Une approche multi-agent pour la segmentation d'images de profondeur à base d'objets polyédriques. Une nouvelle approche de segmentation d'images

Smaine Mazouzi, Zahia Guessoum, Fabien Michel

Une approche multi-agents pour la segmentation d'images de profondeur à base d'objets polyédriques

Une nouvelle approche de segmentation d'images

Smaine Mazouzi — Zahia Guessoum — Fabien Michel

LERI-CReSTIC, IUT de Reims, B.P. 1035, 51687, Reims, France
mazouzi@leri.univ-reims.fr, zahia.guessoum@lip6.fr, fmichel@leri.univ-reims.fr

RÉSUMÉ. *Dans cet article, nous présentons et nous évaluons une approche multi-agents pour la segmentation d'images de profondeur, contenant des objets polyédriques. L'approche consiste en l'utilisation d'une population d'agents autonomes pour la segmentation d'une image de profondeur en ses différentes régions planes. Les agents s'adaptent aux régions sur lesquelles ils se déplacent, puis effectuent des actions coopératives et compétitives produisant une segmentation collective de l'image. Les résultats expérimentaux, obtenus avec des images réelles, montrent le potentiel de l'approche proposée pour l'analyse des images de profondeurs, et ce vis-à-vis de la fiabilité de segmentation et de l'efficacité de calcul.*

ABSTRACT. *In this paper, we present and evaluate a multi-agent approach for polyhedral object-contented range image segmentation. The approach consists in using a population of autonomous agents to segment a range image in its planar regions. The agents adapt to the regions on which they move, then perform cooperative and competitive actions allowing a collective segmentation of the image. The experimental results obtained with real images show the potential of the proposed approach for range image analysis, regarding both segmentation reliability and computation efficiency.*

MOTS-CLÉS : *Segmentation d'images, Systèmes multi-agents, Agents adaptatifs, Image de profondeur, Traitement parallèle d'images.*

KEYWORDS: *Image segmentation, Multi-agent systems, Adaptive agents, Range image, Parallel image processing*

1. Introduction

Segmenter une image est souvent nécessaire pour fournir une description compacte et opportune de son contenu. L'opération consiste en l'affectation des pixels à des sous-ensembles homogènes et disjoints, formant une partition de l'image. Les pixels qui appartiennent à une même région partagent une propriété commune, dite critère d'homogénéité de région. A l'instar des méthodes générales de segmentation d'images, les méthodes de segmentation en imagerie de profondeur se divisent en deux catégories distinctes : les méthodes de segmentation en contours, et les méthodes de segmentation en régions. Pour la première catégorie, les pixels qui correspondent aux discontinuités de profondeur (contour en marche) ou d'orientation de surface (contour en toit) sont sélectionnés et chaînés dans le but de délimiter les régions de l'image (Inokuchi *et al.,* 1982, Fan *et al.,* 1987, Jiang *et al.,* 1999). Les méthodes orientées contour sont bien connues pour leur coût de calcul faible ; cependant elles sont très sensibles au bruit et nécessitent des post-traitements complexes, pour le chaînage et la fermeture de contours (Sappa, 2006). Les méthodes orientées région utilisent les propriétés de surface pour grouper les pixels ayant les mêmes propriétés, dans des parties connexes et disjointes (Kang *et al.,* 1993, Li *et al.,* 2003, Ding *et al.,* 2005, Bab Hadiashar *et al.,* 2006). Comparées aux méthodes orientées contour, les méthodes orientées région sont plus stables et moins sensibles au bruit. Cependant, leur efficacité dépend fortement de la sélection des graines initiales de régions. De plus, cette approche ne facilite pas la parallélisation et la distribution des traitements. Ceci rend le coût de calcul très élevé et ne permet pas à ces méthodes d'être utilisées pour des applications temps réel.

Par ailleurs, la plupart des méthodes développées modélisent les propriétés locales de surfaces en se basant sur le calcul des dérivées de différents ordres de la fonction image. Ceci conduit à une détection très sensible au bruit (Horaud *et al.,* 1995). Ainsi, il est nécessaire d'effectuer des tâches de prétraitements, qui consistent principalement en un lissage de l'image ou un filtrage du bruit, qui est souvent insuffisant pour garantir une segmentation fiable. En effet, dans le cas d'images hautement bruitées, telles que les images de profondeur (Hoover *et al.,* 1996), un fort lissage de bruit peut conduire à l'effacement des contours, notamment les contours en toit et les contours lisses (correspondent à la discontinuité de la courbure de surface). Cependant, si le bruit est sous-lissé, les distorsions qui restent dans l'image engendrent des résultats inexacts ou erronés. Ce problème, qui demeure toujours ouvert en traitement d'images (Li, 2001, Mehrtash, 2006), est dû à la restriction des masques de lissage au voisinage immédiat du pixel traité. Ceci représente une perception locale qui ne permet pas une catégorisation fiable des pixels traités. En imagerie de profondeur, plusieurs méthodes récentes échouent parce qu'elles n'adressent et ne résolvent pas correctement ce problème (Jiang *et al.,* 2000, Bab Hadiashar *et al.,* 2006).

Pour remédier à cette difficulté, certains auteurs tels que Ballet et al. (**?**), Liu et al citeliu99, Richard et al. (Richard *et al.,* 2004) et récemment Mobahi et al. (**?**) et Jones et Saeed (**?**), ont proposé des systèmes multi-agents pour la segmentation d'images (**?**, Liu *et al.,* 1999, Richard *et al.,* 2004). Dans les systèmes multi-agents, un agent

est communément considéré comme une entité software qui, contrairement à un objet, exhibe les caractéristiques suivantes(Wooldridge *et al.,* 1995, Wooldridge, 2002) : 1) L'agent est autonome, et possède son propre "thraed" d'execution. Il n'est par conséquent soumis à un contrôle d'exécution centralisé. 2) Il est situé dans un environnement particulier qui peut sentir et sur lequel il peut agir. 3) Un agent est proactif, et n'agit pas uniquement en réponse aux événements en provenance de son environnement. Il exhibe un comportement dirigé par par un but, où l'agent est incité à prendre de l'initiative (Zambonelli *et al.,* 2003). 4) Un agent est sociable car il interagit avec d'autres agents en utilisant un langage de communication d'agents "Agent Communication Language : ACL" (Genesereth *et al.,* 1994).

Le comportement global dans un système multi-agents est dérivé des interactions au sein des agents qui agissent sur l'environnement dans lequel ils sont situés, et qui notamment interagissent entre eux (Wooldridge *et al.,* 1995). Dans ces système un agent ne possède que des capacités limités de perception et d'action. Il n'est pas conçu donc pour résoudre un problème dans sa totalité. Les agents coopèrent donc pour fournir une solution collective. Contrairement aux systèmes conventionnels, les solutions dans les systèmes multi-agents, émergent des actions collectives effectuées au sein de la population des agents.

Malgré la multitude des travaux ayant proposé des systèmes multi-agents pour la segmentation d'images, la plupart des systèmes proposés sont spécifiques aux contenus des images traitées, et procèdent souvent selon une approche supervisée. Ils sont dédiés en majorité aux images à niveau de gris, où le nombre de régions est préalablement connu, et chacune des région est représentée par une unique grandeur scalaire représentant le niveau de gris, considéré constant à l'intérieur d'une même région (Liu *et al.,* 1999, Richard *et al.,* 2004).

Dans cet article, une approche non supervisée de segmentation d'images de profondeur, basée sur le paradigme agent est présentée et discutée. Elle procède en considérant un nombre quelconques de régions représentant les facettes des objets polyédriques qui existent dans la scene. Chacune des région est représentée à son tour par une équation du plan dont les paramètres sont à calculer lors de l'execution. Le principe de la méthode consiste en l'utilisation d'agents réactifs qui se déplacent sur l'image et agissent sur les pixels situés sur les pourtours des régions. Les agents s'adaptent aux régions planes de l'image, puis lissent les pixels appartenant à ces régions. Par conséquent, ils se trouvent en compétition sur les bordures entre les régions. L'alignement alternatif des pixels des bordures qui résulte des actions des groupes compétitifs d'agents préserve ses bordures contre l'effacement. Les régions de bruit qui sont caractérisées par de faibles tailles ou par des profondeurs aberrantes ou aléatoires, ne permettent pas aux agents de s'adapter. Ces régions se contractent continuellement, et ce par l'alignement de leurs pixels aux régions planes qui les entourent.

L'objectif du travail présenté dans cet article est de surmonter la difficulté due à la restriction de la perception au seul voisinage immédiat du pixel traité. En effet, selon notre approche, un pixel est traité non uniquement en fonction de son voisinage, mais aussi en fonction des états des agents qui visitent ce pixel. La mémoire d'un

agent représente une perception plus large qui, lorsqu'elle est combinée avec l'information locale de l'image, permet une décision plus fiable, en prenant en compte des données globales. Dans le but d'optimiser les déplacements des agents, un champ de potentiel inspiré du champ électrostatique est utilisé. Il permet aux agents de s'auto-organiser et de rationaliser leurs mouvements en se regroupant autour des régions d'intérêt (contours et bruit) et à concentrer leurs actions autour de ces régions.

