
tous	GODOWN	erreur

Pour un mouvement et une position d'outil spécifiés dans l'instruction n, l'indicateur de tangence réfère à la surface d'arrêt de l'instruction n-1.

4.12.2. Syntaxe.

<déclaration de position de l'outil> ::=
<indicateur de position de l'outil>|<indicateur de position de l'outil>,
<instruction de mouvement continu>|<indicateur de position de l'outil>,
<déclaration de position de l'outil>

<indicateur de position de l'outil> ::= TLONPS|TLOFPS|TLNDON|TANCRV|TLON|
TLLFT|TLRGT

<instruction de mouvement continu> ::=
<instruction de mouvement continu intermédiaire>|
<instruction de mouvement continu terminale>

BIBLIOGRAPHIE

1. ACKLEY, D.A. — *Postprocessor for the Seventies. NC Managements's Key to the Seventies. Proceedings of the 7th annual meeting and technical conference of the Numerical Control Society. Numerical Control Society, 1970.*

2. ADEPA — *IFAPT Manuel de référence. Association pour le développement du système de programmation automatique IFAPT, 1969.*

3. ADEPA — *IFAPT Manuel de référence CLDATA. Association pour le développement du système unifié de programmation automatique IFAPT, 1970.*

4. AFNOR — *Codes pour la commande numérique des machines. NF Z 68-010.*

5. AFNOR — *Nomenclature des axes et des mouvements pour la commande numérique des machines. NF Z 68-020.*

6. AFNOR — *Codage des fonctions préparatoires G et auxiliaires M. NF Z 68-030.*

7. AFNOR — *Bandes perforées interchangeables à bloc à format variable pour mise en position et usinage parallèle aux axes sur machines à commande numérique. NF Z 68-031.*

8. AFNOR — *Bandes perforées à bloc à format variable pour mise en position et usinage parallèle aux axes sur machines à commande numérique NF Z 68-032.*

9. AFNOR — *Bandes perforées à bloc à format fixe pour mise en position et usinage parallèle aux axes sur machines à commande numérique. NF Z 68-033.*

10. ALBERT, J. *An adaptable NC programming system.* *NC Management's Key to the Seventies.* *Proceedings of the 7 th annual meeting and technical conference of the Numerical Control Society. Numerical Control Society, 1970.*

11. APT Long Range Program Staff, *IIT Research Institute.* *APT Part Programming.* *Mc Graw Hill Book Company, 1967.*

12. ASA *Fortran versus basic Fortran* *Communications of the ACM.* *Vol. 7, N° 10, October, 1964.*

13. ASA *Appendixes to ASA FORTRANs.* *Communications of the ACM.* *Vol. 8, N° 5, May, 1965.*

14. AUROUX, A., HANS, C. *Le concept de machines virtuelles.* *Revue Française d'Informatique et de Recherche Opérationnelle* *N° 15, 1968.*

15. BEZIER, P. *Emploi des Machines à Commande Numérique* *Masson et Cie, Eyrolles, 1970.*

16. BROWN, S.A., DRAYTON, C.E., MITTMAN, B. *A description of the APT language.* *Communications of the ACM.* *Vol. 6, N° 11, November, 1963.*

17. CARACCIOLO DI FORINO, A. *The formal definition of machine tool languages.* *Proceedings of the PROLAMAT conference.* *North Holland Publishing Company. Amsterdam, 1969.*

18. DEAM, J.D. *The mini computer approach to direct numerical control.* *NC Management's Key to the Seventies.* *Proceedings of the 7th annual meeting and technical conference of the Numerical Control Society. Numerical Control Society, 1970.*

19. DOANE, R.C. *Direct Numerical Control using small computers.* *NC Management's Key to the Seventies.* *Proceedings of the*

7th annual meeting and technical conference of the Numerical Control Society, Numerical Control Society, 1970.

20. DRAYTON, C.E., GRAY, H., HOPEWELL, P., LITTLE, R.N. *Automatically Programmed Tools.* Proceedings of the PROLAMAT conference. *North Holland Publishing Company. Amsterdam, 1969.*

21. GABRINI, P. *Etude d'un système de programmation en commande numérique.* Automatisme. *n° 4, avril, 1970.*

22 GABRINI, P. *Introduction aux langages de programmation des systèmes de commande numérique pour machines-outils.* Revue Française d'Informatique et de Recherche Opérationnelle. *B2, août, 1970.*

23. GABRINI, P. *Réalisation d'un système pour commande numérique sur petits calculateurs.* Congrès D'Informatique AFCET-SICOB. *Paris, septembre, 1970.*

24. GABRINI, P. *Realization of an APT-like numerical control programming system for small computers.* International Symposium on digital computer applications in engineering sciences. *Istanbul, October, 1970.*

