

Contribution à la perception visuelle multi-résolution de l'environnement 3D : application à la robotique autonome

Hossam Fraihat

▶ **To cite this version:**

Hossam Fraihat. Contribution à la perception visuelle multi-résolution de l'environnement 3D : application à la robotique autonome. Traitement du signal et de l'image [eess.SP]. Université Paris-Est, 2017. Français. NNT : 2017PESC1065 . tel-01792544

HAL Id: tel-01792544
https://theses.hal.science/tel-01792544

Submitted on 15 May 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UNIVERSITÉ PARIS-EST

Thèse

Présentée pour l'obtention du titre de

DOCTEUR DE L'UNIVERSITÉ PARIS-EST

Spécialité: Signal, Images et Automatique

Par **Hossam FRAIHAT**

Contribution à la perception visuelle multi-résolution de l'environnement 3D : Application à la Robotique Autonome

Soutenue publiquement le 19 décembre 2017 devant la commission d'examen composée de

Prof.	Hichem	MAAREF	Rapporteur / Université d'Evry-Val d'Essonne
Prof.	Samir	BOUAZIZ	Rapporteur / Université PARIS-SUD
Dr.	Mehdi	EBADZADEH	Examinateur / Amir-Kabir University (Téhéran)
Dr.	Christophe	SABOURIN	Examinateur / Université PARIS-EST Créteil – LISSI
Prof.	Kurosh	MADANI	Directeur de thèse / Université PARIS-EST Créteil – LISSI

Remerciements

J'aimerais exprimer ma plus profonde gratitude au professeur Kurosh MADANI pour son soutien professionnel et moral, sa patience et ses conseils professionnels et aussi personnels. C'est lui qui m'a donné la chance de suivre mes études en doctorat. J'ai appris beaucoup de chose grâce à lui (la façon de chercher l'information et choisir la direction qui m'amène à mon objectif final). Je compte avec un grand plaisir poursuivre mes travaux avec lui dans le futur.

Je tiens également à remercier mon tuteur superviseur, Christophe SABOURIN, Maître de Conférences au LISSI qui m'a apporté une aide précieuse dans le choix des directions et des outils appropriés pour atteindre mon objectif.

Et j'aimerais également remercier les membres du corps professoral de l'IUT de Sénart, en particulier Prof. Yacine AMIRAT, Directeur du laboratoire, Dr. Amine CHOHRA et Dr. Aurélien HAZAN. Mes plus sincères remerciements vont également aux membres du LISSI et spécialement à Monsieur Viachaslau KACHURKA pour son aide.

Je voudrais exprimer ma reconnaissance aux membres de mon Jury de Thèse et plus particulièrement aux Prof. Hichem MAAREF et Prof. Samir BOUAZIZ, les deux rapporteurs de ma thèse, pour avoir accepté de consacrer leur temps précieux et d'évaluer mes travaux de recherche.

Et j'aimerais remercier mes parents, mon père Jawdat et ma mère Husnieh ainsi que mon frère Salam. Ce sont mes premiers maitres dans l'apprentissage de la vie et je suis toujours fier d'eux. Sans leur soutien et leur encouragement, je n'aurais pas pu aller aussi loin.

Enfin, j'aimerais remercier ma future femme Batool pour sa patience et son soutien moral ainsi que tous mes amis.

Table Des Matières

REMERCIEMENT **2**

TABLE DES MATIERES **3**

LISTE DES FIGURES **5**

LISTE DES TABLES **8**

LISTE DES SYMBOLES **9**

INTRODUCTION GENERALE **11**

CHAPITRE 1.ETAT DE L'ART **14**

1.1 INTRODUCTION 14
1.2 LA PERCEPTION VISUALELLE DE L'ENVIRONNEMENT 14
1.3 LES TECHNOLOGIES D'ACQUISITION DE LA PROFONDEUR 15
1.3.1 Les technologies passives *16*
1.3.2 Les technologies Actives *19*
1.3.2.1 La caméra temps de vol TOF 20
1.3.2.2 La caméra 3D Kinect 22
1.3.2.3 La caméra de profondeur ASUS 30
1.3.3 Résumé des caractéristiques des technologies *32*
1.4 LA SAILLANCE 34
1.4.1 Aperçu des techniques existantes dans la saillance 2D *35*
1.4.2 Aperçu des techniques existantes dans la saillance 3D *37*
1.5 CONCLUSION 40

CHAPITRE 2. MODELISATION DE LA SAILLANCE VISUELLE EN 3D **42**

2.1 INTRODUCTION 42
2.2 NOTRE APPROCHE DE LA DETECTION DE LA SAILLANCE EN 3D. 43
2.2.1 La détection de la saillance 2D en couleur et lumière *45*
2.2.2 La détection de la saillance en profondeur *47*
2.2.3 Génération de la carte et le masque de saillance final *51*
2.2.4 Résultat expérimental et validation *53*
2.3 CONCLUSION 58

CHAPITRE 3. EXTENSION ET VALIDATION DU SYSTEME DE PERCEPTION DE LA SAILLANCE VISUELLE 3D PAR UN SYSTEME ROBOTIQUE AUTONOME **61**

3.1 INTRODUCTION 61
3.2 CARACTERISATION DE L'ENVIRONNEMENT UTILISANT LE SOFT COMPUTING 63
3.2.1 Introduction *63*
3.2.2 L'approche expérimentale *64*
3.2.2.1 Capture d'images couleur et profondeur 2D 65
3.2.2.2 Le prétraitement conventionnel des images émises par Kinect 68
3.2.2.3 Soft Computing : Mode Apprentissage et Généralisation 69
3.2.3 Résultat expérimental *71*
3.2.4 Comparaisons d'ANFIS avec d'autres algorithmes d'apprentissage *74*
3.2.5 Discussion et conclusion *74*
3.3 EXTENSION DE NOTRE APPROCHE SUR LE ROBOT PEPPER 76

3.3.1 Introduction *77*
3.3.2 Mise en place de l'approche expérimentale *77*
3.3.2.1 Capture des images *77*
3.3.2.2 Prétraitement des images de la base de données 81
3.3.2.3 Soft Computing: apprentissage et généralisation du model ANFIS 82
3.3.3 Analyse des résultats d'étalonnage *83*
3.3.4 Résultat expérimental de la validation de notre approche avec Pepper *87*
3.3.5 Discussion *89*
3.4 LA CARACTERISATION QUALIFICATIVE 90
3.4.1 La reconnaissance de l'objet *91*
3.4.2 La caractérisation et la localisation de l'objet dans la scène *91*
3.4.3 La réaction du robot dans un environnement réel *96*
3.5 CONCLUSION 101

CONCLUSION GENERALE 102

CONCLUSION 102
PERSPECTIVE 103

PUBLICATIONS 104

ANNEXES 105

ANNEXE1 105
1.1 MLP *105*
1.2 SVR *107*
1.3 Interpolation bilinéaire *110*
ANNEXE2 111
2.1 La fuzzification *111*
2.1.1 La fonction d'appartenance 111
2.1.2 Les différentes formes des fonctions d'appartenance 111
2.2 Les méthodes d'inférences floues *114*
2.2.1 Inférence floue de type Mamdani 115
2.2.2 Inférence floue de type Sugeno 116
2.3 Défuzzification *116*

BIBLIOGRAPHIES 118

RESUME 125

Liste des Figures

Figure 1: Un exemple de perception visuelle (Velmans 2003) 15
Figure 2: Taxinomie des méthodes de mesure de la profondeur 16
Figure 3: Le fonctionnement général de la stéréovision 17
Figure 4: Tsukuba , (a) Gauche (b) Droite (c) carte de disparité (Scharstein et al. 2002) 17
Figure 5: Système de triangulation 18
Figure 6: Camera temps de vol (Gut 2004) 20
Figure 7: Mesure par impulsion et par signal modulé 21
Figure 8: Le principe de la méthode de décalage de phase (Foix et al. 2011) 21
Figure 9: Le capteur Kinect de Microsoft (Lejeune et al. 2012) 22
Figure 10: Images couleur et profondeur capturées respectivement par la camera 2D et IR de la Kinect. 23
Figure 11: Calcul de l'ordonnée *yr* d'un pixel 25
Figure 12: Les coordonnées du système d'étalonnage de la caméra 27
Figure 13: Objet d'étalonnage 'une mire' 28
Figure 14: Une image RGB et une carte de profondeur illustrent le problème de l'encombrement des objets. 29
Figure 15: La position du capteur de la profondeur ASUS dans le robot Pepper (SoftBank 2017) 30
Figure 16 : Une image RGB et une carte de profondeur illustrent le problème des surfaces lisses. 32
Figure 17: L'architecture générale du modèle d'Itti, Koch & Ulmann 1998. Extrait de (Itti et al. 1998) 36
Figure 18: Exemple de résultats du calcul de saillance visuelle d'après (Cheng et al. 2015) .En haut, image originale; au centre, carte de saillance; en bas, masque. 36
Figure 19: Schéma bloc de la méthode de fusion (Desingh et al. 2013). 38
Figure 20: Exemples de détection de la saillance (Cheng et al. 2014). 39
Figure 21 : Schéma bloc de la méthode de la détection de la saillance (Peng et al. 2014). 39
Figure 22 : Comparaisons quantitatives de l'approche proposée avec 8 approches RGBD concurrentes.(Peng et al. 2014) 40
Figure 23: Schéma de principe du système proposé 44
Figure 24 : Le principe de l'algorithme de segmentation MEAN SHIFT (Yilmaz et al. 2006) 48
Figure 25: La segmentation par l'algorithme Mean-shift, a) image couleur, b) image profondeur et c) la segmentation. 48
Figure 26 : L'histogramme d'une région H_K dans l'image de profondeur 49
Figure 27: L'histogramme d'une région H_j dans l'image de profondeur 49
Figure 28: Exemple de CLSM et DSM obtenus à partir d'images RGB et de profondeur fournie par la Kinect. 50
Figure 29: Exemples de FSM (à gauche) et le masque d'extraction de la saillance (à droit). 52
Figure 30: Exemples de courbe ROC et la zone correspondante Areas Under Curve. 53
Figure 31: Les échantillons des résultats obtenus, illustrant : (a) les images RGB. (b) les images de profondeur. (c) les cartes de saillance finale correspondantes. (d) les masques extraites. (e) les éléments ciblent extraits. (f) Ground-true. 55
Figure 32 (suite): Les échantillons des résultats obtenus, illustrant : (a) les images RGB. (b) les images de profondeur. (c) les cartes de saillance finale correspondantes. (d) les masques extraits. (e) les éléments cibles extraits. (f) Ground-true 56
Figure 33: Diagramme de la méthode d'évaluation 57
Figure 34: Comparaisons quantitatives de notre approche avec 9 approches concurrentes pour la détection de la saillance en 3D. 58

Figure 35: Schéma bloc de l'extension de la saillance 3D, implanté sur un capteur 3D 62
Figure 36: Schéma général de l'approche expérimentale proposée pour l'extension de notre approche de la détection de la saillance 3D. 63
Figure 37: Diagramme blocs de l'approche proposée. 65
Figure 38: Etalonnage de la camera couleur et profondeur : a) image couleur, b) image infrarouge, c) image de profondeur 66
Figure 39: Schéma expérimental 67
Figure 40: Exemple d'images capturées pour deux objets donnés avec une forme simple (régulière).. 68
Figure 41: Exemple d'images capturées pour deux objets donnés avec une forme plus complexe (irrégulière). 69
Figure 42: La ligne 1 représente la distance minimale entre deux objets. 69
Figure 43: Architecture d'ANFIS 70
Figure 44: Erreur d'estimation de la distance en mode apprentissage: la base de données2 contenant des objets en forme irrégulière a été utilisée pour l'apprentissage. 72
Figure 45: Erreur d'estimation de distance en mode de généralisation: la base de données1 contenant des objets en forme régulière a été utilisée pour le test. 72
Figure 46: Erreur d'estimation de distance en mode apprentissage: la base de données2 et 50% de la base de données 1 ont été utilisés pour l'apprentissage. 73
Figure 47: Erreur d'estimation de la distance dans la généralisation: 50% de la base de données de repos 1 ont été utilisés pour le test. 73
Figure 48: Erreur de calcul de la distance en utilisant une approche géométrique, la base de données1 a été utilisée pour le test 74
Figure 49: Architecture du protocole proposé par FRAD(Frad 2016) 76
Figure50: Diagramme bloc de l'approche proposée, flux opérationnel montrant le mode d'apprentissage (En haut) et le mode de généralisation (En bas). 78
Figure 51: La méthode de capture des images de profondeur en (mm) : plan observé au niveau de la zone en pointillés (a). Schéma de principe de la manipulation (b). 79
Figure 52: Taux d'erreur d'estimation de la profondeur (Pagliari et al. 2015) 80
Figure 53: Image de profondeur prise en mode de vision nocturne à l'aide du capteur IR de la Kinect. 80
Figure 54: La méthode de capture des images de profondeur (en échelle de gris): plan observé au niveau de la zone en pointillés (a). Schéma de principe de la manipulation (b). 81
Figure 55: (a) Image de couleur: mur plan, (b) image de profondeur, (c) image de profondeur lissée avec le Filtre Médian. 82
Figure 56: La scène vue par le robot Pepper 83
Figure 57: Pour la surface B et pour une profondeur MAX P = 2m, l'erreur d'estimation de la profondeur en mode apprentissage: ¾ de la base de données a été utilisés pour l'apprentissage 83
Figure 58 : Pour la surface B et pour une profondeur Max P = 2m, l'erreur d'estimation de la profondeur en mode généralisation: 1/4 de la base de données a été utilisé pour le test. 84
Figure 59 : Pour la surface A et pour une profondeur Max P = 2m, l'erreur d'estimation de la profondeur en mode apprentissage: ¾ de la base de donnée a été utilisé pour l'apprentissage. 84
Figure 60: Pour la surface A et pour une profondeur Max P = 2m, l'erreur d'estimation de la profondeur en mode apprentissage: ¾ de la base de donnée a été utilisé pour l'apprentissage. 84
Figure 61: Pour la surface B et pour une profondeur Max P = 3.5m, l'erreur d'estimation de la profondeur en mode apprentissage: ¾ de la base de donnée a été utilisés pour l'apprentissage 85
Figure 62: Pour la surface B et pour une profondeur Max P = 3.5m, l'erreur d'estimation de la profondeur en mode généralisation: 1/4 de la base de donnée a été utilisé pour le test. 86
Figure 63: Pour la surface A et pour une profondeur Max P = 3.5m, l'erreur d'estimation de la profondeur en mode apprentissage: ¾ de la base de donnée a été utilisé pour l'apprentissage. 86
Figure 64: Pour la surface A et pour une profondeur Max P = 3.5m, l'erreur d'estimation de la profondeur en mode généralisation: 1/4 de la base de donnée a été utilisé pour le test. 86

Figure 65: Exemples de CLSM, DSM et FSM obtenus à partir d'images RGB et de profondeur fournie par le capteur ASUS de Pepper. 88
Figure 66: Les échantillons des résultats obtenus 89
Figure 67: Schéma global de l'implémentation de l'approche de la détection de la saillance 3D. 92
Figure 68: Objet 'Cornflakes' détecté et cadré en couleur vert par l'algorithme SURF 93
Figure 69: Profondeur des objets 94
Figure 70: Largeur des objets 94
Figure 71: Hauteur des objets 95
Figure 72: Positionnement horizontal des objets 95
Figure 73: Positionnement vertical des objets 95
Figure 74: Les différents aspects du comportement du système perceptuel du robot. 96
Figure 75: Deux scénarios d'exploration de l'environnement. 97
Figure 76: Les objets reconnus dans différents vues 98
Figure 77: Les résultats de la détection de la saillance 3D. 99
Figure 78: L'architecture de MLP avec une couche cachée 105
Figure 79.a : Réglage de la fonction de perte dans le cas d'une SVM linéaire 108
Figure 80: Le diagramme schématique de l'Algorithme d'interpolation bilinéaire. 110
Figure 81: Fonction d'appartenance triangulaire (Ayouni 2012). 112
Figure 82: Fonction d'appartenance trapézoïdale (Ayouni 2012). 113
Figure 83: Fonction d'appartenance gaussienne(Ayouni 2012). 113
Figure 84: Fonction d'appartenance singleton 114
Figure 85: Une partie d'une base des règles sous forme de matrice 115
Figure 86: Méthode du centre de la gravité (Pagliari et al. 2015) 116

Liste des Tables

Tableau 1: Caractéristiques techniques de Kinect 26
Tableau 2 Les caractéristiques du robot Pepper (SoftBank 2017) 31
Tableau 3: Résumé des caractéristiques des technologies 33
Tableau 4: Pourcentage de détections correctes de l'objet sur le test de l'ensemble d'images à l'aide du cadre de détection Viola-Jones 54
Tableau 5: Résultat de l'expérience pour différentes algorithmes d'apprentissage. 74
Tableau 6: Erreur d'estimation de la profondeur fournis par le capteur infrarouge du robot Pepper 87
Tableau 7: Bilan temporel de notre approche 100

Liste des Symboles

D	La distance qui représente la profondeur réel des objets par rapport Kinect.
f	La distance focale de la Kinect.
x_i	L'abscisse du pixel dans l'image.
y_i	L'ordonnée du pixel dans l'image.
x_r	L'abscisse réelle d'un point d'un objet.
y_r	L'ordonnée réelle d'un point d'un objet.
k	La matrice des paramètres intrinsèques de la caméra de profondeur.
(c_u, c_v)	Les coordonnées de la projection du centre optique de la caméra sur le plan image.
(f_u, f_v)	Les distances focales des deux cameras couleur et profondeur.
(X, Y, Z)	Le repère monde.
(u_k, v_k)	Les coordonnées d'un point m_k dans le repère de l'image de la camera de profondeur.
(x_k, y_k, z_k)	Les coordonnées d'un point 3D observé.
T	La distance entre deux cameras.
d	La disparité.
RGB-D	Red Green Blue-Depth
CLSM	Color-luminance Saliency Map
LSM	Local Saliency Map
GSM	Global Saliency Map
DSM	Depth Saliency Map.
FSM	Final Saliency Map
I_{YCC}	L'image représentée dans l'espace couleur YCrCb.
$I_{RGB}(x)$	L'image dans l'espace couleur RGB.
$M_G(x)$	La carte de saillance globale.
My	Carte de luminance
M_{CrCb}	Carte de chrominance
Iy	Intensité de luminance d'un pixel
I_{Cr} I_{Cb}	Chromaticité dans la représentation *Ycc*.
I_R,I_G,I_B	Chromaticité dans la représentation RGB

$\overline{I_y}, \overline{I_{Cr}}, \overline{I_{Cb}}$	Valeurs moyennes dans les canaux Y, Cr et Cb.
C(x)	Un coefficient dépend de la saturation de chaque pixel dans l'espace couleur RGB.
P(x)	Une fenêtre coulissante de taille *P*.
Hc(x)	L'histogramme central dans la fenêtre *P(x)*.
Q(x)	Une autre fenêtre coulissante de taille *Q*.
Hs(x)	L'histogramme central dans la fenêtre *P(x)*.
d_{ch}(x)	La somme de différence à travers tous les 256 histogrammes bin.
$C_u(x)$	Coefficient représente la saturation moyenne en couleur d'une fenêtre *P(x)*.
m (x)	L'emplacement moyen au point x.
C_k	Le contraste représentatif d'une région d'une image segmentée.
C_{max}	Le contraste le plus élevé dans une image.
Z_k	La profondeur métrique du centre d'une région (fournie directement par Kinect).
n_k	Le nombre de pixels d'une région.
H_k	L'histogramme des intensités de pixels d'une région.
D_{kj}	Le produit de l'histogramme H_k et les histogrammes H_j.
S_k	La saillance en profondeur pour une région 'k'.
TP	True-Positive.
FP	False-Positive.
FN	False-négative.
P	La précision.
R	Le Rappel.
AUC	Area Under Curve.
ROC	Receiver Operating Characteristic curve.
f_i	L'inférence flou selon la sortie souhaitée.
$A_i,\ B_i$	Etiquettes d'ensembles flous caractérisés par une fonction d'appartenance appropriée.
μ_{Ai}(x)	La fonction d'appartenance de A_i.
$O_{k,i}$	La fonction nœud, où *k* est le nombre de la couche et *i* est la position du nœud dans la couche.
$\{p_i,\ q_i,\ r_i\}$	Des paramètres set.

Introduction générale

Motivation et objectives

Le travail décrit et les résultats présentés dans ce manuscrit relèvent du domaine du développement d'un système autonome de perception visuelle de l'environnement, destiné à la robotique d'assistance (robots compagnons). Pour être qualifié "d'autonome", un robot doit pouvoir développer sa propre connaissance à partir de la perception de l'environnement dans lequel il est supposé évoluer en utilisant les données sensorielles de bas niveau (images couleurs RGB, information sur la profondeur, etc..). Dans ce contexte, il est donc primordial d'extraire des informations pertinentes de la multitude des données sensorielles perçues.

Concernant la perception visuelle, l'autonomie précitée requière la capacité de la machine (notamment du robot) d'extraire des objets saillants de son environnement. Si une part conséquente des travaux relatifs à saillance visuelle a concerné la prédiction (détermination) des endroits où les êtres humains focalisent leurs regards (ou des objets attirant l'attention visuelle de ces derniers), une autre part importante des travaux a concerné l'extraction des objets saillants du fond de l'image. En effet, la détection d'objets saillants consiste à séparer des objets qualifiés plus pertinents de ceux relevant du « fond de l'image ». Récemment, la détection d'objets saillants à partir d'images RGB-D a suscité un regain d'intérêt des chercheurs en raison de la démocratisation d'une nouvelle génération de technologie de capteurs, telles que la Kinect (Microsoft) ou des capteurs 3D. L'objectif de cette thèse, est une contribution au développement d'un système de détection de la saillance sur la base d'une perception 3D. La motivation du développement de la saillance 3D repose sur le fait qu'elle permet de réduire la complexité computationnelle inhérente à la vision 3D en réduisant celle-ci au niveau des tâches de calculs 2D. Dans cette thèse, nous avons proposé une stratégie fusionnant un modèle de détection de la saillance 2D d'une image RGB avec celui de la détection de la saillance d'une image issue du capteur de profondeur. Plus précisément, la saillance induite par la profondeur est produite par la méthode de calcul des histogrammes, qui calcule le contraste en profondeur d'une région segmentée à partir de sa profondeur par rapport aux autres objets dans la scène.

Contribution

Le travail accompli dans cette thèse a permis d'apporter plusieurs contributions à la conception d'un système de perception artificiel autonome:

- Une étude des technologies 3D disponibles pour l'acquisition d'informations visuelles et une analyse de ces dernières au regard des technologies de vision 3D. En particulier, deux capteurs de l'acquisition de l'information de la profondeur ont été analysés : Kinect de Microsoft et le capteur Xtion équipant le robot Pepper.
- Une étude des techniques de la détection de la saillance visuelle a été menée. Elle a concerné deux aspects : la détection de saillance dans une image RGB et la détection de saillance dans une image de profondeur.
- L'étude, la conception et la réalisation d'un système basé sur la détection de la saillance en 3D. L'évaluation expérimentale du système proposé et l'extension de l'approche proposée au capteur Xtion équipant le robot Pepper.
- La proposition d'une approche pour convertir les informations visuelles du capteur infrarouge du robot Pepper en mesure de la profondeur des objets saillants détectés, exprimée en centimètres.
- Finalement, l'évaluation expérimentale, dans un environnement intérieur réel, du concept proposé équipant un robot Pepper.

Organisation de la thèse

Cette thèse est construite autour de trois chapitres fondamentaux.

Le premier chapitre concerne la présentation de l'état de l'art. Il permet d'introduire, d'une part, les différentes technologies d'acquisition de l'information de la profondeur, et d'autre part, il traite des travaux dans le domaine de la détection de la saillance visuelle en 2D et en 3D. Ce chapitre donne au lecteur un aperçu général des techniques existantes et de la terminologie sur laquelle nous développons encore nos propres recherches décrites dans les chapitres suivant.

Dans le deuxième chapitre, les fondements théoriques de notre travail sont présentés. Une architecture est conçue pour faire face aux problèmes et aux objectifs qui ont été présentés précédemment dans le paragraphe «Motivation et Objective». Ce chapitre fournit un cadre théorique définissant des parties constitutives de nos travaux de recherche, qui sont ensuite

concrétisées et étendues dans le chapitre 3. Nous proposons une approche pour la détection d'objets saillants utilisant des informations sensorielles de la Kinect. Enfin, plusieurs expériences sont décrites afin de valider l'approche proposée.

Le troisième et dernier chapitre présente des expériences réalisées dans un environnement intérieur réel. Dans ce chapitre nous proposons une extension de l'approche de la détection de la saillance en 3D sur d'autres capteurs 3D, comme le capteur de profondeur Xtion de ASUS équipant le robot Pepper. Le capteur de profondeur Xtion deASUS ne fournit pas la valeur de la profondeur métrique directement. Cependant, cette information métrique est nécessaire pour le calcul de la saillance en profondeur. Pour cela nous proposons une approche expérimentale pour étalonner le capteur de profondeur ASUS à l'aide du capteur infrarouge de le Kinect de Microsoft, en utilisant un algorithme d'apprentissage neuro-flou ANFIS.

La faisabilité de cette approche a été vérifiée, d'une part, par une expérience qui consiste en l'estimation métrique de la distance entre deux objets dans l'espace, et d'autre part, en comparant la performance de ce système basé sur l'algorithme ANFIS avec d'autres algorithmes d'apprentissage, tel que SVR (Support Vector Regression), MLP (Multi-Layer Perceptron) et l'Interpolation Bilinéaire.

Cette première phase d'apprentissage étant réalisé, il est alors possible de mettre en œuvre la stratégie présentée dans le chapitre 2 sur le robot Pepper. Nous avons ensuite étendu notre travail à caractérisation spatial qualitative des objets saillants détectés. En effet, nous avons présenté un système de vision artificielle capable :

- d'extraire, de façon autonome, plusieurs objets saillants de son environnement,
- de reconnaitre les objets saillants détectés dans de différents contextes visuels,
- de localiser et caractériser spatialement ces objets.

La prédisposition de distinguer l'information pertinente (la saillance visuelle) et la capacité de la caractérisation spatiales des objets saillants détectés procurent à la machine (le robot) un degré accru de discernement dans sa perception de l'environnement dans lequel elle évolue et de ce fait, augmentent son potentiel d'autonomie d'évolution dans cet environnement. De même, bien que ne faisant pas partie des objectifs des présents travaux de thèse, ces mêmes capacités ouvrent le potentiel, pour un robot doté de telles capacités, de construire une connaissance de son environnement, accroissant aussi son degré d'autonomie dans l'exécution des taches que celui-ci aurait à effectuer dans cet environnement et en interaction avec ce dernier.

Nous terminons ce manuscrit par une conclusion générale permettant de faire une brève synthèse des résultats obtenus et donner quelques perspectives.

Chapitre 1. Etat de l'art

1.1 Introduction

La décennie récente a été un témoignage de nombreux progrès dans les techniques de vision par ordinateur et les capteurs visuels offrant un potentiel attrayant pour examiner le problème d'autonomie des robots dans les environnements complexes. En effet, d'une part, de nombreuses techniques de traitement d'image avec une complexité informatique réduite ont été conçues et, d'autre part, un certain nombre de nouveaux capteurs visuels combinés avec des caractéristiques attrayantes et des prix accessibles ont été présentés, et qui permettent une capture visuelle 3D de l'environnement en fournissant la valeur de la profondeur.

Dans ce chapitre nous allons introduire les différentes technologies d'acquisitions de l'information de la profondeur nécessaire à la détection des objets saillants tels que: les capteurs des profondeurs de Microsoft Kinect, Temps de Vol, ASUS de robot Pepper et la stéréovision. Nous présentons aussi les différentes approches existantes dans le domaine de la saillance visuelle 2D et 3D où la mesure de l'image de la profondeur est faite par les capteurs 3D indépendamment de l'image acquise par la caméra RGB.

La majorité des travaux dans le domaine de la saillance sont basés uniquement sur le calcul de la saillance à partir des images de couleur en deux dimensions, alors que l'humain perçoit le monde sous sa forme tridimensionnelle (image couleur 2D + la dimension de la profondeur) qui est plus riche. Récemment, la détection d'objets saillants 3D à partir d'images RGBD (Red Green Blue Depth) a suscité beaucoup d'intérêt en raison de la commercialisation des capteurs 3D, comme ceux de Microsoft (Kinect) et Asus (Xtion), qui permettent d'obtenir des informations sur la profondeur. Or, certains travaux ont déjà montré l'intérêt d'utiliser les informations relatives à la profondeur afin d'améliorer l'estimation de la saillance des objets dans une image (Scharstein et al. 2002) et plus particulièrement à l'aide des capteurs 3D (Khoshelham et al. 2012; Scharstein et al. 2002) (Ronchetti et al. 2011).

1.2 La perception visuelle de l'environnement

La perception sensorielle est définie comme une action biophysique qui nous permet d'obtenir des informations de l'environnement à travers nos cinq sens (vue, goût, odeur, audition, contact). La perception visuelle, quant à elle, se limite à la capacité du cerveau à

donner un sens à ce que les yeux voient. Il est très important ici de faire la distinction entre la capacité des yeux à capter des images (visualisation) et la capacité du cerveau à interpréter le sens de ces images (Scholl 2001) et comme cela est illustré par la figure 1.

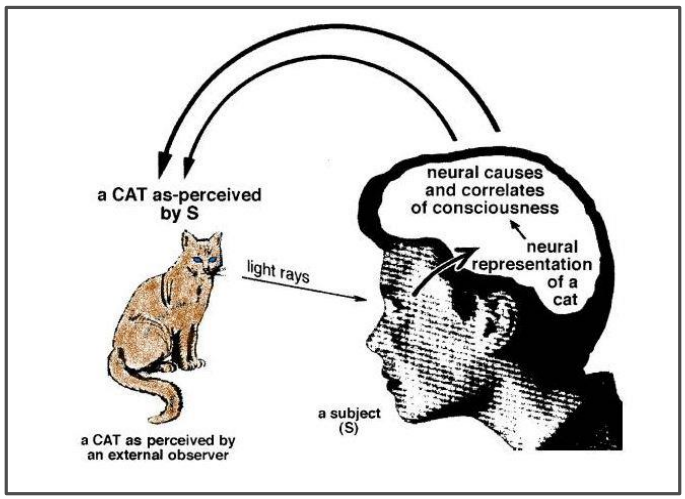


Figure 1: Un exemple de perception visuelle (Velmans 2003)

La perception visuelle est importante pour de nombreuses activités quotidiennes telles que la lecture, l'écriture, la recherche et la manipulation des objets …etc. Dans ce contexte, il est important de souligner que la perception de la profondeur, qui fait référence à notre capacité à voir le monde en trois dimensions, est fondamentale car c'est ce qui nous permet d'interagir avec le monde physique en évaluant avec précision la distance à un objet donné.

La perception visuelle permet généralement aux robots d'extraire des informations métrologique ainsi que d'interpréter l'environnement dans lequel ils évoluent. Pour les robots, la perception visuelle contribue donc à augmenter leur autonomie comme par exemple pour la navigation et la localisation dans un environnement inconnu (Hofmann et al. 2004). Cependant, la complexité de l'environnement et les contraintes de traitement en temps réel inhérentes au domaine de la robotique rendent ces tâches complexes.

1.3 Les technologies d'acquisition de la profondeur

Il existe de nombreuses approches pour percevoir la profondeur d'un objet, parmi lesquelles on trouve les lumières structurées et les rayons laser dans le cas des capteurs actifs ainsi que la stéréovision pour les technologies qualifiées de passives (voir Figure2).

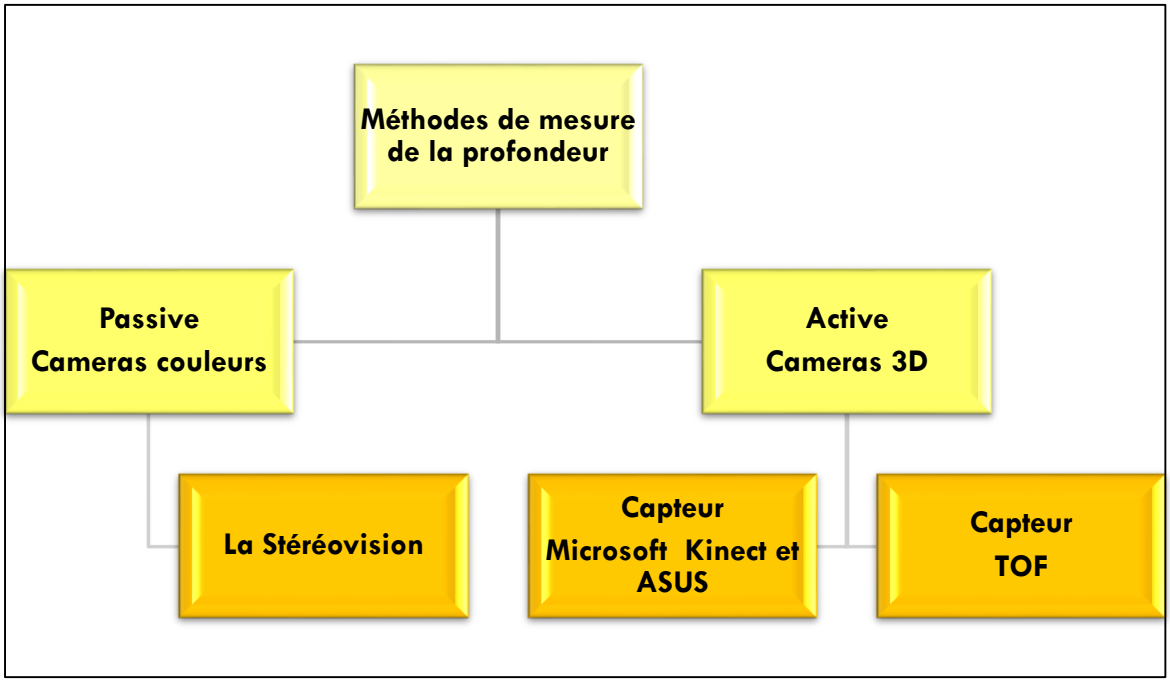


Figure 2: Taxinomie des méthodes de mesure de la profondeur

1.3.1 Les technologies passives

Les technologies passives, contrairement aux technologies actives n'utilisent aucunes mesures directes de la profondeur. Les seules données disponibles sont, une ou plusieurs images couleurs. Parmi les différentes méthodes développées dans cette catégorie, la vision multi-images où la stéréovision sont les plus connues. Les systèmes de stéréovision sont utilisés pour déterminer la profondeur à partir de deux images prises simultanément mais de points de vue légèrement différents à l'aide de deux caméras. Un algorithme de correspondance stéréoscopique est alors utilisé afin de détecter les points des objets similaires dans les deux images capturées. La figure 4c illustre le résultat d'une carte de disparité qui représente l'information de la profondeur obtenue à partir de la stéréovision (Figure 4a et 4b).

La figure 3 montre le principe de fonctionnement de la vision stéréoscopique permettant d'obtenir la carte de disparité (carte de profondeur).

Le calcul de la carte relative à la profondeur nécessite au moins 3 étapes :

- **L'étalonnage stéréoscopique**

  Les paramètres intrinsèques représentent le centre optique et la distance focale de la caméra. Les paramètres extrinsèques représentent l'emplacement de la caméra dans la scène 3D. Ces paramètres calculés dans la phase de l'étalonnage sont nécessaires dans

la phase de l'estimation de la profondeur d'un pixel. Pour estimer ces paramètres, nous devons avoir des points du monde réel (3D) et leurs correspondances sur l'image 2D. Nous pouvons obtenir ces correspondances en utilisant plusieurs images d'un modèle d'étalonnage, tel qu'un damier.

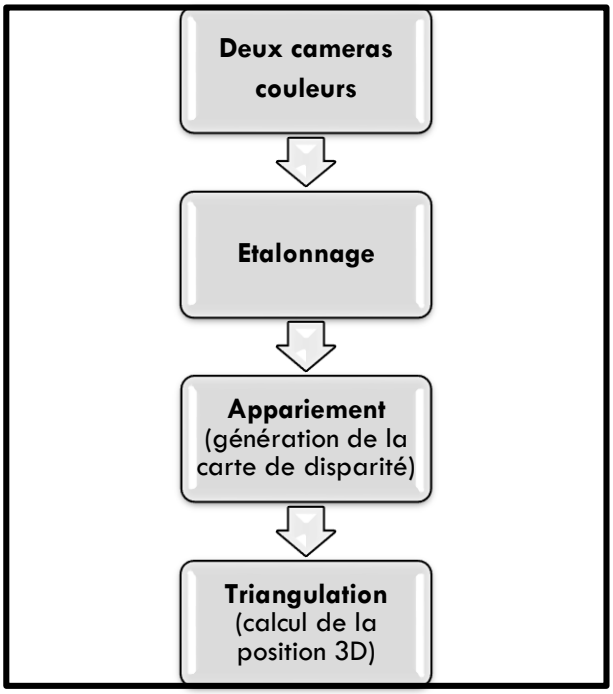


Figure 3: Le fonctionnement général de la stéréovision

- **L'appariement**

L'appariement visuel entre deux images, aussi appelé problème de mise en correspondance, est l'étape qui correspond à la reconstruction tridimensionnelle et la production de la carte de disparité. Ce problème géométrique repose sur le calcul de points correspondances dans les deux images. La différence de position entre les points correspondants est appelée disparité.



Figure 4: Tsukuba , (a) Gauche (b) Droite (c) carte de disparité (Scharstein et al. 2002)

- **Le calcul de la profondeur**

A partir d'une carte de disparité nous calculons et nous récupérons l'ensemble des coordonnées 3D des points de la scène (Scharstein et al. 2002). Ce calcul est une étape indispensable à la reconstruction 3D d'une scène réalisé grâce à la triangulation (voir figure 5). Le calcul de cette triangulation est donné par l'équation 1.

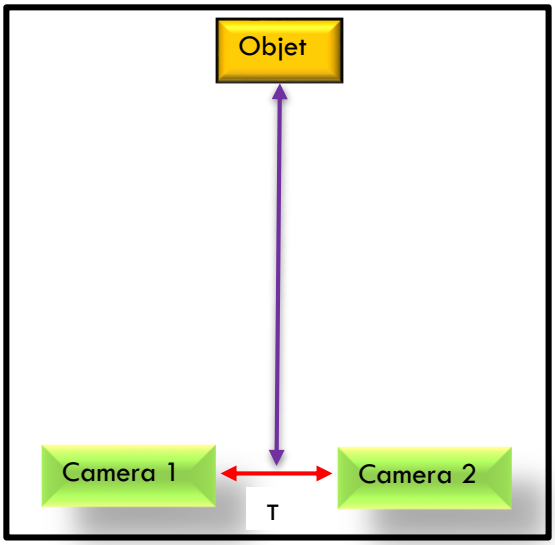


Figure 5: Système de triangulation

$$Z = \frac{f * T}{d} \qquad (1)$$

Z : représente la distance (profondeur) en mètre entre l'objet et le centre des caméras.

f : La distance focale de la camera en mètre.

T: La distance entre deux camera en mètre.

d: La disparité en mètre.

Cependant, l'utilisation de la technologie de la stéréovision pour la détermination de l'information relative à la profondeur a certains inconvénients. Premièrement, il est nécessaire d'utiliser un système à deux caméras qui doit être préalablement étalonné. Deuxièmement, la recherche de la correspondance correcte entre deux points d'une image, qui est lié à l'algorithme de l'appariement, nécessite un temps de calcul important. Troisièmement, cette technologie est moins performante dans le cas de scènes complexes contenant beaucoup d'objets.

La reconstruction 3D consiste à déterminer l'information 3D des objets lors de l'acquisition d'images, c'est-à-dire la projection d'une scène réelle sur un plan d'image. En effet, le déplacement d'une caméra couleur dans un monde 3D collecte un ensemble d'images qui sont des données 2D et porte à la fois une information sur la trajectoire de la caméra et sur le monde dans lequel elle évolue en 3D. La vision 3D n'est souvent possible que si nous avons deux points de vue différents d'une même scène ; on peut alors employer plusieurs caméras (comme en stéréo vision) ou bien utiliser le mouvement d'une seule caméra. Dans ce cas, on parle de "structure à partir du mouvement" ou plus communément le SFM ("Structure From Motion" en anglais). Pour un objet détecté dans une image, il faut alors être capable de dire si cet objet est visible dans une autre image, et quelle est sa nouvelle position dans cette image.

L'objectif de la SFM est de construire un nuage de points d'une scène à partir de correspondances entre des images capturées par une caméra en mouvement. Pour la plupart des applications, telles que la robotique et la conduite autonome, SFM utilise plus de deux vues. Le SFM de vues multiples nécessite un point de correspondances à travers de multiples images, appelées *pistes*. Les erreurs de la reconstruction 3D des images par une caméra en mouvement proviennent d'une correspondance brouillée entre ces images et d'un étalonnage imprécis de la camera. Ces erreurs s'accumulent au fur et à mesure que le nombre des images augmente. Une façon de réduire ces erreurs est d'affiner les prises des images de la caméra et les emplacements des points 3D. L'algorithme d'optimisation non linéaire, appelé *ajustement du faisceau*, peut être utilisé pour le raffinement. L'ajustement de faisceaux consiste à ajuster au mieux deux ensembles de faisceaux : les faisceaux 3D sont les rayons reliant des points de la scène (les primitives 3D) et le centre optique de la caméra, les faisceaux 2D sont les rayons reliant le centre optique de la camera et les points de plan image (Mhiri 2015).

1.3.2 Les technologies Actives

Les technologies actives consistent à combiner une caméra couleur associé à un système permettant de mesurer la profondeur. Pour la mesure de cette profondeur, on distingue deux grandes catégories de caméra de profondeur: la première TOF (Time of flight) est basée sur le temps d'émission et de réception d'un signal lumineux, la deuxième est basée sur l'émission-réception de faisceaux lumineux infrarouges.

Les caméras de profondeur ont été développées depuis de nombreuses années. Par exemples, les sociétés PMDTec (Photonic-Mixer-Devic), Mesa Imaging, 3DV Systems et

Canesta ont commercialisé depuis une dizaine d'années des systèmes basés sur la technologie du TOF. Cependant, les capteurs d'imagerie ToF ont deux problèmes majeurs : une faible résolution ainsi qu'une faible sensibilité ce qui entraîne des niveaux de bruit sur la mesure élevés. Actuellement, la technologie TOF se limite à des résolutions allant jusqu'à 352x222 pixels. En 2010 Microsoft a proposé un nouveau système (la Kinect), qui détermine les disparités entre le faisceau lumineux émis et la position observée du point lumineux avec une imagerie en niveaux de gris de deux mégapixels. En plus de la mesure de la profondeur, la Kinect comprend un capteur d'image couleur RGB ainsi que des microphones.

Dans la suite de cette section, nous présentons les deux principales technologies actives de mesure de profondeur.

1.3.2.1 La caméra temps de vol TOF

La camera 3D temps de vol (voir Figure 6), permet de fournir une information tridimensionnelle d'une scène observée.