Nous montrons dans ce travail que, malgré la simplicité du modèle utilisé pour la représentation des données image, les résultats obtenus sont meilleurs que ceux obtenus avec des approches conventionnelles. Nous croyons que l'interaction entre les agents fournit une solution alternative pour la segmentation d'images, par rapport aux méthodes de segmentation basées sur des modèles complexes et coûteux en temps de calcul (Li, 2001).

L'utilisation d'agents réactifs et faiblement couplés permet de plus à notre approche d'être parallélisable et ainsi d'être utilisée dans des application temps réel. Des expérimentations intensives ont été effectuées en utilisant des images réelles provenant de la base de données standard ABW (Hoover *et al.,* 1996). Les résultats expérimentaux obtenus montrent le potentiel de l'approche proposée pour une segmentation efficace et fiable des images de profondeur.

La suite du papier est organisée comme suit : dans la section 2, nous passons en revue quelques travaux ayant proposé des systèmes multi-agents pour la segmentation d'images. La section 3 présente la méthode de modélisation des propriétés des surfaces. La section 4 est réservée à l'approche proposée. Elle décrit le comportement des agents, et montre le mécanisme collectif sous-jacent permettant la détection de contours et l'élimination de bruit. Les résultats expérimentaux sont présentés à la section 5, dans laquelle nous discutons la sélection des paramètres, et nous présentons et commentons les résultats obtenus. Une analyse de complexité en temps de calcul est également présentée. Finalement, une conclusion résume notre contribution.

2. Etat de l'art

Plusieurs systèmes multi-agents ont été proposés dans le domaine du traitement d'images et de la reconnaissance des formes. Ces systèmes introduisent des solutions intéressantes pour remédier à différents problèmes tels que la manipulation de connaissances hétérogènes, le contrôle de haut niveau sur les traitements de bas niveau, et la parallélisation et la distribution des traitements. Dans cet état de l'art, nous ne considérons que les travaux qui ont adressé une solution basée agent pour la segmentation d'images.

Liu et al. ont présenté un système basé sur des agents réactifs pour la segmentation d'IRM du cerveau (Liu *et al.,* 1999). Quatre types d'agents sont utilisés pour étiqueter les pixels de l'image en fonction de leur degré d'appartenance aux différentes régions. Les agents qui réussissent à trouver des pixels d'une région homogène spécifique, créent des agents clones à l'intérieur de leurs régions voisines. Ces derniers sont ini-

tialisés de telle sorte qu'ils deviennent capables de reconnaître les pixels de la même région, pour laquelle ils ont été créés. Les auteurs affirment que leur système est plus robuste et plus efficace que les algorithmes classiques d'éclatement et de fusion de régions. Cependant, dans ce système les agents n'interagissent pas directement entre eux, et n'agissent pas sur l'image. Leurs actions dépendent uniquement de leur perception locale de l'image. Néanmoins, ils sont créés et placés de telle sorte qu'ils soient susceptibles de trouver plus de pixels de la région pour laquelle ils ont été crées.

Pour le même type d'images, Richard et al. ont proposé une architecture hiérarchique d'agents situés et coopératifs pour la segmentation d'images (Richard *et al.,* 2004). Trois types d'agents ont été utilisés : agent de contrôle global, agent de contrôle local, et agent dédié au tissu. Le rôle de l'agent de contrôle global est de partitionner le volume de données en territoires adjacents et d'affecter à chaque territoire un agent de contrôle local. Le rôle de ce dernier est de créer les agents dédiés au tissu, dont le rôle est d'effectuer un accroissement de région à l'intérieur du volume local. Les paramètres de distribution des données sont mis à jour par coopération entre les agents voisins. Par l'utilisation de différents types d'agents, le système proposé a pris en compte les traitements d'images de bas niveau, ainsi que le contrôle sur les tâches de segmentation. Cependant, il résulte des patterns complexes d'interaction dont la gestion est épineuse et coûteuse.

Les deux systèmes précédents ont été bien optimisés pour la segmentation des IRM du cerveau. Ils peuvent produire de bons résultats, car les caractéristiques de régions dans ce type d'images sont régulières du fait qu'elles correspondent aux différentes structures anatomiques du cerveau. En plus, la plupart des contours dans de telles images sont de type marche. Ces contours sont faciles à détecter, comparés aux contours en toit et contours lisses.

En utilisant le langage multi-agents oRis (Harrouet *et al.,* 2002), Rodin et al. ont proposé un système multi-agents formé d'agents réactifs pour la détection de contours dans des images biologiques (Rodin *et al.,* 2004). En fonction de certains a priori sur le contenu des images, le système vise à produire une détection de contours meilleure que celle obtenue par les détecteurs traditionnels. Deux types d'agents dits respectivement agents de noircissement et agents de luminance suivent respectivement les régions sombres et les régions claires. Leurs actions visent à renforcer les régions par l'accroissement de leur contraste, permettant ainsi une détection fiable de ces régions. Selon cette approche, les agents parcourant les différents contours sont totalement indépendants sans aucune d'interaction entre eux. Le système parait comme un algorithme parallèle de segmentation qui était bien optimisé pour la détection des contours de type toit dans certains types d'images. Cependant, il peut échouer à détecter des contours plus simples tels que les contours de type marche. Il est a noter qu'il n'as été considéré que deux types de régions (claires et sombres) avec une topologie bien déterminée (régions concentriques), ce qui peut limiter l'utilisation du système proposé.

En se basant sur l'architecture cognitive Soar (Newell, 1990), Bovenkamp et al. (Bovenkamp *et al.,* 2004) ont développé un système multi-agents pour la segmentation des images intravasculaires ultrasonores (IVUS). Leur objectif est d'établir un

sous-système basé connaissances, pour le contrôle des algorithmes de traitement de bas niveau. Dans leur système un seul agent est affecté à chaque objet prévu dans l'image. Les agents coopèrent et adaptent dynamiquement les algorithmes de segmentation, en se basant sur la connaissance contextuelle, l'information locale, et les croyances personnelles. Dans ce travail, le problème du contrôle des algorithmes de traitement d'image était bien posé et résolu. Cependant aucun agent ni comportement n'ont été proposés pour remédier au problème d'ambiguïté des données dans l'image. Le couplage fort entre les traitements de haut niveau (spécifiques au domaine) et les traitements de bas niveau (spécifiques à l'image), adopté dans ce système, ne permet pas la réutilisation des méthodes proposées pour d'autres applications ou d'autres types d'images.

Nous pouvons constater que la plupart des systèmes basés agent, proposés pour la segmentation d'images, suivent une approche supervisée, et sont spécifiques au contenu des images traitées. Ces systèmes visent à segmenter les images en régions connues correspondant aux différents objets préalablement prévus. L'approche que nous proposons dans cet article, se veut générale et non supervisée. Elle vise à segmenter une image en ses différentes régions en se basant sur certaines propriétés géométriques des surfaces. Pour ce faire, nous avons doté les agents par un comportement à la fois adaptatif et compétitif, leur permettant de surmonter la contrainte de la restriction de perception au voisinage local du pixel, inhérente à la segmentation d'images.

3. Modélisation et calcul des propriétés locales des surfaces

Une image de profondeur est une image 3-D où à chaque pixel (x, y) correspond la distance $Z(x, y)$ entre le plan du capteur télémétrique et le point correspondant de la scène. Dans une telle image, les régions représentent les parties visibles des surfaces des différents objets figurant dans la scène. Dans le but d'atténuer le bruit Gaussien ainsi que le bruit impulsif, un filtre Gaussien et un filtre médian sont appliqués aux données brutes de profondeur. Soit $Z^*(x, y) \in R^4$, les paramètres de l'équation du plan tangent à la surface en (x, y). L'équation du plan tangent en (x, y) est obtenue par la méthode de régression multiple en utilisant les pixels appartenant au voisinage $\chi(x, y)$ du pixel en question. Le voisinage $\chi(x, y)$ est constitué des pixels (x', y') situés dans une fenêtre de taille 3×3 centrée en (x, y), et dont les profondeurs $Z(x', y')$ sont, à un seuil donné de profondeur (Tr_D), proches. L'équation d'un plan dans un référentiel cartésien 3-D peut être exprimée comme suit :

$$ax + by + cz = d \qquad [1]$$

où $(a, b, c)^T$ est le vecteur unitaire normal au plan ($a^2 + b^2 + c^2 = 1; c < 0$) et d est la distance orthogonale du plan à l'origine du référentiel. En premier, les coefficients α, β et γ de la surface $z = \alpha x + \beta y + \gamma$ au point (x_0, y_0) sont obtenus par la minimisation de la fonction Φ, donnée par :

$$\Phi(\alpha, \beta, \gamma) = \sum_{(x',y') \in \chi(x_0,y_0)} [\alpha x' + \beta y' + \gamma - Z(x', y')]^2 \qquad [2]$$

avec $\chi(x_0, y_0) = \{(x_0 + i, y_0 + j); (i, j) \in \{-1, 0, +1\}$ et $|Z(x_0 + i, y_0 + j) - Z(x_0, y_0)| < Tr_D\}$