25. HOUSTON, G.E. *DNC A user tells it like it is.* NC Management's Key to the Seventies. *Proceedings of the 7th annual meeting and technical conference of the Numerical Control Society. Numerical Control Society, 1970.*

26. IBM *IBM 1130/1800* Basic FORTRAN IV Language. *Form C26-3715-2.*

27. IBM IBM 1130 Disk Monitor System, version 2, Programming and operations guide. *Form C26-3717-4.*

28. IBM — *NC 360 APT Application Description. Form H 20-0181-1.*

29. IBM — *Program Library Numerical Control Graphics. 360 D. 23. 4. 002.*

30. IBM — *NC 360 AUTOSPOT Application Description. Form H 20-0179-1.*

31. IBM — *NC 360 ADAPT Application Description. Form H 20-0180-0.*

32. ISO — *USA Working Paper. APT Language proposed draft proposed standard. USA Standards Institute Task Group X3.4.7., ISO/TC97/SC5/WG1(USA-4)37.*

33. ISO — *Provisional CLDATA Specification. British Standards Institution, ISO/TC97/SC5/WG1(UK15)117.*

34. ISO — *German proposal for an ISO Recommendation. EXAPT1 Programming Language for numerically controlled drilling and boring machines. ISO/TC97/SC5/WG1 (Germany 12) 183, 1967.*

35. KOTSAFTIS, C.J. — *A new APT compiler for a small computer. NC Management's Key to the Seventies. Proceedings of the 7th annual meeting and technical conference of the Numerical Control Society. Numerical Control Society, 1970.*

36. LE BRUSQUE, R. — *Mesures pour l'automatisation des machines-outils. Ingénieurs et Techniciens. n° 225, novembre, 1968.*

37. LOMBARD, J. — *Introduction à la commande numérique des machines-outils. Mécanique Electricité. n° 222-223, juin-juillet, 1968.*

38. MANGOLD, W.E. — *Status of N.C. language standardization in I.S.O. Proceedings of the PROLAMAT conference. North Holland Publishing Company, Amsterdam, 1969.*

39. MARSTON, A.D. *Computerized NC System. NC Management's Key to the Seventies. Proceedings of the 7 th annual meeting and technical conference of the Numerical Control Society. Numerical Control Society, 1970.*

40. MARTIN, M. *Le système de fabrication programmée MECA. Séminaire de programmation de l'IMAG. 17 Mai, 1968.*

41. MÜLLER-TRAUT Hans, OPFERKUCH Heinz. *Generalisierter AUTOSPOT-Postprocessor für numerisch gesteuerte Werkzeugmaschinen. IBM Nachrichten. Heft 173. August 1965.*

42. NEL *2CL Part Programming Reference Manual. National Engineering Laboratory. East Killbride, Scotland. 1967.*

43. NITKIEWICZ, J., WALICKI, A.J. *Computer controlled grinding machines. NC Management's Key to the Seventies. Proceedings of the 7th annual meeting and technical conference of the Numerical Control Society, Numerical Control Society, 1970.*

44. NUSSEY, I.D. *The purpose and future of part programming. Proceedings of the PROLAMAT conference. North Holland Publishing Company. Amsterdam, 1969.*

45. OPITZ, H., BERGER, H., BUDDE, W., ENGELSKIRCHEN, W.-H. *Calculation of technological data for the computer assisted programming of numerically controlled machine tools. Proceedings of the PROLAMAT conference. North Holland Publishing Company. Amsterdam, 1969.*

46. RECKZIEGEL, D., GRUPPE, U. *The programming system of EXAPT. Proceedings of the PROLAMAT conference. North Holland Publishing Company. Amsterdam, 1969.*

47. SCHULMAN, M. *A new computerized monitoring system for the NC machine shop.* *NC Management's Key to the Seventies.* *Proceedings of the 7th annual meeting and technical conference of the Numerical Control Society. Numerical Control Society, 1970.*

48. SIMON, W. *Commande numérique des machines-outils. Eyrolles, 1967.*

49. SOUBIES-CAMY. *Les bases de la programmation en commande numérique.* *Mécanique Electricité. n° 222-223, juin-juillet, 1968.*

50. THILLIEZ, J. *La commande numérique des machines. Dunod, 1967.*

51. WILSON, C.W. *NUMEPS. USASI/X3.4.7. Working Paper CWW/11.*

52. WRIGHT, J.H. *UNIAPT is APT on the PDP-8 computer.* *NC Management's Key to the Seventies. Proceedings of the 7th annual meeting and technical conference of the Numerical Control Society. Numerical Control Society, 1970.*

53. MOREL, J. *Bibliothèque MECALGOL. Communication privée.*

VU,

Grenoble, le

Le Président de la Thèse

Vu,
Grenoble, le

Le Doyen de la Faculté des Sciences

Vu, et permis d'imprimer

Le Recteur de l'Académie
de Grenoble