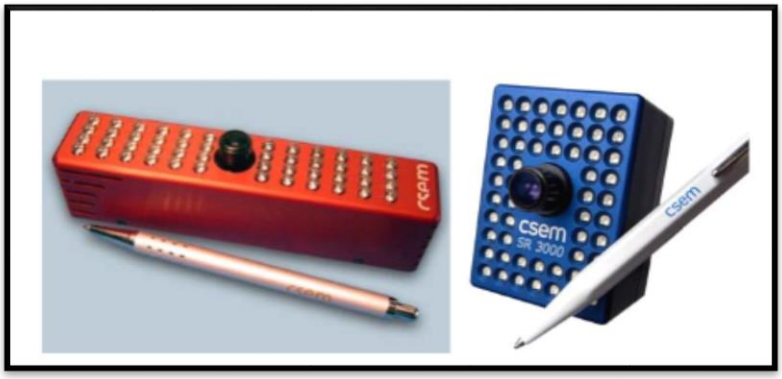


Figure 6: Camera temps de vol (Gut 2004)

La technologie temps de vol est une technologie active et optique, c'est-à-dire qu'elle est basée sur l'émission puis la réception d'un signal lumineux. L'onde utilisée dans cette technologie modulée de deux manières différentes, directes et indirectes (voir Figure 7) :

- ✓ **Mesure direct par impulsion**

Aussi appelé « la lumière pulsée ». La distance est mesurée en appliquant la formule suivante:

$$d = c * \frac{\Delta t}{2} \ , mésuré\ en\ métre(m)$$

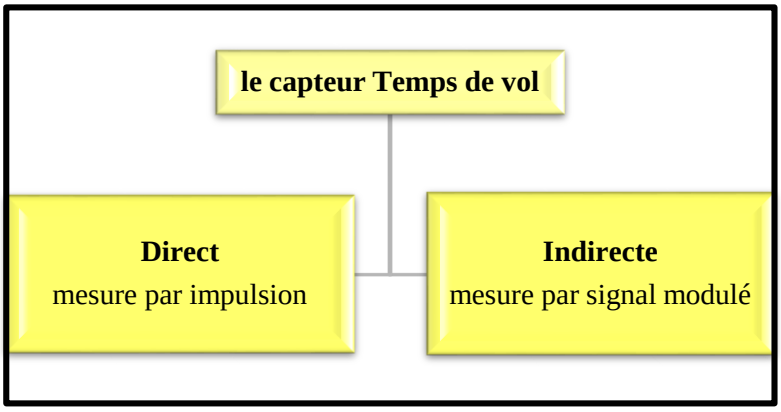


Figure 7: Mesure par impulsion et par signal modulé

avec " c " *(m/s)* est la célérité de la lumière étant connue et constante, "Δt "(s) est le temps parcouru par l'onde, c'est un principe bien connu et employé notamment par les télémètres (Weingarten et al. 2004).

- ✓ **Mesure indirect**

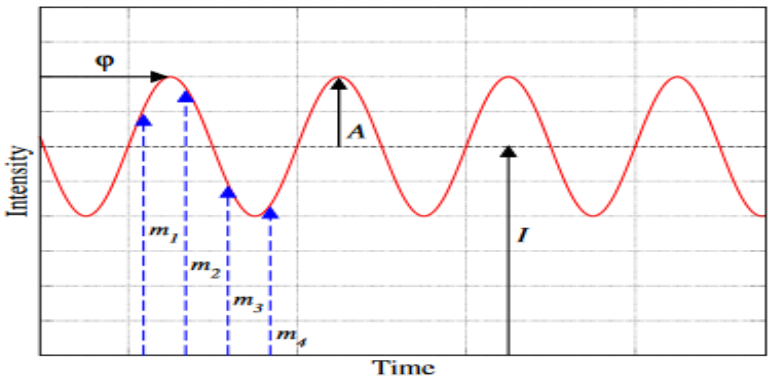


Figure 8: Le principe de la méthode de décalage de phase (Foix et al. 2011)

Dans cette technologie, le capteur TOF échantillonne la quantité de lumière modulée réfléchie pour chaque pixel de la scène. Cela se fait quatre fois par période à intervalles égaux (voir Figure 8). Ces quatre quantités $(m_1..m_4)$ permettent de récupérer le signal entrant sinusoïdal. Le déphasage entre la lumière émise et le signal de retour est calculé par les équations suivantes (Foix et al. 2011):

$$\partial = arctan\frac{m_4 - m_2}{m_1 - m_3} \qquad (2)$$

$$B = \frac{m_1 + m_2 + m_3 + m_4}{4} \qquad (3)$$

$$A = \frac{\sqrt{(m_4 - m_2)^2 + (m_1 - m_3)^2}}{2} \qquad (4)$$

$$\mathrm{D} = L\frac{\partial}{2\pi} \quad , mésurée\ en\ (m) \qquad (5)$$

Cette technique de déphasage permet de calculer facilement la profondeur (D). L'intensité (B) et l'amplitude (A) permettent de prédire la qualité des mesures de la profondeur (D). La fréquence de modulation (f_m) de la lumière émise détermine la plage de distance sans ambiguïté.

$$\mathrm{L} = \frac{c}{2f_m} \quad , mésuré\ en\ (m) \qquad (6)$$

Où "c" : est la vitesse de la lumière dans le vide.

Les principaux inconvénients de cette technologie sont liés à la conception des capteurs (Einramhof et al. 2007). Leur taille réduite conduit directement à une résolution des données acquises faible, ainsi qu'une portée limitée. Aussi, l'augmentation de la taille des capteurs génère la plupart du temps une augmentation du bruit de mesure. En effet, les propriétés de conception et de mesure de ces capteurs sont cause de nombreuses erreurs difficilement évitables, apparaissant notamment lors de l'étape de démodulation du signal (Oggier et al. 2005). Ceci se traduit par une détérioration de la mesure de la profondeur, qui sera touché quelle que soit la taille du capteur par de multiples sources de bruit et d'erreur.

1.3.2.2 La caméra 3D Kinect

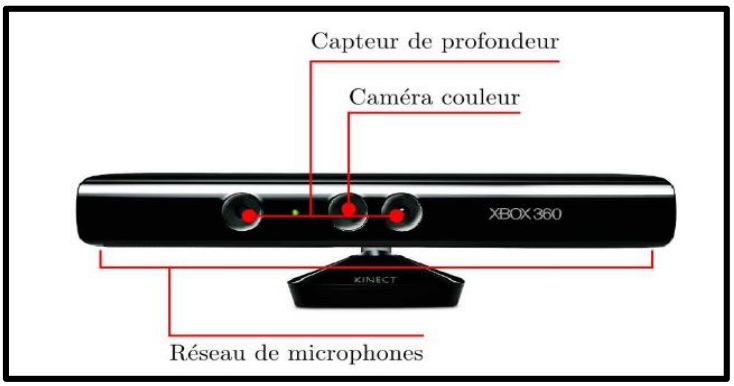


Figure 9: Le capteur Kinect de Microsoft (Lejeune et al. 2012)

En 2010, Microsoft a commercialisé un nouveau périphérique pour sa console de jeux vidéo Xbox 360. Le capteur de profondeur de la Kinect a été développé par une société spécialisée PrimSense, une entreprise qui avait précédemment produit d'autres caméras de profondeur utilisant la même technique de projection infrarouge IR. PrimeSense a donc travaillé en étroite collaboration avec Microsoft pour produire une caméra de profondeur qui fonctionne avec les logiciels et les algorithmes Microsoft. En novembre 2010, la Kinect a été lancé sous le nom de "Microsoft Kinect», et a été un succès commercial majeur. Elle a été vendue jusqu'à 10 millions d'unités dans le premier mois de sa sortie. Elle est devenue le périphérique informatique le plus rapidement vendu dans l'histoire.

Le capteur infrarouge du Microsoft de la Kinect, enregistre la distance des objets placés en face de lui. Il utilise la lumière infrarouge pour créer une image (une image de profondeur) qui ne capture pas ce que ressemblent les objets, mais où ils se trouvent dans l'espace. La lumière infrarouge a une longueur d'onde plus longue que celle de la lumière visible, de sorte que nous ne pouvons pas la voir à l'œil nu. Le projecteur infrarouge du Kinect illumine une grille de points infrarouges sur tout ce qui se trouve devant lui. Ces points sont normalement invisibles pour nous, mais il est possible de capturer une photo d'eux à l'aide d'une caméra infrarouge. La figure 10 montre à gauche une image couleur captée par la camera couleur de la Kinect et à droite l'image de profondeur capturée par la caméra infrarouge de la Kinect.



Figure 10: Images couleur et profondeur capturées respectivement par la camera 2D et IR de la Kinect.

L'information de profondeur est la base des informations les plus intéressantes que la Kinect peut fournir. Elle permet à la Kinect d'agir comme un scanner 3D pour détecter le mouvement des personnes et des objets. Cependant, capturer des images en profondeur n'est

pas la seule chose que Kinect peut faire. La camera couleur de la Kinect dispose d'un capteur numérique similaire à celui de nombreuses webcam et de petits appareils photo numériques. Elle a une résolution relativement faible (640 par 480 pixels).Comme la camera couleur est attachée à la Kinect à une distance connue de la caméra IR, la Kinect peut aligner l'image couleur de cette caméra avec l'information de profondeur capturée par sa caméra IR (Zhang 2012).

En plus de ces cameras couleur et infrarouge, la Kinect dispose de quatre microphones. Ces microphones sont répartis autour de la Kinect autant que nos oreilles sont réparties autour de notre tête. Leur objectif n'est pas seulement de capturer le son, mais aussi de le localiser dans la pièce. À l'intérieur de la base du support en plastique de la Kinect se trouve aussi un petit moteur. Cela permet à la Kinect de pivoter de droite à gauche et de s'incliner de haut en bas.

- ✓ **Mise en fonctionnement de la Kinect**

Pour utiliser la Kinect, il est nécessaire d'avoir recours à une bibliothèque appelée SimpleOpenNI. Cette bibliothèque permet d'accéder à toutes les données de la Kinect dont nous aurons besoin. L'installation de SimpleOpenNI se déroule en deux phases :

- d'abord, nous installons OpenNI, c'est un logiciel fourni par PrimeSense qui communique avec Kinect pour accéder et traiter ses données. Les étapes impliquées dans cette phase diffèrent selon notre système d'exploitation.
- Après avoir installé OpenNI, la dernière étape consiste à installer la bibliothèque de traitement SimpleOpenNI.

Au cours de l'été 2011, cela a changé avec la sortie de SDK Kinect pour Windows (Software Developer Kit). Le SDK a permis aux programmeurs d'écrire des applications exécutées sous Windows à l'aide de la Kinect. Ces programmes sont généralement écrits en utilisant C#, C++, python. Mais ils peuvent être implémentée dans n'importe quelle langage de programmation qui fonctionne avec l'environnement de programmation de Microsoft (Lejeune et al. 2012).

- ✓ **Caractéristique géométrique de la Kinect**

Il est pertinent de noter qu'un certain nombre d'estimations géométriques de la distance entre les objets perçus en utilisant le capteur Kinect sont déjà disponibles (Aptoula et al. 2007; Borenstein 2012). Cependant, ils ont été conçus dans le cadre d'une localisation d'objets par Xbox et les jeux vidéo pris en charge.

Le capteur de la profondeur génère une matrice de 640 x 480 contenant pour chaque pixel la valeur réelle de la profondeur d'un point, d'une scène, par rapport la Kinect. Cette valeur de profondeur (distance) noté « D » présente la coordonnée Z d'un point réel (voir Figure 11).

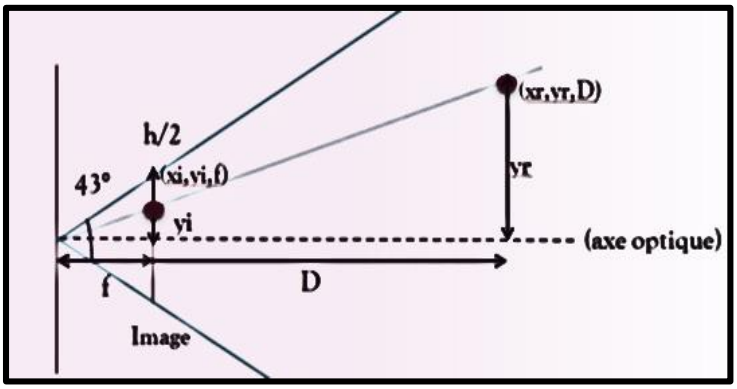


Figure 11: Calcul de l'ordonnée y_r d'un pixel

Avec

$$y_r = \frac{y_i}{f} \qquad (7)$$

$$x_r = \frac{x_i}{f} \qquad (8)$$

$$f = \frac{h/2}{\tan(\frac{43}{2})} \qquad (9)$$

Pour détecter et caractériser les objets réels existants dans l'image couleur, une correspondance entre l'image couleur et l'image de la profondeur est effectuée afin de déterminer les zones contenant l'objet (ou les objets) dans l'image couleur.

Les coordonnées réels d'un point d'un objet (x_r, y_r) sont calculée à partir de la distance focale[1] « f », la valeur de la profondeur « D » et les coordonnées du pixel dans l'image couleur (x_i, y_i) qui correspond à ce point dans l'objet (Ronchetti et al. 2011) (voir Figure 11). Les coordonnées réels (x_r, y_r) servent à calculer les caractéristiques réelles de l'objet, comme :

- La position relative des objets dans la scène au regard du centre de l'image (en haut, en bas, à gauche, à droite…).
- La position des objets les uns par rapport aux autres si plusieurs objets sont détectés.
- La profondeur de l'objet.

[1] La distance focale est la distance qui sépare le centre optique de la lentille du foyer de l'image.

- La hauteur et la largeur de l'objet, définit respectivement par les équations:

$$largeur = \sqrt{(x_{i1} - x_{i2})^2} \qquad (10)$$

$$hauteur = \sqrt{(y_{i1} - y_{i2})^2} \qquad (11)$$

Avec $(x_{i1}, y_{i1}), (x_{i2}, y_{i2})$, les coordonnées de deux points appartenant au contour d'un objet, situés respectivement sur la même ligne pour le calcul de la larguer et sur la même colonne pour le calcul de la même hauteur.

✓ **Les caractéristiques techniques de la Kinect**

Nous citons dans le tableau1 les caractéristiques de la Kinect.

Caractéristiques	Kinect
Aspect	
Alimentation	Prise électrique
La dimension	28 x 6 x 6.5cm
Poids	1,36 kg
Motorisation	Oui
Résolution de base	VGA (640×480) 30 fps QVGA (320×240): 30 fps
Résolution max	SXGA (1280*1024)
Portée de détection	0,5–6 m
Champ de vision	57° H, 43° V
Prix	35 €

Tableau 1: Caractéristiques techniques de Kinect

✓ **Les applications de la Kinect**

Les caméras RGB-D peuvent être utilisées dans plusieurs domaines comme la robotique. Henry dans (Henry et al. 2010) examine la création de la carte de l'environnement intérieur en 3D. Nous citons comme applications, par exemple la cartographie, la navigation du robot et la caractérisation spatiale de l'environnement.

✓ **Etalonnage des caméras couleur et profondeur de la Kinect**

Avant d'acquérir les images de couleur et de profondeur, il est nécessaire d'étalonner les deux cameras couleur et profondeur afin de déterminer les paramètres intrinsèques de chaque caméra et les paramètres extrinsèques entre les deux cameras (Devaux et al. 2014) (Zhang 2000).

- **Paramètres Intrinsèques**

Les paramètres intrinsèques sont les paramètres de passage du repère caméra au repère image (voir figure 12). Ils se traduisent sous la forme d'une matrice appelée matrice caméra k . Ce sont des paramètres fixes, interne à la conception de la caméra et de son focale optique.

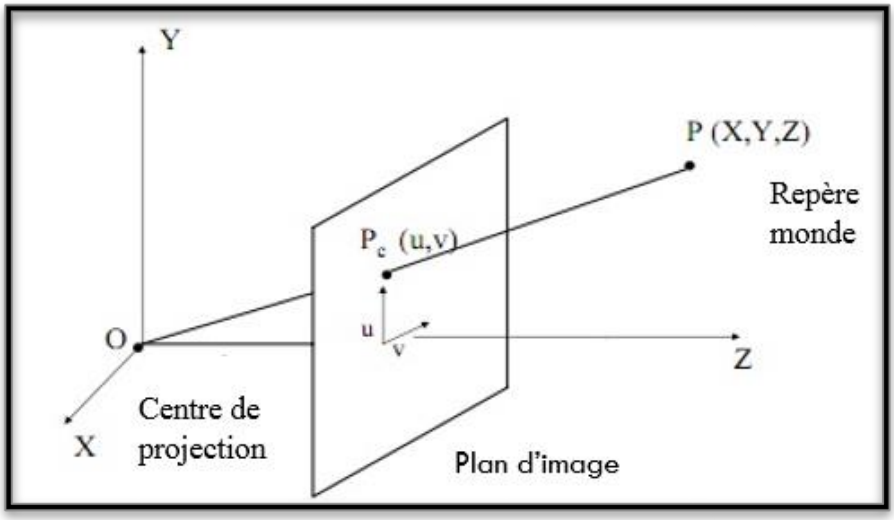


Figure 12: Les coordonnées du système d'étalonnage de la caméra

La matrice intrinsèque de la caméra k, est définie comme:

$$k = \begin{pmatrix} f_u & 0 & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{pmatrix}$$

où (c_u, c_v) représente les coordonnées de la projection du centre optique de la caméra sur le plan image et (f_u, f_v) les distances focales suivant les deux directions $u\ et\ v$. Une observation aura pour coordonnées dans le repère (X, Y, Z) , exprimé dans le repère de l'image de la

camera de profondeur par un point de coordonnées $m_k = (u_k, v_k)$. Les coordonnées (x_k, y_k, z_k) d'un point 3D observé peuvent être calculées comme:

$$x_k = z_k \frac{(u_k - c_u)}{f_u} \qquad (12)$$

$$y_k = z_k \frac{(v_k - c_v)}{f_v} \qquad (13)$$

Avec z_k la valeur de la profondeur qui est fournie directement par la camera de profondeur de la Kinect en centimètre.

- **Paramètres Extrinsèques**

L'estimation des paramètres extrinsèques permet de réunir les nuages de points des deux cameras couleur et profondeur dans un même référentiel, ces paramètre extrinsèques sont les matrices de rotation et translation qui permettent de passer du repère monde au repère lié à la caméra.

L'étalonnage se base sur la détection et la position de l'intersection des carreaux d'une mire : pour n carreaux, il y a (n-1) intersections. Les positions des points d'intersection entre les lignes et les colonnes sont connues avec précision dans le repère attaché à la mire, représenté par des points en couleur rouge sur la Figure13. Les positions de ces points dans l'image sont extraites par le traitement d'image couleur. Une fois ces points déterminés, il ne reste que l'établissement de lien entre le plan 3D du monde réel avec sa projection dans l'image couleur 2D.

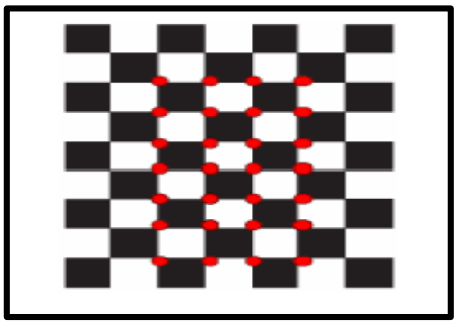


Figure 13: Objet d'étalonnage 'une mire'

✓ **Les limites de la Kinect**

- **Méthode géométrique**

La méthode géométrique est sensible à la calibration entre le capteur RGB et IR, il faudra tenir compte du fait que les images RGB acquises possèdent une résolution (1920 x

1080) bien plus importante et non proportionnelle à la résolution des cartes de profondeur (512 x 424).

- **Le champ de vision**

Le champ de vision de la Kinect est 57,8°, tandis que d'autre capteurs comme le LIDAR a une vue complète à 360°. Un champ de vision plus large permet également au robot de construire efficacement une carte sans trous, un robot avec un champ de vision plus étroit devra constamment manœuvrer pour remplir les trous manquants pour créer une carte complète.

- **La profondeur**

La portée minimale de la Kinect est d'environ 0,6 m et la plage maximale est comprise entre 4 et 8 m. Cette portée est considéré élevé par rapport à d'autre capteurs comme : Le Hokuyo URG-04LX-UG01 qui fonctionne de 0,06 à 4 m et le LIDAR qui fonctionne de 0,2 m à 6 m.

- La présence des trous dans l'image de profondeur. La figure 14, montre à gauche une image RG et à droite une carte de profondeur capturée par Kinect de la même scène. Les pixels en blanc sont des trous dans la carte de profondeur qui surviennent en raison de l'occlusion ou de l'encombrement des objets dans l'image (Fanelli et al. 2011).
- L'éclairage ambiant lumineux peut affecter le contraste des images infrarouges, ce qui entraîne des valeurs aberrantes ou des trous dans la carte de profondeur (voir Figure 14).
- Une configuration spatiale encombrée d'objets peut créer des occlusions et des ombres, ce qui produit également des trous dans l'image de profondeur. En outre, les surfaces lisses apparaissent surexposées dans l'image infrarouge, générant des trous dans la carte de profondeur.



Figure 14: Une image RGB et une carte de profondeur illustrent le problème de l'encombrement des objets.

1.3.2.3 La caméra de profondeur ASUS du robot Pepper

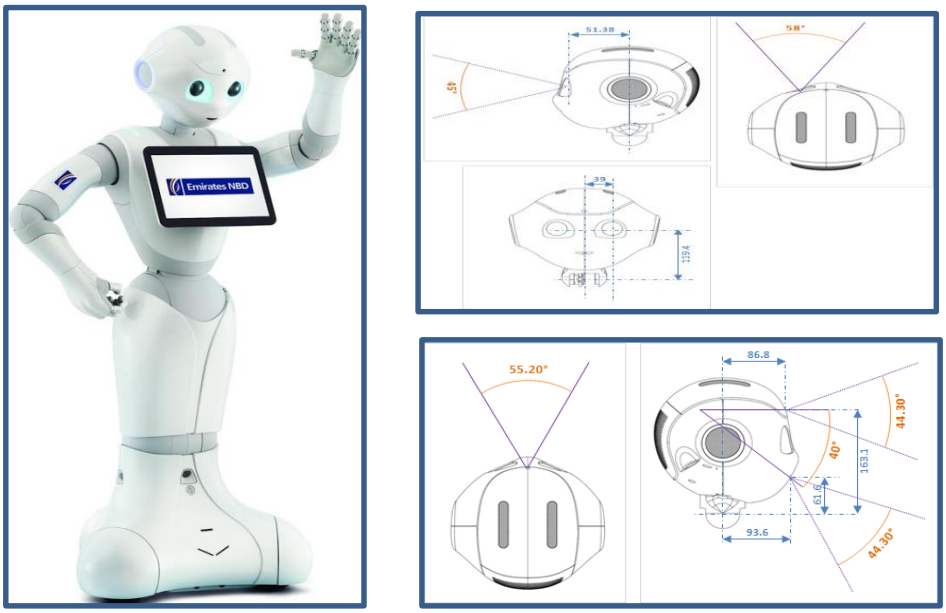


Figure 15: La position du capteur de la profondeur ASUS dans le robot Pepper (SoftBank 2017)

En 2014, SoftBank et Aldebaran Robotics ont annoncé le développement de "Pepper" le premier robot qui a la capacité de lire et exprimer ses émotions. Les ventes étaient initialement réservées aux créateurs, aux développeurs et aux entreprises, mais ont été ouverts au grand public à partir du 20 juin 2015 (SoftBank 2017).

Le capteur ASUS Xtion intégré dans la tête du robot Pepper (voir figure 15), est situé derrière les yeux de Pepper (l'émetteur derrière l'œil gauche et le récepteur derrière l'œil droite). Son intégration dans la tête, permet d'orienter le capteur et d'étendre son champ de vision.

Pour lui permettre d'interagir avec les humains, Pepper est équipé de nombreux capteurs, incluant le capteur de profondeur ASUS, cités dans le tableau 2

dimension	1,210 mm (hauteur) × 480 mm (profondeur) × 425 mm (largeur)
Poids	29 Kg
Batterie	Lithium-ion Capacité: 30.0 Ah/795 Wh Temps operational : Approx. over 12 heures

Capteurs	• Haut-parleurs x2 • Microphones X4 • Camera couleur X2 ➢ Model : OV5640 ➢ Camera output 640*480@30fps or 2560*1920@1fps ➢ Champ de vision : 68.2°DFOV (57.2°HFOV, 44.3°VFOV) ➢ Résolution : 5Mp • Capteur de profondeur ASUS (voir Figure 15) ➢ Model : ASUS XTION ➢ Sensibilité (Signal/bruit)= 45dB ➢ Camera output 320*240@20fps ➢ Image de profondeur : portée de 0.4m à 3.68m Mise à l'échelle [0,255] ➢ Champ de vision 70.0°DFOV (58.0°HFOV, 45.0°VFOV) • Led : dans les yeux 2x8, sur les épaules 2x2, oreilles 2x10 • Unité inertielle : Accéléromètre, gyromètre. • Laser X 6 : longueur d'onde 808 nm. • Infrarouge X 2 • Ultrason x 2 : fréquence 42kHz, Sensibilité -86dB
Vitesse de déplacement	2 km/h
Carte mère	Processeur Atom E3845, CPU Quad core, Ram 4 GB DDR3, MICRO SDHC 16 GB, GPU Intel HD graphique 792 MHz
La mise en réseau	Wi-Fi: IEEE 802.11 a/b/g/n (2.4 GHz/5 GHz) Ethernet port × 1 (10/100/1000 base T)
Plate-forme	NAOqi OS

Tableau 2 Les caractéristiques du robot Pepper (SoftBank 2017)

✓ **Les limites du capteur de profondeur ASUS**

- **Méthode géométrique**

La méthode géométrique est sensible à la calibration entre le capteur RGB et IR, il faudra tenir compte du fait que les images RGB acquises possèdent une résolution 2560*1920 non proportionnelle à la résolution des cartes de profondeur 320 *480.

- **La portée**

La portée minimale pour le robot Pepper est d'environ 0,4 m et la portée maximale est 3.68m, cependant la Kinect a une portée maximale comprise entre 6 à 8 m.

- **La profondeur**

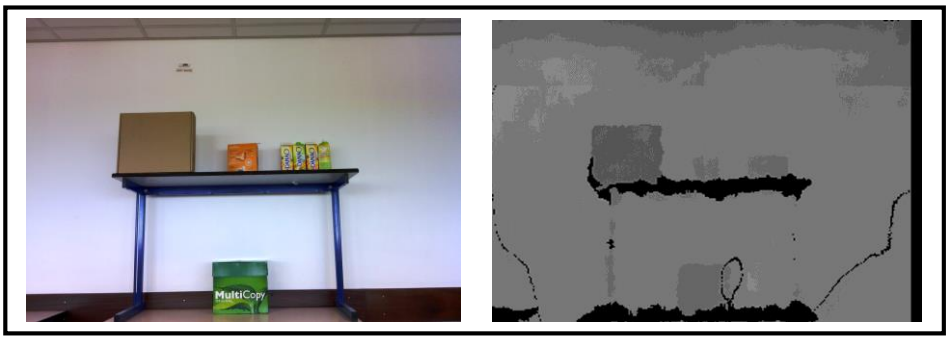


Figure 16 : Une image RGB et une carte de profondeur illustrent le problème des surfaces lisses.

Dans la figure 16, les surfaces lisses (les tables) apparaissent surexposées dans l'image infrarouge, générant des trous dans la carte de profondeur, ce qui signifie un manque d'information de profondeur.

1.3.3 Résumé des caractéristiques des technologies

La méthode de la lumière structurée projette une lumière infrarouge sur toute l'aire d'une pièce dans un environnement. Ces rayons sont réfléchis par l'ensemble des éléments physiques qui sont présents dans la pièce. Cette technique est utilisée dans de nombreuses applications d'environnement intérieur, telles que Microsoft Kinect et le capteur ASUS intégré dans Pepper. Bien que la lumière structurée soit idéale pour les environnements à faible luminosité et les scènes à faible texture, sa plage relativement courte (0,4 m la distance minimal entre l'objet et le capteur, et jusqu'à 3,5 m de distance maximal) l'empêche d'être utilisé dans des applications extérieures comme la navigation par robot.

Caractéristiques	Temps de vol TOF	Capteur de profondeur de Microsoft Kinect	Capteur de profondeur ASUS de Pepper	Stéréovision
Problème de correspondance (disparité)	Non	Non	Non	Oui
Nécessite d'un étalonnage extrinsèque	Non (quand elle est utilisé seul)	Non (quand elle est utilisé seul)	Non (quand elle est utilisé seul)	Oui
AUTO illumination	Oui	Oui	Oui	Non
La portée de la profondeur	De 0.3 à 7.5m	De 0.5 à 8m	De 0.4 à 3.68m	Dépend de la camera couleur
La résolution d'image	204* 204	640*480	320*240	Dépend de la haute résolution de la camera
Image par seconde (fps: frames per second)	25 fps	30 fps	20 fps	Typiquement 25 FPS, dépend de la camera
Nombre des caméras	Une caméra de profondeur et une caméra de couleur	Une caméra de profondeur et une caméra de couleur	Une caméra de profondeur et une caméra de couleur	Deux caméras de couleur minimum
Acquisition de la profondeur Travailler de jour comme de nuit	Oui	Oui	Oui	Non
Sensible à la matière de l'objet (métal, plastique…etc.)	Oui	Oui	Oui	Non
Sensibilité à la lumière	Faible	Elevé	Elevé	Elevé
Sensible aux occlusions et par extension, aux trous.	Faible	Elevé	Elevé	Elevé

Tableau 3: Résumé des caractéristiques des technologies

Le tableau 3 résume les caractéristiques de différentes technologies. La méthode de mesure laser, également connue sous le nom de méthode TOF (Time of Flight), elle calcule le temps nécessaire pour que la lumière ait un aller-retour entre la source lumineuse et l'objet. La stéréovision est une technique de détection complètement passive. Un système de vision stéréo

binoculaire comporte deux caméras parallèles qui prennent des photos de la même scène à partir de positions légèrement différentes. La profondeur dans ce cas-là sera calculée à partir de chaque point de la scène en utilisant la géométrie de triangulation. La stéréovision coûte cher et est sensible à la lumière ambiante et à la texture des scènes.

Chaque technologie d'acquisition de la profondeur a ses avantages et ses inconvénients, d'où la difficulté de choisir la technologie la plus adapté à nos expérimentations. Pour ce faire, nous avons arrêté trois critères pour le choix du capteur : la précision de l'information de la profondeur, une résolution élevée de l'image et une portée de mesure plus large. Ces critères nous ont semblé importants dans la qualité de la détection de la saillance visuelle en 3D. Suivant ces facteurs, le capteur Microsoft Kinect est le plus adapté pour la mise en œuvre de notre approche de la détection de la saillance 3D. Concernant le capteur infrarouge ASUS, intégré au le robot Pepper, même s'il se trouve en seconde position selon les critères arrêtés, il peut également être utilisé pour la mise en œuvre de notre approche de la détection de la saillance 3D. En effet, la ressemblance fonctionnelle de ce capteur avec le Kinect est un facteur intéressant pour envisager une implantation de notre approche de la détection de la saillance 3D sur le robot Pepper.

Dans la suite nous présentons un aperçu de l'état de l'art sur la saillance visuel 2D et 3D et l'influence de ces capteurs de profondeur sur la saillance.

1.4 La saillance

La saillance visuelle (également mentionnée dans la littérature comme l'attention visuelle, l'imprévisibilité ou la surprise) est décrite comme une qualité perceptuelle qui fait ressortir une région d'une image par rapport à son environnement afin d'attirer l'attention de l'observateur (Achanta et al. 2009). Le concept de la saillance visuelle s'inspire de recherche sur le système de la vision humaine. Dans les premières étapes du traitement visuel, le système de vision humain se focalise d'abord, d'une manière inconsciente, sur des régions visuellement « attrayantes » de l'image perçue. Le calcul de la saillance visuelle se base généralement sur des caractéristiques 2D d'une image tels que l'intensité, la luminance, la couleur, mais généralement ne tient pas compte de l'information de la profondeur. Il faut toutefois noter que, à ce jour, l'apparition des capteurs 3D a permis d'initier quelques travaux sur le calcul de la saillance visuelle en tenant compte de cette information. Dans la suite de cette section, nous commencerons par donner un aperçu de l'état de l'art sur le calcul de la saillance qui tient compte exclusivement des informations 2D d'une image puis nous présenterons les résultats de

quelques travaux récents sur le calcul de la saillance à partir d'informations obtenues par des capteurs 3D.

Bien qu'il existe des approches informatiques biologiquement basées sur la saillance visuelle, la plupart des travaux existants ne prétendent pas être biologiquement plausibles: pour cela, ils utilisent des techniques purement informatiques pour atteindre l'objectif. L'une des premières œuvres utilisant la saillance visuelle dans le traitement de l'image a été publiée par Itti, Koch et Niebur (Itti et al. 1998). Les auteurs utilisent une approche biologiquement plausible basée sur un calcul de contraste centre-surround utilisant «Différence de Gaussiens». Publié plus récemment, d'autres techniques communes de calcul de la saillance visuelle basée sur les graphes (Harel et al. 2007), les distances de caractéristique centre-surround (Achanta et al. 2008), le contraste multi-échelle, l'histogramme centre-surround et la distribution spatiale des couleurs ou les caractéristiques de la couleur et de la luminance (Liu et al. 2011). Une approche moins commune est décrite dans (Liang et al. 2012) utilise une représentation et un partitionnement hyper-graphique sensibles au contenu au lieu d'utiliser les fonctionnalités traditionnelles et les paramètres généralement considérés dans les images. Enfin, dans leurs travaux récents, les auteurs de (Liang et al. 2012) et (Maximili et al. 2011) ont étudié un système intelligent de vision 2D pour l'acquisition de connaissances autonomes des robots humanoïdes ainsi qu'une approche de segmentation puissante en profitant de la représentation RGB en coordonnées sphériques(Moreno et al. 2012).

1.4.1 Aperçu des techniques existantes dans la saillance 2D

L'une des premières utilisations de la saillance visuelle dans le traitement d'image a été publié par Itti (Itti et al. 1998). Les auteurs utilisent une approche bio-inspirée qui se base sur le calcul du contraste de la couleur et de l'intensité entre le centre d'une image (ou d'une partie) et de son contour « center-surround ». La figure 17 illustre l'approche proposée par Itti.

Les travaux d'Itti ont été le point de départ de nombreux autres travaux. Dans (Borji et al. 2010), Borji donne un bon aperçu de l'état de l'art des travaux concernant l'attention visuelle. De nombreux modèles et benchmarks peuvent aussi être trouvés à l'adresse suivante. http://saliency.mit.edu/home.html

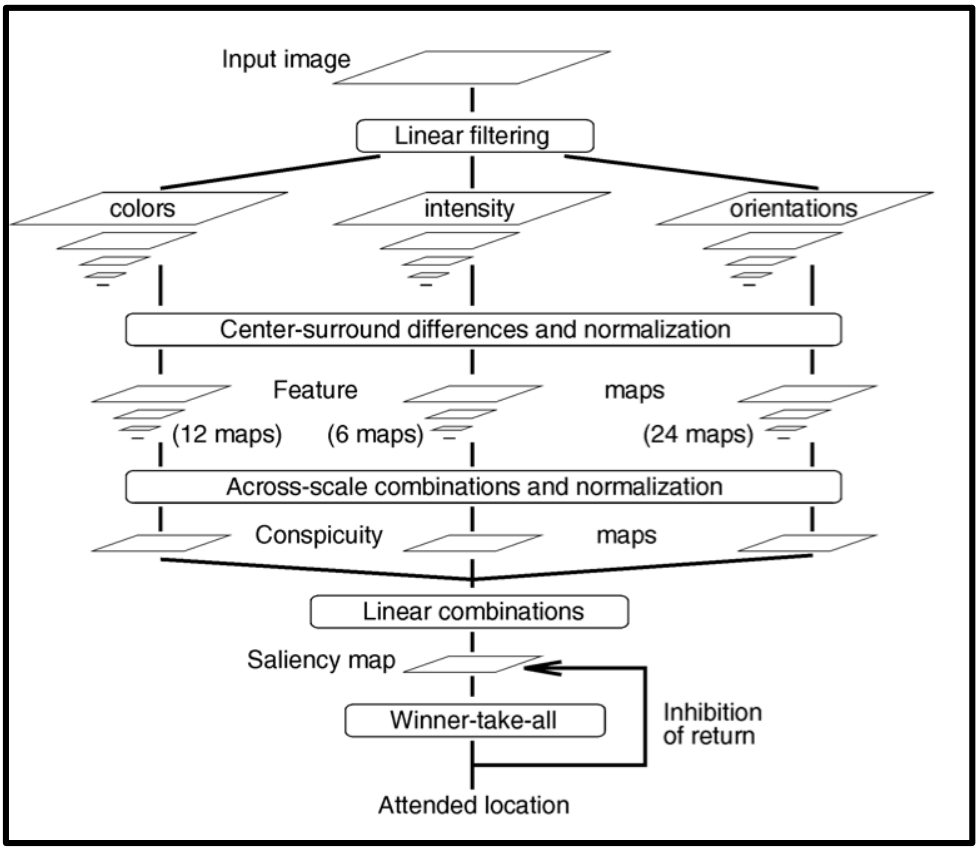


Figure 17: L'architecture générale du modèle d'Itti, Koch & Ulmann 1998. Extrait de (Itti et al. 1998)



Figure 18: Exemple de résultats du calcul de saillance visuelle d'après (Cheng et al. 2015) .En haut, image originale; au centre, carte de saillance; en bas, masque.

La figure 18, extrait de (Cheng et al. 2015), illustre un exemple de calcul de saillance. La carte de la saillance (ligne du centre) est obtenue à partir des caractéristiques 2D d'une image tel que l'intensité, le contraste, la couleur. A partir de la carte de saillance, la sélection des régions (objets) les plus saillantes est ensuite obtenue par application d'un seuil et d'un

algorithme de segmentation (ligne du bas). Il en résulte l'extraction d'un ou de plusieurs objets visuellement important. Il faut aussi noter que certains travaux sur la saillance visuelle se sont basés sur une analyse spectrale comme par exemple(Hou et al. 2007). Dans ces travaux, les auteurs proposent la notion de « spectre résiduel » (spectral residual) qui correspond à une carte de saillance dans le domaine fréquentiel. En traitement de l'image, l'identification des régions visuellement saillantes d'une image est utilisée dans de nombreuses applications comme le redimensionnement intelligent d'une l'image (Avidan et al. 2007), l'affichage adaptatif des images sur les écrans de petits appareils (Chen et al. 2003), l'amélioration de la détection et la reconnaissance d'objets (Navalpakkam et al. 2006) (Rutishauser et al. 2004), la recherche des images sur internet (Wang et al. 2012), la détection de piétons en temps réel (Montabone et al. 2010). Dans (Borba et al. 2006), la saillance est utilisé pour identifier et regrouper des images « similaires », ou des « objets » similaires. Cette approche a notamment été utilisée en robotique autonome. Les résultats expérimentaux montrent que l'utilisation de l'attention visuelle permet d'augmenter le taux de reconnaissance d'objets. Dans (Frintrop et al. 2009), les auteurs ont aussi proposé d'utiliser l'attention visuelle en robotique pour réaliser du suivi d'objet.

L'utilisation de la profondeur pour le calcul de la saillance à fait l'objet, jusqu'à aujourd'hui, de peu de travaux. On peut noter, dans un premier temps, l'utilisation de la stéréovision pour le calcul de la profondeur afin de l'intégrer dans le calcul de la saillance(Niu et al. 2012) et (Maki et al. 2000). Cependant, l'arrivée de capteurs peuso-3D a permis d'obtenir plus facilement cette information. Depuis quelques années, quelques recherches ont fait l'objet de travaux sur le calcul de la saillance à partir de capteurs 3D. L'objectif de la section suivante est de présenter l'état de l'art concernant ces travaux.

1.4.2 Aperçu des techniques existantes dans la saillance 3D

La plupart des méthodes actuelles de saillance reposent sur des données telles que la couleur, la luminance et la texture tout en ignorant les informations de profondeur. Cette information est cependant un facteur important pour la détection des régions visuellement saillantes dans les images notamment lors de la détection de plusieurs zones saillantes (Lang et al. 2012). Il est cependant important de remarquer que dans l'étude proposée par Lang et .al, les auteurs effectuent une étude comparative de la prédiction de la fixation des yeux, plutôt que de la détection des objets saillants, dans les scènes 2D et 3D après avoir collecté une base de 600 paires d'images 2D et 3D correspondantes. Les travaux (Desingh et al. 2013) (Ciptadi et al. 2013) mettent l'accent quant à eux sur la tâche de détection de régions saillantes (autres que

les objets saillants) à partir d'images RGBD: Dans l'article de Desingh (Desingh et al. 2013), les auteurs ont proposé de calculer une carte de saillance en fusionnant une carte de saillance basée sur une image RGB avec une carte de saillance basée sur les informations de profondeur.

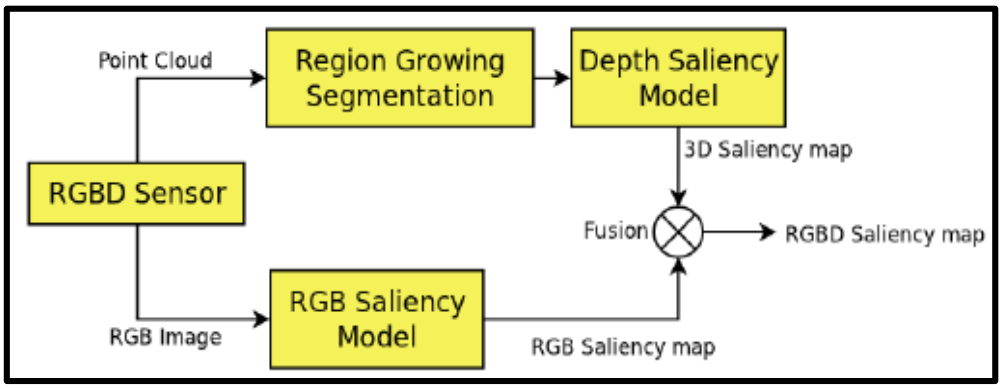


Figure 19: Schéma bloc de la méthode de fusion (Desingh et al. 2013).