Les paramètres a, b, c et d sont donc calculés comme suit :

$$(a, b, c, d)^T = \frac{1}{\sqrt{\alpha^2 + \beta^2 + 1}}(\alpha, \beta, -1, \gamma)^T \quad [3]$$

Les traitements effectués sur l'image se basent sur la comparaison de plans. En effet, deux plans d'équations respectives : $ax + by + cz = d$ et $a'x + b'y + c'z = d'$ sont considérés égaux s'ils ont à des seuils près, la même orientation et la même distance à l'origine du référentiel. Soit θ l'angle entre les deux vecteurs normaux, et D la distance entre les deux plans : $\sin(\theta) = \|(a, b, c)^T \otimes (a', b', c')^T\|$ et $D = |d - d'|$. Les deux plans sont donc considérés égaux si $\sin(\theta) \leq Tr_\theta$ et $D \leq Tr_D$, où Tr_θ et Tr_D sont les seuils respectivement d'angle et de profondeur, dont les valeurs optimales respectives sont expérimentalement sélectionnées. (voir sous-section 6.2). Le test d'égalité de deux plans est utilisé d'abord pour tester si le pixel à la position (x, y) appartient à une région plane, étant donnée son équation de plan. Il est également utilisé pour tester si le pixel en (x, y) est un pixel d'intérêt (de contour ou de bruit) ou non. Dans ce cas, le pixel en question est considéré comme un pixel d'intérêt si au moins un de ses voisins a une équation de plan différente, en se référant aux seuils précédents.

4. Segmentation collective d'images par agents réactifs

4.1. *Comportement des agents*

L'image est considérée comme un environnement dans lequel des agents sont initialisés à des positions arbitraires. Les agents utilisés ont une granularité faible, et leur nombre doit être suffisamment grand. Un agent cherche une région plane autour de sa position courante et s'adapte à cette région en mémorisant son équation de plan. Ensuite, il effectue des actions qui dépendent à la fois de son état et de l'état du pixel sur lequel il se situe. A chaque instant t, un agent est caractérisé par sa position (x_t, y_t) dans l'image, et par sa capacité C_t d'agir sur les pixels rencontrées. Au début du processus, tous les agents sont incapables d'altérer les pixels de l'image. Un agent devient capable de modifier un pixels (C_t=true) lorsqu'il détecte une région plane autour de sa position courante. Par ailleurs, si un agent vient d'altérer un pixel, il perd sa capacité d'altération (C_t=false) et recommence à chercher une nouvelle région plane. Chaque agent ayant modifié un pixel, enregistre dans une structure appropriée I, dite matrice d'états des pixels, à la position (x_t, y_t) le dernier état du pixel visité : $I(x_t, y_t) \in$ {lissé, aligné, non changé}. Au lancement du système, la matrice des états des pixels I est initialisée par "non changé".

Un agent s'adapte à la région de l'image dans laquelle il se déplace en mémorisant les caractéristiques de cette région, et en adoptant le comportement adéquat en fonction des données de l'image. Nous montrons dans la suite de la section, que le comportement simple et réactif des agents permet de faire émerger les lignes de contour et

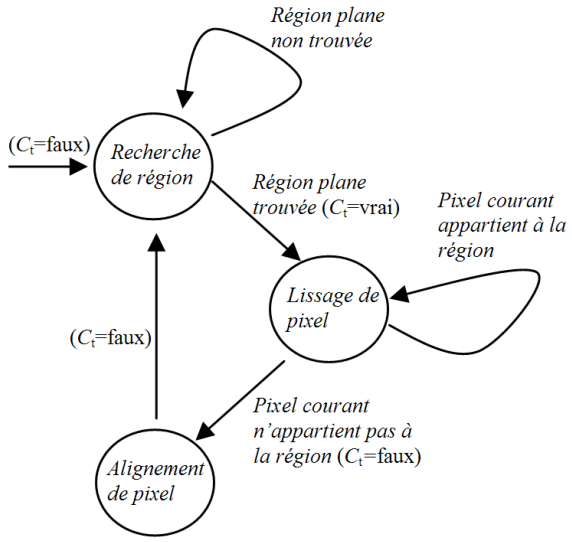


Figure 1. *Le comportement d'un agent en fonction de son état et de sa position*

d'éliminer efficacement le bruit dans l'image. La figure 1 schématise le comportement d'un agent en fonction de son état courant et du pixel sur lequel il est situé.

Le pseudo code exécuté par chaque agent est le suivant :

```
Initialization
Repeat
    Repeat
       Move
       Check Current Position
    Until Planar region found
    Memorize Current Region
    Repeat
       Smooth Current Pixel
       Move
       Check Current Pixel
    Until Current pixel does not belong to the memorized region
    Align current pixel to the Current Planar Region
    Update Potential field
Until Process Termination
```

4.1.1. *Recherche d'une région plane*

Après sa création, un agent se déplace aléatoirement dans l'image et cherche une région plane autour de sa position courante. L'agent utilise les L derniers pixels parcourus pour tester s'il est à l'intérieur d'une région plane ou non. L est appelée la longueur d'adaptation, et représente le degré de confiance que l'agent soit à l'intérieur d'une région plane. L'agent considère qu'il est à l'intérieur d'une région plane si les derniers L pixels parcourus appartiennent au même plan. L'agent mémorise la région trouvée et la considère désormais comme sa région plane courante. Après cela, il devient capable d'altérer le premier pixel rencontré qui n'appartient pas à cette région (C_t=true).

4.1.2. *Lissage des pixels à l'intérieur d'une régions*

Lors de son déplacement dans une région plane, l'agent lisse l'image au niveau du pixel sur lequel il se situe, en mettant à jour les équations respectives du plan de la région mémorisée, et du plan à la position du pixel. Ceci est fait en remplaçant les deux équations par leur moyenne pondérée. Soit (a, b, c, d) et (a', b', c', d') les paramètres respectivement du plan de la région mémorisée, et du plan au pixel courant, les paramètres résultats (a'', b'', c'', d'') du plan moyen pondéré sont obtenus comme suit :

$$(a'', b'', c'', d'') = \frac{1}{1+l}(a' + la, b' + lb, c' + lc, d' + ld) \quad [4]$$

où l est la longueur, en pixels, du chemin parcouru par l'agent dans la région plane.

Pour chaque pixel lissé, l'agent enregistre la valeur "lissé" dans la position correspondante dans la matrice des états des pixels I.

4.1.3. *Alignement des pixels d'intérêt*

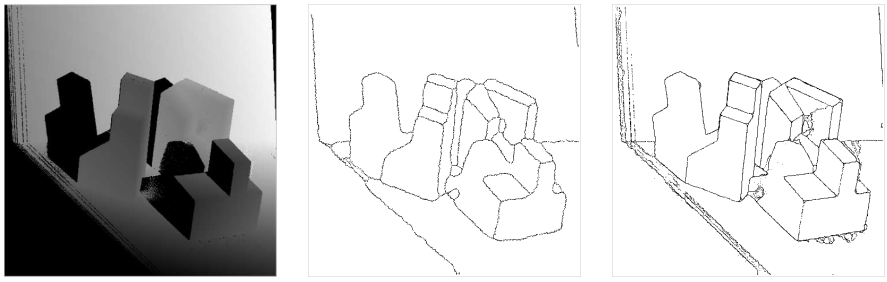
Les pixels d'intérêt sont les pixels de contour ou les pixels à l'intérieur des régions de bruit. Un pixel d'intérêt est reconnu en testant l'égalité du plan au niveau de ce pixel, et celui de la région mémorisée au sein de l'agent (voir section 3). Quand un agent rencontre un pixel d'intérêt (qui n'appartient pas à sa région plane courante), il l'altère pour l'aligner partiellement à la région sur laquelle il se déplace. Les paramètres (a'', b'', c'', d'') de la nouvelle équation de plan à la position du pixel sont obtenus par la combinaison linéaire des anciens paramètres (a, b, c, d) et des paramètres de l'équation de plan de la région mémorisée (a', b', c', d') :

$$(a'', b'', c'', d'') = \frac{1}{1+\xi}(a + \xi a', b + \xi b', c + \xi c', d + \xi d') \quad [5]$$

où ξ est la force d'altération. Après, l'agent enregistre la valeur "aligné" à la position correspondante dans la matrice des états des pixels I.

