SLAM monoculaire et cartographie : vers une mise en cohérence fine

Pierre Lothe[1] Steve Bourgeois[1] Fabien Dekeyser[1] Eric Royer[2] Michel Dhome[2]

[1] CEA LIST

Laboratoire Systèmes de Vision Embarqués
Point Courrier 94, Gif-sur-Yvette, F-91191 France

{pierre.lothe, steve.bourgeois, fabien.dekeyser}@cea.fr

[2] LASMEA

UMR 6602 Université Blaise Pascal/CNRS
Clermont-Ferrand, France

{eric.royer, michel.dhome}@lasmea.univ-bpclermont.fr

Résumé

Les progrès des méthodes de localisation et reconstruction simultanées (SLAM) par vision permettent aujourd'hui d'envisager leur intégration dans de nombreuses applications : trajectométrie, reconstruction 3D, etc. Cependant, les problèmes de dérive de ces méthodes limitent leur utilisation à des séquences courtes. Dans ce papier, nous proposons un post-traitement s'appuyant sur un modèle CAO grossier de la scène pour corriger les reconstructions issues du SLAM. Cette méthode est fondée sur un modèle de déformations spécifique utilisé à travers une méthode ICP non-rigide entre le nuage de points reconstruit et le modèle CAO connu. Les résultats expérimentaux, sur séquences de synthèse et séquences réelles, montrent que la scène reconstruite retrouve sa cohérence spatiale et que la trajectoire de la caméra reconstruite est améliorée.

Mots clefs

SLAM par vision, reconstruction 3D, ICP non-rigide, localisation absolue, trajectométrie.

1 Introduction

Les méthodes de localisation et reconstruction simultanées (SLAM) par imagerie permettent à la fois la reconstruction de l'environnement et de la trajectoire de la caméra à partir d'une séquence vidéo. Cependant, l'importance des calculs nécessaires aux premières méthodes proposées empêche leur utilisation sur des scènes de grande taille. Des travaux récents proposent donc de contourner cette limite en réduisant la complexité du problème tout en conservant l'utilisation de matériel peu coûteux et facilement embarquable.

Par exemple, Nister [1] n'utilise aucune optimisation globale : cela permet de diminuer les temps et les coûts de calcul mais rend la méthode sensible aux erreurs d'accumulation, la géométrie 3D de la scène n'étant jamais remise en cause. Davison [2] propose un algorithme basé sur un filtre de Kalman. Cette méthode permet des traitements temps-réels lorsque le nombre de points considérés est faible. Une autre approche consiste à réaliser une optimisation non-linéaire de la scène : Royer [3] utilise un ajustement de faisceaux hiérarchique afin de réaliser des reconstructions de scènes de grande taille. Par la suite, Mouragnon [4] proposera une méthode de minimisation non-linéaire incrémentale afin d'éviter au maximum les problèmes de mémoires et d'atteindre le temps-réel : seule la géométrie liée aux dernières caméras est optimisée.

Malgré ces progrès, les méthodes de SLAM par vision présentent encore des limitations : la trajectoire ainsi que le nuage de points reconstruits sont connus à une similitude près. En effet, ces données sont calculées dans un repère relatif fixé arbitrairement et il est donc impossible d'obtenir une localisation absolue de chacun des éléments reconstruits. D'autre part, en plus d'être sensibles aux erreurs d'accumulation [2, 4], les méthodes de SLAM par vision monoculaire peuvent présenter une dérive du facteur d'échelle : les reconstructions sont réalisées à un facteur d'échelle près théoriquement constant sur la séquence entière, mais on observe souvent une dérive de ce facteur d'échelle le long de la trajectoire. Même si ces erreurs peuvent être tolérées pour des applications de guidage utilisant des déplacements relatifs locaux, elles deviennent bloquantes pour les applications s'appuyant directement sur les reconstructions des méthodes de SLAM telles que la trajectométrie ou la localisation globale.

Afin d'améliorer les résultats des méthodes de SLAM, une approche possible consiste à injecter des connaissances supplémentaires sur la trajectoire [5] ou sur la géométrie de l'environnement [6]. Cette dernière approche est d'autant plus intéressante que ce type d'information est désormais

facilement accessible grâce aux systèmes de navigation par GPS et aux bases de données SIG.