Le diagramme de la figure 19 résume cette approche. La carte de saillance RGB est obtenue en utilisant des techniques de saillance 'standard fusion' (Desingh et al. 2013). La carte de saillance de la profondeur est quant à elle obtenu par segmentation du nuage de point (image 2D en niveau gris) fourni par la Kinect. Les deux cartes sont ensuite fusionnées. Les expériences sur des ensembles de données de référence (Peng et al. 2014) contenant des images RGB associés à des informations de profondeurs montrent que la méthode proposée entraîne une amélioration de la détection des zones saillantes (augmentation moyenne des scores ROC d'environ 9% par rapport aux modèles de saillance 2D, pour un AUC final = 0.82.). L'approche présentée par (Ciptadi et al. 2013), quant à elle, propose d'utiliser le calcul d'une distance euclidienne pour caractériser une « dissemblance » entre pixels. Cette distance se base notamment sur la couleur et la profondeur. L'approche de Cheng (Cheng et al. 2014) propose une méthode de saillance 3D basée sur l'hypothèse que l'objet le plus proche est plus saillant et positionné au centre de l'image est plus saillant. La figure 20 illustre quelques exemples obtenus par cette méthode tel que : (a) représente l'image RGB, (b) la saillance en 2D, (c) la carte de profondeur, (d) le contraste en couleur met en évidence l'objet le plus saillant dans l'espace de couleur, (e) le contraste de profondeur extrait l'objet sortant de son environnement, (f) la polarisation spatiale extrait l'objet le plus proche au centre. Le résultat final est une fusion utilisant les opérations de multiplication (g), d'addition (h) et max (i).

Dans (Peng et al. 2014), les auteurs ont proposés une solution basée sur l'analyse de contraste « multi-contextuel ». La figure 21 illustre la solution proposée: la saillance de

profondeur est produite par la méthode des contrastes « multi-contextuel », tandis que la saillance RGB est estimée par des méthodes existantes de saillance 2D. Les deux cartes sont ensuite fusionnées. Il faut aussi indiquer que lors de cette étude, les auteurs ont construit une base de données de 1000 images sur laquelle ils ont pu évaluer leur solution.

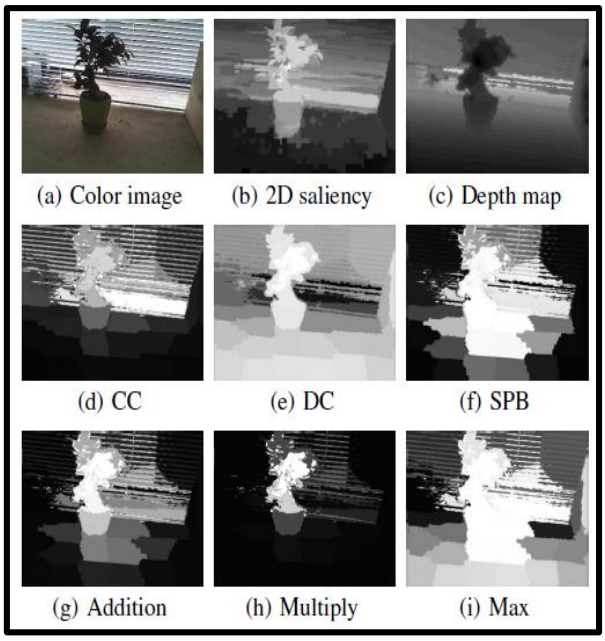


Figure 20: Exemples de détection de la saillance (Cheng et al. 2014).

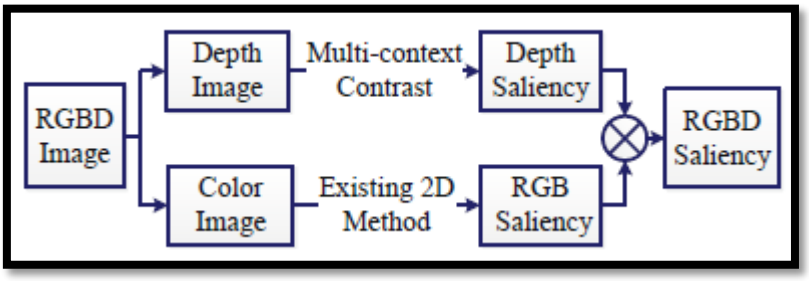


Figure 21 : Schéma bloc de la méthode de la détection de la saillance (Peng et al. 2014).

Le résultat de l'évaluation de cette approche est illustré par la figure 22 qui montre une comparaison des scores (F-measure et AUC) obtenues suivant différentes méthodes.

Les auteurs ont comparé leur modèle de contraste multi-contextuel avec 8 approches de détection de saillance RGBD existante, ils distinguent 6 approches de saillance 2D après fusion de la profondeur: DSR(Li et al. 2013), MR (Yang et al. 2013), HS (Yan et al. 2013), CB (Jiang et al. 2011), PCA (Margolin et al. 2013), LR (Shen et al. 2012) et 2 méthodes de détection de région saillante RGBD récemment proposées qui sont : SVR (Desingh et al. 2013) et LS (Ciptadi et al. 2013).

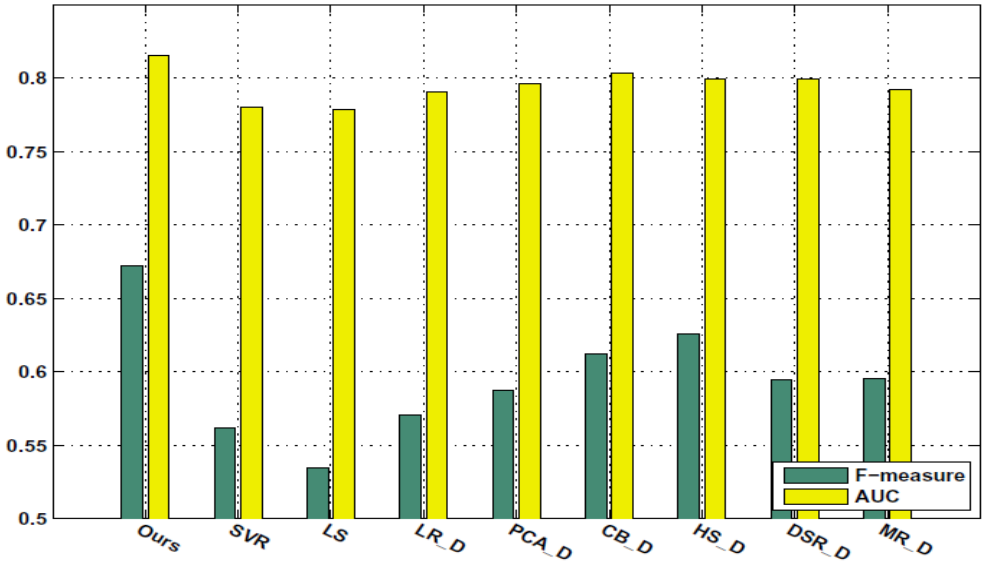


Figure 22 : Comparaisons quantitatives de l'approche proposée avec 8 approches RGBD concurrentes.(Peng et al. 2014)

1.5 Conclusion

Dans ce chapitre, nous avons fourni un aperçu de l'état de l'art sur les différentes technologies d'acquisition de l'information de profondeur, et la détection des objets saillants 2D et RGB-D. La première partie se focalise sur les capteurs d'acquisition de la profondeur et les technologies implémentées dans ces capteurs. Ces capteurs sont conçus pour changer d'une manière significative la vision de l'environnement réel par l'ordinateur. Cependant, les capteurs existants actuellement présentent de réelles limitations comme la présence des points noirs dans les images de profondeurs. Ce manque d'information dans l'image de profondeur rend l'exploitation de l'information de profondeur complexe, il est donc important de s'assurer que l'image de profondeur est une représentation 3D significative de la scène. En revanche, pour notre approche, nous avons opté pour l'utilisation du capteur Kinect. Vu la sensibilité du capteur Kinect dans un environnement encombré (occlusion, présence des trous dans la carte de profondeur...etc.), les informations géométriques (ongle vertical et horizontal) sont influencées par cet environnement.

Dans ce chapitre, nous avons fourni aussi une étude approfondie sur la détection des objets saillants 2D et RGB-D. L'état de l'art montre que la saillance produite en profondeur peut servir de complément utile aux modèles existants de saillance basée sur les couleurs, en

particulier lorsque les objets sont plus près de la caméra, qui vont avoir un contraste de profondeur élevé par rapport à l'arrière-plan.

Nous allons utiliser dans le chapitre 2 la base de donnée de 1000 images proposée dans (Peng et al. 2014) et la base de donnée proposée dans (Cheng et al. 2014), ainsi que comparer notre méthode avec les autres méthodes présentées dans la figure 22.

2 Chapitre 2. Modélisation de la saillance visuelle en 3D

2.1 Introduction

La saillance visuelle caractérise certaines parties d'une scène - pouvant être des objets ou des régions de celle-ci - qui apparaissent à un observateur comme se démarquant du reste de cette scène et notamment par rapport à leur voisinage. En d'autres termes, si une partie de la scène est visuellement plus saillante, intuitivement cela signifie que cette partie est visuellement plus importante ou plus attractive.

Etudiée activement dans le domaine de la psychologie cognitive, la notion de la saillance (visuelle) rejoint la perception et l'attention visuelle de l'humain (Dehaene 2014; Richard 2004). Elle est disséquée sous deux aspects : l'attention visuelle réflexive (liant l'inconscient) et l'attention visuelle intentionnelle (ou consciente). Les travaux de la présente thèse doctorale ne traitent pas ces aspects et se limitent aux contours de cette notion dans le cadre de la science et technologies de l'information et de la communication (STIC).

Dans le domaine STIC et notamment dans son périmètre du traitement de l'information visuelle, la notion de la saillance est liée à la détection et l'extraction d'objets ou régions d'une image représentant une certaine relevance visuelle (importance visuelle) par rapport aux autres objets ou régions de cette même image. Il existe deux principaux mécanismes pour traiter l'information dans le contexte de la saillance visuelle, notamment au regard de l'orientation choisie pour le traitement de l'information : « Botton-up » et « Top down ».

Le mécanisme «Top-down», lequel construit la détection et l'extraction de la saillance visuelle sur la base des notions préalables impliquant la connaissance (préalable) liée à l'environnement observé. En d'autres termes, ce type de mécanismes (ou algorithmes) tente de prendre en compte des phénomènes cognitifs de l'observateur humain : comme la connaissance, l'expérience, sémantique associée aux objets, etc... Ce mécanisme («Top-down») peut, en quelque sorte, rappeler l'aspect « intentionnelle » de l'attention visuelle.

Le mécanisme «Botton-up», lequel contrairement au «Top-down» construit la détection et l'extraction de la saillance visuelle indépendamment de considérations préalables : en d'autres termes, ce type d'approches détecte les objets et/ou des régions potentiellement

saillants sans catégoriser ou graduer leur pertinence par rapport à ce que l'humain aurait considéré comme étant « visuellement pertinent ». Mettant en jeu les caractéristiques bas-niveau de l'image (couleur, lumière et la profondeur des objets par rapport au capteur, etc.), ce mécanisme («Botton-up») rappelle, en quelque sorte, l'aspect « réflexif » de l'attention visuelle.

Dans notre cas, il s'agit de la saillance « Botton-up » excluant toute hypothèse préalable sur la nature, le rôle contextuel ou la sémantique des objets potentiellement saillants.

Dans ce chapitre nous proposons une approche basée sur la détection de la saillance en 3D dans un environnement intérieur. Cette approche utilise les informations d'une image RGB ainsi que celle de la profondeur qui sont fournies par les capteurs de la Kinect. En combinant des algorithmes de détection 3D d'objets saillants, l'approche qui est proposée permet de réaliser une détection d'éléments pertinents dans un environnement 3D. Dans ce contexte, la perception 3D permet de réduire la complexité computationnelle inhérente à la vision 3D en une tâche de calcul 2D.

2.2 Notre approche de la détection de la saillance en 3D

La Kinect de Microsoft est une caméra 3D qui fournit directement des informations de couleur et de profondeur (RGB-D). Même si son champ de vision est limité (environ 60 degrés vertical et 40 degrés horizontal) et que les données sont bruitées, ce capteur est aujourd'hui utilisé dans de nombreux domaines. La Kinect fournit des informations sur la profondeur dans un intervalle de 0,6 à 8 mètres. Bien que les technologies utilisées sur la Kinect limitent son domaine d'utilisation, elle est parfaitement adaptée pour une utilisation dans un environnement intérieur.

Un certain nombre de travaux récents relatif au calcul de la saillance basée sur des informations 3D, ont mis en évidence la pertinence de l'information de la profondeur dans l'amélioration de la détection des éléments saillants dans l'environnement 3D (Desingh et al. 2013; Peng et al. 2014) (Cheng et al. 2014; Tang et al. 2016). Si tous ces travaux s'accordent à montrer la pertinence de l'utilisation de l'information de la profondeur, ils font cependant tous l'hypothèse que les objets les plus saillants se situent généralement au centre de l'image.

Comme cela a déjà été mentionné dans la section introductive, l'approche étudiée tire son premier avantage d'un traitement 2D d'informations visuelles en utilisant conjointement les

informations de la couleur d'une image RGB ainsi que celle de la profondeur fournies par les capteurs de la Kinect. La figure 23 illustre le schéma fonctionnel de l'architecture globale du système de saillance visuelle proposée. Outre le capteur 3D, à savoir Kinect, notre système comprend cinq blocs. Le premier implique l'image RGB fournie par la Kinect et se compose d'un ensemble de tâches de traitement résultant de la détection de la chromacité et de la luminosité de l'image. Le résultat, appelé "Color-luminance Saliency Map" (CLSM) résulte du calcul de deux cartes de saillances élémentaires associant une carte de saillance globale (c'est-à-dire Global Saliency Map) et une carte de saillance locale (Local Saliency Map) (Maximili et al. 2011) (Ramík et al. 2014) (Moreno et al. 2012). Le second implique l'image dite de profondeur (fournie par le capteur de la Kinect). Il extrait une caractéristique de saillance à partir des informations fournies sous la forme d'une image en niveau de gris caractérisant la profondeur. L'image résultante est appelée «Depth Saliency Map» (DSM).

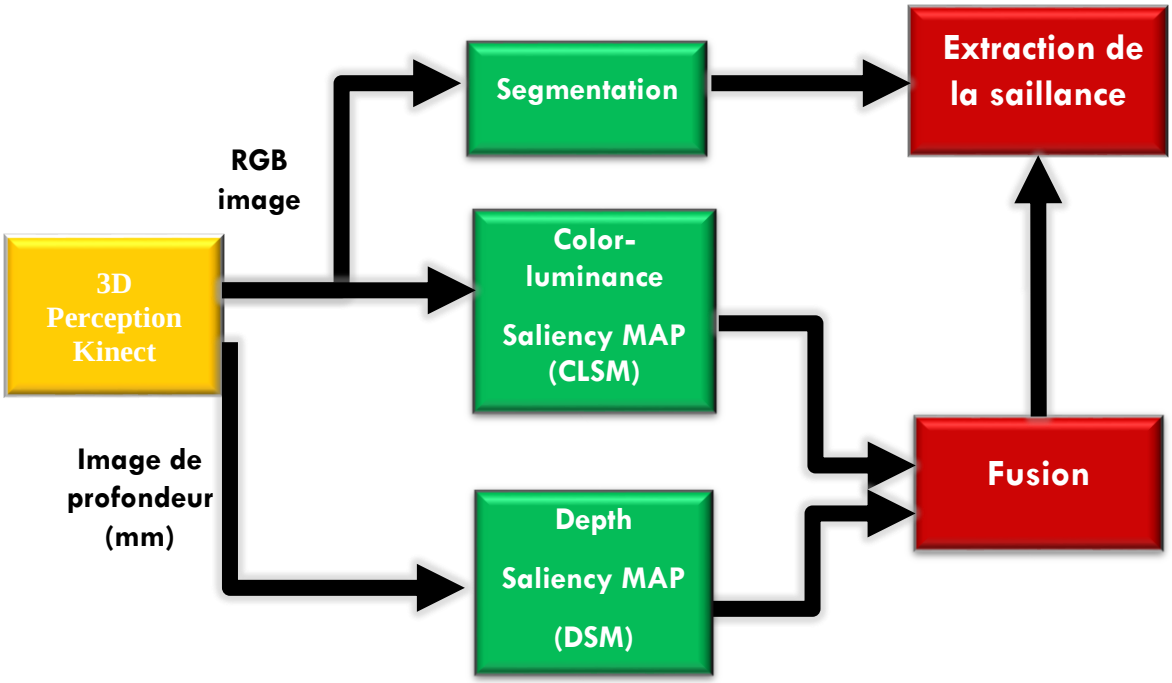


Figure 23: Schéma de principe du système proposé

La troisième unité effectue une segmentation. En principe, toute technique de segmentation d'image peut être utilisée pour effectuer cette tâche. Cependant, la qualité des éléments segmentés influe sur le contour des objets détectés. Nous avons utilisé la technique décrite dans (Moreno et al. 2012) offrant un compromis entre la qualité de segmentation et le temps de calcul. La quatrième opération est consacrée à la fusion non linéaire du CLSM et du

DSM. Le résultat est une image, appelée "Final Saliency Map" (FSM), qui mène à l'opération d'extraction de la saillance. Enfin, la dernière opération concerne l'extraction de masque de la saillance, obtenu à partir de l'image FSM et segmentée.

2.2.1 La détection de la saillance 2D en couleur et lumière

Avant de d'écrire plus précisément les calculs permettant d'obtenir la carte de saillance finale, nous allons commencer par définir quelques notations. Dans la suite de ce mémoire, nous supposerons que I_{YCC} est l'image représentée dans l'espace couleur YCrCb par ses pixels $I_{YCC}(x)$, où $x \in \mathrm{N}^2$ décris la position 2D d'un pixel. $I_Y(x)$, $I_{Cr}(x)$ et $I_{Cb}(x)$ sont les valeurs dans les canaux Y, Cr et Cb respectivement. De même, nous supposerons que $I_{RGB}(x)$ est la même image dans l'espace couleur RGB et où $I_R(x)$, $I_G(x)$ et $I_B(x)$ sont les valeurs des couleurs dans les canaux R, G et B respectivement. Finalement, nous supposerons que $\overline{I_y}$, $\overline{I_{Cr}}, \overline{I_{Cb}}$ sont les valeurs moyennes dans les canaux Y, Cr et Cb. La carte de saillance met en évidence des éléments ou des objets pertinents dans l'image.

La carte de saillance global $M_G(x)$ est le résultat d'une fusion non linéaire de deux cartes élémentaires basées sur les informations de la luminance et la chromaticité et notées respectivement $M_Y(x)$ et $M_{CrCb}(x)$. Les équations (14), (15) et (16) détaillent le calcul de chacune de ces cartes élémentaires ainsi que le calcul de $M_G(x)$.

$$M_Y(x) = \left\| \overline{I_y} - I_Y(x) \right\| \qquad (14)$$

$$M_{CrCb}(x) = \sqrt{\left(\overline{I_{Cr}} - I_{Cr}(x)\right)^2 + \left(\overline{I_{Cb}} - I_{Cb}(x)\right)^2} \qquad (15)$$

$$M_G(x) = \frac{1}{1-e^{-C(x)}}\, M_{CrCb}(x) + \left(1 - \frac{1}{1-e^{-C(x)}}\right) M_y(x) \qquad (16)$$

My : Carte de luminance

M_{CrCb} : Carte de chrominance

Iy : Intensité de luminance d'un pixel

I_{Cr} I_{Cb} : Chromaticité dans la représentation Ycc.

I_R,I_G,I_B : Chromaticité dans la représentation RGB

$C(x)$ est un coefficient calculé dans l'équation 18, dépend de la saturation de chaque pixel

dans l'espace couleur RGB.

$$C_c(x) = Max\ (I_R(x), I_G(x), I_B(x)) - Min\ (I_R(x), I_G(x), I_B(x)) \quad (17)$$

$$C(x) = 10 - 0.5C_c(x) \quad (18)$$

La carte de saillance local ou LSM (Local Saliency Map) (initialement proposée dans (Liu et al. 2011)) est obtenue sur la base du concept d'histogramme « central-surround » qui consiste à comparer une partie de l'image avec son contour.

Soit *P(x)* une fenêtre coulissante de taille *P*, centrée sur un pixel situé en position x, et soit *Q(x)* une autre zone environnante autour du même pixel de taille *Q*, avec $Q - P = P^2$. Définissons l'histogramme central *Hc(x)* comme un histogramme d'intensités de pixels dans la fenêtre *P(x)* avec $h_c(x, k)$ représentant la k–ième valeur de cet histogramme. Enfin, on définit le «Surrounding-Histogram» *Hs(x)* comme un histogramme d'intensités de pixels dans la fenêtre environnante *Q(x)* avec *hs (x, k)* , représentant sa valeur k–ième.

Conformément aux désignations ci-dessus, la fonctionnalité centre-surround $d_{ch}(x)$ est définie comme la somme de différence à travers tous les 256 histogrammes bin.

$$d_{ch}(x) = \sum_{k=1}^{256} \left| \frac{h_c(x,k)}{|H_c(x)|} - \frac{h_s(x,k)}{|H_s(x)|} \right| \quad (19)$$

Le LSM, résultant d'une fusion non linéaire des caractéristiques dites "centre-surround", est obtenu en fonction de l'équation (20), où le coefficient $C_u(x)$ représente la saturation moyenne en couleur d'une fenêtre *P(x),* calculer comme la saturation moyenne de *C(x).*

$$M_L(x) = \frac{1}{1 - e^{-C_u(x)}}\ d_Y(x) + \left(1 - \frac{1}{1 - e^{-C_u(x)}}\right) \mathrm{Max}(\ d_{cr}(x), d_{cb}(x)) \quad (20)$$

$$C_u(x) = \frac{\sum_{l=1}^{p)} C(l)}{p} \quad (21)$$

La carte de saillance finale dont les constituants sont notés $M_f\ (x)$, est une carte résultante d'une fusion conditionnelle impliquant GSM et LSM. Il est calculé en utilisant la relation :

$$M_{CLSM}(x) = \begin{cases} M_L(x) & if\ M_L(x) > M_G(x) \\ \sqrt{M_G(x) M_L(x)} & otherwise \end{cases} \quad (22)$$

2.2.2 La détection de la saillance en profondeur

Le calcul de la carte de saillance en profondeur (DSM : depth saliency map) s'inspire du travail de recherche présenté dans (Desingh et al. 2013) et implique des propriétés statistiques des régions de l'image de profondeur. Cela signifie que l'image de profondeur est divisée en N régions utilisant l'algorithme de segmentation MEAN-SHIFT.

L'étape de prétraitement concerne le traitement des données fournies par les capteurs de Kinect. Les données visuelles (image couleur) sont segmentées et une image binaire résultante est construite(le masque des objets). Il existe plusieurs techniques de segmentation choisies sur la base de la complexité de calcul pour s'adapter aux contraintes de calcul en temps réel (Gonzalez et al. 2002). Cependant, des techniques de traitement plus sophistiquées peuvent être utilisées comme celles proposées par (Moreno et al. 2012) and (Ramik 2012) ou celles de (Ramík et al. 2010).

La méthode de segmentation utilisée dans notre approche s'appelle MEAN SHIFT, elle est basée sur la répartition de la densité des pixels.(Comaniciu et al. 2002) (Comaniciu et al. 2000) (Kheng 2011).

La principale tâche de la méthode MEAN SHIFT est d'estimer la localisation moyenne exacte "m (x)" des données (centre de masse dans la Figure 24) en déterminant le vecteur de décalage à partir de la moyenne initiale (région d'intérêt sur la Figure 24). Le processus sera répété jusqu'à ce que le centre de la région ait une densité de pixels maximale. Le vecteur de décalage moyen suit la direction de l'augmentation maximale de la densité. Le décalage moyen suit la direction du gradient de la densité estimée. Le gradient de la densité estimée indique le nombre de pixels similaires et voisins dans un noyau. La figure 25 montre le résultat d'application de la méthode de segmentation MEAN SHIFT sur une d'image fournie par la Kinect.

Pour calculer l'emplacement moyen m (x) au point x, on utilise l'équation 23 :

$$\mathbf{m}(\mathbf{x}) = \frac{\sum_{i=1}^{n} K(\mathbf{x} - \mathbf{x}_i)\,\mathbf{x}_i}{\sum_{i=1}^{n} K(\mathbf{x} - \mathbf{x}_i)} \qquad (23)$$

n: nombre de points dans le noyau de la région d'intérêt.

x_i: point de données.

x: emplacement moyen initial.

K (x): la fonction du noyau qui indique le nombre d'échantillons x contribue à l'estimation de la moyenne.

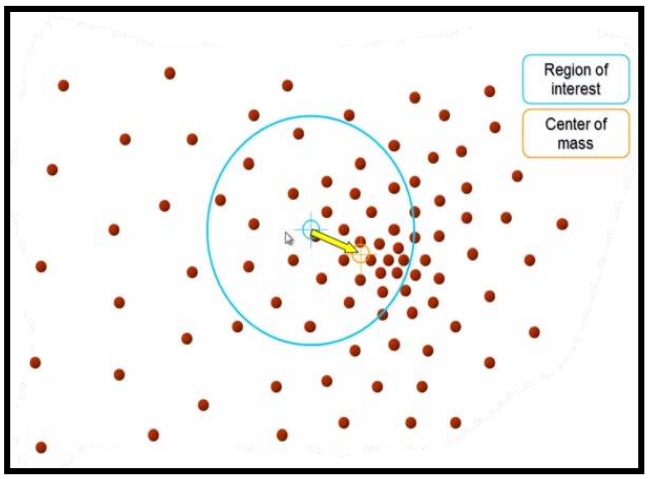


Figure 24 : Le principe de l'algorithme de segmentation MEAN SHIFT (Yilmaz et al. 2006)

Le décalage moyen (MEAN SHIFT) est la différence entre m(x) et x, c'est un algorithme itératif, s'arrête lorsque m(x) = x. Il est calculé itérativement pour obtenir la densité maximale dans le voisinage local.

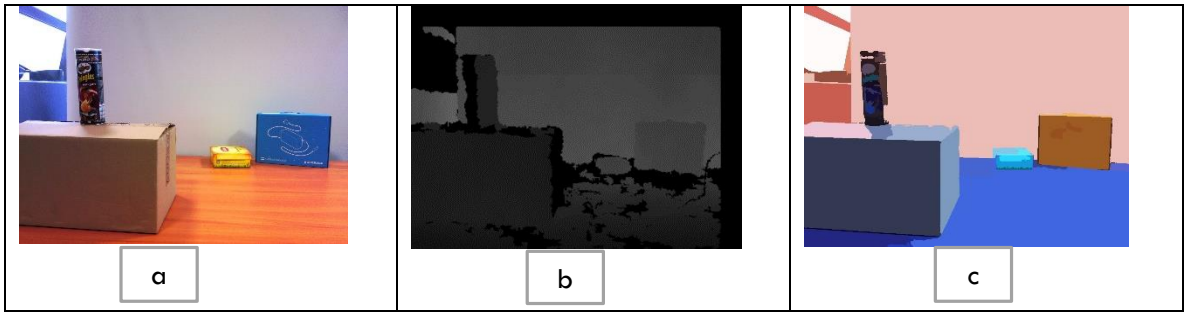


Figure 25: La segmentation par l'algorithme Mean-shift, a) image couleur, b) image profondeur et c) la segmentation.

Une fois que l'image de profondeur est divisée en N régions, nous calculons C_k le contraste représentatif de la k-ieme région (k-ieme parmi les N régions). Z_k correspond à la profondeur du centre de la k-ième région (fournie directement par la Kinect), n_k le nombre de pixels de cette Région et H_k l'histogramme des intensités de pixels de cette région. Soit H_j l'histogramme des intensités de pixels de toutes les autres régions de l'image de profondeur ($j \neq k$) et n_j le nombre de pixels de cette région. En suivant la notation susmentionnée, C_k est

calculé en utilisant l'équation (24), où D_{kj} désigne le produit de l'histogramme H_k et les histogrammes H_j ($j \neq k$).

$$C_k = \frac{2 Z_k n_k \sum_{j \neq k} D_{kj}}{\sum_{j=1}^{N} n_j} \qquad (24)$$

La figure 26 montre la valeur de la profondeur de la région H_k sous forme histogramme, la valeur de la profondeur en échelle de gris est autour de 70. La figure 27 montre la valeur de la profondeur d'une région H_j avec ($j \neq k$), avec j= [1 : n] n : étant le nombre des régions dans l'image.

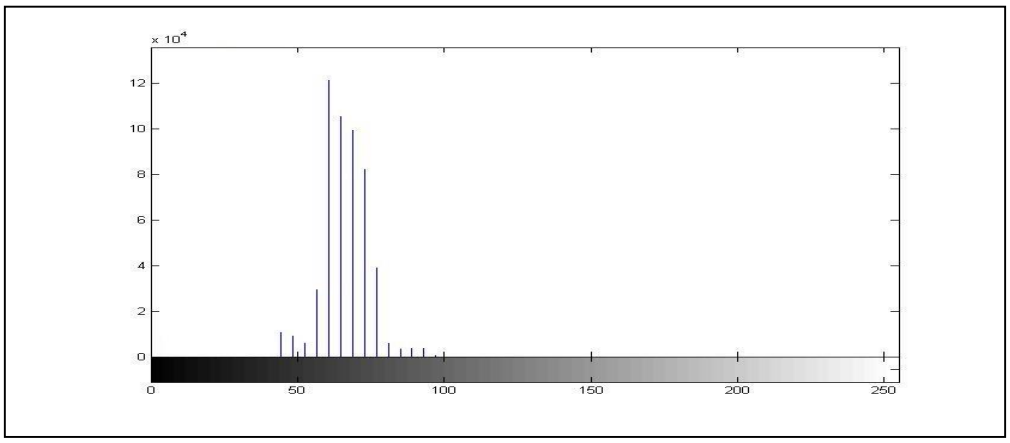


Figure 26 : L'histogramme d'une région H_K dans l'image de profondeur

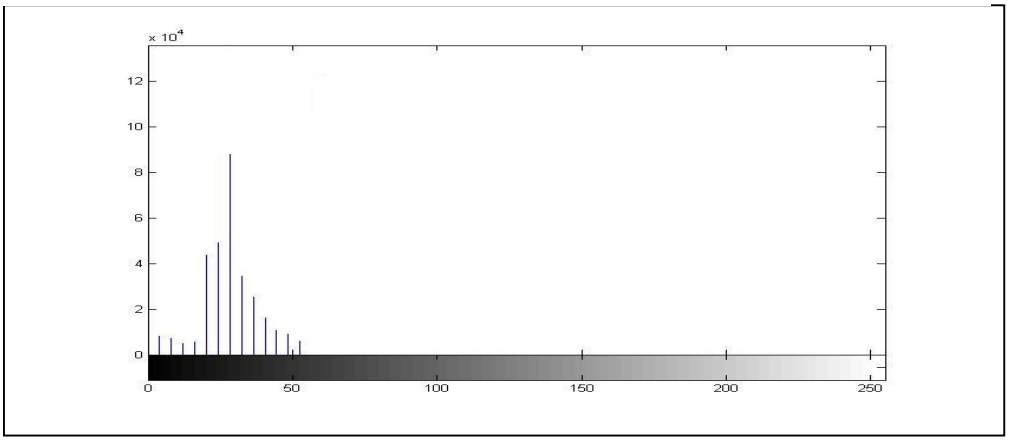


Figure 27: L'histogramme d'une région H_j dans l'image de profondeur

Dans le cas où deux régions se trouvent à la même profondeur par rapport à la Kinect, $D_{kj} = H_k.H_j$ va avoir une valeur très élevée, ainsi que C_k. Ceci implique que S_k va être très faible (voir équation 25), ce qui explique le principe de la saillance en profondeur. En effet, deux régions qui se situent à une même profondeur conduit à une obtention d'une saillance en terme

de profondeur nulle. La saillance de la k-ième région du DSM, désignée par S_k, est alors obtenue à partir de l'équation (25), où C_{max} = Max (C_k) représente le contraste le plus élevé.

$$S_k = 1 - \frac{C_k}{C_{\max}} \qquad (25)$$

Par exemple, dans le cas où H_k= 70 et H_j= 30, la multiplication des deux histogrammes va nous donner une valeur de D_{kj} faible et donc un C_k faible, et par conséquent une valeur de saillance S_k élevée. La valeur Z_k qui représente la profondeur du centre de la région k en centimètres, permets de pondérer la valeur de la profondeur de la région par rapport au capteur Kinect. En effet, une région proche du capteur signifie une valeur de Z_k faible, et dans ce cas le contraste C_k est faible aussi et une valeur de la saillance S_k plus élevée. Ceci implique que plus la région est proche du capteur, plus elle est saillante.

La division de n_k sur $\sum n$ va permettre, quant à elle, de mettre en valeur la superficie d'une région. Plus une région est large et plus elle est saillante. Le contraste C_k valorise donc la profondeur et la surface d'une région dans la scène. Par conséquent, une région large et loin du capteur de la Kinect peut avoir la même saillance qu'une région petite et proche du capteur.

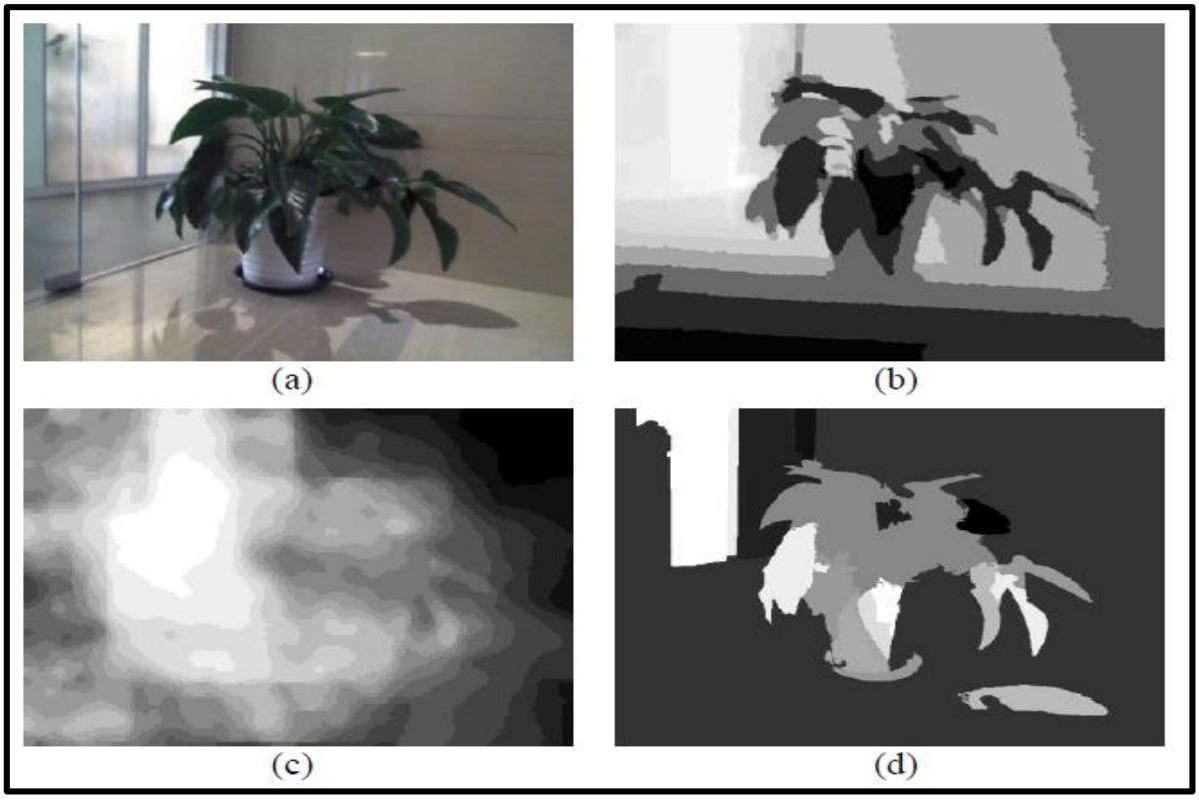


Figure 28: Exemple de CLSM et DSM obtenus à partir d'images RGB et de profondeur fournie par la Kinect.

```
Acquérir une image de couleur et profondeur
Extrait des N régions à partir de l'image couleur

Faire la correspondance de ces régions sur l'image de
profondeur

        Pour k allons de 1 à N

          Calculer le nombre de pixel n de la région k

          Extrait la valeur de Z en centimètres de la région k

          Calculer l'histogramme Hk de la région k avec (j ≠ k)

              Pour j allons de 1 à N  (j ≠ k)

                    Calculer  l'histogramme de Hj

                    Calculer Dkj

         Fin Calculer le contraste Ck de la région k

Calculer le maximum des histogrammes Ck

Calculer la saillance de la région Sk
```

Algorithme 1 : Calcul de la saillance en profondeur

L'algorithme 1 décrit de manière explicite l'ensemble des calculs réalisés pour la saillance Sk. La figure 28 montre un exemple de CLSM et DSM obtenus à partir d'image RGB et de profondeur fournies par la Kinect: (a) image RGB, (b) image de profondeur correspondante, (c) CLSM et (d) DSM.

2.2.3 Génération de la carte et le masque de saillance finale

Comme mentionné précédemment, le FSM résulte de la fusion non linéaire du CLSM et du DSM. La fusion est effectuée région par région (pour toutes les régions N), impliquant les zones correspondantes en images RGB et en profondeur. La FSM résultante, dont les constituants de sa k-ième région sont désignés par M_K^{FSM} (x), est calculée selon l'équation (26), où M_k^{CLSM} (x) est le k-ieme constituant de la région de la CLSM et correspondant à un pixel situé à la coordonnée x, M_k^{DSM} (x) est la DSM de la même région et correspondant au même pixel. À titre d'exemple de FSM, la figure 29 donne la carte finale obtenue issue de CLSM et DSM représentée sur la figure 28.

$$M_k^{FSM}(x)=\left(1-\frac{1}{1+e^{-\frac{C_k}{C_{\max}}}}\right)M_k^{CLSM}(x)+\frac{1}{1+e^{-\frac{C_k}{C_{\max}}}}M_k^{DSM}(x) \quad (26)$$



Figure 29: Exemples de FSM (à gauche) et le masque d'extraction de la saillance (à droit).

Le masque d'extraction d'objets saillants est obtenu à partir d'un filtrage basé sur le seuillage de la FSM. L'image de droite de la figure 29 donne le masque d'extraction de la saillance obtenue à partir du DSM représenté par l'image gauche de la figure 29. Dans le présent travail, le calcul du seuil a été réalisé de manière empirique à l'aide d'un ensemble de 22 images représentatives (22 parmi 1000 images) de la base de données de référence disponibles sur http://sites.google.com/site/rgbdsaliency (Peng et al. 2014). L'ensemble de données de référence comprend 1000 images RGB avec des images de profondeur correspondantes et les éléments saillants cibles (« Ground-truth », éléments devant être détectés) dans chaque paysage. Les objets saillants (ground-truth) sont fournis sous forme de masques binaires, en extrayant précisément l'élément principal cible de chaque image. Chaque image contient un élément saillant unique dans cet ensemble de données de référence. Le seuil a été calculé sur la base du critères F-mesure standard donné par l'équation (27), où *P* désigne la "précision", R signifie "Rappel", TP (True-Positive) pour le nombre d'échantillons de saillance qui ont été correctement détecté, FP (False-Positive) pour le nombre d'échantillons incorrectement détectés dans l'ensemble des éléments saillant) et FN (False-négative) pour le nombre d'échantillons n'appartenant pas à la catégorie correspondante. La valeur de la F-mesure maximisant le seuil a été retenu.

$$F_{Measure} = \frac{(1+0.5)P \times R}{0.5\ P + R} \qquad (27)$$

$$avec$$

$$P = \frac{TP}{TP + FP} \quad et \quad R = \frac{TP}{TP + FN}$$

En d'autres termes, si l'objet saillant de la cible d'une image est correctement détecté (c'est-à-dire la configuration dans l'ensemble des éléments potentiellement saillants), l'échantillon est considéré comme TP autrement il est qualifié de FP. En fait, contrairement aux autres techniques (qui supposent que l'objet unique et pertinent est situé dans la zone centrale de l'image inspecté), l'approche présentée est capable de détecter tout élément potentiel du paysage, sans aucune hypothèse restrictive.

2.2.4 Résultats expérimentaux et validation

Comme mentionné dans la section précédente, la validation a été effectuée en utilisant la base de données de référence fournie par (Peng et al. 2014). Cependant, nous avons étendu la validation expérimentale en utilisant une autre base de données de référence fournie par (Cheng et al. 2014) contenant 135 images. Comme la première base de données, cette deuxième base de données de référence fournit pour chaque image RGB l'image de profondeur correspondante et le « ground-truth ». L'évaluation a été réalisée en utilisant la F-mesure ainsi que l'AUC "Area Under Curve" (Fawcett 2006), mesurant la ressemblance dans le cadre de la "tâche de classification" (Riche et al. 2013).

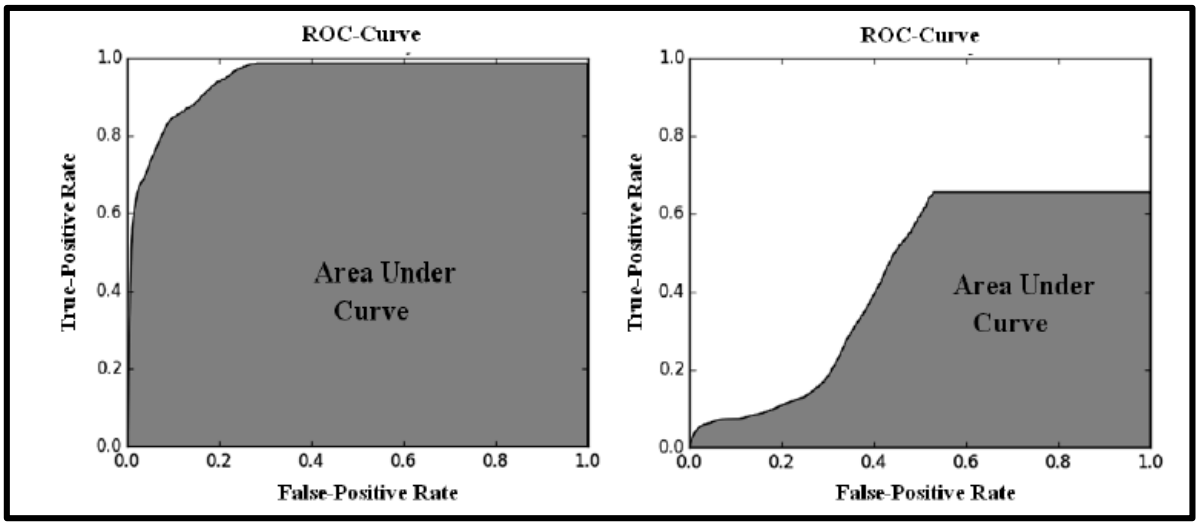


Figure 30: Exemples de courbe ROC et la zone correspondante Areas Under Curve.

	Nombre d'image	AUC score	F-mesure
Base de données 1 (Peng et al. 2014)	1000	0.84	0.86
Base de données 2 (Cheng et al. 2014)	135	0.86	0.88

Tableau 4: Pourcentage de détections correctes de l'objet sur le test de l'ensemble d'images à l'aide du cadre de détection Viola-Jones.