Après cela, l'agent perd sa capacité d'altération (C_t=false) et recommence à chercher une nouvelle région plane. Suite à l'altération d'un pixel, l'agent peut passer dans

une autre région ou rester dans la région précédente. Si le pixel altéré est un pixel de contour, il est plus probable que l'agent passe dans une autre région plane. Par contre, si le pixel altéré appartient à la bordure d'une région de bruit, l'agent traverse cette région et plus probablement retourne à la région plane précédente, sauf si la région de bruit est située entre deux régions planes différentes. La force d'altération ξ est un paramètre critique qui affecte la qualité des résultats et le temps de calcul. En effet, le choix de valeurs élevées pour ξ conduit à une détection rapide de régions. Cependant, les bordures des régions qui en résultent sont déformées (figure 2b). Le choix de valeurs basses de ξ conduit à une détection lente. Néanmoins, les bordures des régions, dans ce cas, sont bien détectées et correctement localisées(figure 2 c). Pour accélérer le processus de segmentation sans déformation des contours, un agent choisit la force d'altération parmi ξ_{min} et ξ_{max} en fonction de l'information enregistrée dans la matrice des états de pixels I. En effet, l'agent suppose que la région courante est adjacente à une région de bruit, et alors utilise ξ_{max} comme force d'altération, s'il existe au moins 3 pixels (x', y') étiquetés "non changé" ($I(x', y')$="non changé") autour de l'agent. Sinon, l'agent suppose que la région plane courante est adjacente à une autre région plane, où des agents ont étiqueté les pixels comme "lissés" ou "alignés". Dans ce cas, l'agent utilise la force d'altération ξ_{min}. Les constantes ξ_{min} et ξ_{max} sont deux paramètres du système dont les valeurs sont expérimentalement sélectionnées (voir sous section 6.2).



(a) (b) (c)

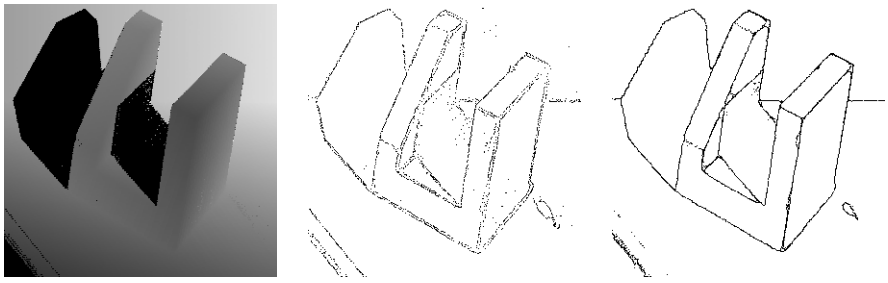
Figure 2. *Impact de la force d'altération sur les résultats de segmentation : (a) Image de profondeur (abw.test.8) ; (b) Résultats de segmentation avec* $\xi_{min} = \xi_{max} = 4$ *à* t*=2500 ; (c) Résultats de segmentation avec* $\xi_{min} = 0.3$ *et* $\xi_{max} = 5$ *à* t*=13000*

4.2. *Détection de contours*

Lors de son déplacement sur l'image, un agent lisse les pixels qui appartiennent approximativement à la région plane sur laquelle il se situe actuellement. Un agent considère les pixels n'appartenant pas à cette région comme des pixels de bruit. Ces derniers sont donc automatiquement alignés à la région plane qui les entoure. Cependant, les pixels des frontières entre les régions planes sont de vrais pixels de contours et donc ne devraient pas normalement être alignés. Néanmoins, sur le segment de

contour entre deux régions adjacentes, deux groupes concurrents d'agents se forment. Chaque groupe est formé d'agents qui transitent d'une région à une autre. Les agents de chaque groupe alignent les pixels de la frontière à leur région plane courante. Ainsi, ces pixels se retrouvent continuellement permutés entre les deux régions adjacentes, et demeurent ainsi distingués dans l'image. Cette compétition entre les agents permet également l'amincissement des contours de type toit et des contours lissés qui existent entre les régions adjacentes d'un même objet. Au début, un tel contour est gras, formé d'une bande de pixels qui n'appartient à aucune des deux régions adjacentes (figure 3b). Durant le processus, les pixels sur chaque coté de la bande sont continuellement alignés à la région dont ils forment la bordure. Les agents qui passent à l'intérieur de cette bande ne peuvent pas s'adapter et quittent la bande sans altérer ses bordures. La bande continue donc à se rétrécir jusqu'au point où il ne reste que les pixels qui sont en son centre. Ces pixels représentent la ligne de compétition entre les deux groupes d'agents, qui ne sera alignée à aucune des deux régions (figure 3c).

Au bout d'un certain temps, il ne reste distingué dans l'image que les pixels de bordure entre les régions adjacentes qui continuent à se permuter entre les régions. Ce schéma d'action compétitif entre les agents permet de faire émerger les lignes de contours. Ce résultat qui n'est codé dans aucun agent, émerge de l'action collective de tous les agents sur les pixels de l'image.



(a) (b) (c)

Figure 3. *Amincissement de contours (image abw.test.22) : (a) Image de profondeur (b) Pixels de contour à t=800 ; (c) Pixels de contour à t= 8000*

4.3. *Elimination du bruit*

Contrairement aux vraies régions de l'image, qui restent préservées contre l'effacement, les régions de bruit se contractent continuellement, et finissent par disparaître. Les bordures de ces dernières sont continuellement alignées aux vraies régions planes environnantes. Chaque agent ayant aligné un pixel appartenant à la bordure d'une région de bruit, et s'est déplacé à l'intérieur de cette région, ne sera pas capable de s'adapter (ne se trouve pas au milieu d'une région plane). Par conséquent, il ne peut aligner aucun pixel en quittant cette région. Ceci se produit dans deux cas différents : 1) lorsque la région de bruit est suffisamment large mais non plane, ou

formée de pixels ayant des profondeurs aléatoires ; 2) lorsque la région est plane mais insuffisamment large pour permettre aux agents de traverser la longueur minimale L nécessaire pour pouvoir s'adapter. Dans les deux cas, l'agent quitte la région de bruit et s'adapte dans la région plane qui entoure cette dernière. A l'intérieur d'une vraie région de l'image, un agent traverse une longueur suffisante lui permettant de s'adapter à cette région. Cependant, un agent ne peut pas s'adapter à l'intérieur d'une région dont la surface n'est pas plane, et ce, quelle que soit la taille de cette région. Il ne peut également pas s'adapter à l'intérieur d'une région plane si la taille de cette dernière ne lui permet pas de traverser une longueur supérieure ou égale à la longueur minimale L. Dans les deux derniers cas, les agents ne peuvent pas s'adapter, et par conséquent ne seront pas capables d'aligner les pixels de bordure en quittant ces régions. Autrement dit, les vraies régions ont des tailles suffisamment grandes qui permettent aux agents qui sont à l'intérieur de s'adapter et d'aligner les pixels de bordure en quittant ces régions. Par contre, les régions de bruit, qui sont des régions non planes ou ayant de faibles tailles ne permettent pas aux agents de s'adapter. Par conséquent les agents n'alignent pas les pixels de bordure de ces régions en les quittant. Dans ce cas, les bordures de ces régions sont continuellement alignées de l'extérieur par inclusion de leurs pixels dans les vraies régions environnantes. Après plusieurs pas d'exécution, ces régions seront complètement effacées.

A la fin du processus, toutes les régions de l'image sont bien lissées et délimitées par les contours détectés. Un simple accroissement de régions, contrôlé par les contours détectés, permet d'extraire les régions de l'image.

4.4. *Coordination d'agents par un champ de potentiel*

Afin de doter les agents d'un mécanisme d'auto-organisation spatiale, un champ de potentiel artificiel inspiré du champ électrostatique est créé et mis à jour autour des pixels alignés. Il permet aux agents de se regrouper autour des régions d'intérêt dans l'image et de concentrer leurs actions sur les contours de ces régions. Contrairement à d'autres travaux où le champ de potentiel est créé à des positions prédéfinies correspondant aux objets de l'environnement (buts et obstacles) (Ferber, 1995, Tsuji *et al.,* 2002, Simonin, 2005), dans notre cas, le champ de potentiel résulte de l'interaction des agents avec les objets de l'environnement (pixels de l'images). L'intensité $\Psi(x, y)$ du champ à la position (x, y), créée par un ensemble de P pixels préalablement alignés $\{(x_i, y_i), i = 1..P \wedge I(x_i, y_i)$=aligné$\}$ est donnée par :

$$\Psi(x, y) = \sum_{i=1}^{P} \frac{k}{\sqrt{(x - x_i)^2 + (y - y_i)^2}}, k \in R^+ \qquad [6]$$

où k est la constante de la force électrostatique, prise égale à 1.

Un agent, capable d'altérer des pixels (C_t=true), et situé à la position (x, y), subit une force d'attraction $\overrightarrow{F}$. Cette force s'exprime à l'aide du vecteur gradient du champ de potentiel :

$$\overrightarrow{F} = \begin{cases} -\overrightarrow{\nabla}\Psi(x,y) & \text{si } C_t\text{=true} \\ \overrightarrow{0} & \text{sinon} \end{cases} \quad [7]$$

Ainsi, les déplacements de l'agent qui sont de nature stochastique sont pondérés par la force attractive exercée par le champ de potentiel. Les agents sont ainsi poussés à se diriger vers les pixels d'intérêt, tout en maintenant un déplacement stochastique qui permet d'explorer toutes les régions de l'image.