Plus précisément, la solution que nous proposons dans ce papier consiste à estimer une déformation non-rigide de la reconstruction issue du SLAM de façon à le mettre en correspondance avec un modèle CAO grossier uniquement composé des plans verticaux représentant les façades des bâtiments. Tout d'abord, une modélisation de l'erreur induite par les méthodes SLAM est proposée et utilisée pour établir un modèle de transformations non-rigides de l'environnement reconstruit (section 2). Les paramètres de ces transformations permettant d'aligner au mieux la reconstruction et le modèle CAO sont estimés à l'aide d'une méthode originale d'ICP non-rigide (section 3). La chaîne de traitement complète est alors évaluée à la fois sur des séquences de synthèse et réelles (section 4).

2 Modèle de transformations

La mise en correspondance entre nuages de points 3D et modèles CAO a déjà été largement étudiée, les méthodes les plus utilisées étant l'ICP (Iterative Closest Point) [7, 8] et le Levenberg-Marquardt [9]. Cependant, ces méthodes sont habituellement utilisées avec des transformations euclidiennes ou des similitudes qui ne sont pas adaptées à notre problème. Des méthodes de mise en correspondance non-rigides sont également proposées de façon à corriger cette limite. En raison de leur grand nombre de degrés de liberté, il est alors nécessaire de leur ajouter des contraintes. Par exemple, Castellani [10] utilise des termes de régularisation pour adapter des transformations non-rigides à son problème. De même, nous allons introduire ci-après le modèle de transformations non-rigides spécifique que nous utiliserons dans notre cas.

2.1 Modèle de transformations non-rigides

Les contraintes que nous utilisons pour limiter les transformations applicables à la reconstruction 3D issue du SLAM sont fondées sur des résultats expérimentaux. Nous observons en effet que le facteur d'échelle est quasi-constant sur les lignes droites alors qu'il a tendance à changer, parfois fortement, dans les virages (voir la figure 5(b)). Nous avons donc décidé de considérer les lignes droites de la trajectoire comme un élément rigide et de placer des articulations à chaque virage : ainsi, les transformations retenues sont des similitudes par morceaux avec contraintes de jointure aux extrémités.

La première étape est donc de segmenter automatiquement la trajectoire reconstruite. Pour cela, nous utilisons l'idée suggérée par Lowe dans [11]. Il propose en effet de découper récursivement un ensemble de points en différents segments en fonction à la fois de leur longueur et de leur déviation. Nous découpons donc la trajectoire reconstruite (représentée par un ensemble de caméras temporellement ordonnées) en m segments $(\mathcal{T}_i)_{1\leq i\leq m}$ dont les extrémités sont notées $(\mathbf{e}_i, \mathbf{e}_{i+1})$.

2.2 Paramétrisation des transformations

Une fois la trajectoire découpée, chacun des points 3D reconstruits doit être associé à un segment de caméras. Nous disons qu'un segment "voit" un point 3D si au moins une caméra de ce segment observe ce point. Deux cas de figure sont alors possibles. Le cas le plus simple apparait quand seulement un segment voit le point : il est alors lié à ce segment. Dans l'autre cas, si le point est observé par plusieurs segments, nous avons testé plusieurs politiques qui fournissent des résultats équivalents et nous avons donc choisi arbitrairement d'associer le point au segment qui l'observe en dernier.

Nous appelons désormais $\mathcal{B}_i$ un fragment composé des caméras de $\mathcal{T}_i$ (c'est à dire celles incluses entre $\mathbf{e}_i$ et $\mathbf{e}_{i+1}$) et des points 3D associés. Il est également à noter que pour $2 \leq i \leq$ m-1, le fragment $\mathcal{B}_i$ partage ses extrémités avec ses fragments voisins $\mathcal{B}_{i-1}$ et $\mathcal{B}_{i+1}$.

Nous avons vu dans la section 2.1 que les transformations utilisées sont des similitudes par morceaux avec contraintes de jointure aux extrémités. En pratique, ces transformations sont paramétrées grâce aux translations 3D des extrémités $(\mathbf{e}_i)_{1\leq i\leq m+1}$. A partir de ces translations, nous pouvons déduire la similitude à appliquer à chacun des fragments (c'est à dire ses caméras et points 3D). Précisément, la caméra étant embarquée sur un véhicule terrestre, l'angle de roulis n'est pas remis en cause : chaque fragment possède donc 6 degrés de liberté codés par les 3 degrés de liberté de chacune de ses extrémités.