La mesure AUC, calcule le score suivant une tâche de classification pour classer les pixels comme appartenant à la classe de "pixels saillants" ou à la classe des "pixels d'arrière-plan». La courbe ROC (Receiver Operating Characteristic curve) est un diagramme des taux de classification reliant la classification des pixels comme "vrais positifs" (TP) et "faux positif" (FP). Dans ces termes, la courbe ROC représente le taux de TP par rapport au taux de FP. Le score de l'AUC est obtenue en calculant la surface de la zone sous la courbe (utilisant par exemple l'intégrale pour le calcul de surface) de la courbe appelée ROC. Le score le plus élevé possible est égal à 1, tandis qu'une carte de saillance uniformément aléatoire atteint le score de 0,5, ce qui signifie qu'une telle mesure évalue la qualité de la technique évaluée par rapport au processus aléatoire. La figure 30 montre deux exemples de courbes ROC avec les zones correspondantes sous la courbe. Le premier (diagramme du côté gauche) correspond à une classification précise correspondant parfaitement au « ground-truth » tandis que le second (diagramme du côté droit) correspond à une très mauvaise classification.

Les figures 31 et 32 donnent quelques échantillons de résultats obtenus, illustrant les images d'entrées RGB (a) et de profondeur (b), les cartes de saillance finale correspondantes (c), les masques d'extraction correspondants (d), les éléments cibles extraits (e) et les « ground-truth ».

L'évaluation des deux ensembles de données de référence a conduit aux scores résumés dans le tableau 4. En se référant aux échantillons représentés à la figure 29, ainsi qu'aux résultats globaux résultant des deux bases de données de référence, on peut noter la très bonne précision de la détection des éléments saillants cibles. Ce fait est confirmé par des scores issus des mesures de l'AUC et de la F-mesure. Les scores élevés (0,86 pour la première base de données et 0,84 pour la deuxième base de données), comparables par rapport aux autres méthodes (voir figure 34), proposé dans approche mettent en évidence les performances de celle-ci pour les deux bases de données.

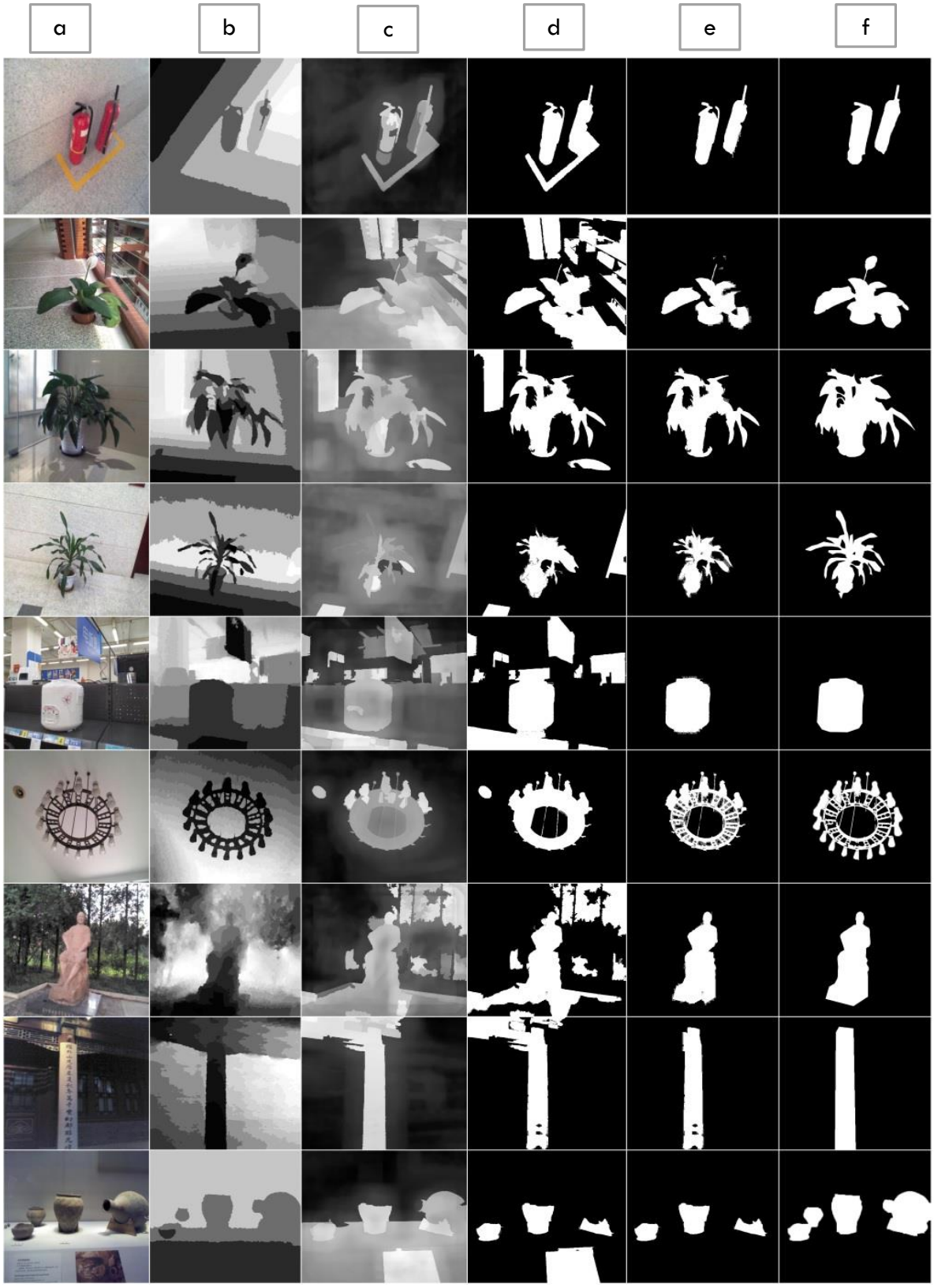


Figure 31: Les échantillons des résultats obtenus, illustrant : (a) les images RGB. (b) les images de profondeur. (c) les cartes de saillance finale correspondantes. (d) les masques extraites. (e) les éléments ciblent extraits. (f) Ground-true.

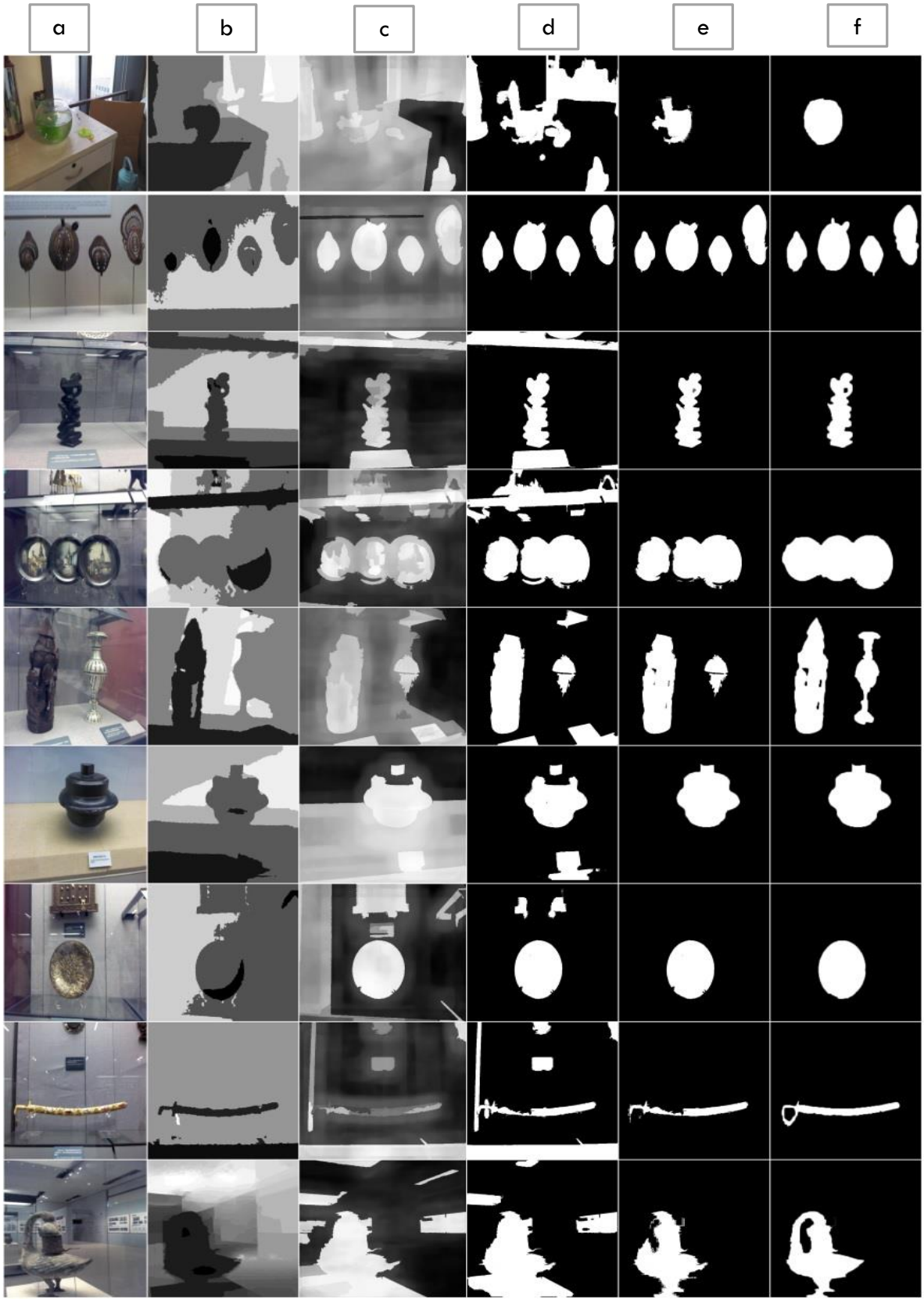


Figure 32 (suite): Les échantillons des résultats obtenus, illustrant : (a) les images RGB. (b) les images de profondeur. (c) les cartes de saillance finale correspondantes. (d) les masques extraits. (e) les éléments cibles extraits. (f) Ground-true

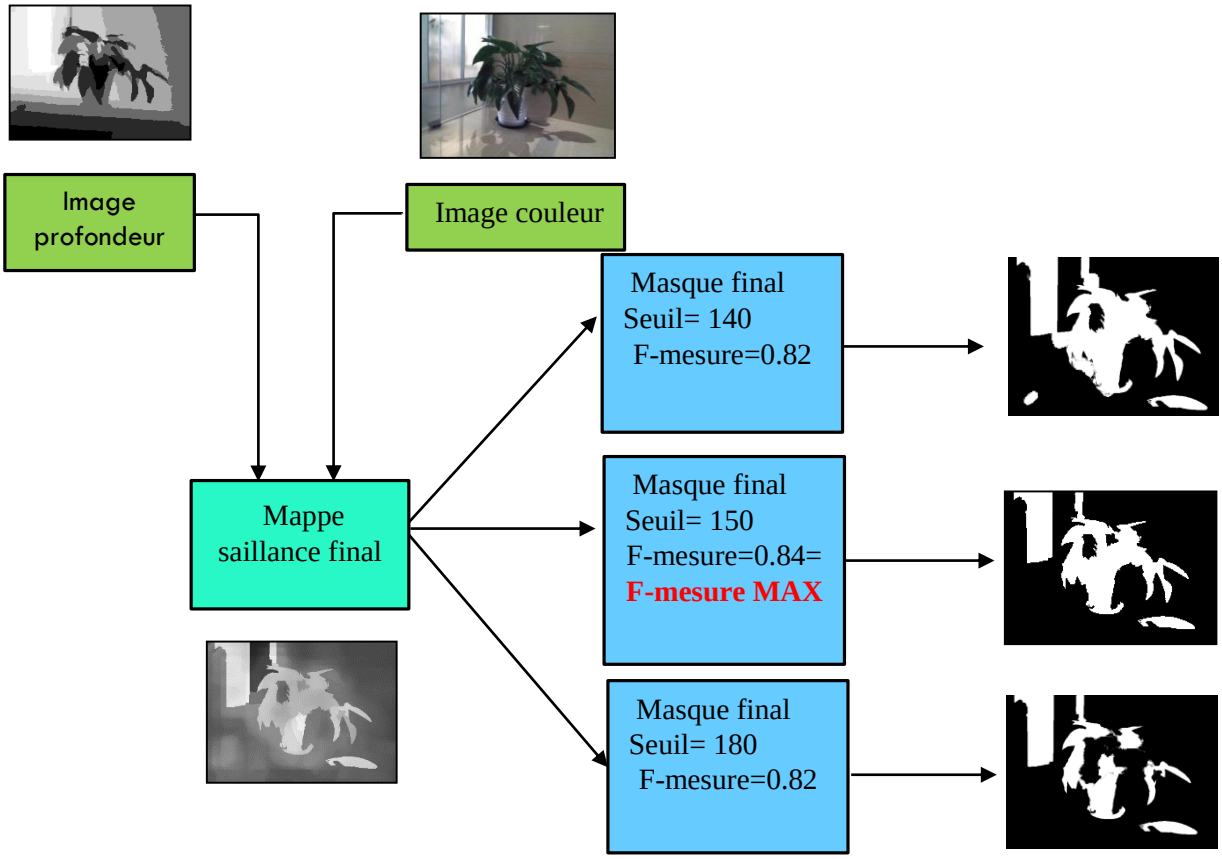


Figure 33: Diagramme de la méthode d'évaluation

Apres avoir obtenu la carte de saillance final, nous générons 26 masques de saillance obtenus grâce aux 26 seuils choisis avec un pas de 10 parmi les 255 niveaux, nous calculons le F-mesure de chaque masque puis nous choisissons notre masque final qui a un F-mesure le plus élevé (voir figure 33 et algorithme2).

```
Acquérir une image de la carte finale de la saillance M^FSM

        Pour seuil allons de 1 : avec un pas de 10 : à 255
            Extrait le masque final de saillance
            Calculer F-mesure
Calculer F-mesure MAX
Déduire le masque de saillance final
```

Algorithme2 : Extraction du masque de saillance finale

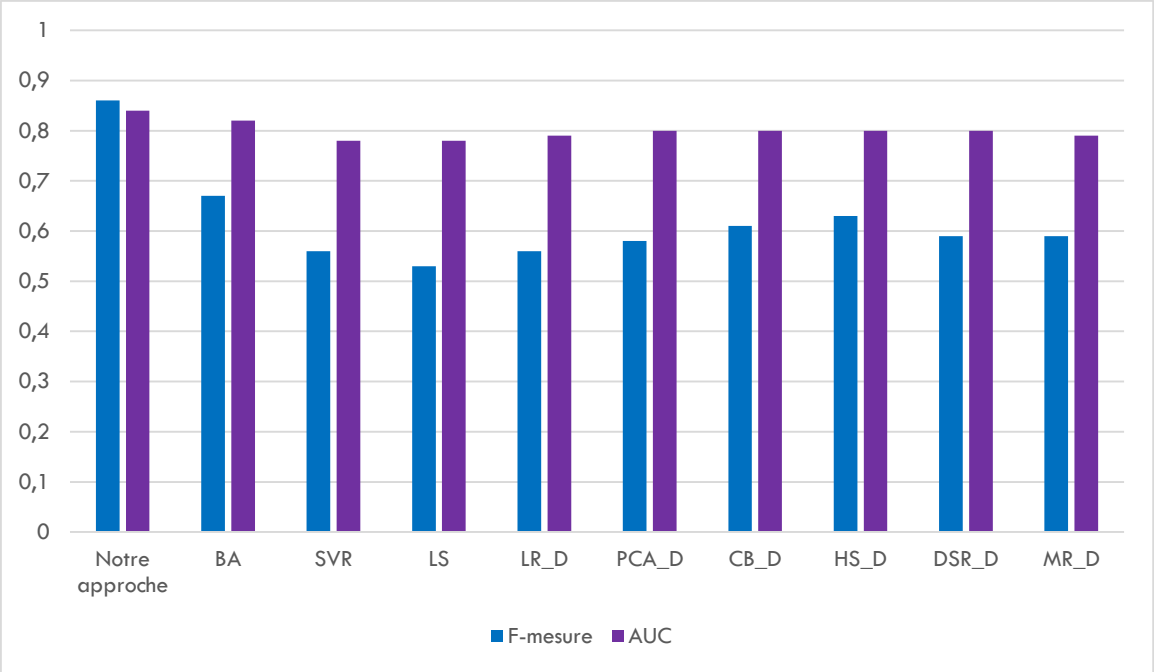


Figure 34: Comparaisons quantitatives de notre approche avec 9 approches concurrentes pour la détection de la saillance en 3D.

Nous avons comparé les scores F-mesure et AUC avec 9 autres méthodes déjà évaluées sur la même base de données de 1000 images. La méthode BA de (Peng et al. 2014) propose une stratégie de fusion simple qui introduit dans les modèles de saillance 2D basés sur RVB une intensité induite par la profondeur. Plus précisément, la saillance induite par la profondeur est produite par la méthode de contraste multi-contextuel. Les autres méthodes étaient déjà comparées dans le benchmark de (Peng et al. 2014) sur la même base de données, nous citons 8 approches de détection de saillance RGBD développés par (Peng et al. 2014) où ils distinguent 6 approches de saillance 2D après fusion de la profondeur: DSR(Li et al. 2013), MR (Yang et al. 2013), HS (Yan et al. 2013), CB (Jiang et al. 2011), PCA (Margolin et al. 2013), LR (Shen et al. 2012) et 2 méthodes de détection de région saillante RGBD proposées qui sont : SVR (Desingh et al. 2013) et LS (Ciptadi et al. 2013). Le résultat de l'évaluation de notre approche comparée avec d'autres approches est illustré dans la figure 34.

2.3 Conclusion

Nous avons présenté une approche basée sur la perception de la saillance 3D dans un environnement intérieur, combinant des algorithmes de détection des objets saillants en couleur-lumière et profondeur.

D’une part, l’approche étudiée recherche une détection autonome d’objets pertinents dans l’environnement. La perception 3D permet de réduire la complexité de calcul de vision 3D en une tâche de calcul 2D en traitant l'information visuelle 3D dans un cadre d'images 2D. D'autre part, le fondement statistique de l'approche étudiée apporte une base théorique solide. En outre, la base statistique susmentionnée soulève une nature ascendante attrayante de la détection de la saillance, ce qui limite l'hypothèse antérieure concernant les éléments potentiellement saillants (hypothèses proposées dans l’état de l’art comme : objet au centre de l’image est plus saillant, pas d’objet saillant dans les coins de l’image...etc.). Enfin, sa construction à base de fusion des fonctionnalités de saillance lui permet de bénéficier d'informations visuelles multi-résolution qui renforcent la précision de l'approche étudiée. L'évaluation du concept proposé a abouti à des scores élevés (AUC=0,86 pour la première base de données et 0,84 pour la deuxième base de données) mettant en évidence une précision élevée de la saillance détectée.

Notre approche prend en compte plusieurs paramètres, les paramètres liés au calcul de la saillance en 2D qui sont la luminance et la chrominance, et les paramètres liés au calcul de la saillance en profondeur qui sont : l’éloignement de l’objet par rapport au capteur Kinect sous l’hypothèse que l’objet le plus proche est le plus saillant, ainsi que le paramètre de surface de l’objet par rapport aux autres objets existants dans la scène. Cependant, elle ne prend pas en compte le paramètre de localisation sous l’hypothèse que l’objet le plus proche du centre de l’image est le plus saillant, ainsi que d’autres paramètres complémentaires comme : l’objet saillant ne peut pas être dans les coins ou le cadre de l’image. Elle détecte plusieurs objets saillants dans l’image contrairement aux autres méthodes qui se limitent à un seul objet saillant dans l’image qui se trouve au centre de l’image généralement.

Les résultats expérimentaux ont démontré que l'information sur la profondeur est un complément utile aux modèles existants basés sur la couleur et la lumière, en particulier lorsque les objets restent plus près du capteur Kinect, ont un contraste de profondeur élevé par rapport à l'arrière-plan où on a une plage de profondeur relativement faible. Par rapport à d'autres modèles 3D concurrents, notre modèle RGB-D proposé atteint des performances supérieures et fournit des résultats plus robustes. Dans le futur, cette approche pourrait évoluer de plusieurs façons. Comme l'intégration d'une stratégie adaptée basée sur le seuil pour la construction des masques d'extraction de la saillance.

Dans le chapitre suivant, nous proposons un système nommé ‘Soft Computing’ basé sur l’algorithme ANFIS pour l’estimation métrique de l’information de profondeur (Z) fournis par le capteur ASUS du robot Pepper. Cette information est utile pour calculer la saillance 3D.

Pour vérifier la faisabilité du système ‘soft Computing’, nous proposons une expérience utilisant le capteur Kinect, consistant à proposer une méthode expérimentale pour l’évaluation des distances métriques entre des objets. La méthode étudiée pointe vers la vision et la métrologie de l'environnement du robot, cette métrologie va nous fournir des informations sur la largeur et la hauteur de l’objet ou la surface de l’objet par rapport à son environnement qui est un facteur important.

3 Chapitre 3. Extension et validation du système de perception de la saillance visuelle 3D par un système robotique autonome

3.1 Introduction

Dans ce chapitre nous proposons une approche plus globale d'extension de l'approche proposée précédemment dans le chapitre 2 sur d'autre capteur 3D, comme le capteur de profondeur ASUS intégré dans le robot Pepper. Notre choix d'un capteur de profondeur intégré dans un robot humanoïde, vient de sa capacité de reconnaitre les objets saillants, les caractériser spatialement et naviguer en toute autonomie dans son environnement pour aller chercher ces objets.

Le problème de l'extension de la saillance 3D est dû au type de l'information de la profondeur, fourni par le capteur ASUS sous forme d'une image de profondeur. Cette information de profondeur n'est pas fournie directement en valeur métrique, comme le fait le capteur infrarouge de la Kinect qui donne la valeur de la profondeur métrique pour chaque pixel de l'image. L'information de la profondeur métrique, cette information est nécessaire pour le calcul de la saillance en profondeur (voir Figure 35).

Une manière de trouver la valeur de la profondeur métrique, nous proposons une approche expérimentale pour étalonner le capteur de profondeur ASUS à l'aide du capteur infrarouge de Microsoft Kinect, utilisant l'algorithme d'apprentissage ANFIS.

Pour vérifier la faisabilité de l'algorithme d'apprentissage ANFIS, nous proposons la méthode d'apprentissage illustrée dans la figure 36-Etape1 du système 'Soft Computing' qui vérifie l'estimation métrique de la distance entre deux objets dans l'espace, puis nous comparons la performance de l'algorithme ANFIS avec d'autres algorithmes d'apprentissage, tel que SVR (Support Vector Regression), MLP (MultiLayer Perceptron) et l'Interpolation Bilinéaire.

Dans la figure 36-Etape 2, le système 'Soft Computing' est encore utilisé pour vérifier l'estimation métrique de l'information de la profondeur Z fournie par le capteur ASUS du robot Pepper. Cette information de profondeur Z est utile pour calculer la saillance 3D. En plus, ce

système ouvre d'autres perspectives notamment l'estimation des caractéristiques spatiales, comme la distance métrique entre deux objets, la largeur et la hauteur de l'objet …etc.

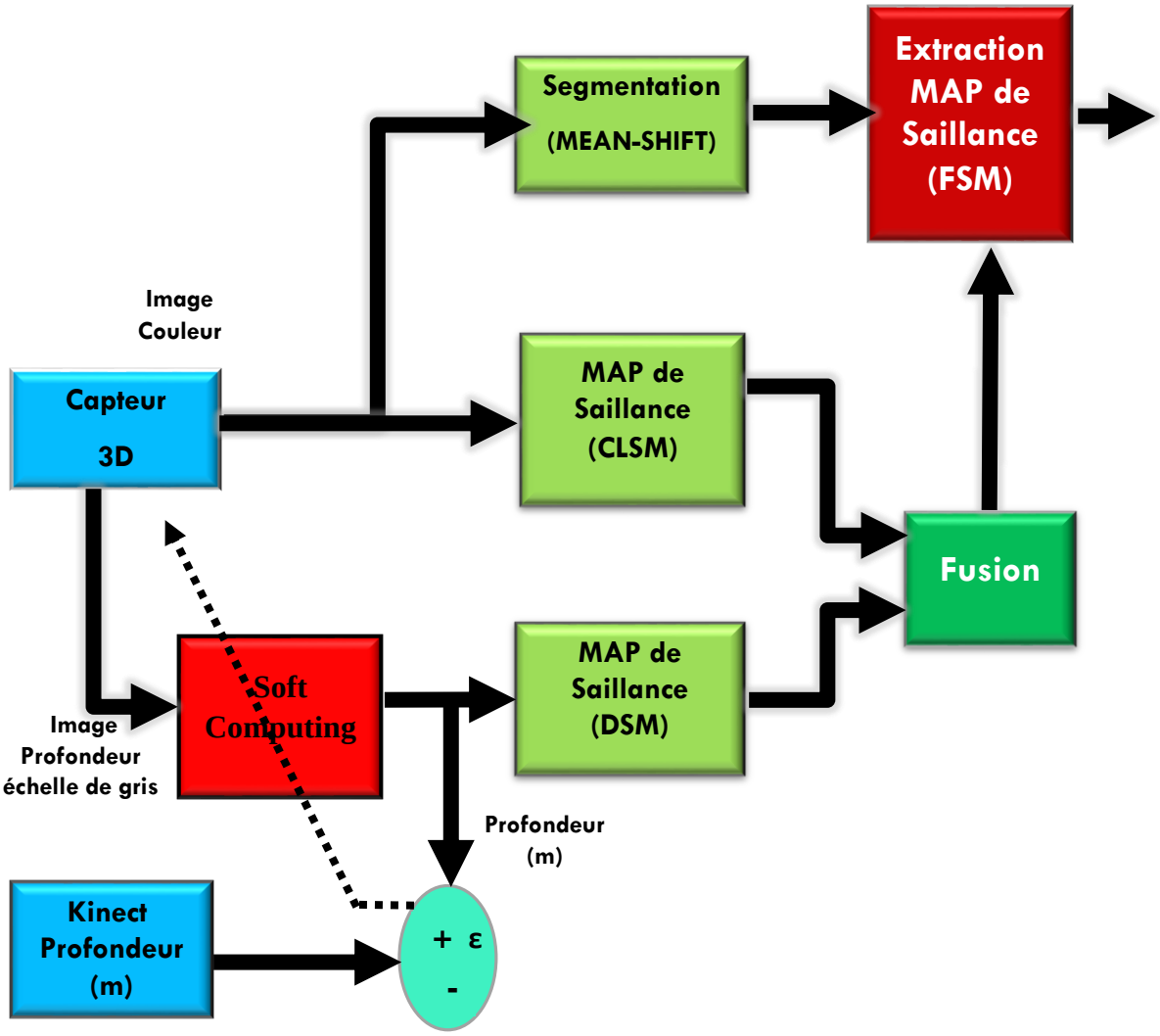


Figure 35: Schéma bloc de l'extension de la saillance 3D, implanté sur un capteur 3D

A la fin de ce chapitre, un test préliminaire est réalisé pour évaluer la capacité d'un certain nombre de techniques de reconnaissance et de caractérisation spatiale des objets dans leur environnement.

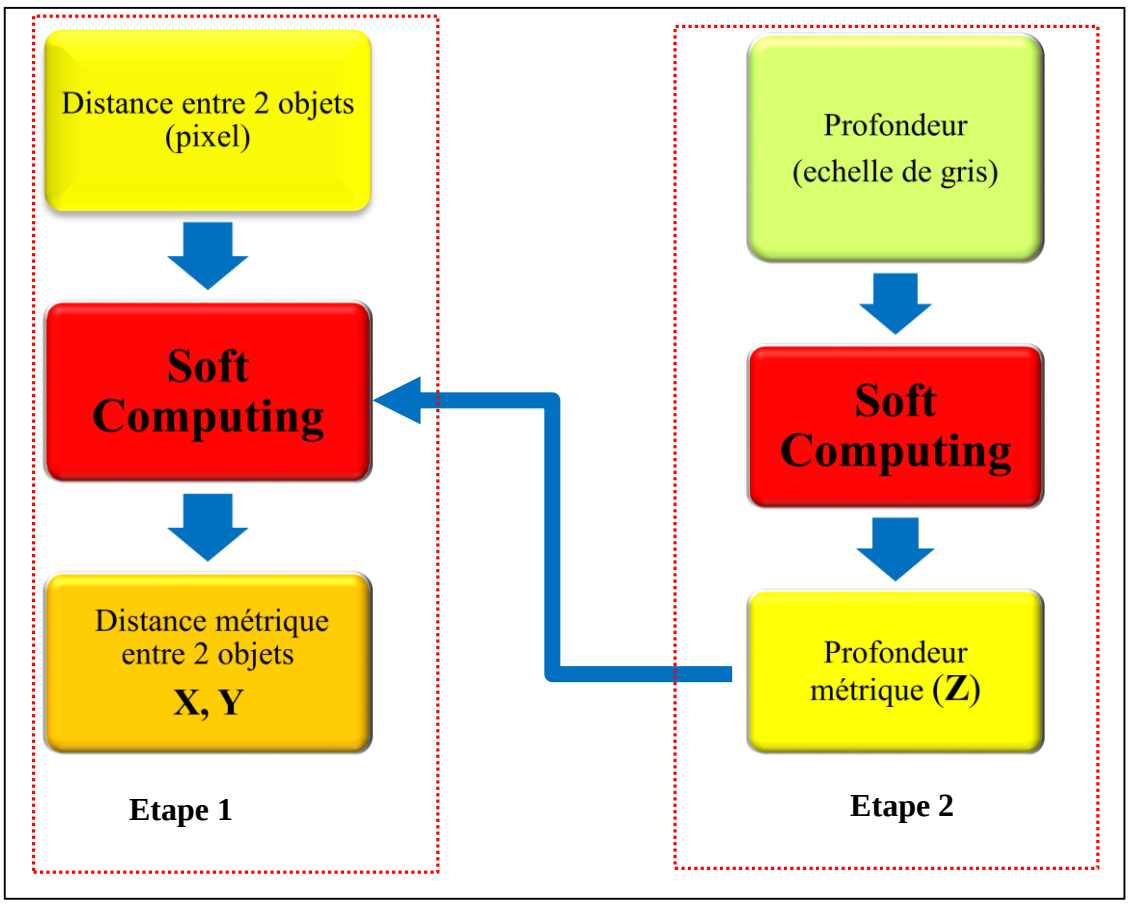


Figure 36: Schéma général de l'approche expérimentale proposée pour l'extension de notre approche de la détection de la saillance 3D.

3.2 Caractérisation de l'environnement utilisant le Soft Computing

3.2.1 Introduction

En tenant compte du fait que le capteur d'image couleur de Kinect offre une image de 640 pixels par 480 pixels avec une visualisation verticale de 57 ° et 43° de vision horizontale, les coordonnées spatiales (en mètres) d'un objet perçu dans le référentiel de la Kinect pourraient être estimées par les équations (28) et (29), où z_m est la distance de l'objet par rapport à la Kinect (fournie directement par le capteur infrarouge en mètres), $p(x)$et $p(y)$ sont des positions horizontales et verticales des pixels dans l'image, respectivement et f est la distance focale. Ensuite, la distance entre deux points caractérisés par leurs coordonnées $(x_{m1}, y_{m1,}z_{m1,})$ et $(x_{m2}, y_{m2,}z_{m2,})$ appartient à l'objet 1 et 2 successivement, est estimée par l'équation (30).

$$x_m = z_m \frac{1}{f(x)} \left[p(x) - \frac{640}{2} \right] \qquad (28)$$

$$y_m = z_m \frac{1}{f(x)} \left[p(y) - \frac{480}{2} \right] \qquad (29)$$

$$Distance = \sqrt{(x_{m1} - x_{m2})^2 + (y_{m1} - y_{m2})^2 + (z_{m1} - z_{m2})^2} \qquad (30)$$

Une portée attrayante des données complexes, incomplètes ou imprécises (fuite) est de tirer parti de la capacité d'estimation (généralisation) des approches basées sur Soft-Computing. Cette technique, y compris les réseaux neuronaux artificiels (ANN), Fuzzy Logic (FL) et Algorithmes évolutifs (AL).En outre, les surfaces lisses et spéculaires apparaissent surexposées dans l'image infrarouge, générant des trous dans la carte de profondeur (voir figure 14) ce qui influence sur les données géométriques fournies par la Kinect, ces données sont nécessaire pour le calcul de distance entre deux objets et donc le calcul de la hauteur ou la largeur d'un objet.

Dans cette section, nous décrirons une approche basée sur le Soft-Computing (Karray et al. 2004) qui estime les distances relatives des objets utilisant les données de la Kinect et des techniques d'apprentissages. Nous montrerons que l'approche proposée pourrait être étendue pour l'estimation des tailles des objets en utilisant le même signal et les mêmes données visuelles.

3.2.2 L'approche expérimentale

L'approche étudiée, basée sur les techniques de Soft-Computing et de traitement d'image conventionnel des images en couleurs 2D ainsi que des informations de profondeur fournies par la Kinect (voir figure 36-Etape1), se compose de trois phases (voir la figure 37):

1. **Phase1**: Capture d'images couleur et profondeur 2D.
2. **Phase2**: Le prétraitement conventionnel des images délivrées par la Kinect extrait les caractéristiques appropriées. (la phase de segmentation)
3. **Phase3**: Apprendre les fonctionnalités extraites (en mode apprentissage) puis l'estimation de la distance entre les objets (en mode généralisation).

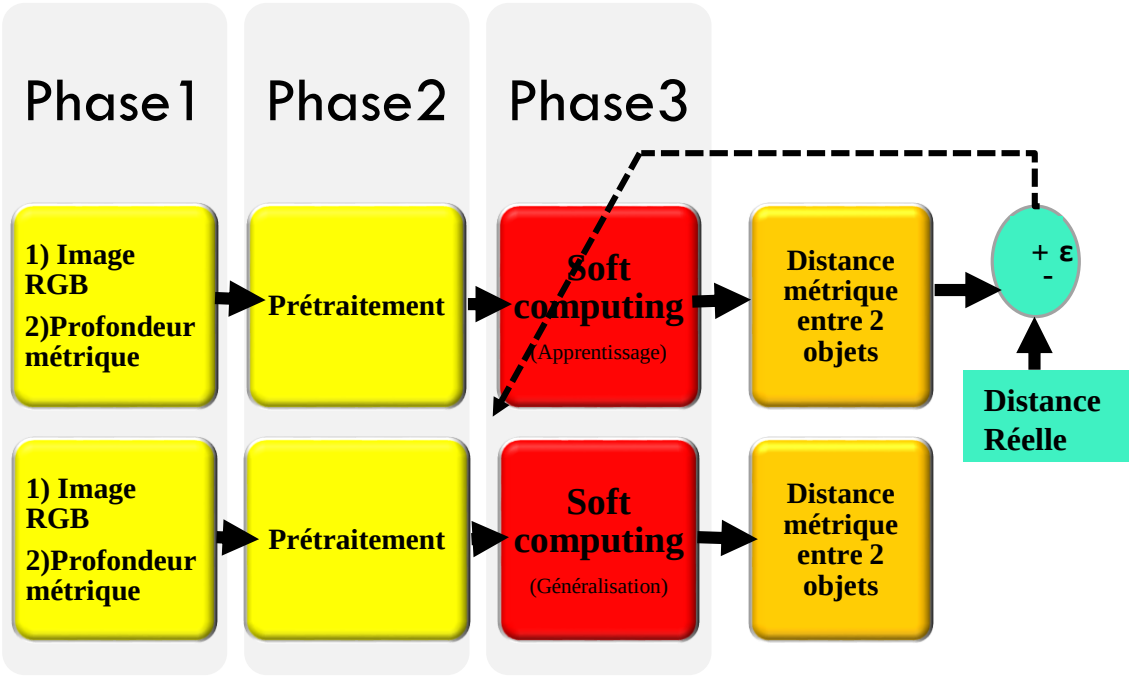


Figure 37: Diagramme blocs de l'approche proposée.

3.2.2.1 Capture d'images couleur et profondeur 2D

Avant de procéder aux captures des images pour créer nos bases de données (phase1), nous commençons par l'étalonnage des capteurs images de couleur et de profondeur.

- **Etalonnage du capteur Kinect**

Pour déterminer les paramètres des caméras RGB et profondeur IR, nous utilisons les fonctions de la bibliothèque open cv 'calibrate.py', cette fonction nous a permis de calculer les paramètres de la matrice 'k', les matrices de rotation R et translation T de la caméra couleur et profondeur respectivement (voir section 1.3.2.2). Nous commençons par réaliser une prise des images de couleurs et d'infrarouge d'un damier de différentes ongles et positions (voir figure 38). Le programme nécessite un grand nombre d'image couleurs avec ses correspondants en images infrarouge (environ 60 images), la taille des carreaux horizontaux et verticaux doit être identiques, ces carreaux doivent être alternés en noir et blanc car l'étalonnage se base sur la détection et la position de l'intersection de ces carreaux, pour n carreaux on obtient (n-1) intersections. Chaque carreau à une dimension de 2.5cm.

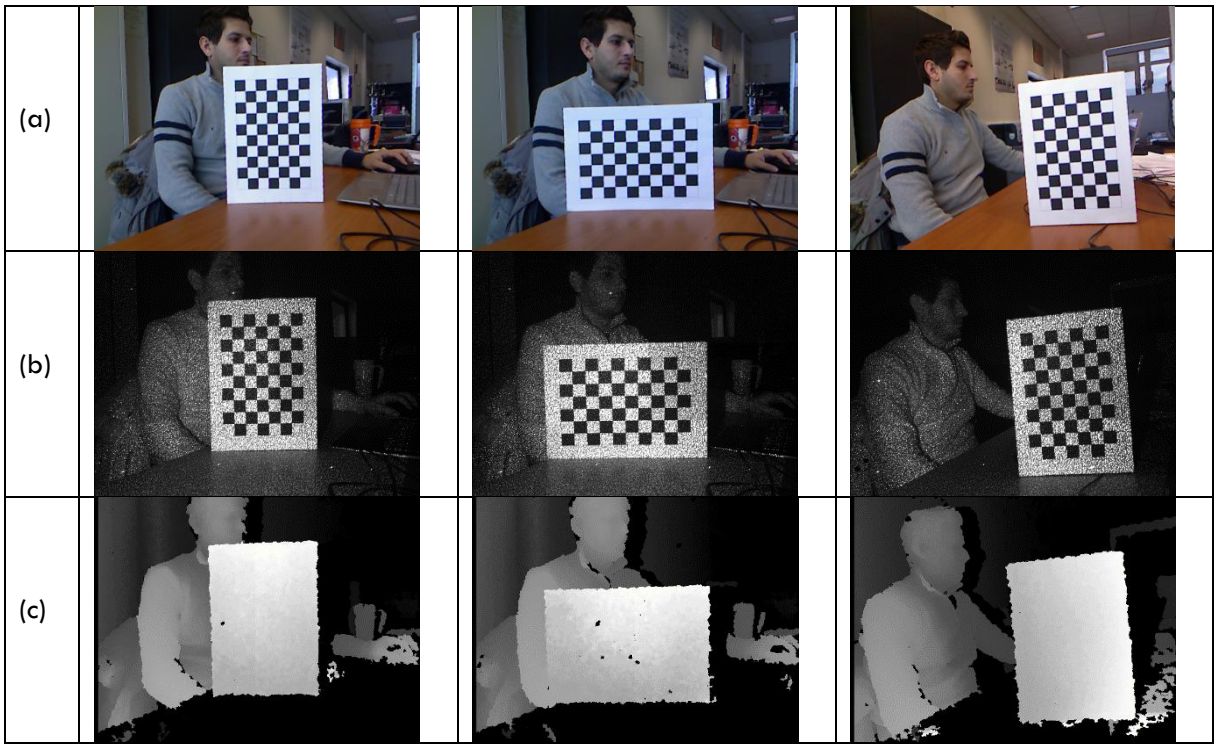


Figure 38: Etalonnage de la camera couleur et profondeur : a) image couleur, b) image infrarouge, c) image de profondeur

Nous utilisons les paramètres intrinsèques de la caméra de profondeur. Chaque pixel de la caméra en profondeur peut être projeté dans l'espace métrique 3D, ensuite nous pouvons projeter chaque point de l'image en profondeur dans l'image de couleur pour déterminer sa couleur dans l'image initiale. Les paramètres de rotation R et de translation T sont estimés lors de l'étalonnage stéréo entre la caméra couleur et la caméra de profondeur, utilisant la fonction 'Stereo-Calibrate' de la bibliothèque OPENCV (Madani et al. 2017).

La technique de l'étalonnage stéréo se fait en utilisant deux représentations matricielles des coordonnées des pixels : la matrice x_{mp} obtenue à partir de l'équation (28) et la matrice y_{mp} à partir de l'équation (29). La correspondance est établie en deux étapes : La première étape utilise l'équation (31) impliquant les matrices R et T calculées à partir de la procédure d'étalonnage de la Kinect, pour obtenir ce que nous appelons « les coordonnées translation-rotation » x_{TRm} et y_{TRm}. La deuxième étape calcule les « coordonnées d'étalonnage » x_{cm} et y_{cm} en utilisant l'équation (32) et (33), où c_x et c_y sont les paramètres intrinsèques obtenus à partir de l'étalonnage de la caméra couleur. f_x et f_y sont les distances focales de la caméra couleur (Burrus 2014; Madani et al. 2017).

$$\begin{pmatrix} x_{TRm} \\ y_{TRm} \\ z_{TRm} \end{pmatrix} = R^T \begin{pmatrix} x_m \\ y_m \\ z_m \end{pmatrix} - R^T T \quad (31)$$

$$x_{cm} = [x_{TRm} - f_x] \frac{1}{z_{TRm}} + c_x \quad (32)$$

$$y_{cm} = [y_{TRm} - f_y] \frac{1}{z_{TRm}} + c_y \quad (33)$$

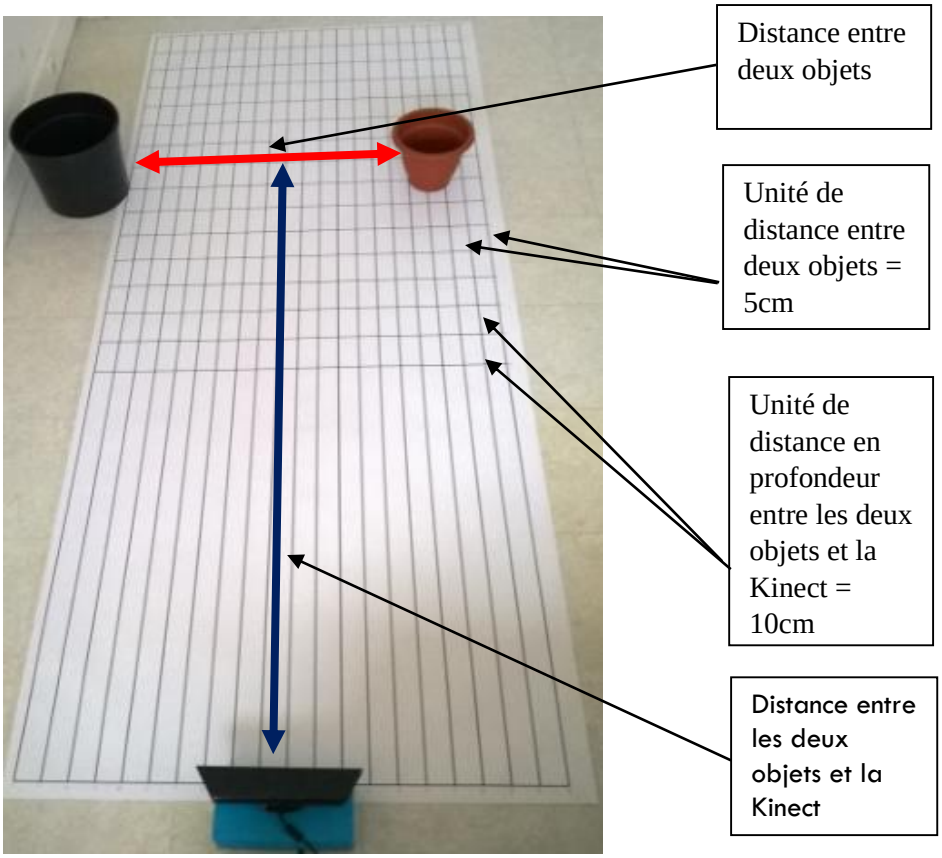


Figure 39: Schéma expérimental

- **Création des bases des données**

La préparation des bases de données consiste à capturer des images suivant deux cas de figures (voir Figure 39):

1) Nous déplaçons les deux objets par rapport à l'échelle de profondeur Kinect de 100 cm à 250 cm, en gardant une distance fixe entre les deux objets.