Un mécanisme de relaxation du champ de potentiel est introduit. Il permet aux agents groupés autour des pixels d'intérêt de se libérer et d'explorer d'autres régions dans l'image. Autour d'un pixel donné, l'intensité du champ décroît après chaque altération de ce pixel. L'équation de la dynamique de relaxation est exprimée comme suit :

$$\Psi_{t+1}(x, y) = \mu \times \Psi_t(x, y), \mu < 1 \quad [8]$$

$\Psi_0(x, y)$ correspond au champ créé après la première altération du pixel. La constante μ initialisée à 0.9, représente le taux de décroissance de l'intensité du champ. Après plusieurs altérations d'un pixel donné, le champ généré autour de ce pixel tend à s'annuler. A la limite de la dynamique, le champ de potentiel est complètement relaxé dans toute l'image. Ceci représente l'état final du processus pour lequel l'intérieur des régions est totalement lissé et les contours sont précisément détectés.

5. Implementation

Plusieurs plates-formes ont été proposées pour le développement des systèmes multi-agents (**?**, **?**, **?**, **?**), dont la plupart sont implémentées en JAVA. Le langage JAVA permet la portabilité des solutions et la facilité d'intégration. Cependant, il n'est pas recommandé en cas d'applications nécessitant un minimum de temps d'exécution, telles que l'interprétation d'images en vision robotique. Certaines plates-formes sont spécialement orientées simulation, telle que SWARM (**?**). Swarm existe en objective-C, cependant cette version est de moins en moins supportée, et l'expérience s'accumule de plus en plus avec la version en JAVA. Nos expérimentations avec quelques une de ces plates forme ont révélé des temps de calcul excessifs. Par ailleurs, le C++ reste le langage qui regroupe la plupart des développeurs en traitement d'images et reconnaissance d'objets. En plus de l'expérience accumulée avec ce langage, le C++ génère un code dont le temps d'exécution est plus rapide. De ce fait, nous avons opté pour le développement d'un framework dédié à notre approche multi-agents pour la segmentation d'images, en utilisant le langage de programmation C++. Le framework était utilisé pour la segmentation d'images de profondeur. Néanmoins, sa décomposition

fonctionnelle permet sa réutilisation pour la segmentation de d'autres types d'images. Par ailleurs, ce framework est facile à utiliser par des développeurs qui n'ont pas nécessairement une compétence particulière en multi-agents. Le développeur est appelé à instancier uniquement les éléments qui sont spécifiques aux images traitées. Le framework développé est composé de plusieurs classes, dont les principales sont : la classe Pixel et la classe Agent. Le Sheduler est implémenté sous forme d'une classe qui existera lors de l'execution en une unique instance.

5.1. *Pixel*

Un pixel est caractérisé par une profondeur (z), les paramètres de l'équation du plan correspondant (a, b, c, d), un état ($state$) (lissé, aligné, inchangé), et l'intensité du champ de potentiel psi à la position du pixel. Deux méthodes, à savoir $isInPlane()$ et $isPixelOfInterest()$, sont utilisés pour déterminer l'état du pixel en cours. Elle permettent respectivement de tester si le pixel en question appartient à un un plan, et s'il est un pixel d'intérêt ou non. Deux autres méthodes, à savoir $smoothPixel()$ et $alignPixel()$ permettent respectivement et en fonction de l'état du pixel de le lisser, ou de l'aligner à une région plane.

5.2. *Agent*

Chaque agent est caractérisé par sa position courante sur l'image (x, y) et la capacité d'altération de pixels (c). L'activité d'un agent est initialisée par la méthode $create()$. La méthode $step()$ permet à un agent de se déplacer d'un pas sur l'image et d'exécuter l'action appropriée sur le pixel courant. Elle permet aussi de créer et de mettre à jour l'intensité du champ de potentiel aux pixels voisins du pixel en cours.

5.3. *Scheduler*

L'option du multi-threading a été aussitôt abandonnée dès les premières expérimentations. En effet, nous avons constaté qu'il n'était pas possible de lancer un nombre de threads suffisamment élevé (le nombre utilisé pour les images de test était 2500 ; voir section 6.2). Nous avons donc opté pour l'activation des agents par un scheduler d'exécution. Il s'agit du pilote du système qui permet d'initialiser les différents objets, et d'activer les agents. Le scheduler, après l'initialisation de tous les agents, sélectionne à chaque itération un agent et lui passe le contrôle pour effectuer un seul pas. Aussitôt le pas effectué, le contrôle est retourné au scheduler pour sélectionner et activer un autre agent. L'ordre de sélection des agents peut être séquentiel ou aléatoire. Nos expérimentations ont montré que les résultats étaient indépendants de l'ordre choisi. Les principaux attributs de la classe Scheduler sont :

– le nombre d'agents ($nbrAgents$) ;

– la longueur de réadaptation L ;

– les forces d'altération ξ_{min}, et $\{xi_{max}$;

– les seuils Tr_θ et Tr_D) ;

– le nombre maximal d'itérations ($maxIterations$) ;

– l'ordre de sélection ($selectionOrder$) ;

La principale méthode du scheduler est *launch()* qui permet d'initialiser et de lancer le scheduler.

6. Expérimentation et discussion

6.1. *Le framework d'évaluation*

Un framework d'évaluation et de comparaison, dédié aux méthodes de segmentation d'images de profondeur, a été proposé par Hoover et al. dans (Hoover *et al.,* 1996) et est largement utilisé par les auteurs d'algorithmes dans le domaine (Jiang *et al.,* 1999, Jiang *et al.,* 2000, Li *et al.,* 2003, Ding *et al.,* 2005, Bab Hadiashar *et al.,* 2006). Le framework consiste en une base d'images de profondeur, et un ensemble de métriques objectives de performance. Il permet de comparer une segmentation générée par une machine de segmentation (MS) à une segmentation manuelle, supposée idéale, et représentant la réalité terrain (GT). Les métriques de performance les plus importantes sont le nombre de régions correctement détectées, de régions sur-segmentées, de régions sous-segmentées, de régions omises et de régions de bruit. La classification de régions est effectuée en fonction d'un seuil de tolérance T ; $50\% < T \leq 100\%$, qui reflète la rigueur de la classification. Les 40 images de la base ABW, de tailles de 512×512 pixels, ont été divisées en deux sous-ensembles : un sous-ensemble de 10 images d'apprentissage, et un sous-ensemble de 30 images de test (Hoover *et al.,* 1996). Les images d'apprentissage sont utilisées pour estimer les paramètres d'une méthode de segmentation donnée. Suite à cela, la méthode est appliquée à toutes les images de test. Les différentes valeurs des métriques de performance sont calculées et stockées pour être utilisées pour la comparaison des différentes méthodes. Nous avons utilisé ce framework pour évaluer notre approche et comparer nos résultats aux quatre méthodes : USF, WSU, UB et UE, référencées dans (Hoover *et al.,* 1996). Avant de présenter les résultats expérimentaux, nous décrivons la phase d'apprentissage qui permet la sélection des valeurs optimales des différents paramètres.

6.2. *Sélection de paramètres*

Pour l'approche proposée, nommée SIBA pour Segmentation d'Images à Base d'Agents, six paramètres doivent être fixés : ξ_{min}, ξ_{max}, Tr_θ , Tr_D, N, et L. Cet ensemble de paramètres est divisé en deux sous-ensembles. ξ_{min}, ξ_{max}, Tr_θ, et Tr_D représentent respectivement les forces d'alignement et les seuils d'angle et de profondeur, et sont utilisés dans les opérations de test et d'alignement des pixels.

Les paramètres N et L représentent respectivement le nombre d'agents, et la longueur d'adaptation. Ces deux paramètres influencent la dynamique des agents. Pour le premier sous-ensemble de paramètres, nous avons considérés 256 combinaisons à savoir (ξ_{min}, ξ_{max}, Tr_θ et Tr_D) $\in \{0.5, 0.3, 0.1, 0.05\} \times \{1.0, 3.0, 5.0, 7.0\} \times \{15^\circ, 18^\circ, 21^\circ, 24^\circ\} \times \{12, 16, 20, 24\}$, pour faire exécuter sur les images d'apprentissage. Le critère de performance choisi pour ces quatre paramètres est le nombre moyen de régions correctement détectées dans les images d'apprentissage, avec le seuil de tolérance T=80%. Les forces d'alignement ξ_{min} et ξ_{max} ont été initialisées respectivement à 0.3 et 5.0. Ces valeurs assurent une bonne localisation des pixels de contour en un temps d'exécution raisonnable. Le seuil Tr_θ est initialisé à 21°. Nous avons constaté que des valeurs plus grandes de ce paramètre sous différencient les pixels selon leurs vecteurs normaux, et conduisent à une sous-segmentation. Cependant, des valeurs significativement inférieures à 21° sur différencient les pixels et conduisent à une sur-segmentation de l'image. Ceci se traduit par la génération d'un nombre élevé de fausses petites régions. Finalement, le seuil Tr_D est initialisé à 16. Les valeurs significativement supérieures à 16 peuvent conduire à une fausse fusion de régions parallèles et proches. Cependant, si la valeur de ce paramètre est significativement inférieure à 16, les régions hautement inclinées ne peuvent pas être détectées comme régions planes (Jiang *et al.,* 1999). Ceci conduit à un taux élevé de régions omises.