La figure 1 illustre cette paramétrisation.

3 ICP non-rigide

Une fois les transformations non-rigides (section 2.1) et leur paramétrisation définies, nous pouvons utiliser les informations supplémentaires fournies par le modèle CAO pour améliorer les résultats issus du SLAM. Ainsi, nous proposons dans cette partie une méthode de type ICP non-rigide entre le nuage de points 3D reconstruit et le modèle CAO : tout d'abord, nous présentons la fonction de coût que nous minimisons, puis les différentes étapes d'optimisation et enfin l'initialisation de la méthode.

Fonction de coût. Les étapes de pré-traitement étant réalisées, nous disposons de la reconstruction fragmentée et d'un modèle CAO simple de la scène.

Dans nos travaux, nous souhaitons mettre en correspondance le nuage de points 3D avec le modèle CAO. La fonction de coût de base ϵ est donc la distance orthogonale d entre un point 3D et le modèle CAO $\mathcal{M}$, i.e. la distance entre un point 3D $\mathcal{Q}_i$ et le plan $\mathcal{P}_{g_i}$ auquel il appartient :

$$\begin{aligned} \epsilon(\mathcal{Q}_i, \mathcal{M}) &= d(\mathcal{Q}_i, \mathcal{M}) \\ &= d(\mathcal{Q}_i, \mathcal{P}_{g_i}) \end{aligned} \quad (1)$$

Minimisation non-linéaire. Le but de cette étape est de transformer les différents fragments pour minimiser la distance entre le nuage de points reconstruit et le modèle

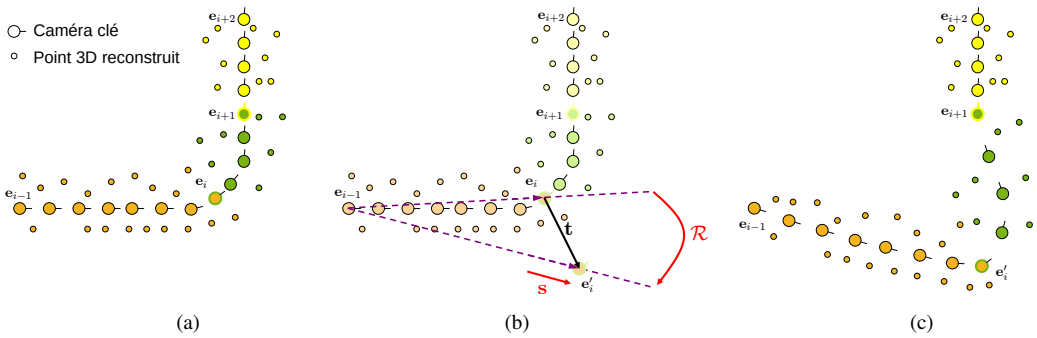

Figure 1 – ***Exemple de transformation basée fragment d'une reconstruction SLAM.*** *(a) La segmentation de la reconstruction originale. (b) L'extrémité* $\mathbf{e}_i$ *subit une translation. La similitude* $\mathcal{S}(\mathbf{e}_{i-1}, s, \mathcal{R})$ *est déduit de ce déplacement et est appliquée au fragment* $\mathcal{B}_{i-1}$*. Un traitement équivalent est réalisé sur* $\mathcal{B}_i$*. (c) Résultat de la transformation : les deux fragments liés à l'extrémité déplacée ont été modifiés.*

CAO, c'est à dire résoudre :

$$\min_{\mathbf{e}_1,\dots,\mathbf{e}_{m+1}} \sum_i d(\mathcal{Q}_i, \mathcal{M})^2 \tag{2}$$

Association point-plan. Le problème de cette fonction est que nous ne savons pas à quel plan $\mathcal{P}_{g_i}$ du modèle CAO le point reconstruit appartient réellement. Nous faisons donc l'hypothèse qu'il s'agit du plan le plus proche. L'équation 1 devient alors :

$$\epsilon(\mathcal{Q}_i, \mathcal{M}) = \min_{\mathcal{P}_j \in \mathcal{M}} d(\mathcal{Q}_i, \mathcal{P}_j) \tag{3}$$