2) Nous varions la distance entre les deux objets suivant la même plage de profondeur à chaque fois.

Nous avons créé deux bases de données: Base de données 1 avec 495 images de couleur des objets de forme régulière (rectangulaire) et Base de données 2 avec 304 images de couleur des objets de forme irrégulière (trapézoïdale).

Ces bases de données contiennent des images relatives à diverses positions des objets considérés. D'une part, différentes distances entre les objets sont définies entre 4 cm et 100 cm pour la base de données 1 et entre 1,7 cm et 91,7 cm pour la base de données2 (voir figure 39). D'autres autre part, différentes positions par rapport à la position du Kinect sont définies entre 100 cm et 263 cm pour la base de données 1 entre 100 cm et 250 cm pour la base de données2 (voir figure 39). Ces bases de données seront utilisées pour construire le modèle de prédiction de distance.

3.2.2.2 Le prétraitement conventionnel des images émises par Kinect

La figure 40 et la figure 41 montrent les résultats d'application de la méthode de segmentation MEAN SHIFT (phase2) sur un ensemble d'images différentes. Ces images fournies par la camera couleur de la Kinect pour une même distance entre deux objets, plus en plus loin de la Kinect. Dans les figures 40 et 41, à gauche: les objets sont proches de la Kinect, au centre: les objets sont un peu plus loin de la Kinect, à droite: les objets sont plus loin de la Kinect, la deuxième ligne donne les images segmentées. Les objets sont simples (objets de forme réguliers) dans la figure 40, tandis que dans la figure 41, les deux objets sont plus complexes en ce qui concerne leurs formes.

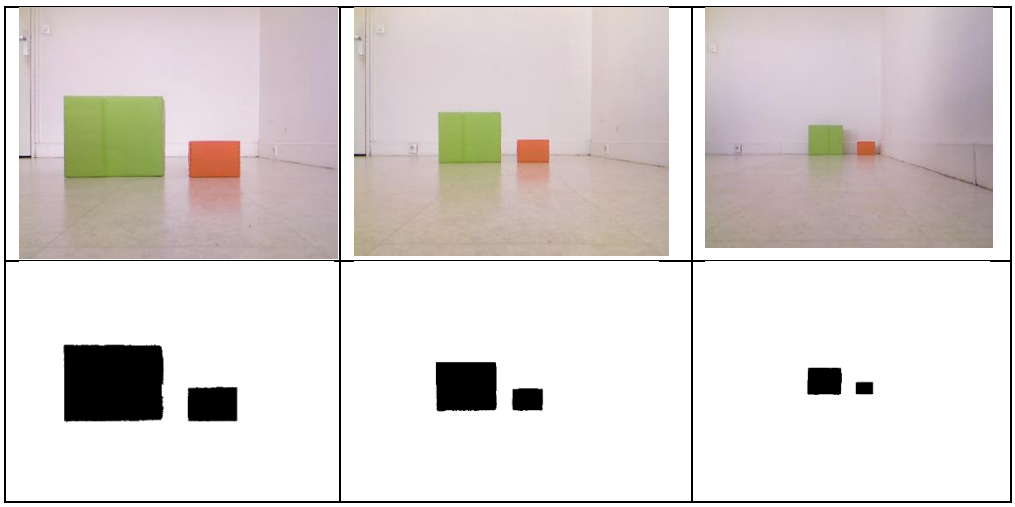


Figure 40: Exemple d'images capturées pour deux objets donnés avec une forme simple (régulière).

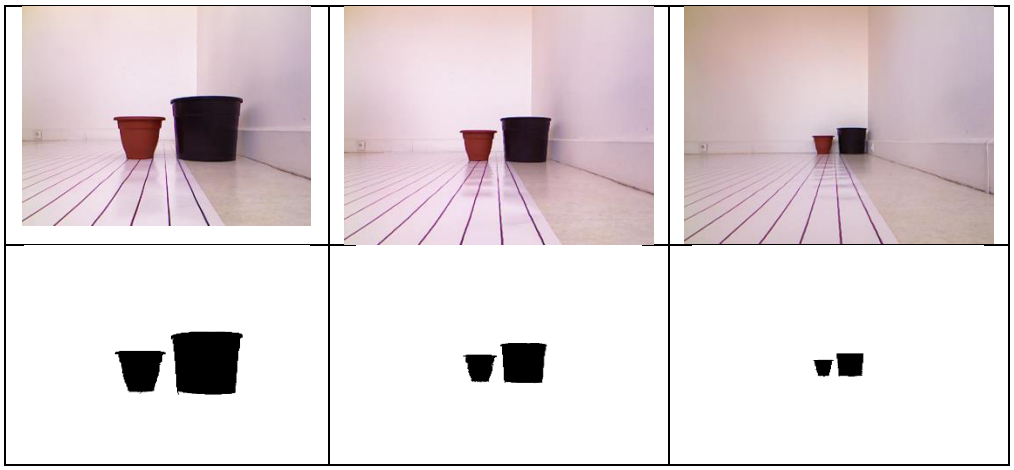


Figure 41: Exemple d'images capturées pour deux objets donnés avec une forme plus complexe (irrégulière).

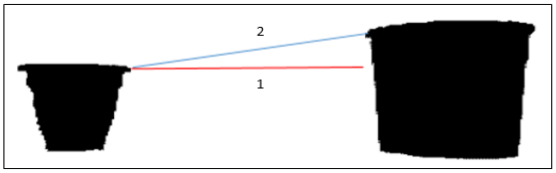


Figure 42: La ligne 1 représente la distance minimale entre deux objets.

Lorsque la segmentation est effectuée, nous calculons la distance minimale entre les objets (nombre de pixels). Les fonctions de calcul de distance entre deux objets sont basées sur le calcul de la distance entre les deux pixels, des deux objets, les plus proches (représenté par la ligne 1 de la figure 42).

3.2.2.3 Soft Computing : Mode Apprentissage et Généralisation

Le module basé sur le Soft-Computing (Phase3) estime la distance en fonction du Système d'inférence floue (FIS) et utilisant le réseau neuronal artificiel, à savoir ANFIS (Adaptive Neuro-Fuzzy Inference Systems) (Jang et al. 1995) (voir la figure 43). L'estimation de la distance entre deux objets en centimètres à partir de deux entrées qui sont : X représente la distance entre ces deux objets en pixels obtenue par traitement d'image couleur segmentés, et Y représente la profondeur entre ces deux objets et le capteur Kinect. Cette estimation est un problème non linéaire, qui nécessite une solution non linéaire telle qu'ANFIS était plus appropriée.

La base des règles contient deux règles 'si'-'donc' floues de type Takagi et Sugeno (Jang 1993). Nous citons brièvement la signification de chaque couche (Layer) et nous détaillons dans la section suivante la conception d'un système flou. Pour deux entrées **X** : qui représente la distance entre deux objets en pixel et **Y** qui représente la profondeur en centimètre entre la Kinect et ces deux objets et une sortie *f* qui représente la distance entre deux objets en centimètres, on peut distinguer deux règles :

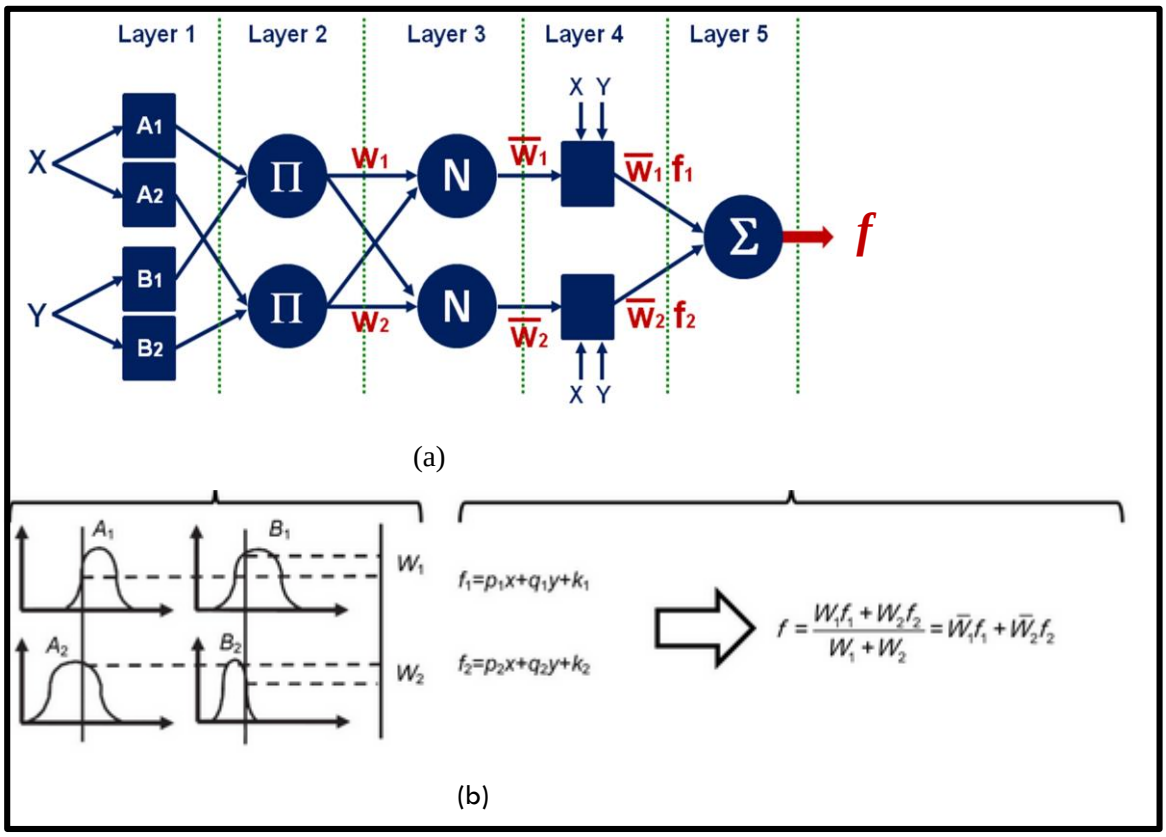


Figure 43: Architecture d'ANFIS

Règle 1: si **x** est **A1** et y est **B1,** donc ***f1*** = **p1x** + **q1y** + **r1**.

Règle 2: si **x est A2 et y est B2,** donc ***f2*** = **p2x** + **q2y** + **r2.**

Où: *fi* représente l'inférence flou selon la sortie souhaitée, A_i, B_i sont des étiquettes d'ensembles flous caractérisés par une fonction d'appartenance appropriée, $\mu_{Ai}(x)$ est la fonction d'appartenance de A_i et {a,b,c} : sont des paramètres set.

$$\mu_A(x) = \frac{1}{1+\left|\frac{x-c_i}{a_i}\right|^{2b_i}}$$

Nous montrons brièvement la signification des différentes couches du système ANFIS, donnée par les équations suivantes :

Couche 1: Génération du degré d'appartenance.

$$O_{1,i} = \mu_{Ai}(x)\,, \qquad i = 1,2 \tag{34}$$

Où:

$O_{k,i}$: Est la fonction nœud, où *k* est le nombre de la couche et *i* est la position du nœud dans la couche.

Couche 2: Fuzzyfication.

$$O_{2,i} = w_i = \mu_{Ai}(x).\mu_{Bi}(x)\,, \qquad i = 1,2 \tag{35}$$

Couche 3: Normalisation.

$$O_{3,i} = \overline{w_i} = \frac{w_i}{w_1 + w_2} \qquad i = 1,2 \tag{36}$$

Couche 4: Defuzzyfication

$$O_{4,i} = \overline{w_i} f_i = \overline{w_i}(p_i x + q_i y + r_i) \tag{37}$$

$Ou\ \{p_i\,,\ q_i\,,\ r_i\}$: Sont des paramètres set.

Couche 5: La sortie final

$$O_{5,i} = \sum_i \overline{w_i} f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{38}$$

Les différents étapes qui constitue notre système flou composé sont: premièrement la fuzzification (voir figure 43, couche 1,2 et 3) qui permet de convertir des entrées en valeurs floues utilisant des fonctions d'appartenance, deuxièmement l'évaluation des règles (voir figure 43, couche4) définissent la relation entre les entrées et les sorties, troisièmement la défuzzification (voir figure 43, couche 5) qui permet de convertir le résultat flou des règles en une valeur se sortie précise.

3.2.3 Résultat expérimental.

Les processus de capture et de segmentation ont été développés en langage de programmation PYTHON, et la construction du modèle de prédiction de la distance a été réalisée en langage Matlab R2011a dans un premier temps, puis nous l'avons traduit en python en utilisant la fonction ANFIS fournis par la bibliothèque OPENCV.

La figure 44 et la figure 45 montrent les résultats d'estimation de la distance en centimètres, indiquant l'erreur d'estimation pour le cas où l'apprentissage a été effectué en

utilisant la deuxième base de données, et le test a été effectué en utilisant la première base de données. La figure 46 et la figure47 montrent les résultats de l'estimation de la distance indiquant l'erreur d'estimation pour le cas où l'apprentissage a été effectué en utilisant la base de données2 et 50% de l'échantillonnage aléatoire de la base de données1, le test a été effectué en utilisant les 50% de la base de données1 restante.

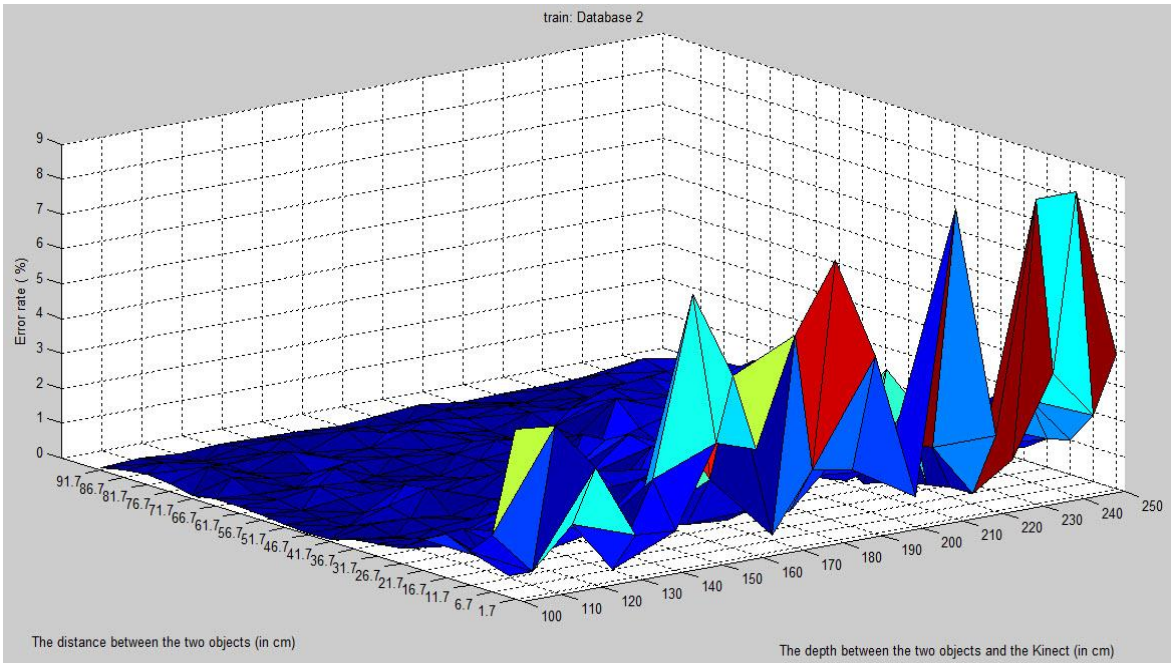


Figure 44: Erreur d'estimation de la distance en mode apprentissage: la base de données2 contenant des objets en forme irrégulière a été utilisée pour l'apprentissage.

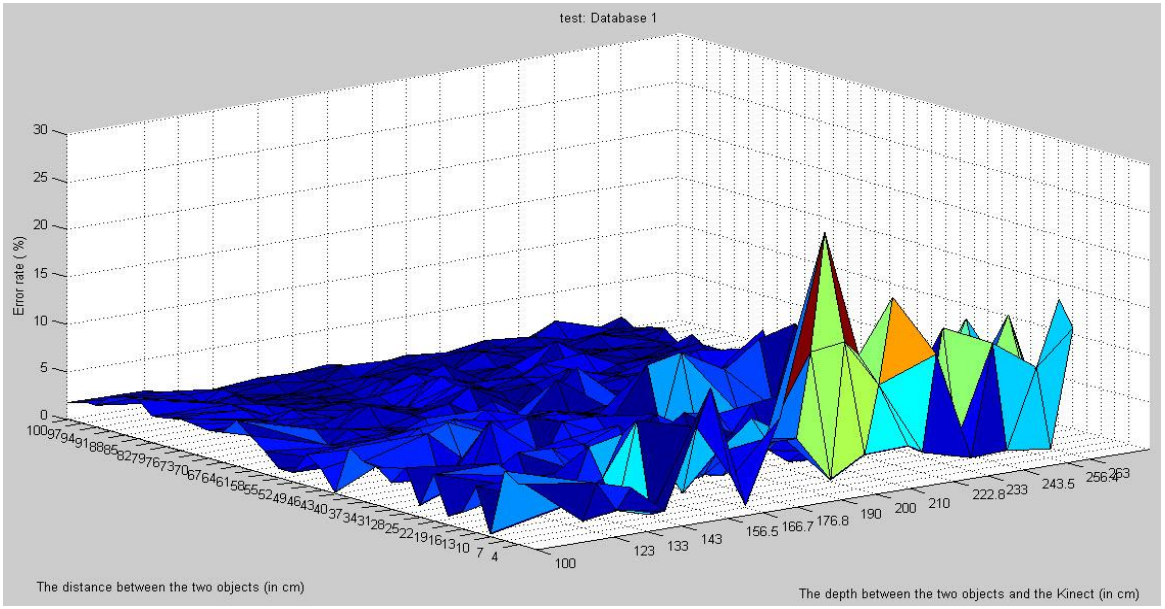


Figure 45: Erreur d'estimation de distance en mode de généralisation: la base de données1 contenant des objets en forme régulière a été utilisée pour le test.

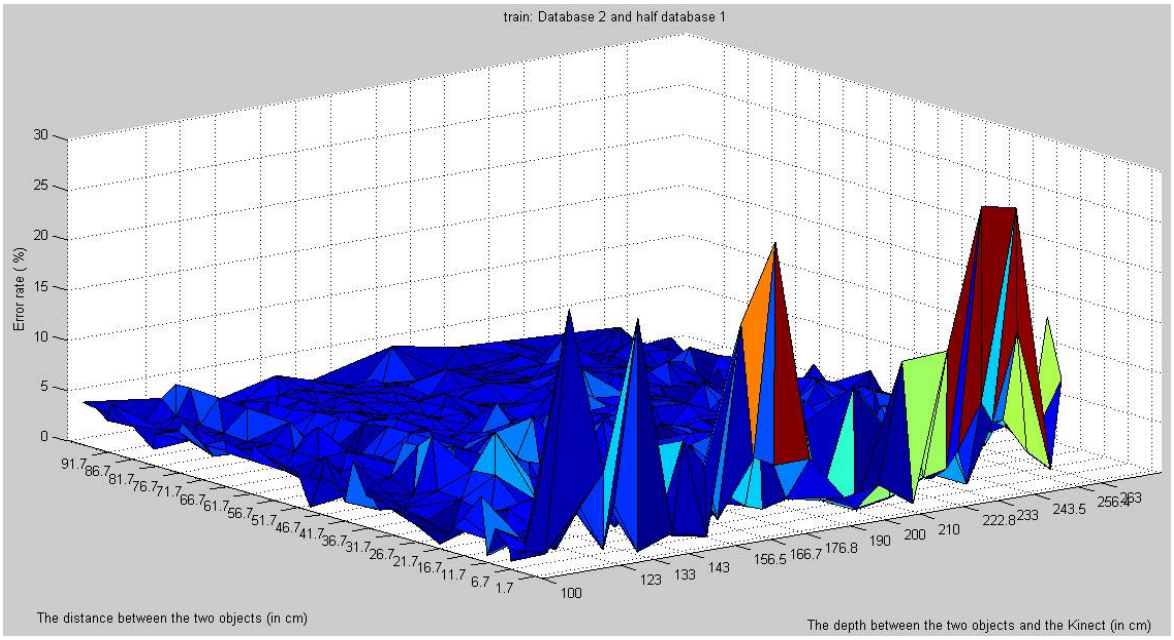


Figure 46: Erreur d'estimation de distance en mode apprentissage: la base de données2 et 50% de la base de données 1 ont été utilisés pour l'apprentissage.

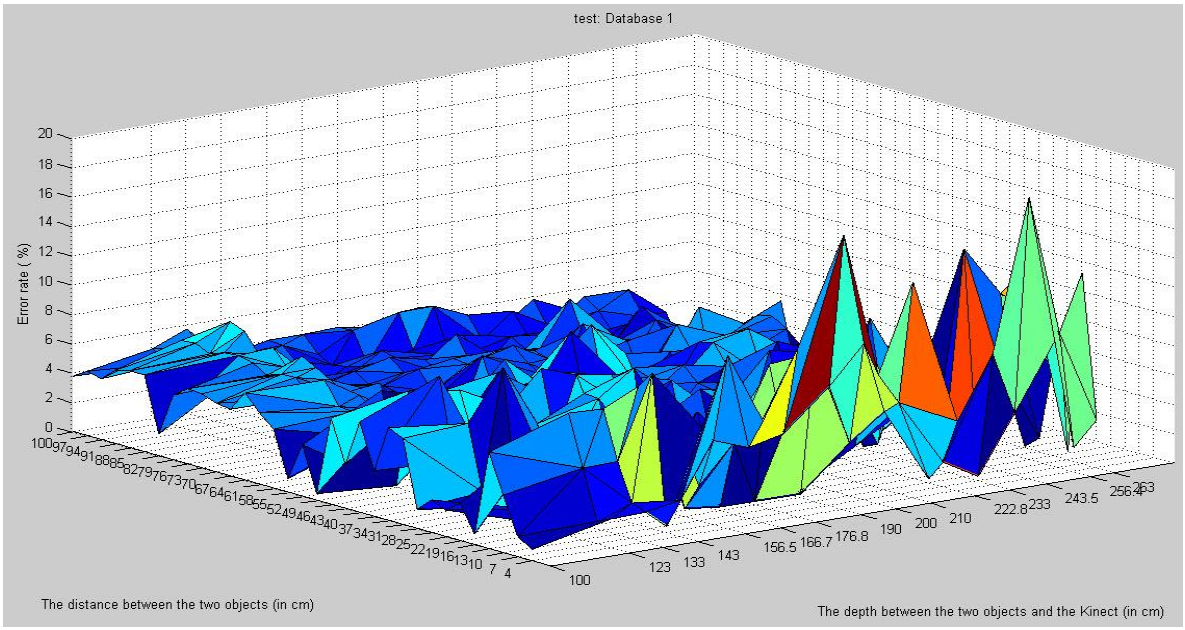


Figure 47: Erreur d'estimation de la distance dans la généralisation: 50% de la base de données de repos 1 ont été utilisés pour le test.

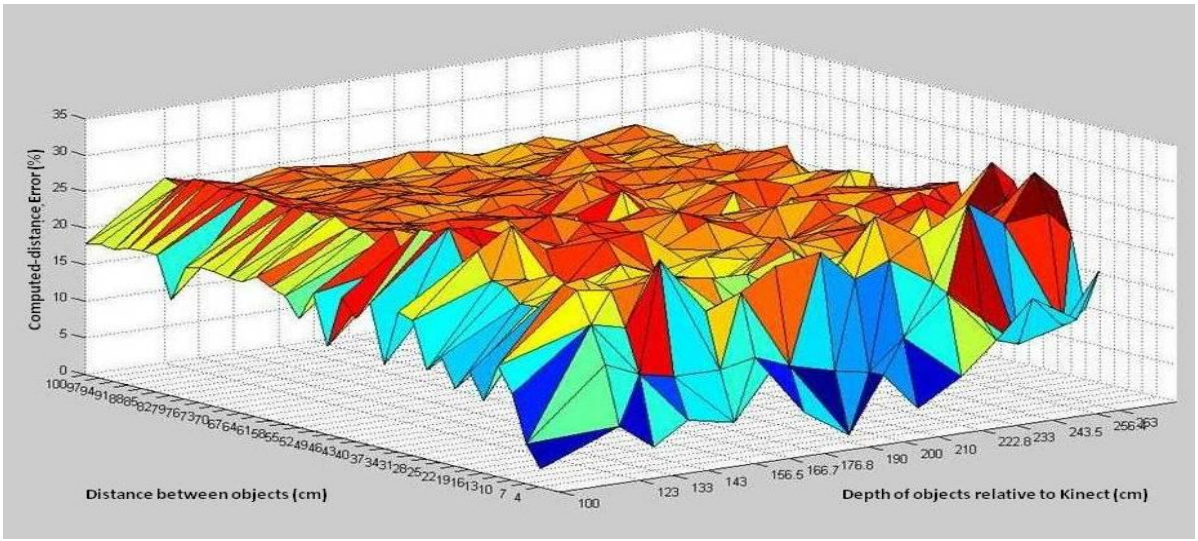


Figure 48: Erreur de calcul de la distance en utilisant une approche géométrique, la base de données1 a été utilisée pour le test

3.2.4 Comparaisons d'ANFIS avec d'autre algorithme d'apprentissage

La même procédure d'apprentissage a été appliquée pour MLP, SVR et interpolation bilinéaire (voir Annexe1). Nous avons utilisé la bibliothèque Scikit-Learn en python pour l'implémentation de l'algorithme SVR, et le langage MATLAB pour MLP et l'interpolation bilinéaire. Le tableau 5 résume le résultat des différentes expériences d'apprentissage et de test pour chaque algorithme d'apprentissage.

	ANFIS	MLP	SVR	Interpolation bilinéaire
Résultat de l'apprentissage (taux d'erreur max d'estimation de la distance)	30%	60%	80%	35%
Résultat du test (taux d'erreur max d'estimation de la distance)	20%	80%	100%	25%

Tableau 5: Résultat de l'expérience pour différentes algorithmes d'apprentissage.

3.2.5 Discussion et conclusion

Une approche d'estimation de distance basée sur Soft-Computing a été proposée, mise en œuvre et validée expérimentalement. Profitant de Soft-Computing, l'approche d'estimation de distance visuelle proposée associée au capteur Kinect permet une perception 3D de l'environnement avec une erreur d'estimation raisonnable, par conséquent, elle fournit une solution peu coûteuse pour la vision des robots nécessitant des compétences métrologiques

(comme par exemple la navigation autonome). Les concepts étudiés peuvent être facilement étendus à l'estimation des tailles des objets. Les résultats obtenus montrent que le travail actuel ouvre une perspective qui consiste à améliorer l'estimation des «distances courtes» où l'erreur résiduelle reste encore plus élevée que celle relative à l'estimation des distances entre des objets plus éloignés. Cela pourrait se faire en introduisant des fonctions supplémentaires extraites de l'image couleur de la Kinect. Cependant, l'extraction ne doit pas affecter les contraintes en temps réel, ce qui signifie que les fonctionnalités supplémentaires doivent être assez simples.

Le tableau 5 montre les résultats du taux d'erreur pour l'estimation de la distance entre deux objets. Comme nous pouvons le voir dans ce tableau, le réseau de neurone ANFIS pourrait produire l'erreur de prédiction la plus faible dans les ensembles de données. Il semble clairement qu'ANFIS est plus efficace que MLP et SVM si le manque de données existe ou si la taille de la base de données est faible. ANFIS devrait être préféré en raison de sa capacité à gérer l'incertitude des données floues, ambiguës ou incomplètes. La taille des règles de base est cruciale pour la charge de calcul, pour cette raison, cette méthode est appropriée pour les problèmes ayant relativement un petit nombre de variables d'entrée. Le réseau MLP, est le meilleur choix si des données suffisantes existent pour caractériser le comportement cible. Le SVM dans l'original utilisé pour la classification, il peut résoudre les problèmes de régression qui représente un cas particulier de classement: classe continue. La régression SVM (SVR) est basée sur la stratégie d'approximation locale, elle est basée sur l'utilisation de fonctions du noyau qui permettent une séparation optimale des données, présente un inconvénient lorsqu'il existe un conflit entre les classes. Le SVM est habituellement beaucoup plus rapide que l'ANFIS et le MLP. L'estimation de la distance entre deux objets en centimètres utilisant l'interpolation bilinéaire donne un meilleur résultat que le MLP et SVR (voir tableau 5). Cette méthode est basée sur la stratégie d'approximation locale, l'inconvénient de cette méthode est que la distance est calculée à partir des quatre valeurs de distance qui cadre la distance a testée et dépend de la précision de ces quatre distances, sans possibilité de correction ou de réglage (voir Annexe1). Le taux d'erreur d'estimation de distance obtenue entre deux objets est respectivement de 20% et 25% avec ANFIS et interpolation bilinéaire. Ces résultats fournissent des informations sur l'entropie[2] de notre base de données.

Dans cette section, nous avons présenté un aperçu de notre système 'Soft Computing' basé sur l'algorithme d'apprentissage ANFIS pour la prédiction de la distance.

[2] L'entropie permet de mesurer la quantité d'information moyenne d'un ensemble de données et de mesurer son incertitude

Dans la section suivante, nous entamons l'étape d'extension de notre approche de détection de la saillance 3D (voir figure 36-Etape2). Le système 'Soft Computing' va nous servir pour résoudre le problème de l'étalonnage de l'image de profondeur fourni par le capteur ASUS du robot Pepper à l'aide de l'image de profondeur fourni par le capteur Microsoft de Kinect.

3.3 Extension de notre approche sur le robot Pepper

Il existe plusieurs approches basées sur l'étalonnage sensoriel des capteurs via d'autre capteur de même fonctionnalité. Nous citons comme exemple le travail de (Frad 2016) pour l'étalonnage d'un capteur Scalabe-SPIDAR via le système de tracking optique infrarouge ARTtrack1. L'auteur a collecté deux types de données pour l'étalonnage, des données de positions fournies par un système de tracking optique infrarouge ARTtrack1 et d'autres données fournies par le Scalabe-SPIDAR (voir figure 49). Le protocole proposé utilise les techniques de réalité virtuelle pour récupérer les positions spatiales renvoyées par le Scalable-SPIDAR et le système ARTtrack1 dans un espace de travail bien défini.

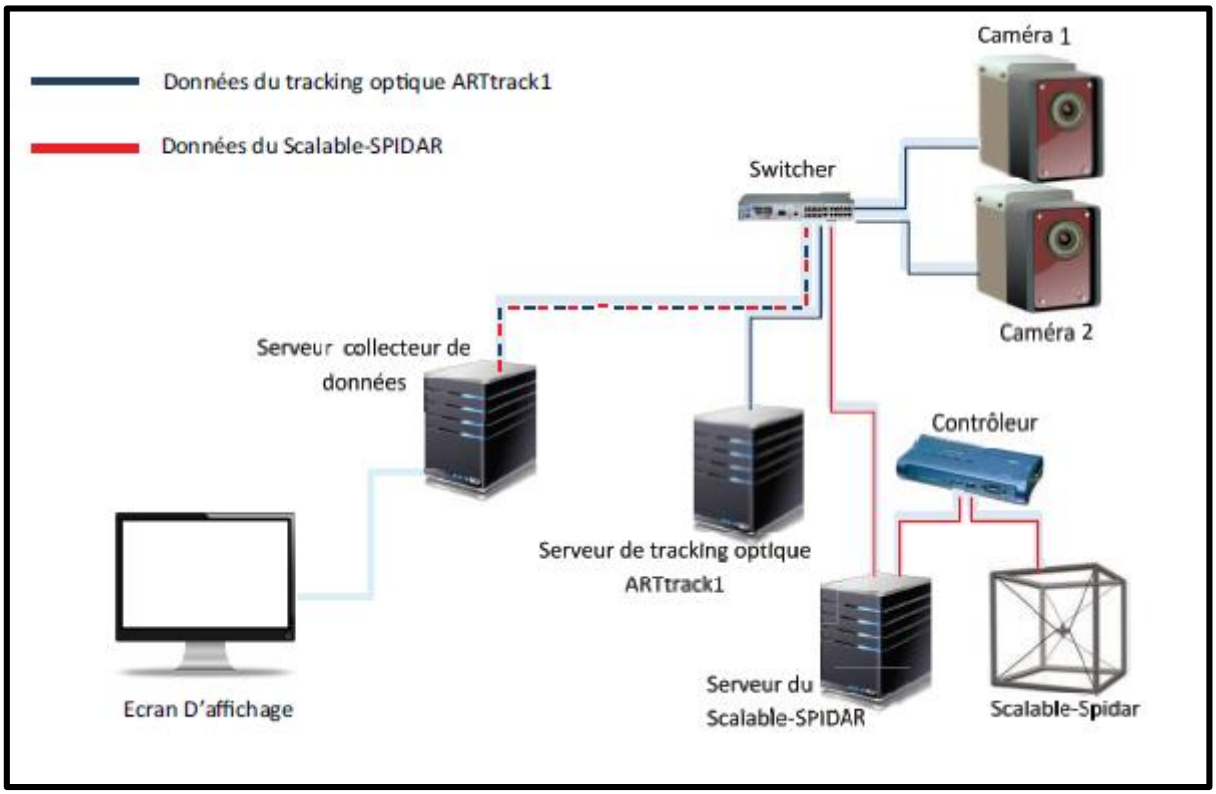


Figure 49: Architecture du protocole proposé par FRAD(Frad 2016)

3.3.1 Introduction

Des nombreux problèmes, d'origine technologique dans les capteurs 3D, sont souvent présents et peuvent pénaliser la qualité de la perception visuelle de l'environnement. Nous avons rencontré un problème de l'acquisition de l'information de profondeur avec le capteur ASUS du robot Pepper. Cette information est fournie sous forme d'image en échelle de gris, or que notre méthode de détection de saillance en profondeur a besoin de cet information en centimètres Z , d'où l'intérêt d'utiliser le système 'Soft Computing' pour l'étalonnage du capteur de profondeur ASUS de Pepper à l'aide du capteur de profondeur de Kinect.

3.3.2 Mise en place de l'approche expérimentale

L'approche expérimentale est basée sur: 1) le système Soft-Computing utilisé dans la section 3.2 (voir figure 36-Etape2) le traitement d'image conventionnel des images en couleurs 2D, 3) des informations de profondeur fournies par le capteur ASUS du robot Pepper et le capteur de profondeur de Microsoft Kinect. L'approche expérimentale est composée de trois phases (voir la figure 50):

Phase1 : Capture d'images couleur et profondeur 2D de Kinect et Pepper.

Phase2 : Prétraitement des images de profondeur capturées par le capteur ASUS de Pepper.

Phase3 : Création et test d'un modèle d'apprentissage. Pour ce faire, dans la phase d'apprentissage, nous utilisons la méthode supervisé d'apprentissage tel que, les images de profondeur fournis en échelle de gris par le capteur ASUS représentent l'entrée d'apprentissage du model, et les valeurs de profondeur en (mm) fournies par la Kinect représentent la sortie souhaitée. Dans la phase de généralisation, le model est testé à partir des nouvelles images de profondeur fournies par le capteur ASUS.

3.3.2.1 Capture des images

Pour capturer les images (phase1), nous avons procéder de la manière suivante : 1) Capturer des images de profondeur sur un mur plan (surface plane) par Kinect, depuis différents portées. La Kinect se déplace sur un axe virtuel perpendiculaire au mur observé (voir la ligne rouge dans la Figure 51 (b)). Les images de profondeur sont prises avec un pas de 2cm entre la Kinect et le mur plan. L'intervalle de mesure opérationnel de la Kinect s'étend de 0.5 à 6 mètres. 2) de la même manière que 1), nous avons capturé les images de profondeurs avec le capteur

ASUS (Voir la figure 53). Au total nous avons construit une base de données de 150 images de profondeur.

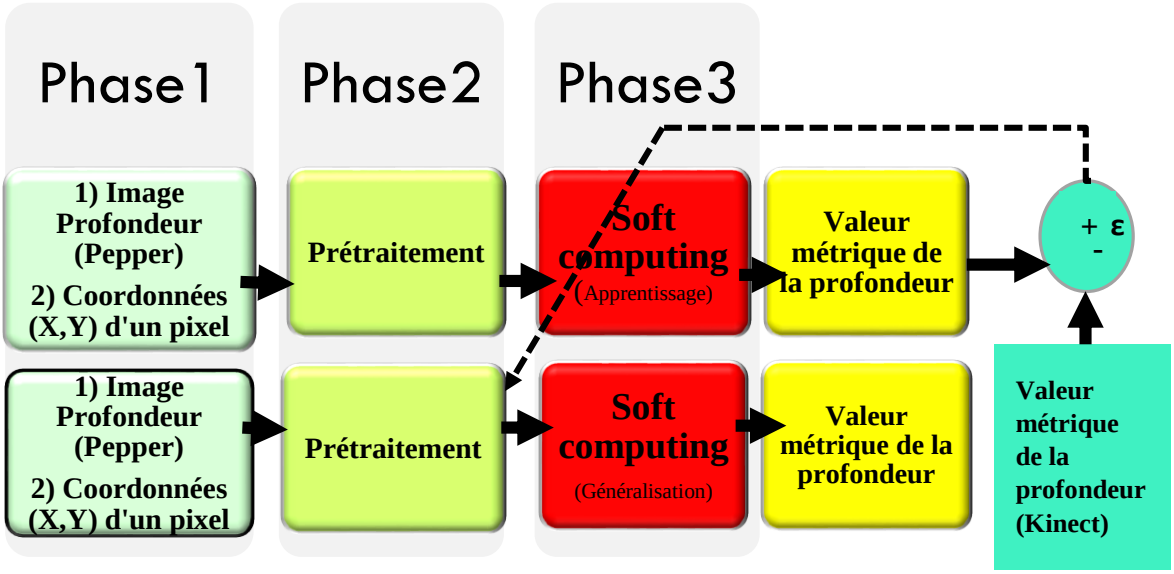
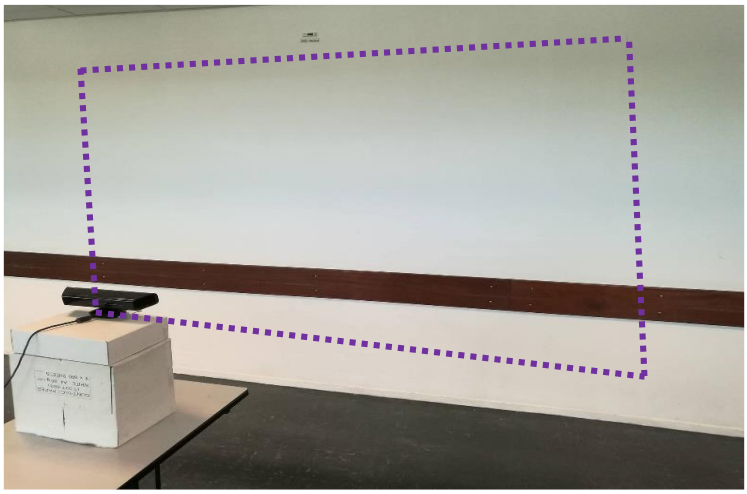


Figure50: Diagramme bloc de l'approche proposée, flux opérationnel montrant le mode d'apprentissage (En haut) et le mode de généralisation (En bas).

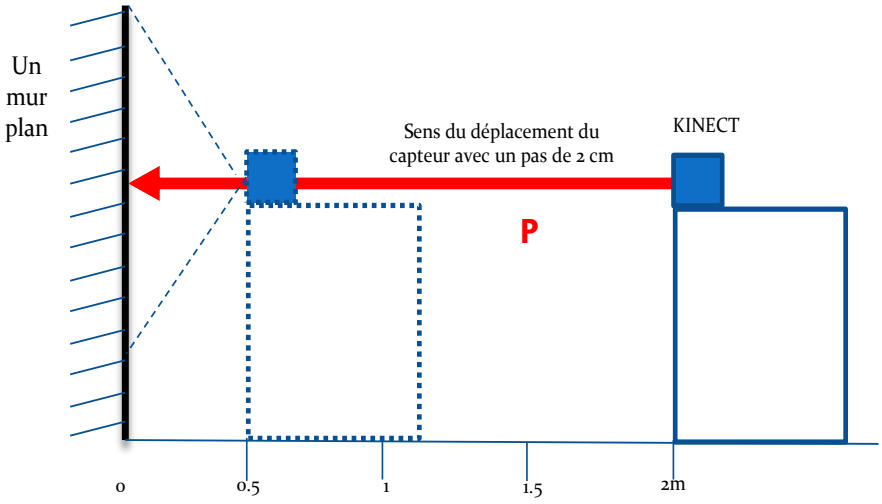
Nous avons constaté que les erreurs de mesures de la profondeur pouvaient être acquises avec une erreur maximale de 0.1% (voir figure 52) pour une distance de 3.5m entre la Kinect et le mur.

La construction du modèle d'apprentissage utilisant les images capturées par la Kinect a été effectuée par l'algorithme d'apprentissage ANFIS sous MATLAB. Cependant, comme Pepper supporte le langage de programmation Python, nous avons reconstruit le modèle d'apprentissage, avec les images capturées cette fois par Kinect et Pepper, en utilisant l'algorithme d'apprentissage ANFIS en Python et la bibliothèque OPENCV.