Le nombre d'agents utilisés N, et la longueur d'adaptation L sont critiques et doivent être minutieusement sélectionnés. Un mauvais choix de ces deux paramètres peut conduire à un taux élevé d'erreurs de segmentation. En effet, un nombre insuffisant d'agents, pour une taille donnée des images traitées (512×512 pixels), conduit à un sous-lissage de l'image et à un sous-amincissement des contours. Les régions qui en résultent sont amputées d'un nombre important de pixels qui auraient dû être inclus dans ces régions. Une faible valeur de la longueur d'adaptation L conduit à la prise en compte de petites régions planes qui devraient être considérées comme régions de bruit. Par contre, une valeur élevée de L peut conduire à l'omission de certaines vraies régions planes et qui sont relativement étroites (voir section 4.3). Pour fixer les paramètres N et L, 25 combinaisons de ces paramètres, soit $(N, L) \in \{1500, 2000, 2500, 3000, 3500\} \times \{3, 5, 7, 9, 11\}$ ont été exécutées sur les images d'apprentissage. Dans ce cas, le critère de performance choisi est le nombre moyen des régions de bruit, avec le seuil de tolérance T=80%. Le calcul a aboutit aux valeurs de N=2500 et L=7.

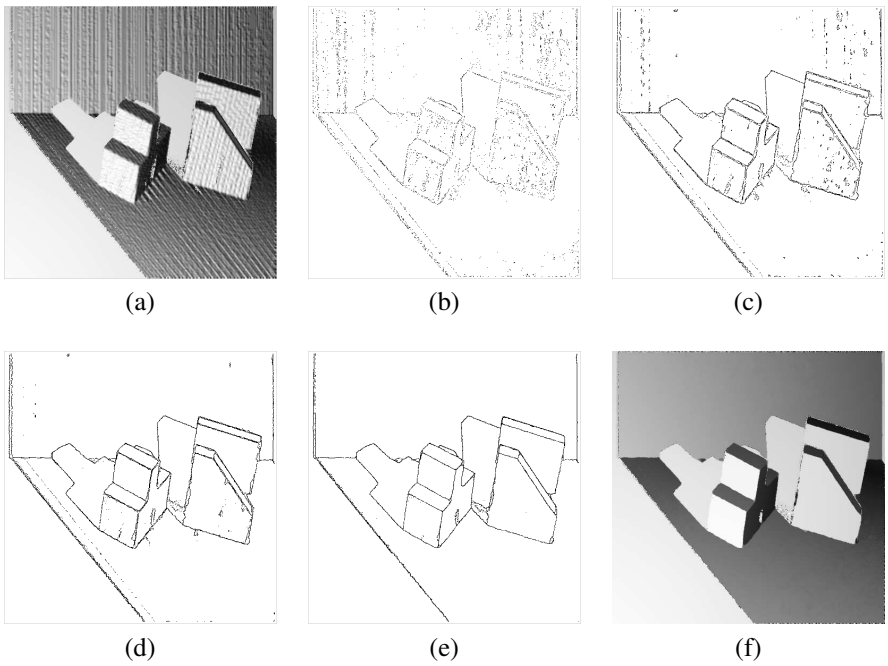
6.3. *Résultats expérimentaux*

La figure 4 montre un exemple de l'évolution du processus de segmentation dans le temps. Le temps t représente le nombre de pas effectués par chaque agent depuis le début du processus. L'affichage d'image de profondeur par un simple algorithme de rendu réaliste (figure 4a), permet de constater le niveau élevé de bruit dans les images utilisées. Les figures 4b, 4c, 4d et 4e montrent l'ensemble des pixels d'intérêt (pixels

de contours ou de bruit) respectivement à t=1000, 5000, 9000 et 13000. Les régions sont progressivement lissées par l'alignement des pixels de bruits qui sont à l'intérieur de ces régions, aux régions environnantes. Les contours entre les régions adjacentes sont progressivement amincis. A la fin, ces contours sont formés de lignes minces de 1 pixel de largeur (figure 4e).

La figure 5 montre les résultats de segmentation de l'image abw.test.8 obtenus par les différentes méthodes, avec le seuil de tolérance T=80%. Cette image était prise comme une image typique pour comparer les méthodes impliquées (Hoover *et al.,* 1996, Ding *et al.,* 2005). Les figures 5a, 5b et 5c montrent respectivement l'image de profondeur, l'image rendue réaliste, et la segmentation manuelle (GT).

Les figures 5d, 5e, 5f et 5g montrent respectivement les résultats de segmentation des méthodes USF, WSU, UB, et UE, tandis que la figure 5h présente le résultat de segmentation obtenu par notre méthode. Les métriques correspondantes dans la table 1 montrent que toutes les régions de l'image détectées par la meilleure méthode (UE) ont été détectées par notre méthode. A l'exception de la région ombrée qui n'a été détectée par aucune des méthodes, toutes les régions d'objets ont été bien détectées. Les régions incorrectement détectées dans cette image sont dues à une sur-segmentation



(a) (b) (c)

(d) (e) (f)

Figure 4. *Progression de segmentation. (a) Image de profondeur rendue réaliste (abw.test.6) ; (b) à t=1000, (c) à t=5000 ; (d) à t=9000 ; (e) à t=13000 ; (f) Image reconstruite rendue réaliste*

de la région plane représentant le support de la scène. Comparées aux autres méthodes, les différentes valeurs des métriques de détection incorrecte sont excellentes avec cette image.

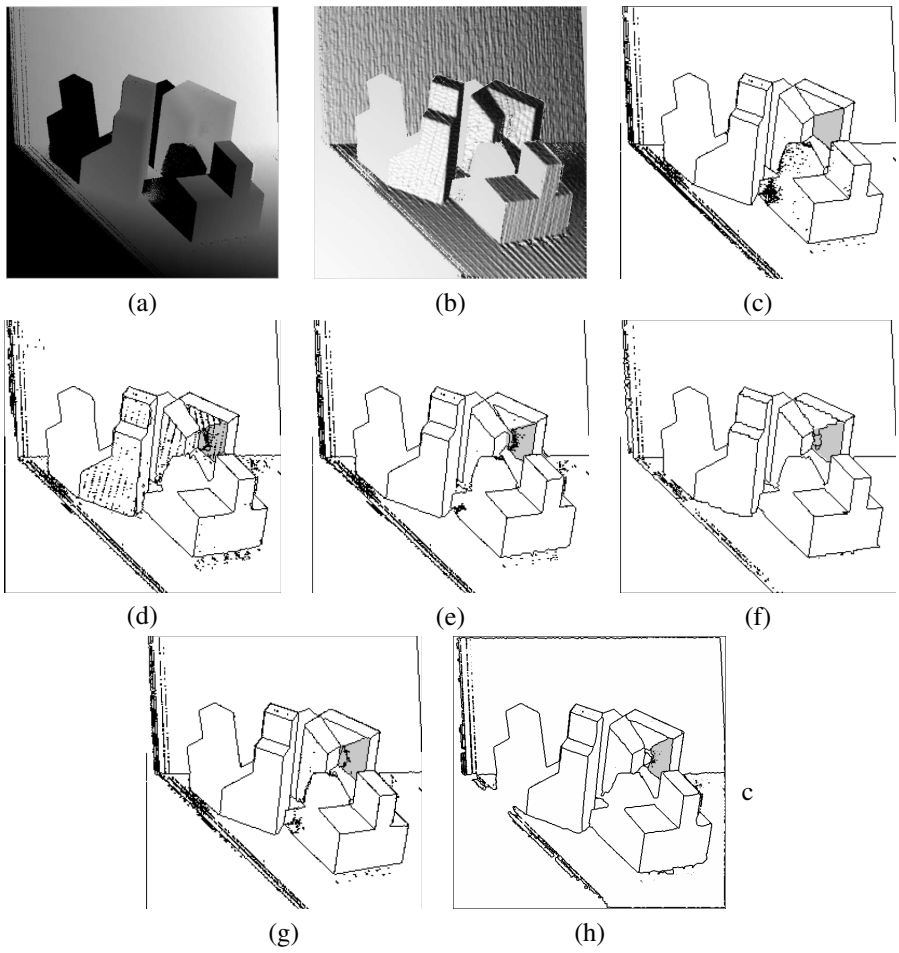


Figure 5. *Résultats de segmentation de l'image abw.test.8. (a) Image de profondeur ; (b) Image de profondeur rendue réaliste ; (c) Segmentation selon la Réalité terrain(GT) ; (d) Résultat de USF ; (e) Résultat de WSU ; (f) Résultat de UB ; (g) Résultat de UE ; (h) Résultats de SIBA (Régions extraites)*