Afin de réduire la complexité de l'algorithme, nous considérons alors que pour chacun des points $\mathcal{Q}_i$, le plan le plus proche $\mathcal{P}_{h_i}$ ne change pas au cours de la minimisation. De ce fait, l'association entre chaque point 3D et le plan $\mathcal{P}_{h_i}$ peut être réalisée hors de la minimisation :

$$\forall \mathcal{Q}_i, \mathcal{P}_{h_i} = \underset{\mathcal{P} \in \mathcal{M}}{argmin}\ d(\mathcal{Q}_i, \mathcal{P}) \tag{4}$$

Il est à noter que la distance d prend en compte que les plans 3D sont des plans finis : pour être associé au plan $\mathcal{P}$, un point 3D $\mathcal{Q}$ doit avoir sa projection orthogonale à l'intérieur des limites de $\mathcal{P}$.

Estimation robuste. L'association $(\mathcal{Q}_i, \mathcal{P}_{h_i})$ peut être erronée pour deux raisons : si $\mathcal{Q}_i$ est initialement trop éloigné de sa position réelle ou si ce point n'est pas sur un des plans du modèle CAO dans la réalité (i.e. n'appartient pas à une façade). Dans ces deux cas, le terme $d(\mathcal{Q}_i, \mathcal{P}_{h_i})$ peut empêcher l'obtention du minimum recherché dans notre problème. Pour limiter cet effet, nous utilisons un M-estimateur robuste ρ dans l'équation (2) :

$$\min_{\mathbf{e}_1,\dots,\mathbf{e}_m} \sum_i \rho(d(\mathcal{Q}_i, \mathcal{P}_{h_i})) \tag{5}$$

Nous avons décidé d'utiliser le M-Estimateur de Tukey [12]. Le seuil des M-estimateurs peut être choisi automatiquement grâce au MAD (Median Absolute Deviation). Le MAD fonctionne dans l'hypothèse où les données étudiées suivent une distribution Gaussienne autour du modèle. Cette hypothèse peut être considérée correcte sur chacun des fragments pris séparément mais pas sur leur ensemble. Nous utilisons donc un seuil différent ξ_j par fragment. Nous avons donc à normaliser les valeurs du Tukey sur chaque segment :

$$\rho'_{l_i}(d(\mathcal{Q}_i, \mathcal{P}_{h_i})) = \frac{\rho_{l_i}(d(\mathcal{Q}_i, \mathcal{P}_{h_i}))}{\max_{\mathcal{Q}_j \in \mathcal{B}_{l_i}} \rho_{l_i}(d(\mathcal{Q}_j, \mathcal{P}_{h_j}))} \tag{6}$$

où l_i est l'indice du fragment contenant $\mathcal{Q}_i$ et ρ_{l_i} le M-estimateur de Tukey utilisé avec le seuil ξ_{l_i}.

Pondération des fragments. Avec cette fonction de coût, chaque fragment aura un poids dans la minimisation proportionnel au nombre de points 3D qu'il contient. Dans ce cas, les fragments possédant peu de points pourraient ne pas être optimisés en faveur des autres. Afin de donner le même poids à chacun des segments, nous avons décidé d'unifier les résidus de leurs points 3D en fonction de leur cardinal :

$$\rho^*_{l_i}(d(\mathcal{Q}_i, \mathcal{P}_{h_i})) = \frac{\rho'_{l_i}(d(\mathcal{Q}_i, \mathcal{P}_{h_i}))}{card(\mathcal{B}_{l_i})} \tag{7}$$

et le problème s'écrit finalement :

$$\min_{\mathbf{e}_1,\dots,\mathbf{e}_{m+1}} \sum_i \rho^*_{l_i}(d(\mathcal{Q}_i, \mathcal{P}_{h_i})) \tag{8}$$

Enfin, nous le résolvons à l'aide de l'algorithme de Levenberg-Marquardt [13].

Optimisation itérative. En pratique, nous réalisons plusieurs fois cette minimisation non-linéaire en remettant en cause l'association point-plan avant chacune d'elles. Cela permet aux points 3D de changer le plan auquel ils sont associés sans perte de temps de calcul.