La figure 53, montre une image (a) de profondeur prise en mode de vision nocturne à l'aide du capteur IR de la Kinect. Les images (b) et (c) montrent deux parties de l'image (a) avec une intensité de projection des faisceaux IR différentes. Nous pouvons constater une forte intensité de faisceaux IR dans la surface (c) par rapport à la surface (b), qui signifie que la projection de ces faisceaux n'est pas distribuée uniformément dans toute l'image capturée, et cela dû au problème technologique de fabrication de la Kinect.



(a)



(b)

Figure 51: La méthode de capture des images de profondeur en (mm) : plan observé au niveau de la zone en pointillés (a). Schéma de principe de la manipulation (b).

Suite à ce problème, nous distinguons deux surfaces sur le mur (voir figure 51 et 54), une surface coloriée en vert nommé ‘surface A’ qui a une forte intensité de projection des faisceaux. Et une autre surface nommée ‘surface B’ qui représente le reste de l’image avec moins d’intensité que la surface A.

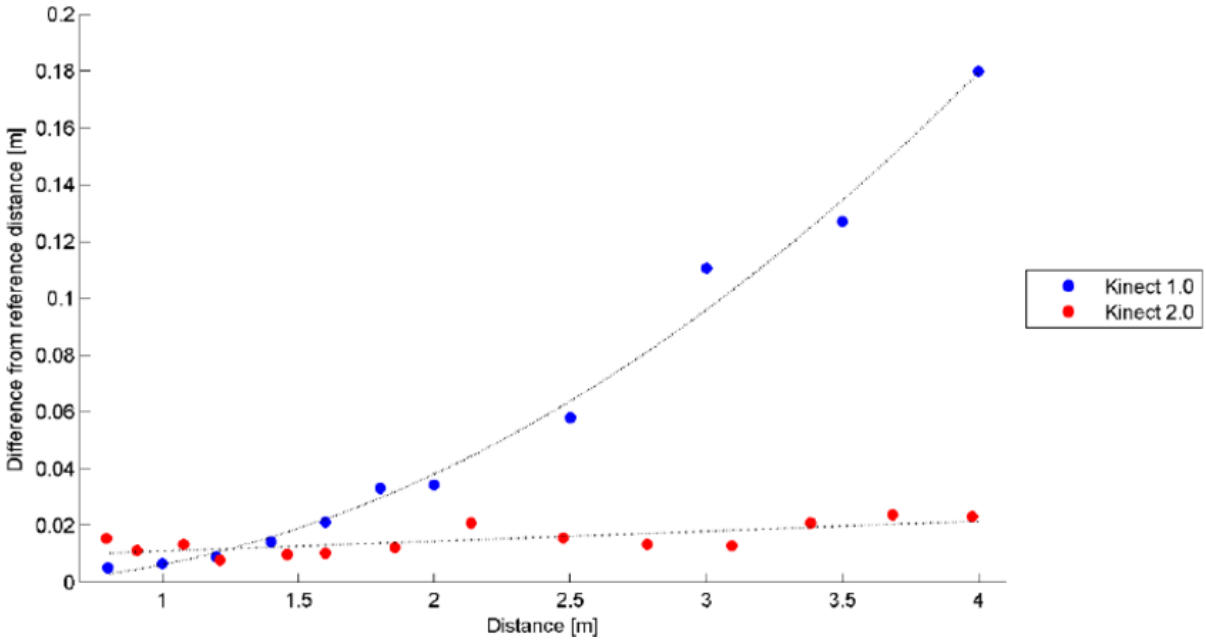


Figure 52: Taux d'erreur d'estimation de la profondeur (Pagliari et al. 2015)

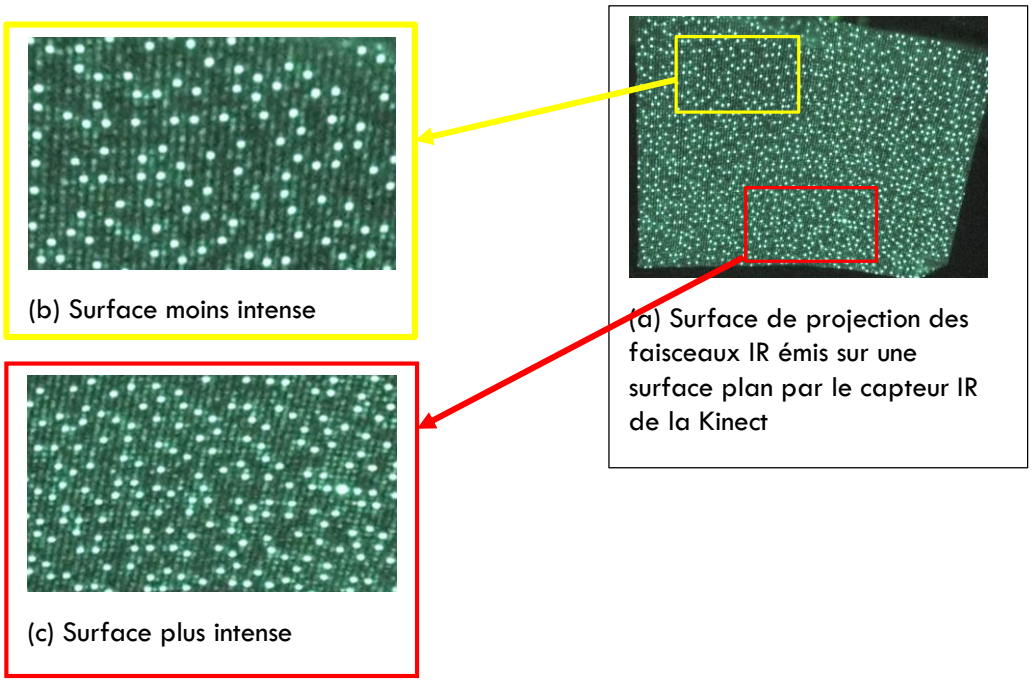
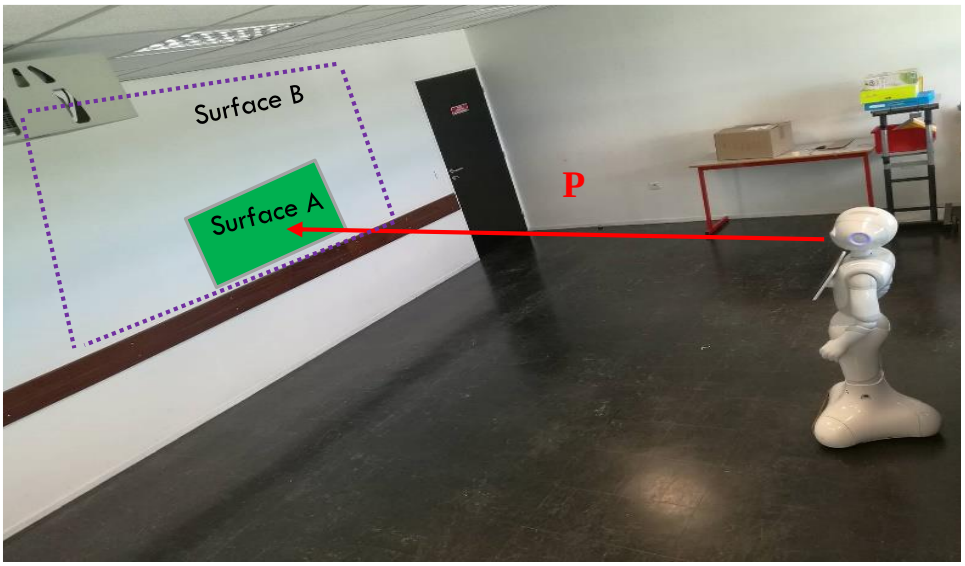
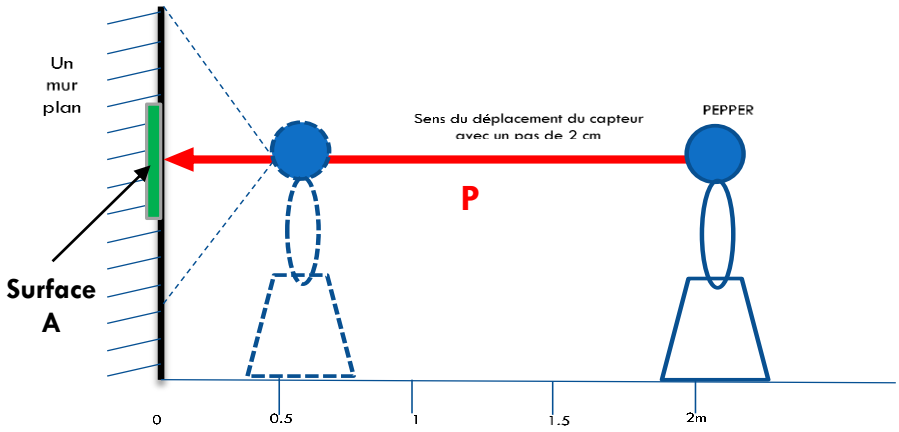


Figure 53: Image de profondeur prise en mode de vision nocturne à l'aide du capteur IR de la Kinect.



(a)



(b)

Figure 54: La méthode de capture des images de profondeur (en échelle de gris): plan observé au niveau de la zone en pointillés (a). Schéma de principe de la manipulation (b).

3.3.2.2 Prétraitement des images de la base de données

Nous avons effectué un lissage (phase2) sur les images de profondeur du capteur ASUS en utilisant le filtrage médian (medianBlur.py) d'OPENCV pour corriger les points noir généré par le capteur de profondeur ASUS (voir figure 55).

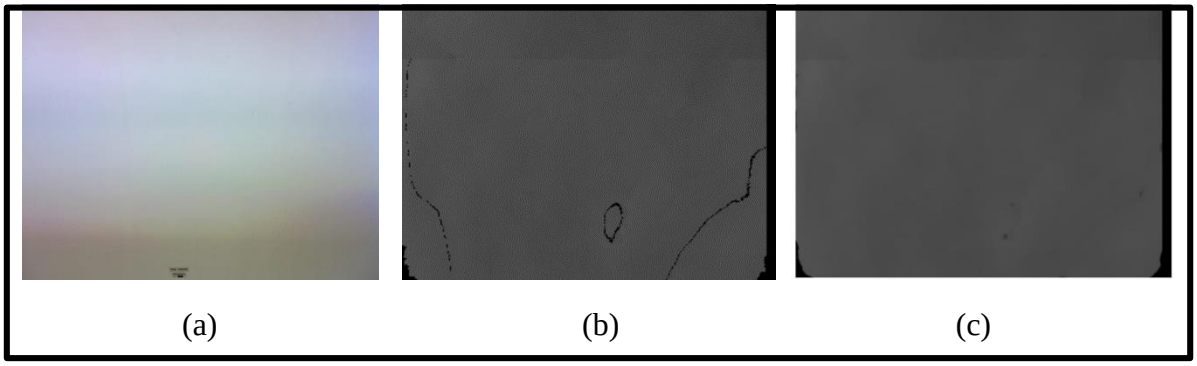


Figure 55: (a) Image de couleur: mur plan, (b) image de profondeur, (c) image de profondeur lissée avec le Filtre Médian.

L'application du filtre médian permet d'éliminer le bruit existant dans l'image sans contaminer les valeurs voisines aux valeurs incohérentes. Le filtre médian respecte le contraste de l'image et la luminosité de l'image. Dans les surfaces de l'image où l'intensité est monotone le filtre laisse l'image inchangée, en respectant les contours, et en éliminant les valeurs extrêmes.

3.3.2.3 Soft Computing: apprentissage et généralisation du model ANFIS

Le système Soft-Computing (phase3) estime la valeur de la profondeur métrique de chaque pixel dans une image à partir de deux entrées qui sont (voir Figure 43) :

X : représente la valeur de la profondeur fournie par le capteur ASUS en échelle de gris.

Y : représente les coordonnées des axes d'abscisse et ordonné d'un pixel dans une image.

Pour les deux paramètres d'entrée **X** et **Y**, la sortie f (voir Figure 43) représente la valeur métrique de la profondeur de chaque pixel dans une image, généré par le modèle.

3.3.3 Analyse des résultats d'étalonnage



Figure 56: La scène vue par le robot Pepper

La figure 56 montre la scène vue par le robot Pepper, tel que: la surface A représente la zone avec une forte intenté des faisceaux lumineux (voir figure 54-(a)), la surface B représente la zone avec une faible intensité des faisceaux lumineux.

L'erreur d'estimation de la profondeur augmente lorsque la mesure s'éloigne de la surface A vers le reste de l'image (surface B).

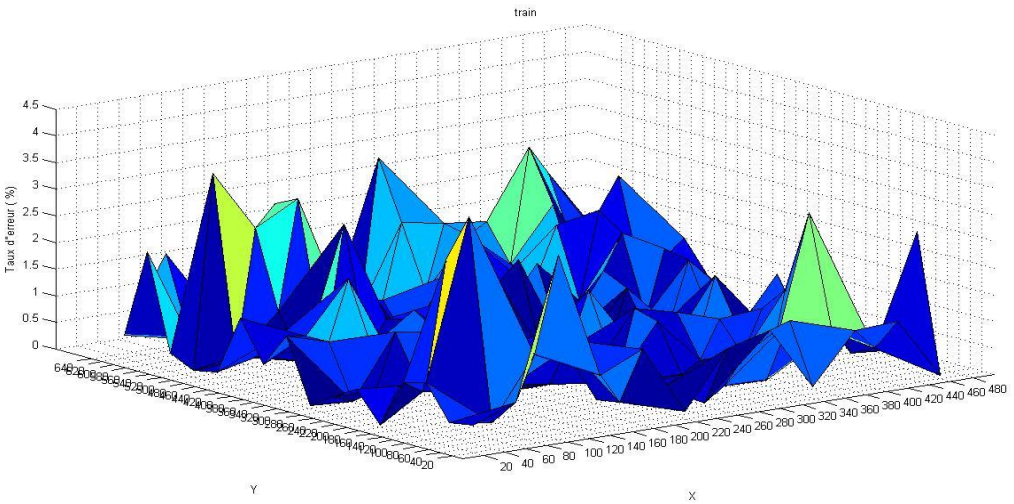


Figure 57: Pour la surface B et pour une profondeur MAX P = 2m, l'erreur d'estimation de la profondeur en mode apprentissage: ¾ de la base de données a été utilisés pour l'apprentissage.

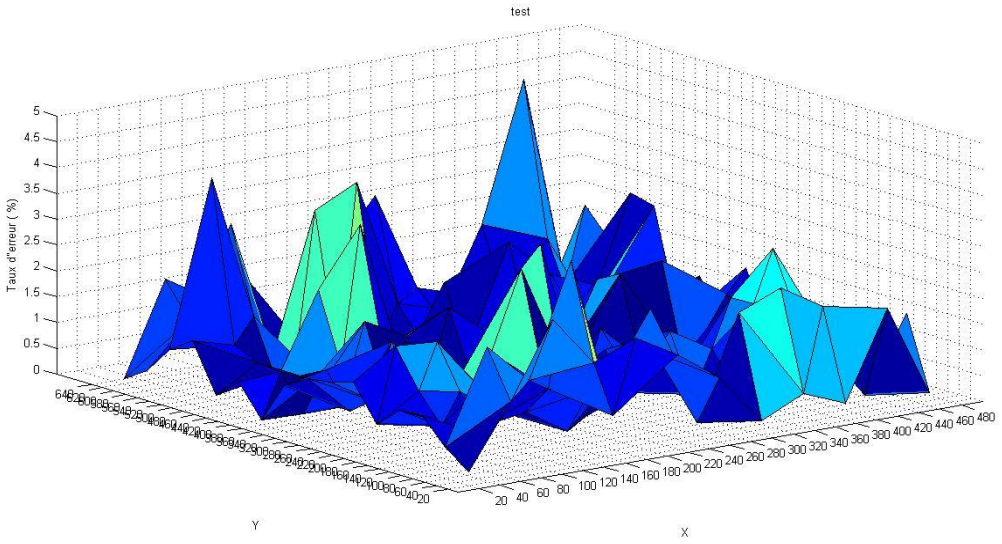


Figure 58 : Pour la surface B et pour une profondeur Max P = 2m, l'erreur d'estimation de la profondeur en mode généralisation: 1/4 de la base de données a été utilisé pour le test.

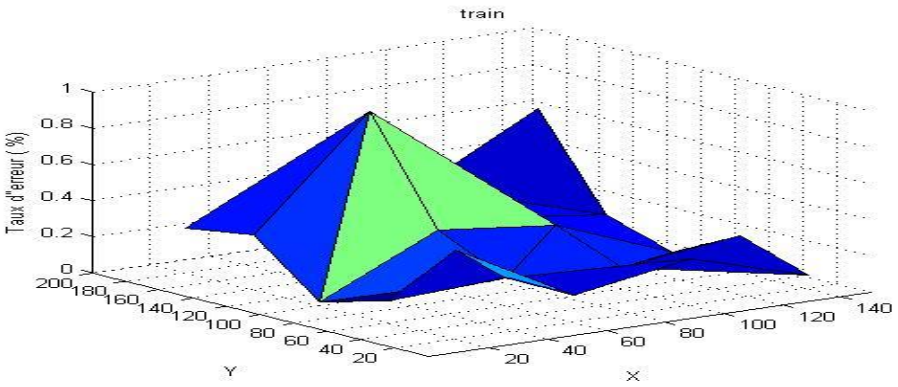


Figure 59 : Pour la surface A et pour une profondeur Max P = 2m, l'erreur d'estimation de la profondeur en mode apprentissage: ¾ de la base de donnée a été utilisé pour l'apprentissage.

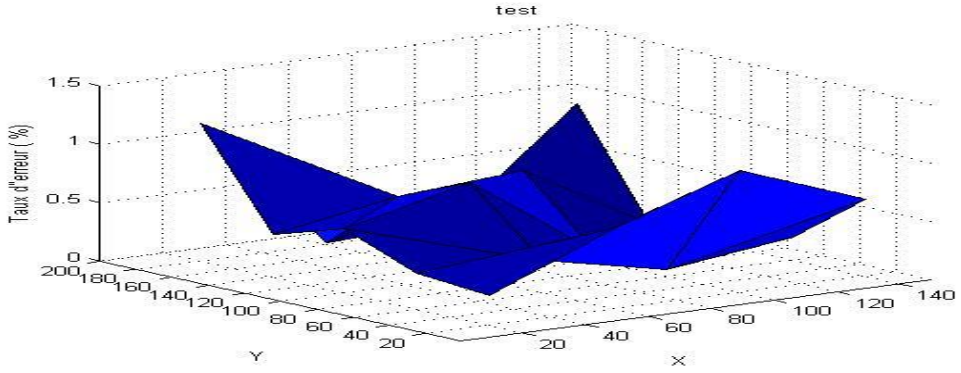


Figure 60 : Pour la surface A et pour une profondeur Max P= 2m, l'erreur d'estimation de la profondeur en mode généralisation: ¼ de la base de donnée a été utilisé pour le test

Les figures 57 et 58 montrent les résultats de l'estimation de la profondeur en centimètres en fonction des coordonnées (x, y) de chaque pixel sur le mur, indiquant l'erreur d'estimation pour un apprentissage qui a été effectué en utilisant ¾ de la base de données et un test qui a été effectué en utilisant le reste de la base de données.

Pour une profondeur MAX entre le capteur ASUS et le mur **P** = 2m, pour la surface B, l'erreur en mode apprentissage Max= 4.5% et en mode généralisation= 5%, tant dit que dans les figures 59 et 60, pour la surface A, l'erreur en mode apprentissage Max= 1% et en mode généralisation= 1.5%.

L'erreur d'estimation de la profondeur augmente lorsque le robot s'éloigne de mur.

Les figures 61 et 62 montrent l'erreur d'estimation de la profondeur, pour une profondeur max **P**= 3.5m, pour la surface B, l'erreur en mode apprentissage Max= 12% et en mode généralisation= 14%, tandis que dans les figures 63 et 64, pour la surface A, l'erreur en mode apprentissage Max= 2% et en mode généralisation= 4%.

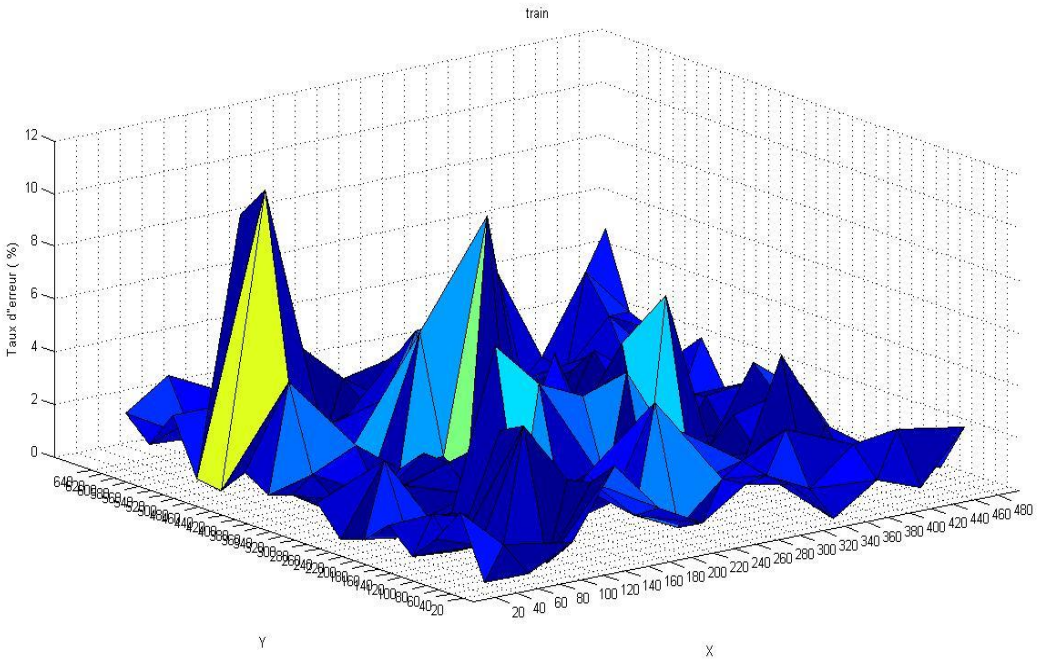


Figure 61: Pour la surface B et pour une profondeur Max P = 3.5m, l'erreur d'estimation de la profondeur en mode apprentissage: ¾ de la base de donnée a été utilisés pour l'apprentissage

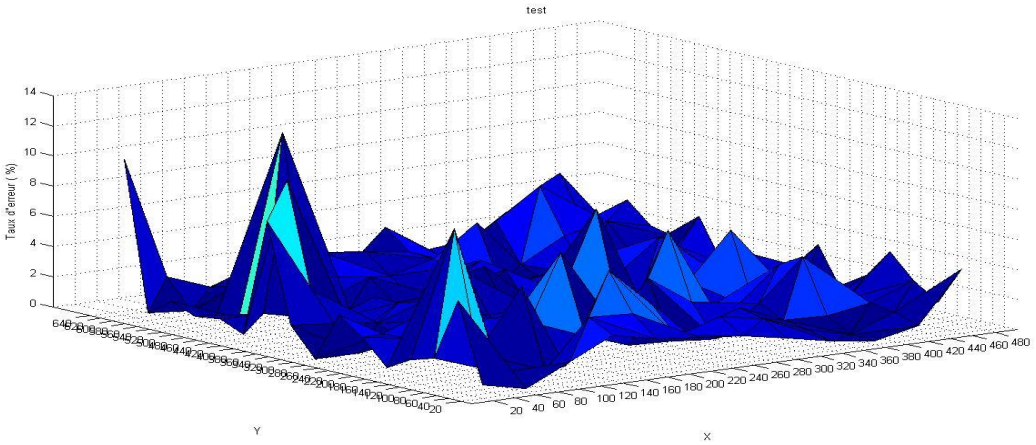


Figure 62: Pour la surface B et pour une profondeur Max P = 3.5m, l'erreur d'estimation de la profondeur en mode généralisation: 1/4 de la base de donnée a été utilisé pour le test.

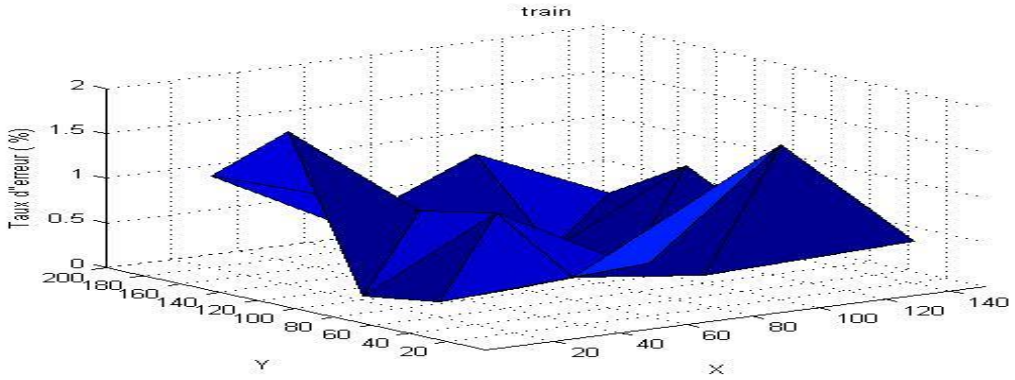


Figure 63: Pour la surface A et pour une profondeur Max P = 3.5m, l'erreur d'estimation de la profondeur en mode apprentissage: ¾ de la base de donnée a été utilisé pour l'apprentissage.

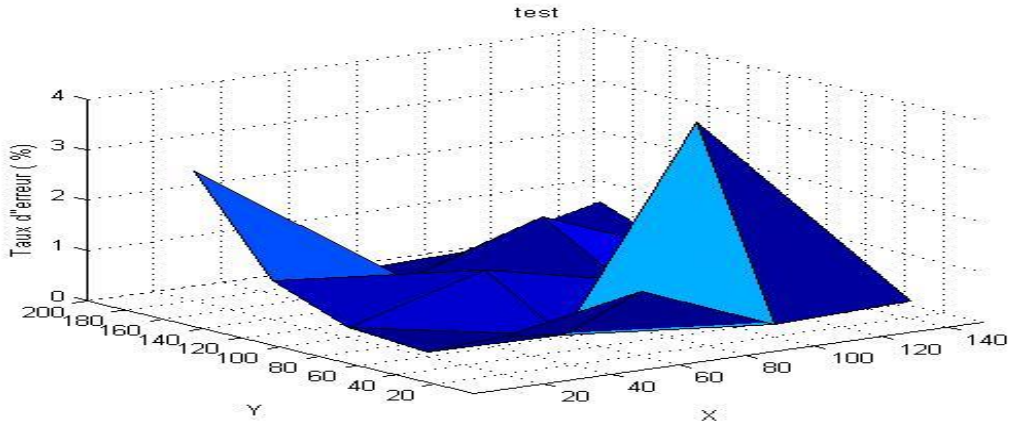


Figure 64: Pour la surface A et pour une profondeur Max P = 3.5m, l'erreur d'estimation de la profondeur en mode généralisation: 1/4 de la base de donnée a été utilisé pour le test.

Le tableau 6 résume les résultats des erreurs Max d'estimations de la profondeur dans les deux modes (apprentissage et généralisation), pour les deux surfaces (A, B) et pour deux valeurs de profondeur P_{max} (P_{max}=2m et P_{max}=3.5m). L'erreur Max d'estimation de la profondeur pour la surface B est plus élevée que la surface A, cette augmentation est dû aux données moins précises fournies par le capteur infrarouge de la Kinect (intensité des faisceaux lumineux sont plus fort dans la surface A que la surface B (voir figure 53)). Tandis que l'erreur Max d'estimation de la profondeur dans la surface B pour P_{max} =3.5m est beaucoup plus élevée que pour un P_{max}=2m, car la profondeur acquies pour P_{max}=3.5m sur la surface A est plus élevé dans la surface B (P_{max}>3.5m), sachant que la limite de mesure de la profondeur du capteur ASUS du robot Pepper est égale à 3.6m. Pour cela, nous recommandons P_{max}=2m comme profondeur Max d'acquisition de l'information de profondeur avec le capteur ASUS de Pepper.

	Surface A		Surface B	
La profondeur $\mathbf{P_{max}}$	2m	3.5m	2m	3.5m
L'erreur Max d'estimation de la profondeur en Mode apprentissage	1%	2%	4%	12%
L'erreur Max d'estimation de la profondeur en Mode généralisation	1.5%	4%	5%	14%

Tableau 6: Erreur d'estimation de la profondeur fournis par le capteur infrarouge du robot Pepper

L'erreur Max totale de notre approche pour l'estimation de la profondeur en cm, regroupe l'erreur Max de l'apprentissage égale à 5%, en ajoutant l'erreur de l'estimation de la profondeur générée par la technologie utilisée dans la Kinect1 est égale à 2% (0.04m/2m= 2%, voir figure 52), pour avoir au final une erreur Max égale à 7%.

Apres avoir estimé la valeur de profondeur en centimètres quasiment pour chaque pixel fournis par le capteur infrarouge du robot Pepper, nous procédons au calcul de la saillance en profondeur (DSM) ainsi qu'à la détermination de la carte de saillance finale (FSM).

3.3.4 Résultat expérimental de la validation de notre approche avec Pepper

La figure 65 donne un échantillon de résultats obtenus, illustrant les images d'entrée RGB (a) et de profondeur (b), la carte de saillance en 2D (CLSM) (c), la carte de saillance en profondeur DSM (d) et la carte de saillance finale FSM (e). Les objets saillants (habituellement,

plusieurs objets saillants à cause de la couleur, la lumière, la profondeur et la surface) ont été extraits de la scène. Une extraction réussie a été réalisée sur l'ensemble des images, les objets saillants extraits avec notre approche inclus l'objet saillant CLSM en couleur et lumière (boite de thé, boite en plastique rouge et le sac en jaune), ainsi que l'objet saillant en profondeur DSM (le carton) qui représente l'objet le plus proche au capteur infrarouge de Pepper.

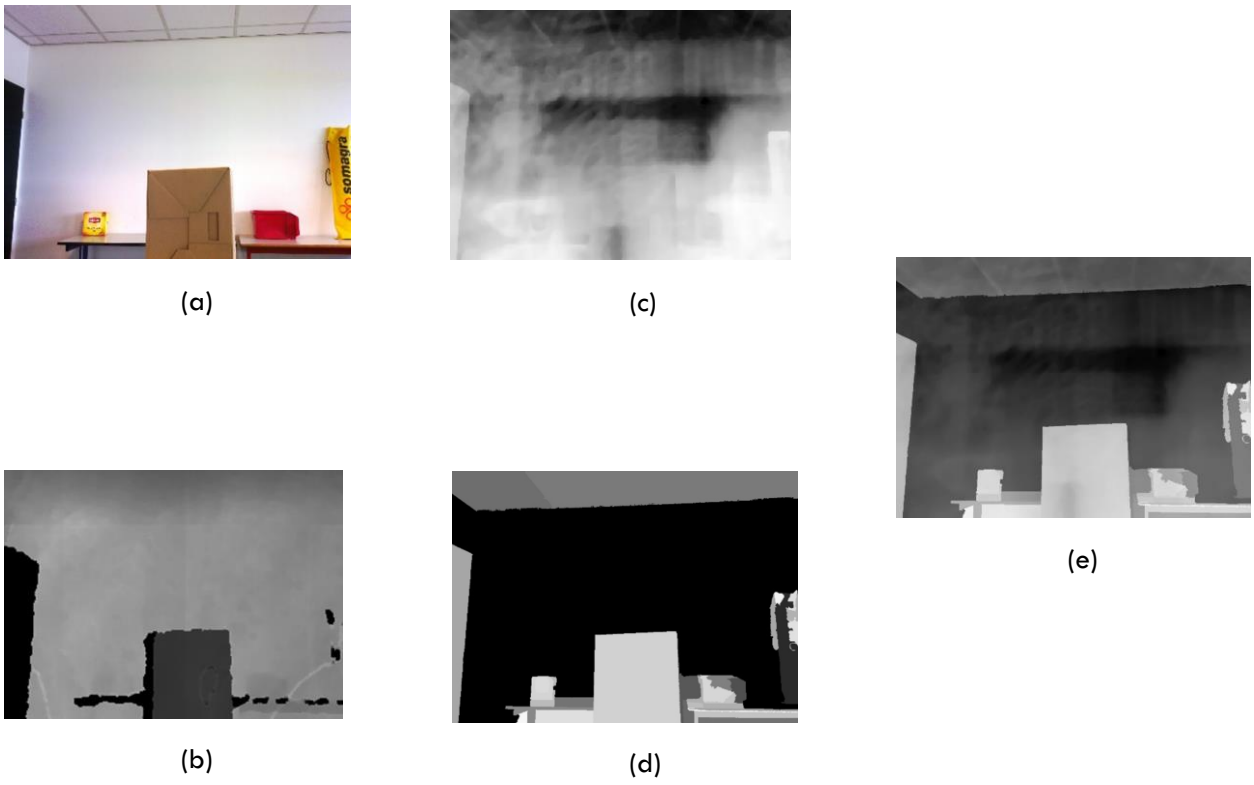
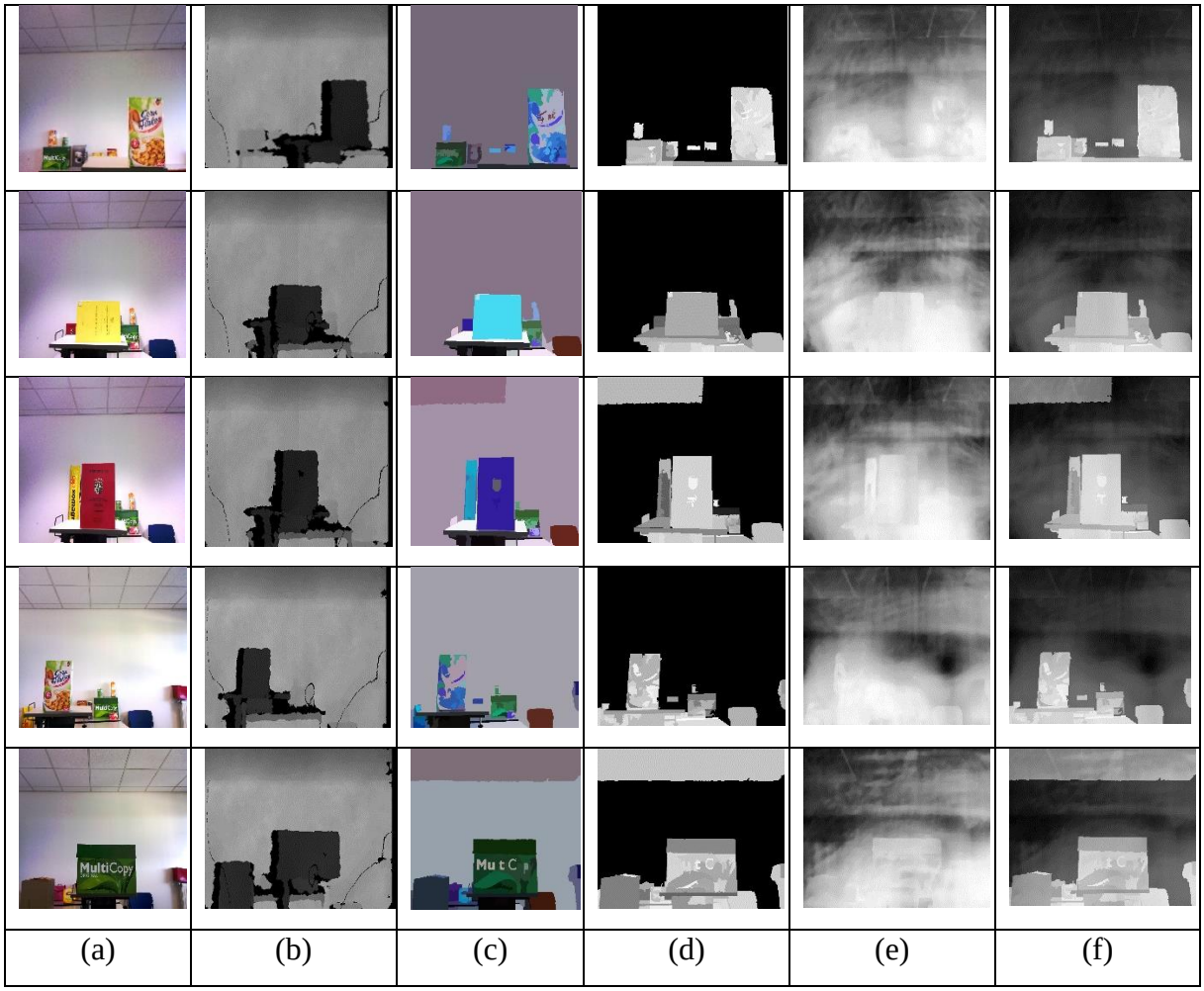


Figure 65: Exemples de CLSM, DSM et FSM obtenus à partir d'images RGB et de profondeur fournie par le capteur ASUS de Pepper.

La figure 66 donne quelques échantillons de résultats obtenus, illustrant les images d'entrée RGB (a), la profondeur (b), les images RGB segmentées (c), les cartes de saillance en profondeur DSM (d), les cartes de saillance en 2D CLSM (e) et les cartes de saillance finale correspondantes FSM (f). Les objets saillants (habituellement, plusieurs objets saillants à cause de la couleur, la lumière, la profondeur et la surface) ont été extraits de chaque scène.

L'avantage principal de notre approche vis-à-vis des autres approches de la détection de la saillance 3D, réside sur le fait que nous détectons plusieurs objets saillants pour chaque différente expérimentation, contrairement aux approches basées sur l'hypothèse qui favorise la localisation de l'objet qui se trouve majoritairement au centre de l'image.



(a) (b) (c) (d) (e) (f)

Figure 66: Les échantillons des résultats obtenus

Dans le chapitre 2, l'évaluation de la base de données de 1000 images RGBD, a été effectué, en déterminant le masque de la saillance final extrait de la carte de saillance finale, par le calcul de F-mesure Max grâce à l'information Ground-Truth. Nous avons constaté que le meilleur seuil choisi pour extraire le masque de saillance finale est pour un interval qui varie de 140 à 180.

Dans nos expériences avec Pepper, le manque de l'image de Ground-Truth nous oblige à varier le seuil de 140 à 180 pour avoir le meilleur masque de saillance.

3.3.5 Discussion

Le système 'Soft Computing' utilisé dans notre méthode d'étalonnage du capteur de profondeur du robot Pepper, permet en sortie de fournir la valeur de la profondeur en centimètres pour chaque pixel. L'erreur d'estimation de la profondeur métrique de notre méthode qui englobe l'erreur de l'apprentissage 5% (voir tableau 7), plus l'erreur d'estimation

de la profondeur du capteur Kinect utilisé comme outil de référence 2% (voir la figure 52), pour avoir au final un erreur égale à 7% , qui est considéré raisonnable.

Le résultat expérimental démontre que l'information sur la profondeur est un complément utile aux modèles existants basés sur la couleur et la lumière, en particulier lorsque les objets restent plus près du robot Pepper (voir figure 65: objet en carton), a un contraste de profondeur élevé par rapport à l'arrière-plan où nous avons une plage de profondeur relativement faible en saillance DSM.

Nous avons constaté aussi, que notre approche est très sensible à la qualité de la segmentation générée par l'algorithme Mean-Shift qui est sensible de son coté à la luminosité de la scène, l'augmentation du nombre des régions segmentés augmente le temps de traitement. Nous visualisons cette constatation dans la troisième image (voir figure 66-ligne3 et 5), ou le plafond apparait dans notre DSM car il était segmenté avec l'algorithme Mean-Shift.
La précision de l'étalonnage entre la camera RGB et le capteur infrarouge influence sur les résultats de notre approche, l'objectif de l'étalonnage est d'associer pour chaque pixel de l'image de profondeur avec son correspondant dans l'image de couleur. Comme nous avons mentionné précédemment, l'étalonnage entre les deux images couleur et profondeur de Pepper a été établie linéairement par manque d'outil d'étalonnage des images de profondeur fournies par le constructeur du robot Pepper.

Dans la section suivante, nous utilisons le système flou pour permettra au robot Pepper de transmettre ces observations en terme de reconnaissance et de positionnement d'un objet dans l'environnement. Pour la reconnaissance, nous allons utiliser l'algorithme SURF (Speeded Up Robust Features) (Viola et al. 2001), et la logique floue pour le positionnement de l'objet. L'image couleur de la camera RGB du robot Pepper est utile pour la reconnaissance ainsi que pour notre système floue qui positionne l'objet dans son environnement (objet en haut, en bas, à gauche et à droite). L'information de profondeur fournie par le capteur infrarouge du robot Pepper est utile pour positionner l'objet en terme de profondeur (objet proche ou loin du robot Pepper).

3.4 La caractérisation qualificative

Comme cela a déjà été mentionné dans la section précédente, suite à la détection des objets saillants, nous nous s'inspirons de la méthode de (Hassan et al. 2015) pour la caractérisation qualificative de ces objets dans leur environnement.

La figure 67 montre les différentes étapes de traitement de la perception visuelle et la caractérisation qualificative de l'environnement, qui sont :

1. La détection des objets saillants 3D.
2. La reconnaissance de ces objets saillants utilisant l'algorithme SURF.
3. Le calcul de la profondeur de l'objet par rapport à la position du Pepper, en appliquant notre méthode d'étalonnage 'Soft Computing'.
4. Le calcul de la hauteur et la largeur de l'objet utilisant l'algorithme de (Hassan et al. 2015)
5. L'utilisation du système floue mise en œuvre par (Hassan et al. 2015) pour le positionnement spatiale de l'objet par rapport à ses cordonnées x, y, z .

3.4.1 La reconnaissance de l'objet

Inspiré du travail de (Hassan et al. 2015)), nous adaptons son travail sur nos images de couleur et de profondeur acquises par les capteurs du robot Pepper au lieu Kinect. Nous avons en premier lieu créé pour chaque objet détecté une base de donnée d'image 'boite de cornflakes', la détection est meilleure si nous disposons d'un grand nombre d'images prises pour le même objet dans différentes conditions de luminosité et d'orientation. En deuxième lieu, nous avons appliqué l'algorithme SURF pour la reconnaissance de l'objet 'Cornflakes', qui est correctement détecté dans la figure 68.

3.4.2 La caractérisation et la localisation de l'objet dans la scène

Une fois que la reconnaissance est validée, nous procédons à la caractérisation spatiale en utilisant la méthode géométrique pour calculer les coordonnées spatiales de l'objet x_m ,y_m et z_m (décrite dans la section 3.2.1) à partir de l'image de couleur et profondeur fournies par le capteur infrarouge de robot Pepper.

Nous avons adapté pour le robot Pepper le calcul des coordonnées spatiales de l'objet x_m , y_m et z_m, définis initialement pour la Kinect. Cette adaptation est établie à cause du manque des paramètres intrinsèques d'étalonnage de la camera de profondeur de Pepper, qui est utile pour le calcul des équations x_m , y_m et z_m.