La table 2 présente les résultats moyens obtenus avec toutes les images de test et pour toutes les métriques de performances. Le seuil de tolérance étant fixé à la valeur typique 80%. La figure 6 montre les nombres moyennes des nombres de régions correctement détectées pour toutes les images de test aux différentes valeurs du seuil de tolérance T ; $T \in \{51\%, 60\%, 70\%, 80\%, 90\%, 95\%\}$. Les résultats obtenus montrent que le nombre de régions correctement détectées par notre méthode est en moyenne

Tableau 1. *Comparaison des résultats de segmentation de l'image abw.test.8 avec le seuil de tolérance T=80%*

Méthode	Régions GT	Détection Correcte	Sur-segmentation	Sous-segmentation	Omise	Bruit
USF	21	17	0	0	4	3
WSU	21	12	1	1	6	4
UB	21	16	2	0	3	6
UE	21	18	1	0	2	2
SIBA	21	18	2	0	1	1

Tableau 2. *Résultats moyens des cinq méthodes au seuil de tolérance T=80%*

Méthode	Régions GT	Détection Correcte	Sur-segmentation	Sous-segmentation	Omise	Bruit
USF	15.2	12.7	0.2	0.1	2.1	1.2
WSU	15.2	9.7	0.5	0.2	4.5	2.2
UB	15.2	12.8	0.5	0.1	1.7	2.1
UE	15.2	13.4	0.4	0.2	1.1	0.8
SIBA	15.2	13.0	0.5	0.1	1.7	0.9

meilleur que celui des méthodes USF, UB et WSU. Par exemple, notre méthode enregistre un score meilleur que celui de WSU pour toutes les valeurs du seuil de tolérance T. Elle enregistre un score meilleur que celui de la méthode USF pour le seuil de tolérance $T \in \{80\%, 90\%, 95\%\}$, et meilleur que celui de UB pour $T \in \{51\%, 60\%, 70\%, 80\%\}$. Pour toutes les métriques de détection incorrecte, notre méthode a enregistré des scores équivalents à ceux de UE et USF, qui enregistrent des scores meilleurs que ceux des méthodes UB et WSU (figure 7).

6.4. *Analyse du coût de calcul*

En considérant la dynamique des agents, nous pouvons constater que le temps de calcul, représenté par le nombre de pas effectués par chaque agent, dépend uniquement de la densité de distribution des agents sur l'image. En effet, les agents nécessitent un minimum de temps de traitement pour pouvoir complètement amincir les contours des vraies régions, et lisser les régions de bruit. En dehors de l'influence du champ de potentiel, nous pouvons supposer que les agents restent uniformément distribués dans l'image. Dans ce cas, le temps de traitement est indépendant du nombre et de la distribution des contours et des régions de bruit.

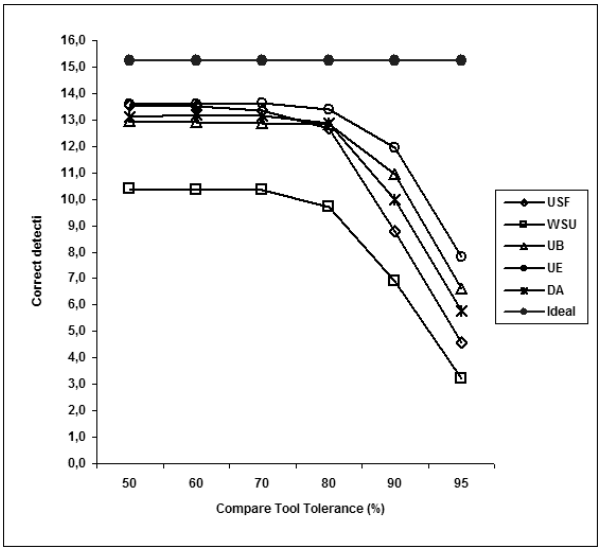


Figure 6. *Résultats moyens des régions correctement détectées de toutes les méthodes au différentes valeurs du seuil de tolérance T ;* $0.5 < T \leq 1.0$

Pour illustrer l'estimation du coût de calcul, nous considérons une image de synthèse de taille $W \times H = 256 \times 256$, contenant une unique région de bruit A de forme circulaire et d'un rayon initial (à t=0), $R_0 = 25$ pixels (figure 8a). Cette region est entourée d'une unique région plane B qui ne contient aucun pixel d'intérêt. Soit le nombre d'agents $N = 800$ et soit $\xi_{min} = \xi_{max} = \infty$. Ces deux dernières valeurs de ξ_{min} et ξ_{max} permettent d'aligner un pixel à une région plane après une seule altération par un agent (alignement total). Sous ces conditions d'exécution, les agents alignent les pixels de la bordure de la région A lorsqu'ils se déplacent à l'intérieur de cette région. Comme il s'agit d'une région de bruit, les agents qui transitent à l'intérieur ne peuvent pas s'adapter et par conséquent n'alignent pas les pixels de la bordure lorsqu'ils quittent cette région. La région A se contracte donc continuellement jusqu'à ce qu'elle ait complètement disparu (figure 8b, 8c, 8d). Le nombre de pas effectués par les agents avant l'effacement total de la région A représente le temps total de calcul, noté Γ. Au temps t, le nombre d'agents situés à l'intérieur de la région A est : $\pi\rho R_t^2$; où R_t est le rayon de la région au temps t et ρ est la densité de distribution des agents. Cette distribution dépend de la taille de l'image et du nombre d'agents utilisés :$\rho = \frac{N}{WH}$. Ainsi, le nombre d'agents situés sur la bordure de la région au temps t, peut être approximativement estimé par $2\pi\rho R_t$, exprimant le nombre d'agents sur la circonférence d'un cercle de rayon R_t, avec une densité d'agents ρ. Evidement, seulement un taux τ de ces agents alignent des pixels sur la bordure de la région A. Il s'agit des agents avec une capacité C_t=true et passant de la région B vers la région A. Nous déduisons donc, le nombre de pixels alignés au temps t : $2\pi\tau R_t\rho$, et par consé-

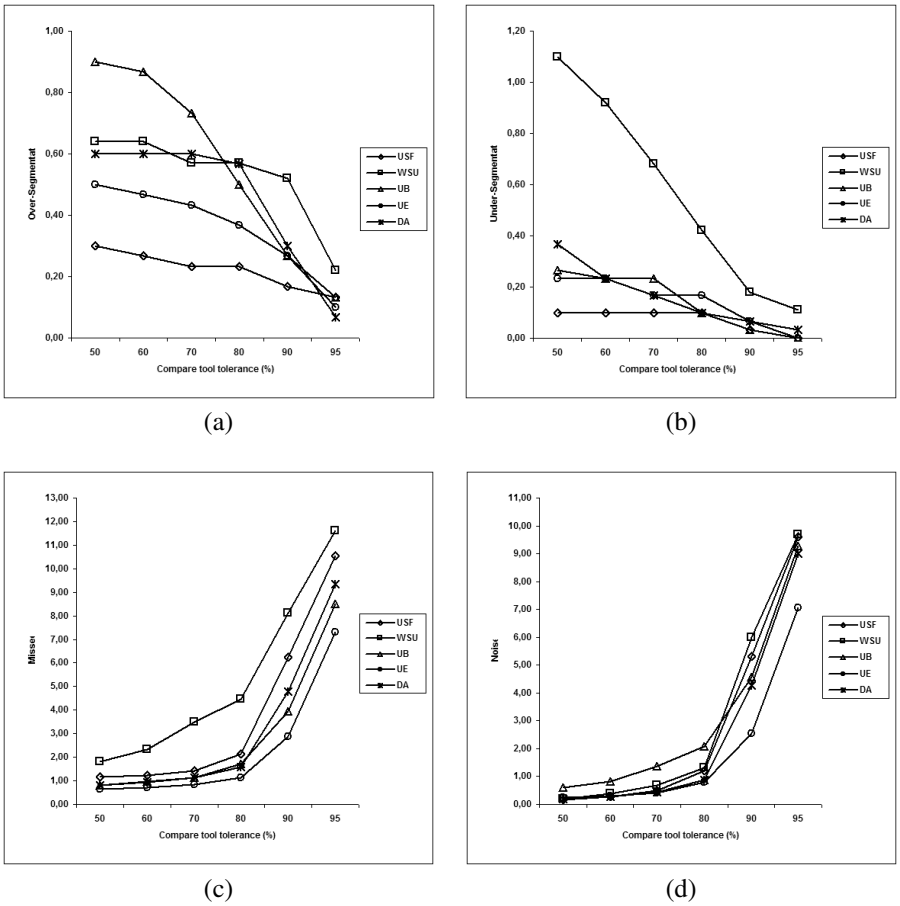


Figure 7. *Résultats moyens des métriques de détection incorrecte : (a) Sur-segmentation ; (b) Sous-segmentation ; (c) Regions omises ; (d) Regions de bruit*

quent, la nouvelle taille de la région A deviendra : $\pi R_{t+1}^2 = \pi R_t^2 - 2\pi\tau\rho R_t$. Le taux de décroissance du rayon R_t de la region de bruit A peut être exprimé comme suit : $R_{t+1}^2 = R_t^2 - \eta R_t$, où $\eta = \frac{2\tau N}{WH}$. Le temps minimum Γ nécessaire pour complètement lisser la région de bruit est obtenu lorsque $R_\Gamma \leq 0$.