Initialisation. Les algorithmes de minimisation non-linéaires nécessitent une bonne initialisation des paramètres : la reconstruction 3D doit donc au moins être placée dans le même repère que le modèle CAO. Pour cela,

estimer une transformation rigide est suffisant lorsque la dérive de la reconstruction 3D est faible. Cependant, la dérive du facteur d'échelle fréquemment observée dans les reconstructions de SLAM peut impliquer des déformations géométriques très importantes. Pour assurer la convergence de l'algorithme, nous avons donc décidé de placer grossièrement chaque extrémité $\mathbf{e}_i$ autour du modèle CAO. Cela peut être fait manuellement grâce à une interface graphique utilisateur ou automatiquement si la séquence est synchronisée avec des données d'un système de navigation GPS grand public par exemple.

4 Résultats expérimentaux

Dans cette partie, nous présentons une séquence de synthèse de façon à obtenir des résultats numériques ainsi qu'une séquence réelle. L'algorithme de SLAM utilisé ici est celui proposé par Mouragnon [4].

4.1 Séquence de synthèse

La figure 4 présente les différentes étapes de notre méthode. La séquence synthétique (basée pour le modèle CAO de la figure 4(a)) a été créée afin d'utiliser l'algorithme de SLAM dans un monde 3D texturé de synthèse réalisé à l'aide d'un logiciel de dessin 3D. La trajectoire suivie est indiquée par les flèches rouges sur la figure 4(a).

La figure 2 souligne la dérive de la méthode de SLAM originale. Le facteur d'échelle est calculé comme étant le rapport entre la distance entre deux caméras reconstruites et la distance entre ces deux caméras dans la vérité terrain. Nous pouvons observer que le facteur d'échelle diminue au fil du parcours, ce qui rend la reconstruction incorrecte (figure 4(b)). En particulier, la boucle originale sur la trajectoire de la caméra n'est pas respectée.

Pour l'initialisation de l'ICP, nous avons simulé manuellement les résultats d'un système GPS bas coût : chacune des extrémités a été déplacée suivant l'erreur de ce type de système (figure 4(c)).

Nous observons qu'après notre traitement (figure 4(d)) la boucle est restaurée alors qu'aucune contrainte de boucle n'est directement insérée dans notre méthode. De plus, le nuage de points 3D retrouve sa cohérence spatiale. Cette amélioration est confirmée par les résultats numériques du tableau 1 : la distance moyenne entre les caméras reconstruites et la vérité terrain passe de plus de 4 mètres à environ 50 centimètres. Les statistiques avant l'ICP ont été calculées sur les 5591 points 3D reconstruits (parmi les 6848 proposés par le SLAM) considérés comme non-aberrants par l'étape d'ICP. De plus, il apparaît dans la figure 3 que seules les erreurs suivant la direction de la trajectoire restent signifiantes. Cela est directement lié au fait que les transformations non-rigides que nous utilisons supposent que le facteur d'échelle est strictement constant dans les lignes droites, ce qui est uniquement une approximation grossière.

Tableau 1 – ***Résultats numériques sur la séquence de synthèse.*** *Chaque valeur est une valeur moyenne sur l'ensemble de la reconstruction.*

	Avant ICP	Après ICP
Distance moyenne caméras-vérité terrain (m)	4.61	**0.51**
Ecart type (m)	2.25	**0.59**
Distance moyenne points 3D-modèle CAO (m)	3.37	**0.11**
Ecart type	3.9	**0.08**
Seuil de Tukey	×	0.38

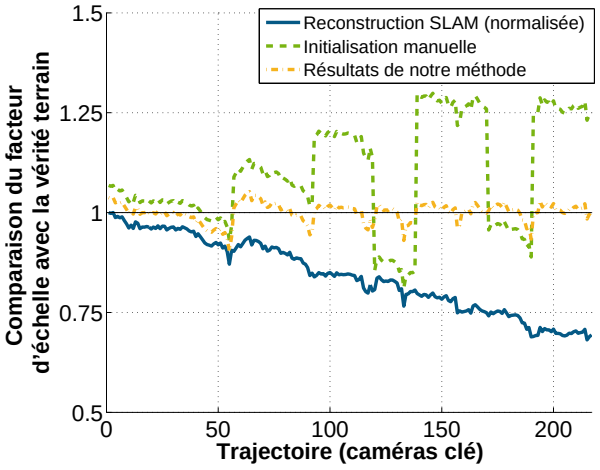

Figure 2 – ***Evolution du facteur d'échelle.*** *Nous observons la dérive du facteur d'échelle sur la reconstruction originale. Après notre méthode, le facteur d'échelle est centré autour de 1.*