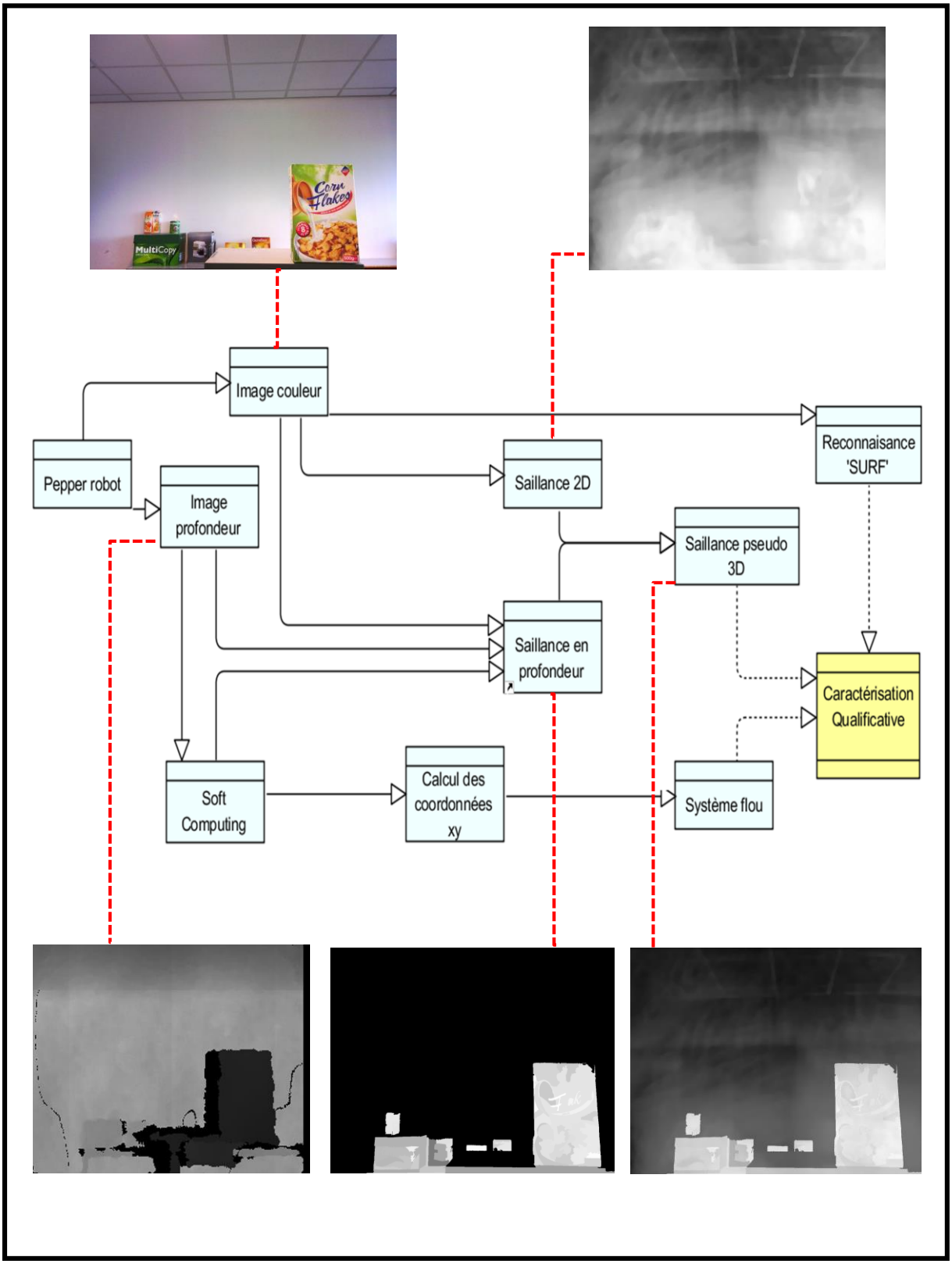


Figure 67: Schéma global de l'implémentation de l'approche de la détection de la saillance 3D.

Nous calculons ensuite la largeur et la hauteur de l'objet (section 3.2.1). Nous transformons les données métriques de l'objet détecté en données spatiales (objet proche et loin, à gauche et droite, en haut et en bas), pour cela nous utilisons la logique floue implémentée par (Hassan et al. 2015).



Figure 68: Objet 'Cornflakes' détecté et cadré en couleur vert par l'algorithme SURF

Le système flou utilise la bibliothèque <skfuzzy>. Pour la fuzzification, Hassan a utilisé la fonction d'appartenance trapézoïdale, elle a définie des règles de surface et de localisation correspondant à chaque combinaison possible de la hauteur et la largeur, la profondeur et l'emplacement de l'objet dans l'image. Hassan a utilisé l'inférence floue de type Mamdani qui permet une description linguistique du système par une base des règles floues de la forme : **SI** antécédent 1 **ET** antécédent 2 **ALORS** consequence1.

Les règles de la base de connaissances sont listées ci-dessous, où la largeur $\in \{P, M, G, TG\}$, la hauteur $\in \{P, M, G, TG\}$ et la surface $\in \{P, M, G, TG\}$, avec $\{P, M, G, TG\}$ correspond à $Petite, Moyenne, Grande, Très\ Grande$:

SI largeur est P **ET** hauteur est P **ALORS** surface est P.

SI largeur est P **ET** hauteur est M **ALORS** surface est P.

SI largeur est P **ET** hauteur est G **ALORS** surface est M.

SI largeur est P **ET** hauteur est TG **ALORS** surface est M.

SI largeur est M **ET** hauteur est P **ALORS** surface est P.

SI largeur est M **ET** hauteur est M **ALORS** surface est M.

SI largeur est M **ET** hauteur est G **ALORS** surface est G.

SI largeur est M **ET** hauteur est TG **ALORS** surface est G.

SI largeur est G **ET** hauteur est P **ALORS** surface est M.

SI largeur est G **ET** hauteur est M **ALORS** surface est G.

SI largeur est G **ET** hauteur est G **ALORS** surface est G.

SI largeur est G **ET** hauteur est TG **ALORS** surface est TG.

SI largeur est TG **ET** hauteur est P **ALORS** surface est M.

SI largeur est TG **ET** hauteur est M **ALORS** surface est G

SI largeur est TG **ET** hauteur est G **ALORS** surface est TG.

SI largeur est TG **ET** hauteur est TG **ALORS** surface est TG.

Les figures 70, 71 et 72 donnent des fonctions d'appartenance pour " La profondeur ", " La largeur " et " La hauteur ", respectivement.

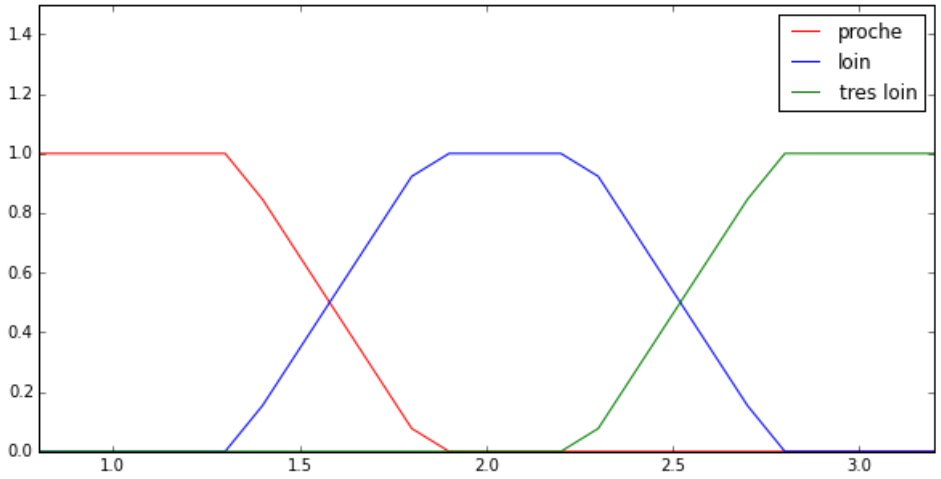


Figure 69: Profondeur des objets

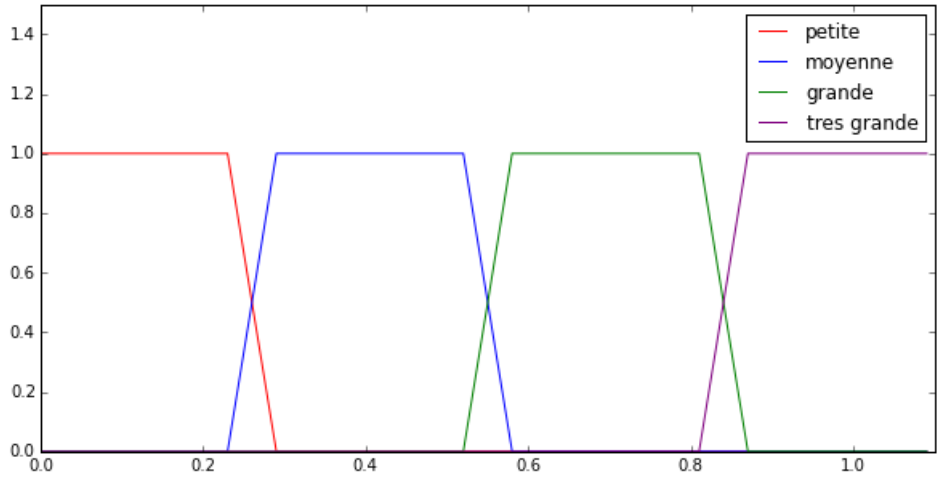


Figure 70: Largeur des objets

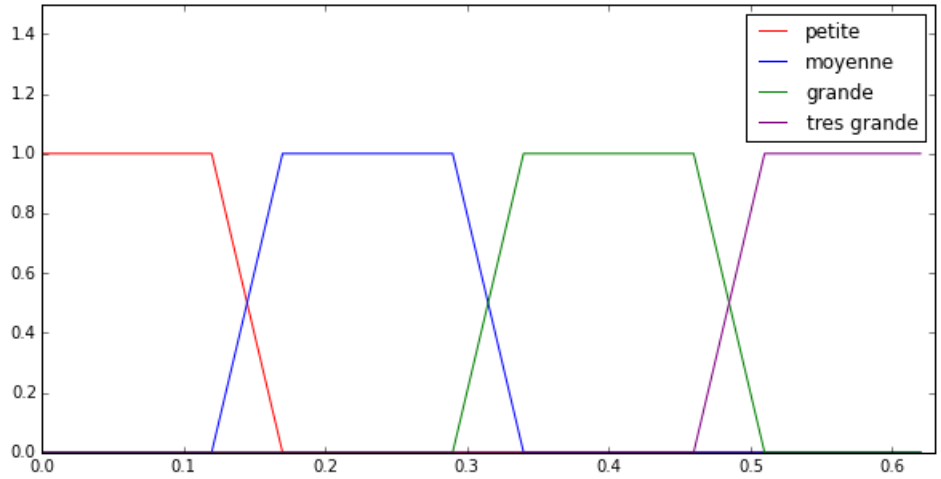


Figure 71: Hauteur des objets

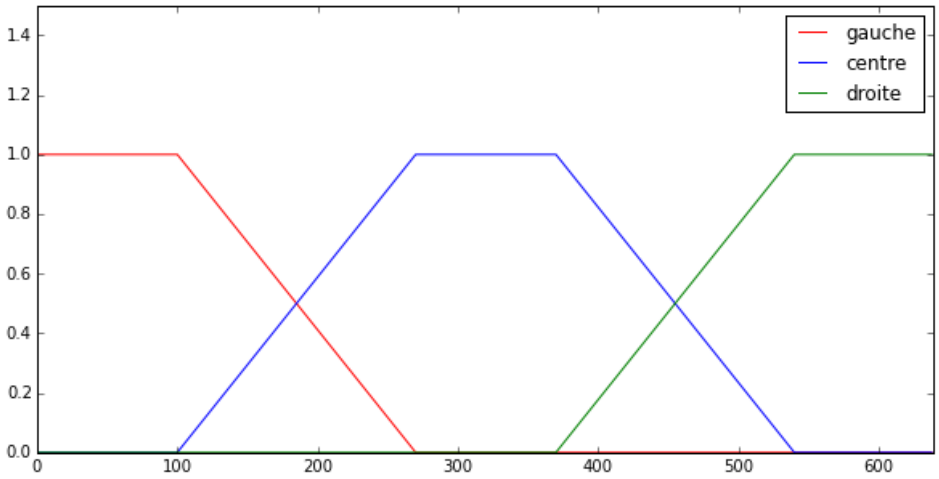


Figure 72: Positionnement horizontal des objets

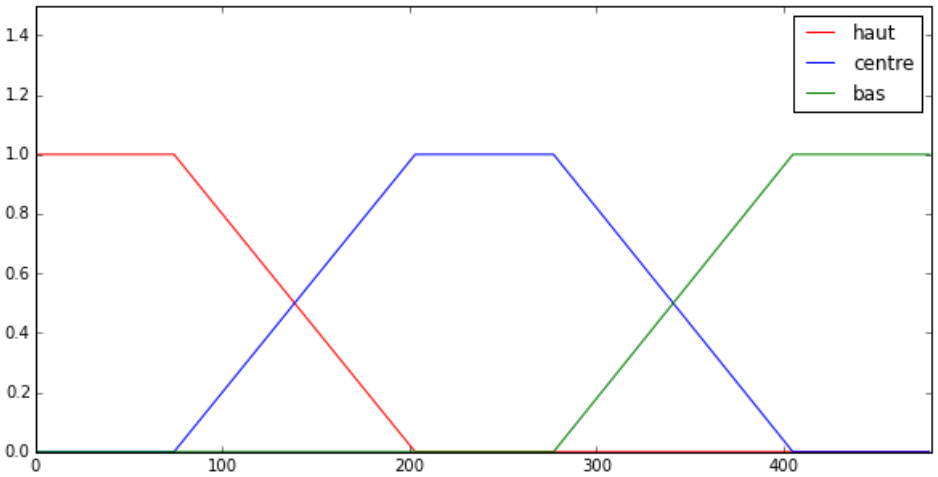


Figure 73: Positionnement vertical des objets

De la même manière, les fonctions d'appartenance décrivant la caractérisation spatiale des objets (par exemple leur emplacement, etc.) sont obtenues par la détermination de la

position de pixel du centre de masse d'un objet par rapport au centre de l'image. En tenant compte de la résolution spatiale de la caméra couleur de Pepper (640x480), les figures 73 et 74 donnent les fonctions d'appartenance des positions spatiales horizontales et verticales, respectivement.

Dans la section suivante, nous clôturons notre travail par un exemple de réaction du robot humanoïde Pepper vers son environnement.

3.4.3 La réaction du robot dans un environnement réel

La figure 75, montre les différents aspects du comportement du système perceptuel du robot Pepper sous forme organigramme

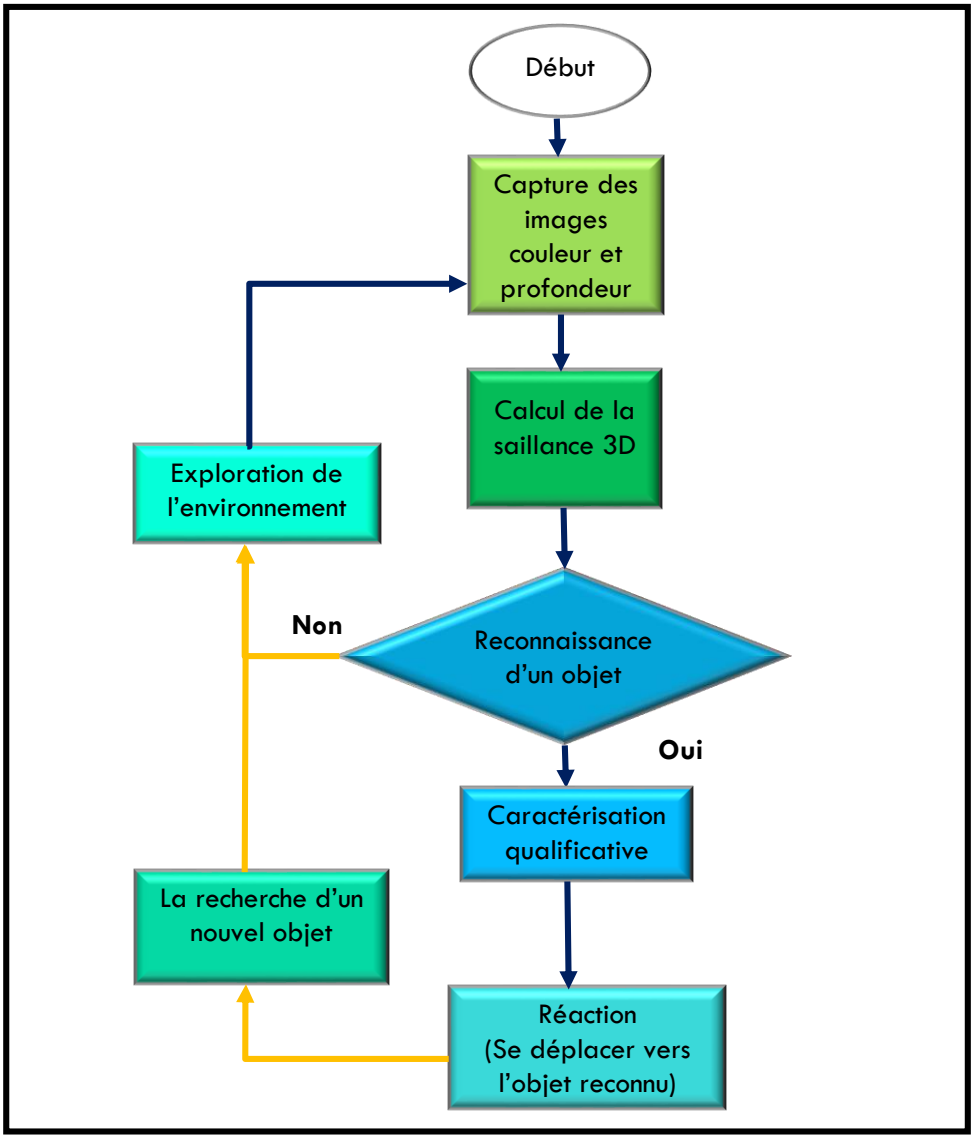


Figure 74: Les différents aspects du comportement du système perceptuel du robot.

Au début, le robot Pepper commence par un comportement d'exploration libre de son environnement, il navigue librement et capture des images de couleur et de profondeur qui servent à calculer la saillance 3D pour la reconnaissance des objets. Si un objet est reconnu, le robot caractérise qualificativement l'objet (détermination de l'emplacement de l'objet dans l'espace (x_m,y_m,z_m)), puis il se déplace vers l'objet reconnu. Par la suite, le robot continu sa navigation pour reconnaitre un nouvel objet. Si aucun objet n'est reconnu, le robot continue son exploration de l'environnement.

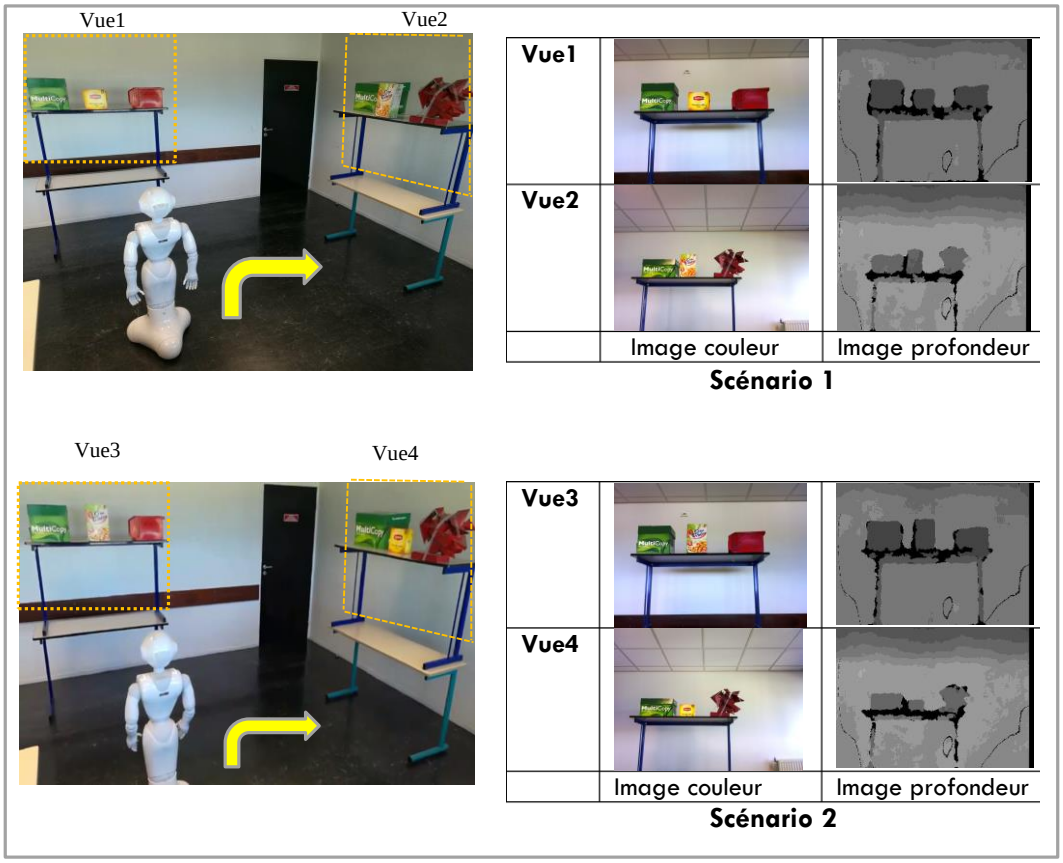


Figure 75: Deux scénarios d'exploration de l'environnement.

La figure76 montre deux scénarios d'exploration de l'environnement, où des différents objets du quotidien ont été distribués dans l'environnement de différentes manières afin de tester la capacité du robot à reconnaitre un objet ou deux objets dans la même vue prise pendant l'exploration. Dans le scénario 1, nous avons placé les deux objets à reconnaitre 'Cornflakes' et 'Sculpture' dans la vue2. En revanche, dans le scénario 2 nous avons placé un seul objet à reconnaitre dans chaque vue. La flèche jaune indique approximativement le mouvement du robot dans l'environnement. Dans la figure 77, les objets 'Cornflakes' et 'Sculpture' reconnus pendant l'exploration sont montrés et cadrés en couleur verte.

Vue1	
Vue2	
Vue3	
Vue4	

Figure 76: Les objets reconnus dans différents vues.

Comme nous l'avons expliqué, le robot commence par la capture des images de couleur et de profondeur, détecte les objets saillants en 3D. Si le robot reconnait des objets, il les caractérise qualificativement puis il les communique par la voix vocale et par l'affichage d'un message dans l'écran, comme exemple : « Bonjour, j'ai détecté l'objet cornflakes, il est situé en bas, à droite, cet objet a une petite surface et il est loin de moi ».

La figure 78 présente les résultats de la détection de la saillance obtenus, illustrant pour chaque vue (voir figure 76) les images d'entrée RGB (a), la profondeur (b), les images RGB segmentées (c), les cartes de saillance en profondeur DSM (d), les cartes de saillance en 2D CLSM (e) et les cartes de saillance finale correspondantes FSM (f).

La détection des objets saillants est sensible à :

1) La qualité de l'image de profondeur, par exemple dans les 4 vues, nous observons un manque de l'information dans les images de la profondeur (figure78-b) au niveau de la table, cela est dû à la nature de sa surface lisse.
2) L'étalonnage entre la caméra couleur et profondeur pour faire correspondre chaque pixel de l'image couleur avec son correspondant dans l'image de la profondeur.
3) La qualité de l'image couleur capturé peut dégrader la qualité de la segmentation.

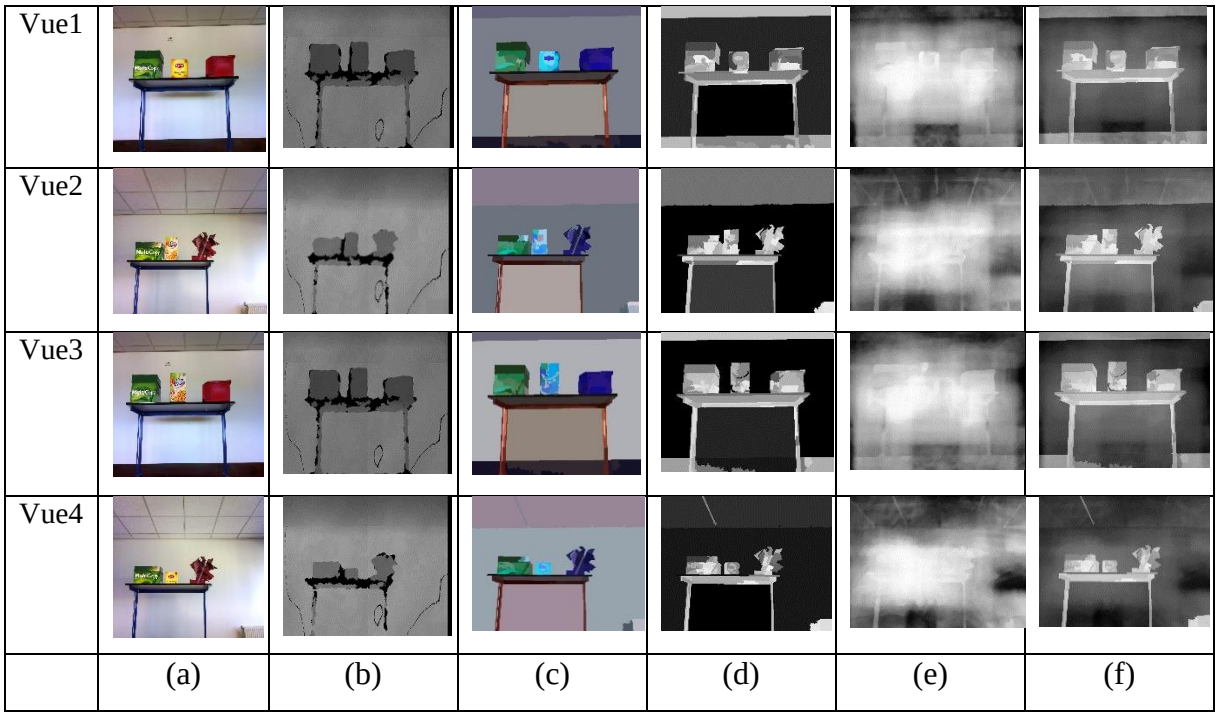


Figure 77: Les résultats de la détection de la saillance 3D.

Nombre de région dans une image	22	31	38	40	45	46	61
Capture des images couleur et profondeur par le robot Pepper.	0.5s	0.5s	0.5s	0.5s	0.5s	0.5s	0.5s
Calcul de la saillance 2D (Ramik algorithme)	9s	9s	9.5s	9.5s	9.5s	9.5s	9.5
Segmentation de l'image couleur avec MEAN-SHIFT	19s	38s	39s	50s	60s	60s	61s
Calcul de la saillance en profondeur (calcul d'histogramme)	1.5s	2.5s	3s	3s	3.5s	4s	4.5s
Calcul de la saillance 3D (fusion)	0.05s	0.05s	0.05s	0.05s	0.05s	0.05s	0.05s
Total	30.05s	50.05s	52.05s	63.05s	73.55s	74.05s	75.55s

Tableau 7: Bilan temporel de notre approche

Les temps d'exécution des étapes des calculs contribuant à la détection de la saillance et à la caractérisation spatiale de l'environnement sont représentés dans le tableau 7. Il est pertinent de rappeler que les expérimentations rapportées ont essentiellement concerné la validation des concepts étudiés et plus particulièrement la qualité des résultats. Notamment, pour ce qui concerne l'extension (portage) des concepts sur le robot Pepper, l'objectif était de montrer la portabilité des techniques (algorithmes) mise au point sans l'optimisation de l'implantation de ces algorithmes. Ainsi, le bilan temporel de traitement de la détection de la saillance 3D pour différentes images, sur la base de l'utilisation d'un processeur 1.6GHz et une mémoire RAM de 4Go, montre des disparités importantes de temps d'exécution notamment pour ce qui concerne la segmentation (l'algorithme de segmentation MEAN-SHIFT) et le calcul de la saillance 2D.

En effet, nous remarquons que l'algorithme de segmentation MEAN-SHIFT est un algorithme couteux en terme de l'espace mémoire utilisé et le temps de traitement, occupant jusqu'à 80% du temps d'exécution. Le choix de cet algorithme avait été motivé par la qualité de la segmentation des objets. Cependant, l'utilisation de cet algorithme ne semble pas le plus approprié pour ce qui concerne le portage sur le robot Pepper. Par ailleurs, le calcul de la saillance 2D a été implanté sans l'optimisation du code pour un portage sur le Pepper. En effet, le code initial avait été optimisé pour des images de plus petites tailles (et de ce fait, pour des résolutions plus faibles). C'est pourquoi, une optimisation de cet algorithme conduirait aux temps d'exécution de l'ordre de 0.5s à 1s pour la détection de la saillance 2D. Il est donc envisageable d'obtenir (avec une segmentation plus rapide et une détection de la saillance 2D optimisée) des temps d'exécution de l'ordre de une à deux secondes par image.

3.5 Conclusion

Dans ce chapitre, nous avons proposé deux expériences utilisant de 'Soft Computing':

Dans la première expérience, la faisabilité du système 'Soft Computing' a été vérifiée grâce au calcul de la distance entre deux objets en centimètres, bénéficiant de la machine ANFIS pour résoudre le problème de l'apprentissage d'une faible base de données.

Dans la deuxième expérience, nous avons utilisé le système 'Soft Computing' pour générer l'information de profondeur en centimètres en utilisant le capteur infrarouge du robot Pepper, afin de calculer la saillance en profondeur et la saillance finale. Notre système a prouvé sa précision de mesure pour avoir une erreur d'estimation maximale de 2% dans la surface A et 6% dans la surface B. Dans ce contexte, nous avons proposé une approche utilisant la reconnaissance et la caractérisation qualificative de l'objet saillant, en s'appuyant sur le travail de (Hassan et al. 2015) avec un portage des concepts sur le robot Pepper. Cependant, les expérimentations rapportées ont essentiellement concerné la validation des concepts étudiés et plus particulièrement la qualité des résultats, notamment pour ce qui concerne l'extension (portage) des algorithmes sur le robot Pepper. Ainsi, le bilan temporel de traitement de la détection de la saillance 3D pour différentes images, sur la base de l'utilisation d'un processeur 1.6GHz et une mémoire RAM de 4Go, montre des disparités importantes de temps d'exécution notamment pour ce qui concerne la segmentation (l'algorithme de segmentation MEAN-SHIFT) et le calcul de la saillance 2D.

Conclusion générale

Conclusion

Dans cette thèse, nous avons fourni une étude approfondie sur les différentes technologies d'acquisition de l'information de profondeur, en justifiant le choix de la Kinect puis le robot Pepper comme implémentation de notre travail. Aussi nous avons fourni une étude approfondie sur la détection des objets saillants 2D et 3D.

Nous avons proposé une approche pour la détection de la saillance 3D. Les expériences vérifient que la saillance produite en profondeur peut fonctionner comme un complément utile aux modèles existants basés sur la couleur et la lumière, surtout lorsque les objets restent plus proche du capteur de profondeur, qui ont un contraste de profondeur élevé par rapport à l'arrière-plan. Nous avons testé notre approche sur la base de données de 1000 images proposée dans (Peng et al. 2014) et la base de données proposée dans (Cheng et al. 2014), et nous avons comparé nos résultats avec ceux montrés dans la Figure 34. Notre approche atteint des performances élevées et produit des résultats robustes.

La détection de la saillance 3D a été étendue sur d'autres capteurs de profondeur telle que le capteur ASUS du robot Pepper. Pour cela nous avons proposé le système 'Soft Computing' basé sur l'algorithme d'apprentissage ANFIS, utilisé pour estimer la valeur de la profondeur métrique fournie par le capteur ASUS. Cette estimation est conçue par l'étalonnage de l'information de profondeur du capteur ASUS à l'aide de l'information métrique de profondeur de la Kinect.

Pour valider la faisabilité de notre système 'Soft Computing', nous avons proposé une approche expérimentale permettant d'estimer la distance entre deux objets en centimètres. L'utilisation d'ANFIS a permis une meilleure estimation en comparaison d'autres méthodes comme SVR, MLP, et l'interpolation bilinéaire. Notre système produit des résultats robustes et précis.

L'approche de la détection de la saillance en 3D avec le robot Pepper, est la première partie clé de l'unité d'acquisition des connaissances de l'environnement pour le robot. Nous avons pu démontrer que les objets saillants détectés peuvent être utilisés dans la deuxième partie

qui garantit une détection et une reconnaissance d'objet par le détecteur SURF. Par la suite, nous avons adapté la méthode de (Hassan et al. 2015) pour la caractérisation spatiale des objets détectés avec le robot Pepper au lieu de la Kinect. Les résultats expérimentaux montrent une précision élevée d'estimation des caractéristiques spatiales. Cependant, la qualité d'étalonnage de la camera de profondeur du robot Pepper a une influence importante sur la précision de nos résultats.

Nous clôturons notre travail, par un test pratique sur le robot Pepper qui traduit son comportement dans des conditions réelles. Toutes les parties constitutives du notre système ont été assemblés et exploités comme une seule structure. Cela a été couronné de succès et toutes les unités du système collaborent efficacement pour acquérir des connaissances spatiales de l'environnement. Le système dans son état complet a montré qu'il est capable de remplir les tâches pour lesquelles il a été conçu. Une vidéo montrant différents aspects de ce qui a été décrit dans cette thèse est disponible.

Perspectives

Notre travail de thèse ouvre deux perspectives :

- Compte tenu des temps d'exécutions relevés, la première perspective à court terme, est l'optimisation des deux étapes les plus couteuses en temps de calcul de la chaine du traitement.
- Une seconde perspective, à moyen terme, est de lier la détection de la saillance (et la caractérisation spatiale des objets saillants détectés) à la stratégie de navigation du robot établissant ainsi une stratégie visuelle pour une navigation autonome (ou semi-autonome) du robot. Cette étape nécessite une classification des caractéristiques qualitatives décrivant la localisation spatiale des objets ainsi qu'une association de ces caractéristiques à des scénarios (stratégies) de navigation.

Espérant rester (continuer) dans les domaines de la perception et de l'intelligence artificiel, comme perspective personnel de carrière de recherche, je vais profiter de l'expérience de Monsieur MADANI dans le domaine du traitement des signaux physiologiques pour développer un système de rééducation chez les personnes affectées par l'AVC.

Publications

Chapitres d'ouvrages et articles longs dans des recueils avec comité de lecture :

(Madani et al. 2015) : Madani, K., Fraihat, H., Sabourin, C. 2016. "Machine-Learning-Based Visual Objects' Distances Evaluation: a Comparison of ANFIS, MLP, SVR and Bilinear Interpolation Models", *Computational Intelligence*, Studies in Computational Intelligence, Vol. 669, Springer, ISBN: 978-3-319-48504-1, pp. 462 – 479.

Communications dans congrès internationaux avec acte et comité de lecture :

(Fraihat et al. 2015b) : Fraihat, H., Madani, K., and Sabourin, C. 2015. "Soft-Computing Based Fast Visual Objects' Distance Evaluation for Robots' Vision", in *Proc. of the International Conference on Intelligent Data Acquisition and Advanced Computing Systems (IEEE/IDAACS 2015)*, Warsaw, Poland, September 24 – 26, 2015, Vol. 1, pp. 81 – 86.

(Fraihat et al. 2015a) : Fraihat, H., Sabourin, C. and Madani, K. 2015. "Learning-Based Distance Evaluation in robot vision: A Comparison of ANFIS, MLP, SVR and Bilinear Interpolation Models", in Proc. of the International Conference on Neural Computation Theory & Applications (NCTA 2015), International Joint Conference on Computational Intelligence (IJCCI 2015), Lisbon, Portugal, November 12 – 14, 2015, ISBN: 978-989-758-157-1, pp. 168 – 173.

(Madani et al. 2015) : Fraihat, H., Madani, K., Sabourin, C. 2017. "A Pseudo-3D Vision-Based Dual Approach for Machine-Awareness in Indoor Environment Combining Multi-Resolution Visual Information", in *Proc. of the International Work-conference on Artificial Neural Networks (IWANN 2017)*, Cadiz, Spain, June 14 – 16, 2017, LNCS series, Vol. 10306, Part II, Springer, ISBN 978-3-31959146-3,pp. 644 – 654.

(Madani et al. 2015) : Madani, , K., Fraihat, H., Sabourin, C. 2017. "Machine-Awareness in Indoor Environment: A Pseudo-3D Vision-Based Approach Combining Multi-Resolution Visual Information", in *Proc. of the IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems (IEEE/IDAACS 2017)*, Bucharest, Romania, September 21 – 23, 2017, Vol. 1, pp. 419 – 424.

Communications orales ou posters dans des colloques ou journées nationaux ou locaux :

H. Fraihat, C. Sabourin, **K. Madani**, "Estimation de distance entre objets à partir d'un capteur pseudo-3D et des techniques connexionnistes (Soft-Computing)", Journée Recherche IUT de Sénart-FB, Lieusaint, 30 juin 2015. (Poster)

Annexes

Annexe1

1.1 MLP

Le réseau de perceptron multicouches est un neurone artificiel organisé en couches où l'information se déplace dans une direction, de la couche d'entrée à la couche de sortie (Bardos 1997; Hérault et al. 1994; Parizeau 2004; Touzet 1992) . La figure 79 donne un exemple d'un réseau contenant une couche d'entrée, des couches cachées et une couche de sortie. La couche d'entrée représente une couche virtuelle associée aux entrées de données. Il ne contient pas de neurone. Les couches cachées suivantes sont des couches de neurones. Les sorties des neurones de la dernière couche correspondent toujours aux sorties de données souhaitées. Un perceptron multicouches peut avoir n'importe quel nombre de couches et nombre de neurones (ou entrées) dans n'importe quelle couche. Les neurones sont reliés ensemble par des connexions pondérées. C'est le poids w_{ij} de ces connexions qui gère le fonctionnement du réseau et assure la transformation des données d'entrée en données de sorties.

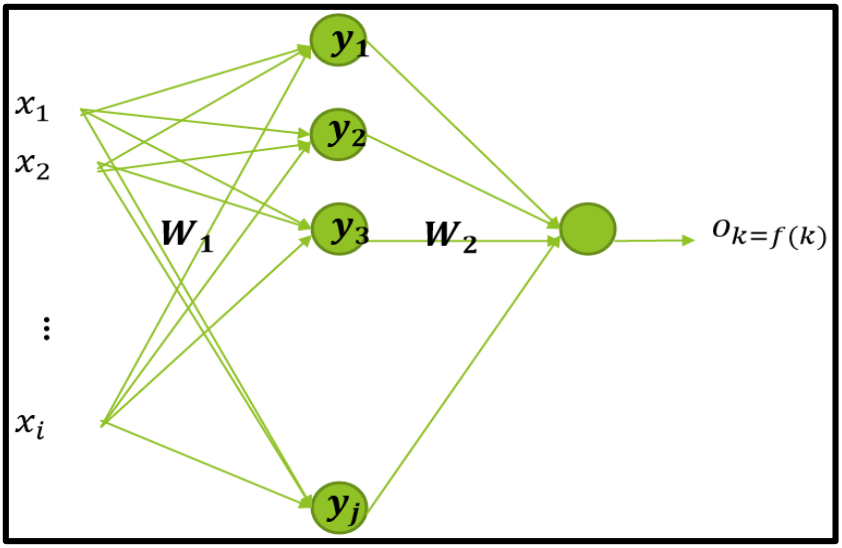


Figure 78: L'architecture de MLP avec une couche cachée

Par exemple, les neurones de la couche niv produisent une réponse sur chaque neurone dans la couche niv + 1 en calculant une somme pondérée des neurones dans le niveau niv auquel il est connecté. Cette somme est ensuite transformée par une fonction tangente g avec g(x) =tan(x).

Les neurones y_j dans la couche cachée et le neurone O_k dans la couche de sortie sont calculés en utilisant les équations 39 et 40

$$y_j = g\left[\sum_{i=1}^{n} w_{1(i,j)} g(x_i) + seuil1(j)\right] \quad (39)$$

$$o_k = \sum_{j=1}^{h} w_{2(j,k)} g\left[\sum_{i=1}^{n} w_{1(i,j)} g(x_i) + seuil1(j)\right] + seuil2(k) \quad (40)$$

Où

n : nombre de donnée d'entrée.

h: nombre de neurones dans la couche cachée.

Dans le mode d'apprentissage supervisé, l'algorithme de rétro-propagation du gradient d'erreur est utilisé pour estimer le poids w1 (i, j) et w2 (j, k) et le seuil1 (j) et le seuil2 (k).

L'algorithme de rétro-propagation est utilisé pour minimiser l'erreur E calculée entre la sortie o_k et la sortie souhaitée d_k (voir Eq38).

$$E = \sum_{k=1}^{m} (o_k - d_k)^2 \quad (41)$$

Initialement, les poids sont sélectionnés au hasard, puis le gradient de E est propagé à nouveau en changeant la valeur des poids pour minimiser l'erreur E. Les poids à déterminer sont

W1 (i, j) et W2 (j, k).

$$W(t) = w(t-1) + \Delta w(t)$$

Avec $\Delta w(t) = -\varepsilon \frac{\partial E}{\partial W}$ (42)

t: le nombre d'itération de l'algorithme.

Le coefficient ε affecte la rétro-propagation de la vitesse de convergence de la qualité et de l'algorithme qui optimise les poids W et les seuils, il aide à accélérer l'apprentissage et à lisser la sortie d'erreur carrée.

Dans notre travail, nous utilisons un MLP avec une couche cachée, avec 304 variables d'entrée, 100 neurones sur la couche cachée et 19 neurones sur la couche de sortie. Le problème qui a rencontré la phase d'apprentissage des données est lorsque le nombre de pixels entre deux objets est fixé (entrée = x1 = x2 = xi) pour différente valeur de profondeur, ce problème est dû à la

résolution de la caméra et à la précision de l'algorithme de segmentation d'image. Ce problème complique l'apprentissage et peut affecter négativement les performances du réseau.

1.2 SVR

L'algorithme SVM, initialement développé pour les problèmes de classification, a été étendu à des problèmes de régression. Nous nous concentrerons uniquement sur la régression SVM que nous présentons dans les principes de base suivants. Cependant, une représentation détaillée peut être trouvée dans (Smola et al. 2004).

Compte tenu d'un ensemble de données $D = \{(\boldsymbol{x}_i, y_i) | 1 \leq i \leq N\}, \quad \boldsymbol{x}_i \in R^n, \quad y_i \in R$ (43)

Dans la régression ε-SVM (Cherif 2013; Li et al. 2000; Üstün et al. 2006; Vapnik 1995), l'objectif est de trouver une fonction f (x) qui s'écarte au plus de ε dans la cible réelle y_i pour toutes les données d'entraînement et, en même temps, être aussi régulière que possible. En d'autres termes, les erreurs qui sont inférieures à ε sont tolérées, alors qu'un écart plus important que ε est pénalisé. Nous commençons par décrire le cas des fonctions linéaires F sous la forme:

$$f(\boldsymbol{x}_i) = \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b \tag{44}$$

$\langle \cdot, \cdot \rangle$ Indique le produit dot en R^n

La "planéité" dans ce cas signifie un petit w (moins sensible aux perturbations des caractéristiques).