Dans le but de comparer le temps de calcul théorique au temps de calcul observée, nous avons enregistré la taille de la région A après chaque pas d'exécution. La figure 9 montre les courbes respectives de la variation estimée et observée de la taille de la région de bruit, avec le paramètre $\tau = 0.24$. La corrélation des résultats nous permet de valider le modèle théorique d'estimation du coût de calcul.

Le temps moyen de segmentation par image sur l'ensemble des 30 images de test était de 14 secondes sur un mono-processeur PC Compaq 8220. Cette moyenne

Figure 8. *Variation de la taille de la région de bruit (A) : (a) Image de profondeur rendue réaliste ; (b) à t=5000 ; (c) à t=10000 ; (d) à t=15000*

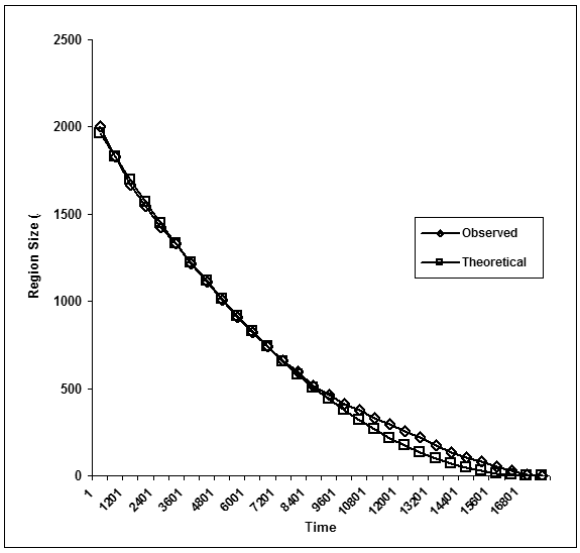


Figure 9. *Variations théorique et observée de la taille de la région de bruit (A)*

de temps de segmentation est meilleure que celles de la majorité des méthodes publiées (Hoover *et al.,* 1996). Par ailleurs, grâce au couplage faible des agents et au fait qu'ils ne communiquent pas directement, il est possible de paralléliser les traitements en les distribuant sur des unité de calcul autonomes partageant une mémoire commune qui contiendra l'image à segmenter. Ceci permettra une segmentation d'images rapide, souhaitable pour les application temps réel.

7. Conclusion

Dans cet article, nous avons proposé une approche multi-agents pour la segmentation d'images de profondeur. Les interactions indirectes entre les agents autonomes

qui se déplacent dans l'image permettent à la fois, une élimination efficace du bruit, et une extraction fiable des régions. La compétition entre les agents qui s'auto-organisent autour des bordures des régions ont permis de faire émerger les lignes de contours dans l'image. Ces contours, dont aucune détection explicite n'est codée dans les agents, émergent des interactions et des actions collectives au sein de la population d'agents. Nos résultats sont meilleurs que ceux obtenus par des algorithmes conventionnels de segmentation, tel que l'accroissement de région, ou la détection explicite de contours. Les agents utilisés sont faiblement couplés et communiquent indirectement via leur environnement (l'image). Ceci permet des implémentations parallèles souhaitables pour une haute efficacité de calcul, nécessaire pour l'interprétation en temps-réel d'images, notamment en vision robotique. Les résultats d'expérimentation obtenus avec les images réelles de la base ABW ont été comparés à ceux obtenus par quatre algorithmes typiques de segmentation d'images de profondeur. Les résultats de comparaison révèlent une excellente performance de la méthode proposée et ce, à la fois, en terme d'efficacité de détection et d'exactitude des résultats. Notre approche s'apprête à être généralisée à la segmentation d'images contenant des surfaces courbées. Pour ce faire, il est nécessaire de modéliser les surfaces dans ces images, et d'adapter le comportement des agents en fonction de la nature de ces surfaces.

8. Bibliographie

Bab Hadiashar A., Gheissari N., « Range Image Segmentation Using Surface Selection Criterion », *IEEE Transactions on Image Processing*, vol. 15, n° 7, p. 2006-2018, July, 2006.

Bovenkamp E. G. P., Dijkstra J., Bosch J. G., Reiber J. H. C., « Multi-agent segmentation of IVUS images », *Pattern Recognition*, vol. 37, n° 4, p. 647-663, 2004.

Ding Y., Ping X., Hu M., Wang D., « Range image segmentation based on randomized Hough transform. », *Pattern Recognition Letters*, vol. 26, n° 13, p. 2033-2041, 2005.

Fan T., Medioni G., Nevatia R., « Segmented Description of 3-D Surfaces », *IEEE J. Robotics Automat.*, vol. 3, n° 6, p. 527-538, December, 1987.

Ferber J., *Les systèmes multi-agents : vers une intelligence collective*, Informatique, Intelligence Artificielle, InterÉditions, 1995.

Genesereth M. R., Ketchpel S. P., « Software agents », *Commun. ACM*, vol. 37, n° 7, p. 48-ff., 1994.

Harrouet F., Tisseau J., Reignier P., Chevaillier P., « oRis : un environnement de simulation interactive multi-agents », *Technique et Science Informatiques*, vol. 21, n° 4, p. 499-524, 2002.

Hoover A., Jean-Baptiste G., Jiang X., Flynn P. J., Bunke H., Goldgof D. B., Bowyer K. W., Eggert D. W., Fitzgibbon A. W., Fisher R. B., « An Experimental Comparison of Range Image Segmentation Algorithms », *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, n° 7, p. 673-689, 1996.

Horaud R. P., Monga O., *Vision par ordinateur : outils fondamentaux*, Editions Hermès, Paris, 1995.

Inokuchi S., Nita T., Matsuda F., Sakurai Y., « A Three Dimensional Edge-Region Operator for Range Pictures », *6th International Conference on Pattern Recognition*, Munich, p. 918-920, 1982.

Jiang X., Bowyer K. W., Morioka Y., Hiura S., Sato K., Inokuchi S., Bock M., Guerra C., Loke R. E., du Buf J. M. H., « Some Further Results of Experimental Comparison of Range Image Segmentation Algorithms », *International Conference on Pattern Recognition*, vol. 4, p. 4877-4882, 2000.

Jiang X., Bunke H., « Edge Detection in Range Images Based on Scan Line Approximation. », *Computer Vision and Image Understanding*, vol. 73, n° 2, p. 183-199, 1999.

Kang S., Ikeuchi K., « The Complex EGI : A New Representation for 3-D Pose Determination », *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, n° 7, p. 707-721, 1993.

Li S. Z., *Markov random field modeling in image analysis*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2001.

Li S., Zhao D., « Gradient-based polyhedral segmentation for range images », *Pattern Recognition Letters*, vol. 24, n° 12, p. 2069-2077, 2003.

Liu J., Tang Y. Y., « Adaptive Image Segmentation With Distributed Behavior-Based Agents », *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, n° 6, p. 544-551, 1999.

Mehrtash A., « Fuzzy Edge Preserving Smoothing Filter Using Robust Region Growing », *IEEE International Conference on Fuzzy Systems*p. 1748-1755, 2006.

Newell A., *Unified theories of cognition*, Harvard University Press, Cambridge, MA, USA, 1990.

Richard N., Dojat M., Garbay C., « Automated segmentation of human brain MR images using a multi-agent approach », *Artificial Intelligence in Medicine*, vol. 30, n° 2, p. 153-176, 2004.

Rodin V., Benzinou A., Guillaud A., Ballet P., Harrouet F., Tisseau J., Bihan J. L., « An immune oriented multi-agent system for biological image processing », *Pattern Recognition*, vol. 37, n° 4, p. 631-645, 2004.

Sappa A., « Unsupervised Contour Closure Algorithm for Range Image Edge-Based Segmentation », *IEEE Transactions on Image Processing*, vol. 15, n° 2, p. 377-384, February, 2006.

Simonin O., « Construction of numerical potential fields with reactive agents », *AAMAS '05 : Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, ACM Press, New York, NY, USA, p. 1351-1352, 2005.

Tsuji T., Tanaka Y., Morasso P., Sanguineti V., Kaneko M., « Bio-mimetic trajectory generation of robots via artificial potential field with time base generator. », *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 32, n° 4, p. 426-439, 2002.

Wooldridge M., *Introduction to MultiAgent Systems*, John Wiley & Sons, June, 2002.

Wooldridge M., Jennings N. R., « Intelligent Agents : Theory and Practice », , vol. 10, n° 2, p. 115-152, 1995.

Zambonelli F., Jennings N. R., Wooldridge M., « Developing multiagent systems : The Gaia methodology », *ACM Trans. Softw. Eng. Methodol.*, vol. 12, n° 3, p. 317-370, 2003.