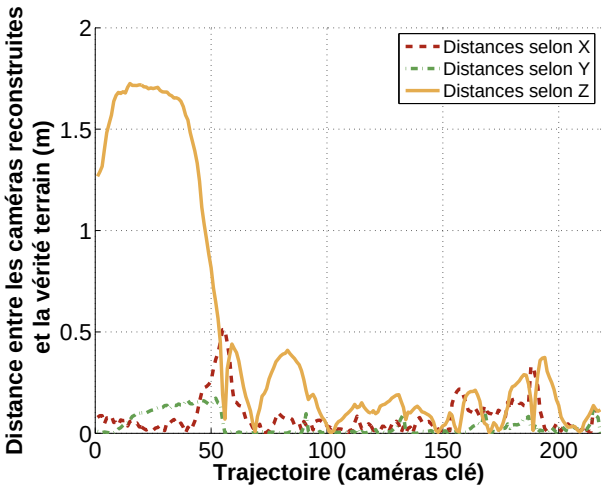

Figure 3 – ***Erreurs résiduelles sur les distances entre caméras reconstruites et vérité terrain.*** *Les coordonnées (X, Y, Z) sont relatives à chaque caméra : Z correspond à l'axe optique, X à la latitude et Y l'altitude.*

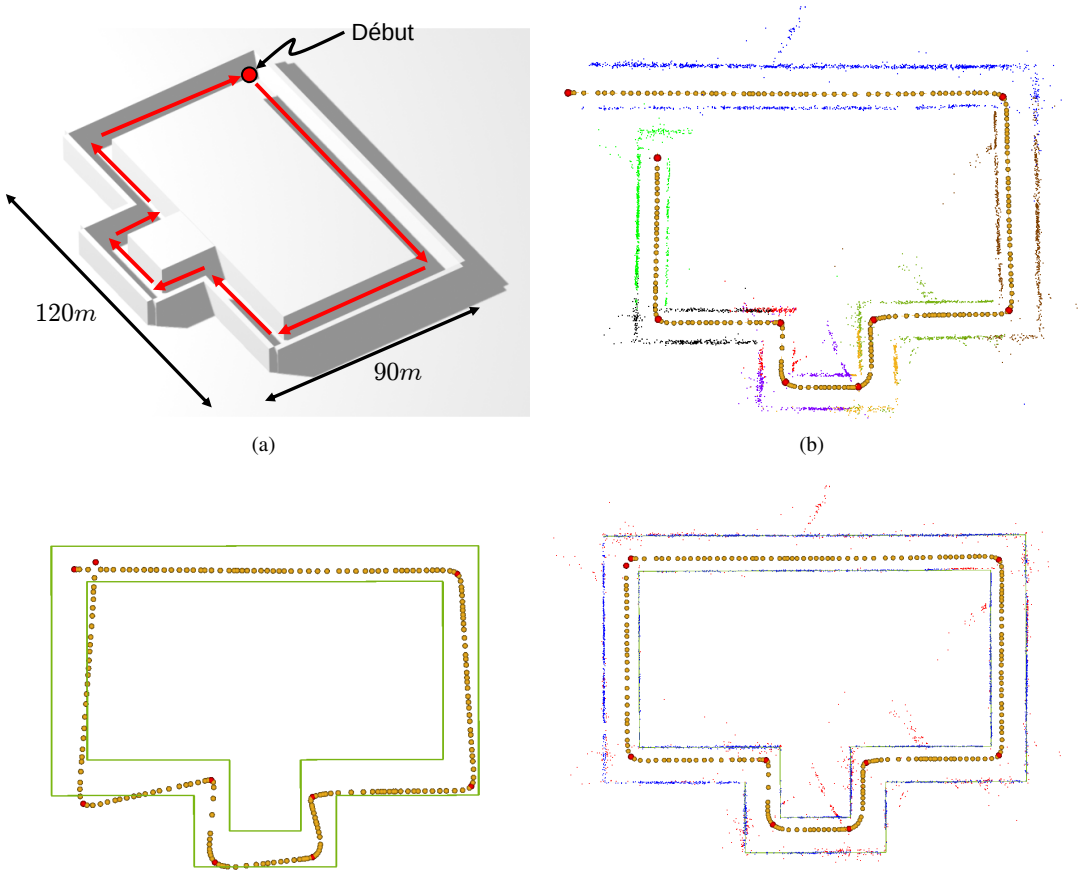

Figure 4 – ***Traitement de la séquence de synthèse.*** *(a) Modèle CAO de synthèse. (b) Vue de dessus de sa reconstruction 3D en utilisant la méthode de Mouragnon [4]. Les sphères oranges sont les caméras et les rouges correspondent aux extrémités des segments. La segmentation automatique de la reconstruction est également représentée : les points 3D reconstruits sont colorés selon leur appartenance à un fragment. La figure (c) représente l'initialisation utilisateur de la trajectoire autour du modèle CAO (en vert) avant l'ICP. La figure (d) présente la reconstruction 3D après notre méthode : les points 3D bleus sont les points conservés et les rouges les points aberrants. (Voir la version électronique du papier pour les figures en couleur.)*