Par conséquent, nous pouvons écrire le problème comme suit :

$$\min \frac{1}{2} \|\boldsymbol{w}\|^2$$

Sujet à $\begin{cases} y_i - \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle - b \leq \varepsilon \\ \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b - y_i \leq \varepsilon \end{cases}$ (45)

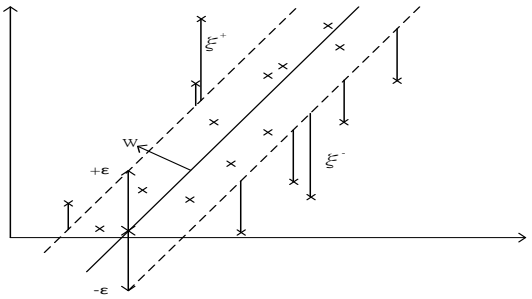


Figure 79.a : Réglage de la fonction de perte dans le cas d'une SVM linéaire

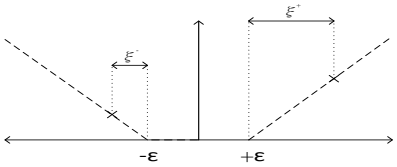


Figure 79. b. Ɛ sensibilité de la fonction de perte

F Approximante de toutes les paires (x_i, y_i) avec la précision ε, mais nous pouvons également autoriser certaines erreurs. Dans certains cas où nous voulons tolérer certaines erreurs dans l'estimation, les variables de relaxation ($\xi_i^+ + \xi_i^-$) peuvent être introduites pour traiter les données que nous ne pouvons pas aborder linéairement ε- dans l'optimisation du problème (Eq 41).

Ainsi, nous obtenons la formulation proposée dans (Vapnik 1995)

$$\min \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}(\xi_i^+ + \xi_i^-) \qquad (46)$$
$$\begin{cases} y_i - \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle - b \le \varepsilon + \xi_i^+ \\ \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b - y_i \le \varepsilon + \xi_i^- \\ \xi_i^+, \xi_i^- \ge 0 \end{cases}$$

C : Le paramètre de régularisation détermine le compromis entre la planéité et la tolérance des erreurs.

ξ_i^+, ξ_i^- : Les variables lâches déterminent le degré d'erreur loin du tube insensible ε.

La figure 80 illustre le cas d'une fonction linéaire. Seuls les points en dehors de la zone (entre + ε et -ε) sont pénalisés de manière linéaire (la pénalité est proportionnelle à la distance entre le point en question de la limite de tolérance ε).

Problèmes doubles et programmes quadratiques :

Nous associons un multiplicateur Lagrange pour chaque contrainte décrite ci-dessus, le problème original peut être décrit par son double problème, qui est un problème d'optimisation quadratique sans contraintes. Le lagrangien du système s'exprime comme suit:

$$L_{primal}(w, b, \xi_i^+, \xi_i^-) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}(\xi_i^+ + \xi_i^-) - \sum_{i=1}^{N}\alpha_i^+(\varepsilon + \xi_i^+ - y_i + \langle \boldsymbol{w}, \boldsymbol{x}_i\rangle + b) - \sum_{i=1}^{N}\alpha_i^-(\varepsilon + \xi_i^- + y_i - \langle \boldsymbol{w}, \boldsymbol{x}_i\rangle - b) - \sum_{i=1}^{N}(\mu_i^+\xi_i^+ + \mu_i^-\xi_i^-) \quad (47)$$

$$\alpha_i^{+(-)} \geq 0 \text{and } \mu_i^{+(-)} \geq 0$$

Où $\mu_i^+, \mu_i^-, \alpha_i^+, \alpha_i^-$ sont les multiplicateurs Lagrange. A partir de cette formulation et en utilisant les dérivées partielles du Lagrange L. La fonction f peut être écrite:

$$f(\boldsymbol{x}) = \sum_{i=1}^{N}(\alpha_i^+ - \alpha_i^-)\langle \boldsymbol{x}, \boldsymbol{x}_i\rangle + b \quad (48)$$

$$\boldsymbol{w} = \sum_{i=1}^{N}(\alpha_i^+ - \alpha_i^-)\boldsymbol{x}_i$$

C'est ce qu'on appelle "Support Vector" dans lequel w peut être complètement décrit comme une combinaison linéaire de $\boldsymbol{x}_i$.

Calcul de b:

Le paramètre b de l'équation 48 peut être calculé par les conditions de Karush-Kuhn-Tucker:

$$\begin{cases}\alpha_i^+(\varepsilon + \xi_i^+ - y_i + \langle \boldsymbol{w}, \boldsymbol{x}_i\rangle + b) = 0 \\ \alpha_i^-(\varepsilon + \xi_i^- + y_i - \langle \boldsymbol{w}, \boldsymbol{x}_i\rangle - b) = 0 \\ \begin{cases}\mu_i^+\xi_i^+ = (\mathrm{C} - \alpha_i^+)\xi_i^+ = 0 \\ \mu_i^-\xi_i^- = (\mathrm{C} - \alpha_i^-)\xi_i^- = 0\end{cases}\end{cases} \quad (49)$$

Nous exploitons les équations du système que nous obtenons:

$$\max\{y_i - \langle \boldsymbol{w}, \boldsymbol{x}_i\rangle + \varepsilon \,|\, \alpha_i^+ < C \text{ or } \alpha_i^- > 0\} \leq b \leq \min\{y_i - \langle \boldsymbol{w}, \boldsymbol{x}_i\rangle - \varepsilon \,|\, \alpha_i^+ > 0 \text{ or } \alpha_i^- < C\}$$

Non linéarité et noyau

L'étape suivante consiste à adapter le SVM à un cas non linéaire pour approximer les données fortement non linéaires. Cela peut se faire en utilisant une transformation de l'espace de recherche d'origine à l'aide d'une fonction φ. Le choix de la fonction φ est crucial, nous restreignons la fonction Φ pour laquelle il existe un noyau tel que $K(\boldsymbol{x}, \boldsymbol{x}_i) = \langle \varphi(\boldsymbol{x}), \varphi(\boldsymbol{x}_i) \rangle$.

L'extension de l'équation 48 dans le cas non-linéaire utilisant le noyau K est ainsi écrit:

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} (\alpha_i^+ - \alpha_i^-) \langle \varphi(\boldsymbol{x}), \varphi(\boldsymbol{x}_i) \rangle + b \qquad (50)$$

$$\boldsymbol{w} = \sum_{i=1}^{N} (\alpha_i^+ - \alpha_i^-) \varphi(\boldsymbol{x}_i)$$

1.3 Interpolation bilinéaire

Nous connaissons les valeurs des points P1, P2, P3 et P4 (Figure 81) qui représentent les profondeurs et les distances en centimètres entre deux objets en pixels, on recherche la distance bilinéaire intermédiaire entre deux classes, chaque classe représente une distance entre deux objets dans Centimètres. Cette distance intermédiaire (P) est donnée par l'équation 51 (Chen et al. 2010; Cok 1987; Lu et al. 2008) :

$$P = (1- \lambda). [(1- \mu).P1 + \mu.P3] + \lambda. [(1- \mu).P2 + \mu.P4] \qquad (51)$$

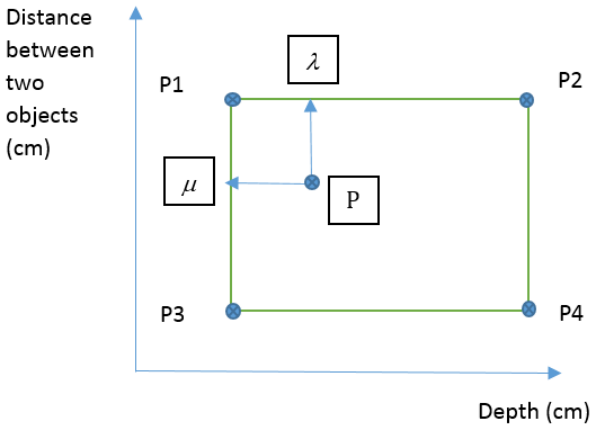


Figure 80: Le diagramme schématique de l'Algorithme d'interpolation bilinéaire.

Annexe2

2.1 La fuzzification

Le premier traitement qui entre en compte dans la conception d'un système flou est la fuzzification, son rôle est de rendre floues (fuzzifier) les entrées et sorties du système, elle se compose de plusieurs étapes :

- Etablir les variables linguistiques
- Déterminer le nombre de valeurs linguistiques.
- Attribuer un sens numérique à chaque quantificateur flou grâce à la fonction d'appartenance.

2.1.1 La fonction d'appartenance

La fonction d'appartenance décrit le comportement d'un système qualitativement, $\mu_A(x)$ donnée par l'équation 52, qui décrit le degré avec lequel l'élément x appartient à A telle que (Flaus 1994):

$$\mu: x \in A \rightarrow \mu_A(x) \in [0,1] \begin{cases} \mu_A(x) = 1 \; si \; x \; est \; complétement \; dans \; A \\ 0 < \mu_A(x) < 1 \; si \; x \; est \; partiellement \; dans \; A \\ \mu_A(x) = 0 \; si \; x \; est \; \text{à} \; l'extérieur de \; A \end{cases} \quad (52)$$

2.1.2 Les différentes formes des fonctions d'appartenance

La fonction d'appartenance est présentée par plusieurs formes, les formes les plus connue sont triangulaire, trapézoïdales et gaussienne.

- **Fonction d'appartennace tringulaire**

Telle que montrée dans la figure 82, la fonction triangulaire est caractérisé par trois paramètres x1, x2, et x3, définit comme suit :

$$\mu_A(x) = \begin{cases} \dfrac{x - x_1}{x_2 - x_1} & si\ x_1 \leq x \leq x_2 \\ \dfrac{x - x_3}{x_2 - x_3} & si\ x_2 \leq x \leq x_3 \\ 0 & sinon \end{cases} \quad (53)$$

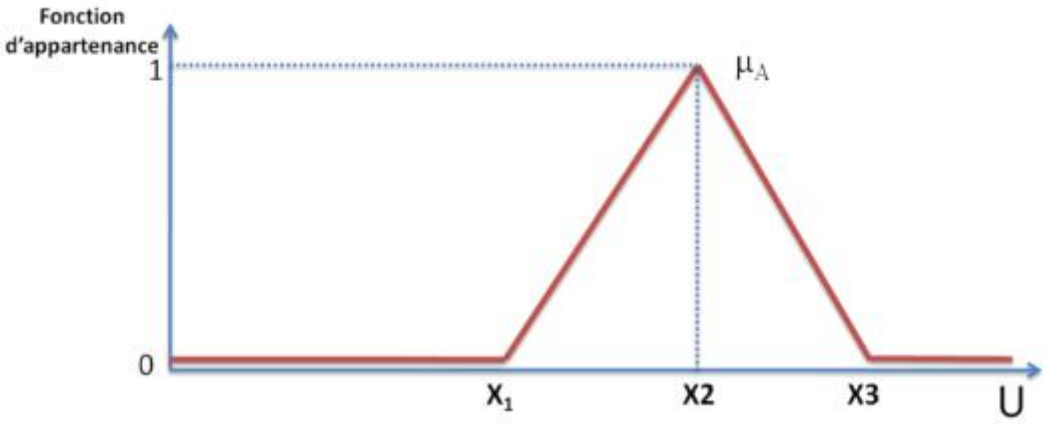


Figure 81: Fonction d'appartenance triangulaire (Ayouni 2012).

- **Fonction d'appartenance trapézoïdale**

Telle que montrée dans la figure 83, la fonction trapézoïdale est caractérisé par trois paramètres x1, x2, x3 et x4 définit comme suit :

$$\mu_A(x) = \begin{cases} \dfrac{x - x_1}{x_2 - x_1} & si\ x_1 \leq x \leq x_2 \\ 1 & si\ x_2 \leq x \leq x_3 \\ \dfrac{x - x_4}{x_3 - x_4} & si\ x_3 \leq x \leq x_4 \\ 0 & sinon \end{cases} \quad (54)$$

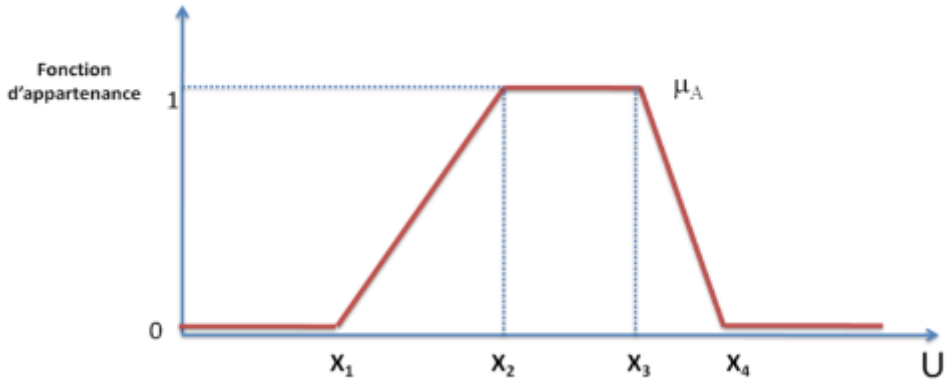


Figure 82: Fonction d'appartenance trapézoïdale (Ayouni 2012).

- **Fonction d'appartenance gaussienne**

Telle que montrée dans la figure 84, la fonction gaussienne utilisé dans notre système flou est caractérisé par sa valeur centrale m et son écart type ϭ, définit comme suit :

$$\mu_A(x) = e^{-\frac{1}{2}(\frac{x-m}{\sigma})^2} \quad (55)$$

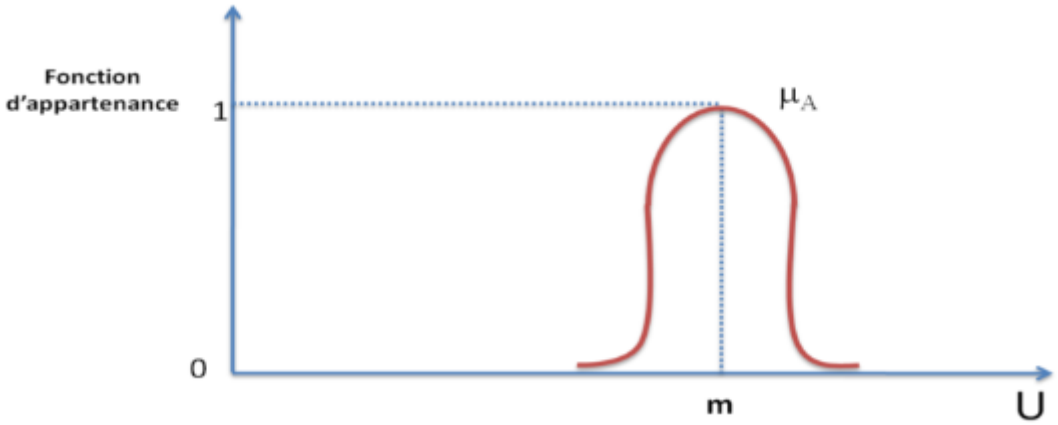


Figure 83: Fonction d'appartenance gaussienne(Ayouni 2012).

Ces dernières formes sont utilisées car elles comportent des zones où la notion est vraie, des zones où elle est fausse.

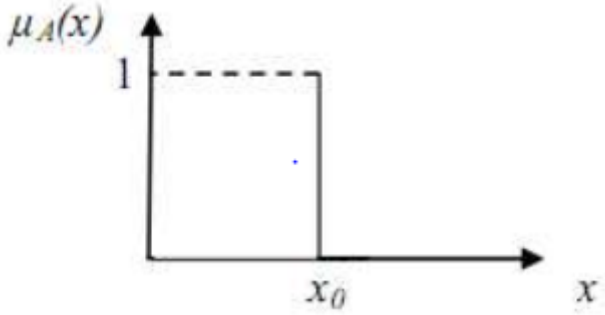
Nous avons présenté ci-dessus les types e fonctions d'appartenance les plus utilisées, en outre il existe d'autre formes telles que la fonction singleton.

- **Fonction d'appartenance singleton**

On a un singleton lorsqu'une fonction d'appartenance est partout nulle sauf en un point, cette fonction d'appartenance est décrit le degré avec lequel l'élément x appartient à A telle que :

$$\mu_{Fx}(u) = 1 \quad si \quad \mu = \mu_0$$
$$\mu_{Fx}(u) = 0 \quad si \quad \mu \neq \mu_0 \qquad (56)$$

La figure 85 illustre la fonction d'appartenance de type singleton.



(57)

Figure 84: Fonction d'appartenance singleton

2.2 Les méthodes d'inférences floues

La première étape dans le système d'inférence floue consiste à créer des règles dont la syntaxe est linguistique :

SI antécédent 1 **ET** antécédent 2 **ALORS** consequence1

Antécédent : variable d'entrée, dans notre model c'est X (les coordonnées (x, y) d'un pixel de l'image de profondeur fournis par le robot Pepper) et Y (la valeur de profondeur de ce pixel en échelle de gris) (voir section 3.3).

Conséquence : variable de sortie, dans notre model représente la valeur de la profondeur en centimètre.

Exemple :

- **SI** la coordonné X= 80 **ET** la cordonnée Y= 77 **ET** la valeur de profondeur en échelle

de gris = 133 **ALORS** la valeur de la profondeur = 350 centimètres (objet plus proche au centre de l'image est plus sombre)

- **SI** la coordonné X= 80 **ET** la cordonnée Y= 200 **ET** la valeur de profondeur en échelle de gris = 166 **ALORS** la valeur de la profondeur = 320 centimètres.

On peut représenter la base des règles sous forme un tableau ou une matrice (voir figure 86). Un exemple de la base des règles est le suivant :

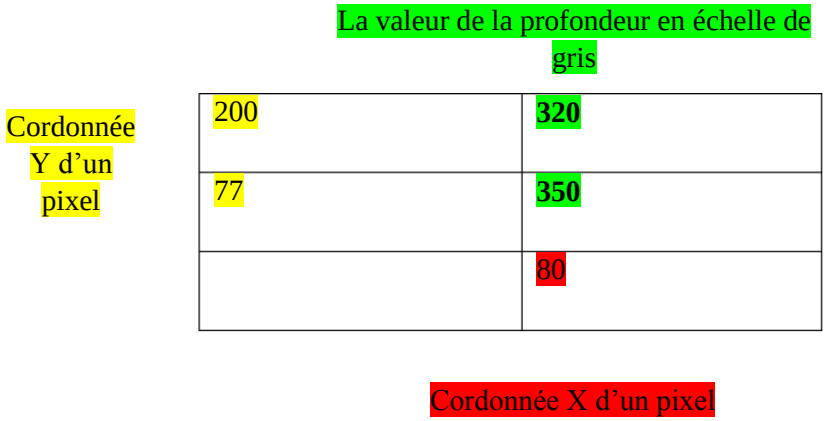


Figure 85: Une partie d'une base des règles sous forme de matrice

La quantité des règles dépend du nombre d'entrées et du nombre de sortie de chacune d'elles. Plusieurs types de systèmes d'inférence floue existent dans la littérature qui peuvent être mis en œuvre dans les systèmes flous, parmi ces types, la méthode de Mamdani et la méthode de Sugeno utilisé dans notre model.

2.2.1 Inférence floue de type Mamdani

Le model de Mamdani décris le système flou utilisant une base des règles suivent le model de Zadeh (Nakoula et al. 1997). Dans ce modèle, la conjonction des antécédents sont interprétée par l'opération min et la disjonction des regles comme le max, appelée la méthode min-max, la t-norme et la t-conorme sont définis comme :

t-norme: $\mu_{A \cap B}(x) = min(\mu_A(x), \mu_B(x))$

t-conorme : $\mu_{A \cup B}(x) = max(\mu_A(x), \mu_B(x))$

2.2.2 Inférence floue de type Sugeno

Notre système qui utilise ANFIS (Artificial Neuro-Fuzzy Inference Systems) est de type Sugeno utilisé pour l'apprentissage de nos données d'entrée X et Y (voir figure 42). La règle typique du processus d'inférence floue à la forme mentionné précédemment comme suit (Sugeno et al. 1988) :

SI *antécédent I = x* **ET** *antécédent 2 =y* **ALORS** Sortie $z = ax + by + c$.

Pour un model sugeno d'ordre zéro, dans ce cas la sortie z=c (a=b=0), l'ensemble flou de l'inférence des conséquences sera un ensemble discret avec un nombre bien fini et précise qui facilite le calcul de l'algorithme de défuzzification.

2.3 Défuzzification

Plusieurs stratégies de défuzzification existent, les plus utilisées sont la méthode de centre de gravité, la méthode du maximum et la méthode de la moyenne des maxima. Nous avons utilisé dans notre model la méthode de centre de gravité.

Le centre de gravité est calculé en utilisant la méthode d'inférence de type Sugeno d'ordre zéro.

La méthode du centre de gravité (CDG) détermine la cordonnée x du centre de gravité de la surface de l'ensemble flou. La figure 87 est un exemple d'une solution floue donc la solution z_0 est déterminé par la méthode du centre de gravité.

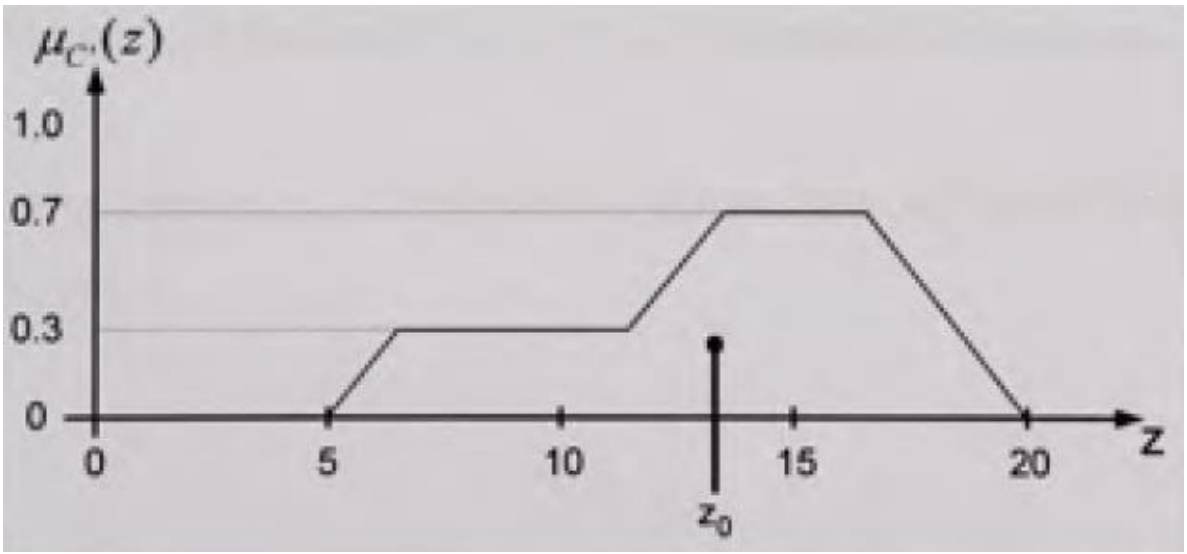


Figure 86: Méthode du centre de la gravité (Pagliari et al. 2015)

Le calcul de z_0 utilisant la méthode du centre de gravité est définit par :

$$z_0 = \frac{\int_z f_\beta(z).z\,dz}{\int_z f_\beta(z)\,dz} \quad (58)$$

Notre model utilise une méthode d'inférence de type sugeno d'ordre zéro, la défuzzification est donnée par la moyenne pondéré, le calcul de la moyenne pondérée est donné par l'équation 59:

$$y(x) = \frac{\sum_{k=1}^{N} \mu_k(x).f_k(x)}{\sum_{k=1}^{N} \mu_k(x)} \quad (59)$$

Avec :

$$\mu_k(x) = \prod_{i=1}^{n} F_i^k,\ F_i^k \in \{F_i^1, F_i^2, \dots, F_i^{m_i}\}\ repr\acute{e}sente\ le\ degr\acute{e}\ d'activationde\ la\ r\grave{e}gle\ R_k.$$

Bibliographies

Achanta, R., Estrada, F., Wils, P., and Süsstrunk, S. 2008. "Salient region detection and segmentation," *Computer Vision Systems*), pp 66-75.

Achanta, R., Hemami, S., Estrada, F., and Susstrunk, S. Year. "Frequency-tuned salient region detection," Computer vision and pattern recognition, 2009. cvpr 2009. ieee conference on, IEEE2009, pp. 1597-1604.

Aptoula, E., and Lefèvre, S. 2007. "A comparative study on multivariate mathematical morphology," *Pattern Recognition* (40:11), pp 2914-2929.

Avidan, S., and Shamir, A. Year. "Seam carving for content-aware image resizing," ACM Transactions on graphics (TOG), ACM2007, p. 10.

Ayouni, S. 2012. "Etude et extraction de regles graduelles floues: définition d'algorithmes efficaces," *These de doctorat, Université Montpellier* (2).

Bardos, M. 1997. "WH ZHU," *Revue de statistique appliquée* (45:4), pp 65-92.

Borba, G. B., Gamba, H. R., Marques, O., and Mayron, L. M. Year. "An unsupervised method for clustering images based on their salient regions of interest," Proceedings of the 14th ACM international conference on Multimedia, ACM2006, pp. 145-148.

Borenstein, G. 2012. *Making things see: 3D vision with kinect, processing, Arduino, and MakerBot*, (" O'Reilly Media, Inc.".

Borji, A., Ahmadabadi, M. N., Araabi, B. N., and Hamidi, M. 2010. "Online learning of task-driven object-based visual attention control," *Image and Vision Computing* (28:7), pp 1130-1145.

Burrus, N. 2014. "Kinect Calibration."

Chen, D., Ou, T., Gong, L., Xu, C.-Y., Li, W., Ho, C.-H., and Qian, W. 2010. "Spatial interpolation of daily precipitation in China: 1951–2005," *Advances in Atmospheric Sciences* (27:6), pp 1221-1232.

Chen, L.-Q., Xie, X., Fan, X., Ma, W.-Y., Zhang, H.-J., and Zhou, H.-Q. 2003. "A visual attention model for adapting images on small displays," *Multimedia systems* (9:4), pp 353-364.

Cheng, M.-M., Mitra, N. J., Huang, X., Torr, P. H., and Hu, S.-M. 2015. "Global contrast based salient region detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence* (37:3), pp 569-582.

Cheng, Y., Fu, H., Wei, X., Xiao, J., and Cao, X. Year. "Depth enhanced saliency detection method," Proceedings of international conference on internet multimedia computing and service, ACM2014, p. 23.

Cherif, A. 2013. *Réseaux de neurones, SVM et approches locales pour la prévision de séries temporelles*, Tours.

Ciptadi, A., Hermans, T., and Rehg, J. M. Year. "An in depth view of saliency," Georgia Institute of Technology2013.

Cok, D. R. 1987. "Signal processing method and apparatus for producing interpolated chrominance values in a sampled color image signal," Google Patents.

Comaniciu, D., and Meer, P. 2002. "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on pattern analysis and machine intelligence* (24:5), pp 603-619.

Comaniciu, D., Ramesh, V., and Meer, P. Year. "Real-time tracking of non-rigid objects using mean shift," Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on, IEEE2000, pp. 142-149.

Dehaene, S. 2014. *Le Code de la conscience*, (Odile Jacob.

Desingh, K., Krishna, K. M., Rajan, D., and Jawahar, C. Year. "Depth really Matters: Improving Visual Salient Region Detection with Depth," BMVC2013.

Devaux, J.-C., Hadj-Abdelkader, H., and Colle, E. Year. "Toolbox d'étalonnage pour Kinect: Application à la fusion d'une Kinect et d'un télémètre laser," Reconnaissance de Formes et Intelligence Artificielle (RFIA) 20142014, pp. -.

Einramhof, P., Olufs, S., and Vincze, M. 2007. *Experimental evaluation of state of the art 3d-sensors for mobile robot navigation*, (na.

Fanelli, G., Gall, J., and Van Gool, L. Year. "Real time head pose estimation with random regression forests," Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, IEEE2011, pp. 617-624.

Fawcett, T. 2006. "An introduction to ROC analysis," *Pattern recognition letters* (27:8), pp 861-874.

Flaus, J.-M. 1994. *La régulation industrielle: régulateurs PID, prédictifs et flous*, (Hermes.

Foix, S., Alenya, G., and Torras, C. 2011. "Lock-in time-of-flight (ToF) cameras: A survey," *IEEE Sensors Journal* (11:9), pp 1917-1926.

Frad, M. H. 2016. *Étude et mise en œuvre d'un système d'interaction adaptatif pour les applications de réalité virtuelle*, Université Paris Saclay; Université d'Evry Val-d'Essonne; Ecole Nationale d'Ingénieurs de Monastir. Tunisia.

Fraihat, H., Madani, K., and Sabourin, C. Year. "Learning-based distance evaluation in robot vision: a comparison of ANFIS, MLP, SVR and bilinear interpolation models," Computational Intelligence (IJCCI), 2015 7th International Joint Conference on, IEEE2015a, pp. 168-173.

Fraihat, H., Sabourin, C., and Madani, K. Year. "Soft-computing based fast visual objects' distance evaluation for robots' vision," Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), 2015 IEEE 8th International Conference on, IEEE2015b, pp. 81-86.

Frintrop, S., and Kessel, M. Year. "Most salient region tracking," Robotics and Automation, 2009. ICRA'09. IEEE International Conference on, IEEE2009, pp. 1869-1874.

Gonzalez, R. C., and Woods, R. E. 2002. "Processing," Prentice-Hall.

Harel, J., Koch, C., and Perona, P. Year. "Graph-based visual saliency," Advances in neural information processing systems2007, pp. 545-552.

Hassan, D., Madani, K., and Sabourin, C. Year. "Dual 2-d images-based approach for objects'3-D characterization and localization for Machine-Awareness in indoor environment," Awareness Science and Technology (iCAST), 2015 IEEE 7th International Conference on, IEEE2015, pp. 201-206.

Henry, P., Krainin, M., Herbst, E., Ren, X., and Fox, D. Year. "RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments," In the 12th International Symposium on Experimental Robotics (ISER, Citeseer2010.

Hérault, J., and Jutten, C. 1994. "Réseaux neuronaux et traitement de signal: Hermes édition," *Traitement du signal*).

Hofmann, J., Jüngel, M., and Lötzsch, M. Year. "A vision based system for goal-directed obstacle avoidance used in the rc'03 obstacle avoidance challenge," Proc. of 8th Int. Workshop on RoboCup2004, pp. 418-425.

Hou, X., and Zhang, L. Year. "Saliency detection: A spectral residual approach," Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on, IEEE2007, pp. 1-8.

Itti, L., Koch, C., and Niebur, E. 1998. "A model of saliency-based visual attention for rapid scene analysis," *IEEE Transactions on pattern analysis and machine intelligence* (20:11), pp 1254-1259.

Jang, J.-S. 1993. "ANFIS: adaptive-network-based fuzzy inference system," *IEEE transactions on systems, man, and cybernetics* (23:3), pp 665-685.

Jang, J.-S., and Sun, C.-T. 1995. "Neuro-fuzzy modeling and control," *Proceedings of the IEEE* (83:3), pp 378-406.

Jiang, H., Wang, J., Yuan, Z., Liu, T., Zheng, N., and Li, S. Year. "Automatic salient object segmentation based on context and shape prior," BMVC2011, p. 9.

Karray, F. O., and De Silva, C. W. 2004. *Soft computing and intelligent systems design: theory, tools, and applications*, (Pearson Education.

Kheng, L. W. 2011. "Mean shift tracking," *Technical report, School of Computing, National University of Singapore*).

Khoshelham, K., and Elberink, S. O. 2012. "Accuracy and resolution of kinect depth data for indoor mapping applications," *Sensors* (12:2), pp 1437-1454.

Lang, C., Nguyen, T. V., Katti, H., Yadati, K., Kankanhalli, M., and Yan, S. 2012. "Depth matters: Influence of depth cues on visual saliency," in *Computer vision–ECCV 2012*, Springer, pp. 101-115.

Lejeune, A., Piérard, S., Van Droogenbroeck, M., and Verly, J. 2012. "Utilisation de la Kinect," *Linux Magazine France* (151), pp 16-29.

Li, X., Lu, H., Zhang, L., Ruan, X., and Yang, M.-H. Year. "Saliency detection via dense and sparse reconstruction," Proceedings of the IEEE International Conference on Computer Vision2013, pp. 2976-2983.

Li, Y., Gong, S., and Liddell, H. Year. "Support vector regression and classification based multi-view face detection and recognition," Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on, IEEE2000, pp. 300-305.

Liang, Z., Chi, Z., Fu, H., and Feng, D. 2012. "Salient object detection using content-sensitive hypergraph representation and partitioning," *Pattern Recognition* (45:11), pp 3886-3901.

Liu, T., Yuan, Z., Sun, J., Wang, J., Zheng, N., Tang, X., and Shum, H.-Y. 2011. "Learning to detect a salient object," *IEEE Transactions on Pattern analysis and machine intelligence* (33:2), pp 353-367.

Lu, G. Y., and Wong, D. W. 2008. "An adaptive inverse-distance weighting spatial interpolation technique," *Computers & Geosciences* (34:9), pp 1044-1055.

Madani, K., Fraihat, H., and Sabourin, C. 2015. "Machine-Learning-Based Visual Objects' Distances Evaluation: A Comparison of ANFIS, MLP, SVR and Bilinear Interpolation Models," in *Computational Intelligence*, Springer, pp. 462-479.

Madani, K., Hassan, D., and Sabourin, C. 2017. "A dual approach for machine-awareness in indoor environment combining pseudo-3D imaging and soft-computing techniques," *International Journal of Machine Learning and Cybernetics* (8:6), pp 1795-1814.

Maki, A., Nordlund, P., and Eklundh, J.-O. 2000. "Attentional scene segmentation: integrating depth and motion," *Computer Vision and Image Understanding* (78:3), pp 351-373.

Margolin, R., Tal, A., and Zelnik-Manor, L. Year. "What makes a patch distinct?," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition2013, pp. 1139-1146.

Maximili, D., Sabourin, C., and Madani, K. Year. "Hybrid salient object extraction approach with automatic estimation of visual attention scale," Signal-Image Technology and

Internet-Based Systems (SITIS), 2011 Seventh International Conference on, IEEE2011, pp. 438-445.

Mhiri, R. 2015. *Approches 2D/2D pour le SFM à partir d'un réseau de caméras asynchrones*, Rouen, INSA.

Montabone, S., and Soto, A. 2010. "Human detection using a mobile platform and novel features derived from a visual saliency mechanism," *Image and Vision Computing* (28:3), pp 391-402.

Moreno, R., Grana, M., Ramik, D., and Madani, K. 2012. "Image segmentation on spherical coordinate representation of RGB colour space," *IET Image Processing* (6:9), pp 1275-1283.

Nakoula, Y., Galichet, S., and Foulloy, L. 1997. "Identification of linguistic fuzzy models based on learning," *Fuzzy model identification*), pp 281-319.

Navalpakkam, V., and Itti, L. Year. "An integrated model of top-down and bottom-up attention for optimizing detection speed," Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, IEEE2006, pp. 2049-2056.

Niu, Y., Geng, Y., Li, X., and Liu, F. Year. "Leveraging stereopsis for saliency analysis," Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE2012, pp. 454-461.

Oggier, T., Büttgen, B., Lustenberger, F., Becker, G., Rüegg, B., and Hodac, A. 2005. "SwissRanger SR3000 and first experiences based on miniaturized 3D-TOF cameras," *Proc. of the First Range Imaging Research Day at ETH Zurich*).

Pagliari, D., and Pinto, L. 2015. "Calibration of kinect for xbox one and comparison between the two generations of microsoft sensors," *Sensors* (15:11), pp 27569-27589.

Parizeau, M. 2004. "Le perceptron multicouche et son algorithme de rétropropagation des erreurs," *département de génie électrique et de génie informatique, Université de laval*).

Peng, H., Li, B., Xiong, W., Hu, W., and Ji, R. Year. "Rgbd salient object detection: a benchmark and algorithms," European conference on computer vision, Springer2014, pp. 92-109.

Ramik, D. M. 2012. *Contribution to complex visual information processing and autonomous knowledge extraction: application to autonomous robotics*, Université Paris-Est.

Ramík, D. M., Sabourin, C., and Madani, K. Year. "On Human Inspired Semantic SLAM's Feasibility," ANNIIP2010, pp. 99-108.

Ramík, D. M., Sabourin, C., Moreno, R., and Madani, K. 2014. "A machine learning based intelligent vision system for autonomous object detection and recognition," *Applied intelligence* (40:2), pp 358-375.

Richard, J.-F. 2004. "Paysages," *Revista de ciencia y tecnología de la información geográfica* (2).

Riche, N., Duvinage, M., Mancas, M., Gosselin, B., and Dutoit, T. Year. "Saliency and human fixations: State-of-the-art and study of comparison metrics," Proceedings of the IEEE international conference on computer vision2013, pp. 1153-1160.

Ronchetti, M., and Avancini, M. 2011. "Using kinect to emulate an interactive whiteboard," *MS in Computer Science, University of Trento*).

Rutishauser, U., Walther, D., Koch, C., and Perona, P. Year. "Is bottom-up attention useful for object recognition?," Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on, IEEE2004, pp. II-II.

Scharstein, D., and Szeliski, R. 2002. "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International journal of computer vision* (47:1-3), pp 7-42.

Scholl, B. J. 2001. "Objects and attention: The state of the art," *Cognition* (80:1), pp 1-46.

Shen, X., and Wu, Y. Year. "A unified approach to salient object detection via low rank matrix recovery," Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE2012, pp. 853-860.

Smola, A. J., and Schölkopf, B. 2004. "A tutorial on support vector regression," *Statistics and computing* (14:3), pp 199-222.

SoftBank 2017. "Pepper - Documentation."

Sugeno, M., and Kang, G. 1988. "Structure identification of fuzzy model," *Fuzzy sets and systems* (28:1), pp 15-33.

Tang, Y., Tong, R., Tang, M., and Zhang, Y. 2016. "Depth incorporating with color improves salient object detection," *The Visual Computer* (32:1), pp 111-121.

Touzet, C. 1992. *les réseaux de neurones artificiels, introduction au connexionnisme*, (EC2.

Üstün, B., Melssen, W. J., and Buydens, L. M. 2006. "Facilitating the application of support vector regression by using a universal Pearson VII function based kernel," *Chemometrics and Intelligent Laboratory Systems* (81:1), pp 29-40.

Vapnik, V. 1995. "The nature of statistical learning theory Springer New York Google Scholar,").

Velmans, M. 2003. "Is the world in the brain, or the brain in the world?," *Behavioral and Brain Sciences* (26:4), pp 427-429.

Viola, P., and Jones, M. Year. "Rapid object detection using a boosted cascade of simple features," Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on, IEEE2001, pp. I-I.

Wang, P., Wang, J., Zeng, G., Feng, J., Zha, H., and Li, S. Year. "Salient object detection for searched web images via global saliency," Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE2012, pp. 3194-3201.

Weingarten, J. W., Gruener, G., and Siegwart, R. Year. "A state-of-the-art 3D sensor for robot navigation," Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on, IEEE2004, pp. 2155-2160.

Yan, Q., Xu, L., Shi, J., and Jia, J. Year. "Hierarchical saliency detection," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition2013, pp. 1155-1162.

Yang, C., Zhang, L., Lu, H., Ruan, X., and Yang, M.-H. Year. "Saliency detection via graph-based manifold ranking," Proceedings of the IEEE conference on computer vision and pattern recognition2013, pp. 3166-3173.

Yilmaz, A., Javed, O., and Shah, M. 2006. "Object tracking: A survey," *Acm computing surveys (CSUR)* (38:4), p 13.

Zhang, Z. 2000. "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence* (22:11), pp 1330-1334.

Zhang, Z. 2012. "Microsoft kinect sensor and its effect," *IEEE multimedia* (19:2), pp 4-10.

Résumé

Le travail de recherche effectué dans le cadre de cette thèse concerne le développement d'un système de perception de la saillance en environnement 3D en tirant l'avantage d'une représentation 3D. Notre contribution et concept issue de celle-ci part de l'hypothèse que la profondeur de l'objet par rapport au robot est un facteur important dans la détection de la saillance. Sur ce principe, un système de vision saillante de l'environnement 3D a été proposé, conçu et validée sur une plateforme comprenant un robot équipé d'un capteur 3D. La mise en œuvre du concept précité et sa conception ont été d'abord validés sur le système de vision 3D KINECT. Puis dans une deuxième étape, le concept et les algorithmes mis aux points ont été étendus à la plateforme précitée. Les principales contributions de la présente thèse peuvent être résumées de la manière suivante : A) Un état de l'art sur les différents capteurs d'acquisition de l'information de la profondeur ainsi que les différentes méthodes de la détection de la saillance 2D et 3D. B) Etude d'un système basé sur la saillance visuelle 3D réalisée grâce au développement d'un algorithme robuste permettant la détection d'objets saillants dans l'environnement 3D. C) réalisation d'un système d'estimation de la profondeur en centimètres pour le robot Pepper. D) La mise en œuvre des concepts et des méthodes proposés sur la plateforme précitée. Les études et les validations expérimentales réalisées ont notamment confirmé que les approches proposées permettent d'accroitre l'autonomie des robots dans un environnement 3D réel.

Mots clés: Perception visuelle, Saillance 3D, Calcul léger, Algorithme d'apprentissage, Logique floue, Kinect-Robot Pepper

Abstract

The research work, carried out within the framework of this thesis, concerns the development of a system of perception and saliency detection in 3D environment taking advantage of a 3D representation. Our contribution and the issued concept derive from the hypothesis that the depth of the object with respect to the robot is an important factor in the detection of the saliency. On this basis, a salient vision system of the 3D environment has been proposed, designed and validated on a platform including a robot equipped with a 3D sensor. The implementation of the aforementioned concept and its design were first validated on the 3D KINECT vision system. Then, in a second step, the concept and the algorithms have been extended to the aforementioned robotic platform. The main contributions of the present thesis can be summarized as follow: A) A state of the art on the various sensors for acquiring depth information as well as different methods of detecting 2D salience and 3D. B) Study of 3D visual saliency system based on benefiting from the development of a robust algorithm allowing the detection of salient objects. C) Implementation of a depth estimation system in centimeters for the Pepper robot. D) Implementation of the concepts and methods proposed on the aforementioned platform. The carried out studies and the experimental validations confirmed that the proposed approaches allow to increase the autonomy of the robots in a real 3D environment.

Keywords: Visual perception, Saliency 3D, Soft Computing, Machine learning, Fuzzy logic, Kinect-Pepper robot.