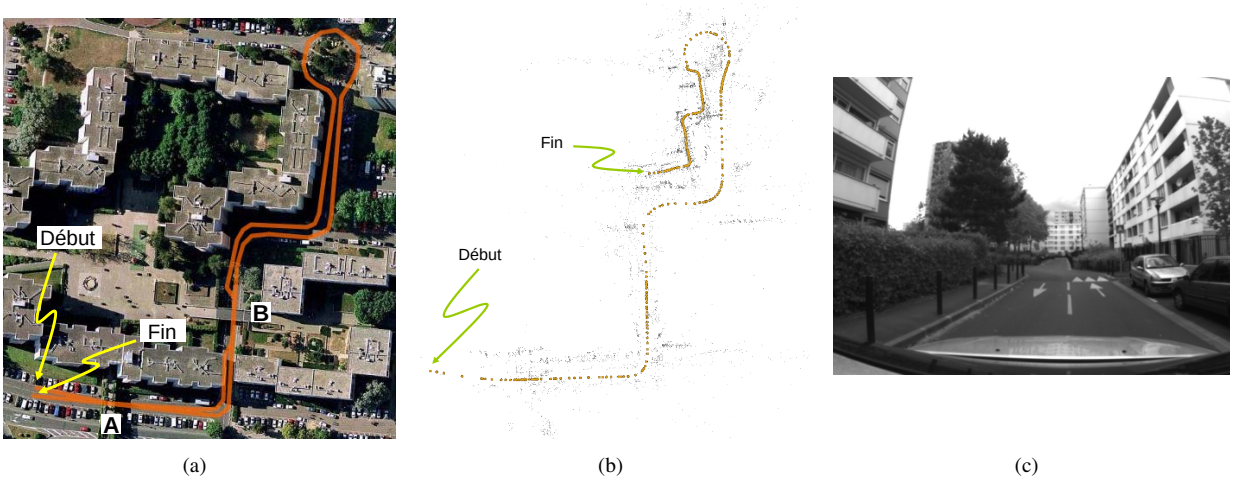

Figure 5 – ***Reconstruction de la séquence réelle par la méthode de Mouragnon [4].*** *(a) est la vue satellite de la trajectoire reconstruite dans (b). (c) est une image de la séquence vidéo utilisée.*

4.2 Séquence réelle

La séquence réelle (figure 5) est une boucle d'environ 600 mètres. La reconstruction SLAM (figure 5(b)) est un bon exemple de la dérive du facteur d'échelle : nous observons qu'il y a un changement radical du facteur d'échelle après le rond-point et que de plus la boucle n'est pas respectée.

Aucune vérité terrain n'étant disponible pour cette séquence, nous ne pouvons avoir qu'une appréciation visuelle de nos résultats. La figure 6 tend à montrer la cohérence de notre méthode. En effet, deux places spécifiques indiquées dans les figures 5(a) et 6(c) sont aisément reconnaissables : "A" est un croisement de la trajectoire dû à la présence d'un tunnel à une seule voie et nous pouvons voir en "B" que nous sommes passés deux fois sur la même voie à cause d'un véhicule en panne sur la chaussée.

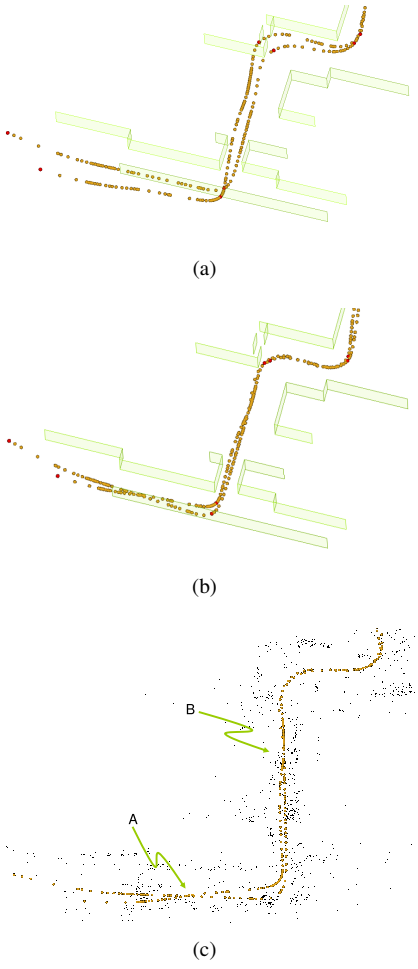

Figure 6 – ***Traitement sur la séquence réelle.*** *(a) Trajectoire à l'initialisation de l'ICP. (b) Trajectoire corrigée après l'ICP non-rigide. (c) Reconstruction SLAM après notre méthode. Les points "A" et "B" correspondent aux points "A" et "B" de la figure 5(a).*

5 Conclusion et perspectives

Nous avons présenté dans ce papier un algorithme post-traitement visant à améliorer les résultats issus des méthodes de reconstruction de type SLAM. Notre méthode utilise une fragmentation automatique de la scène reconstruite puis un ICP non-rigide entre celle-ci et un modèle CAO simple de l'environnement. Les expériences sur des séquences de synthèse et réelles montrent que notre méthode permet de réduire de façon significative les erreurs sur les positions 3D. Les boucles des trajectoires sont corrigées et les erreurs sur la position des caméras sont réduites jusqu'à obtenir des erreurs de l'ordre de 50 centimètres (erreur essentiellement située le long de la trajectoire) pour des séquences de plusieurs centaines de mètres.
Nos recherches actuelles et futures visent à relâcher les contraintes de notre modèle d'erreur en incluant directement les informations relatives au modèle CAO dans l'ajustement de faisceaux du SLAM.

Références

[1] D. Nister, O. Naroditsky, et J. Bergen. Visual odometry. Dans *CVPR*, pages 652–659, 2004.

[2] A. Davison, I. Reid, N. Molton, et O. Stasse. MonoSLAM : Real-time single camera SLAM. *PAMI*, 26(6) :1052–1067, 2007.

[3] E. Royer, M. Lhuillier, M. Dhome, et T. Chateau. Localization in urban environments : Monocular vision compared to a differential gps sensor. Dans *CVPR*, pages 114–121, 2005.

[4] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, et P. Sayd. Real time localization and 3d reconstruction. Dans *CVPR*, pages 363–370, 2006.

[5] A. Levin et R. Szeliski. Visual odometry and map correlation. Dans *CVPR*, pages 611–618, 2004.

[6] G. Sourimant, L. Morin, et K. Bouatouch. Gps, gis and video fusion for urban modeling. Dans *CGI*, may 2007.

[7] S. Rusinkiewicz et M. Levoy. Efficient variants of the ICP algorithm. Dans *3DIM*, pages 145–152, 2001.

[8] W. Zhao, D. Nister, et S. Hsu. Alignment of continuous video onto 3d point clouds. *PAMI*, 27(8) :1305–1318, 2005.

[9] A. Fitzgibbon. Robust registration of 2d and 3d point sets. Dans *BMVC*, pages 411–420, 2001.

[10] U. Castellani, V. Gay-Bellile, et A. Bartoli. Joint reconstruction and registration of a deformable planar surface observed by a 3d sensor. Dans *3DIM*, pages 201–208, 2007.

[11] D. G. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31(3) :355–395, 1987.

[12] P. Huber. *Robust Statistics*. Wiley, New-York, 1981.

[13] K. Levenberg. A method for the solution of certain non-linear problems in least squares. *Quart. Appl. Math.*, 2 :164–168, 1944.