SLAM visuel temps réel pour l'estimation précise de plan

Datta Ramadasan, Clement Deymier, Marc Chevaldonné, Thierry Chateau

SLAM visuel temps réel pour l'estimation précise de plan

D. Ramadasan[1] C. Deymier[1] M. Chevaldonné[2] T. Chateau[1]

[1] Insitut Pascal UMR 6602 CNRS-UBP-IFMA
[2] ISIT UMR 6284 Uda-CNRS

datta.ramadasan@univ-bpclermont.fr

Résumé

Ce papier présente un algorithme de localisation et cartographie simultanée (ou $SLAM$ pour Simultaneous Localization And Mapping) utilisant des contraintes issues de modèles observés dans un environnement inconnu, on parle alors de $SLAM$ contraint ou $CSLAM$. L'objectif est l'estimation précise des paramètres d'un objet présent dans la scène pour la réalité augmentée. Nous proposons pour cela d'inclure, dans une méthode d'ajustement de faisceaux incrémental, les paramètres de l'objet au même titre que les poses caméras et les points 3D nécessaires au $CSLAM$. Nous montrons, à travers l'exemple d'un modèle de plan 3D initialisé en ligne, que l'optimisation conjointe des paramètres permet, non seulement de contraindre les points 3D à se rapprocher du plan, mais également de contraindre le plan à se rapprocher des points 3D. Des expérimentations mettront en évidence la précision et le gain en temps de calcul de notre approche comparativement au $CSLAM$ classique.

Mots Clef

SLAM temps réel, ajustement de faisceaux incrémental, contrainte planaire.

Abstract

This paper presents an algorithm for simultaneous localization and mapping (or SLAM) using constraints from patterns observed in an unknown environment, this is called contrained SLAM or CSLAM. The aim is an accurate estimation of an object's parameters in the scene. We propose to include in an incremental bundle adjustment method, the parameters of the object as well as the camera poses and 3D points used by the CSLAM algorithm. We show, through the example of a 3D model of plane initialized online, that the joint optimization of parameters allow the process to force the 3D points to approach the plane, and also to force the plane to approach 3D points. Experiments will demonstrate the accuracy and the gain in computation time of our approach compared to conventional CSLAM.

Keywords

Real-time SLAM, incremental bundle adjustment, planar constraint.

1 Introduction

Les avancées en matière de $SLAM$ *(Simultaneous Localization And Mapping)* monoculaire de la dernière décennie ont permis aux méthodes basées ajustement de faisceaux [4][7] d'atteindre des performances compatibles avec des applications temps réel. Des travaux plus récents [6][12] prennent en compte des *a priori* sur la géométrie de la reconstruction (contraintes sur les poses caméras et/ou sur les points 3D) et ainsi de gagner en robustesse et performance. Toutefois, ces *a priori* sont souvent connus approximativement car ils sont estimés en ligne. Les contraintes appliquées sont alors incohérentes avec les données déjà reconstruites. C'est pourquoi nous proposons dans ce papier d'inclure les contraintes dans les paramètres estimés par le processus d'optimisation.

Nous utilisons une méthode de SLAM basée image clé semblable à celles dévelopées par Mouragnon *et al.* [7] et Klein *et al.* [4]. Ces méthodes sous-échantillonnent le flux vidéo en images clés (considérées pertinentes) telles que les différences entre chacune d'elles permettent l'estimation de la trajectoire caméra. L'originalité de ces méthodes réside dans l'optimisation locale qui ne remet en cause qu'un sous-ensemble de paramètres. Celui-ci correspond aux 3 dernières images clés dans [7] et aux 5 plus proches dans [4] avec les points 3D associés. Ces deux approches incluent également dans la fonction de coût les erreurs liées aux points 3D vus dans des poses caméras non optimisées. Ceci a pour effet d'ancrer la partie plus ancienne de la trajectoire afin de limiter la dérive induite par l'accumulation d'erreur. Ces méthodes fonctionnent en environnement inconnu, mais il est souvent possible d'utiliser des *a priori*.

Plus récemment, Tamaazousti *et al.* [12] proposent d'ajouter à un ajustement de faisceaux incrémental introduit dans [7] des contraintes sur les poses et points 3D. Celles-ci proviennent d'observations liées à un objet dont on connaît le modèle et qui est présent dans la scène. L'objet est détecté dès l'initialisation du $CSLAM$. Lorsque l'objet est suffisamment texturé pour y détecter des points d'intérêt, les points 3D associés à l'objet sont reconstruit sur la surface de celui-ci. Lorsque l'objet est peu texturé, les contours de l'objet sont détectés et la pose de la caméra est contrainte pour que la projection dans l'image du modèle 3D coïncide avec les contours de l'objet observé. Tamaazousti *et al.*

montrent ainsi que l'ajout de contraintes sur les poses et les points 3D permet une reconstruction de meilleure qualité et un recalage plus précis pour des applications de réalité augmentée comparativement au $SLAM$ classique.
De façon similaire, Lee *et al.* [6] détectent et optimisent un plan avec un ajustement de faisceaux sur les 2 premières poses de la séquence. La suite du processus est un $CSLAM$ avec contrainte planaire, le plan n'étant plus remis en question. Ils montrent que l'utilisation de la contrainte planaire permet la réduction de l'ensemble des poses à optimiser à la dernière image clé tout en conservant une précision suffisante.
Les solutions temps réel décrites nécessitent que le plan soit parfaitement connu dès les premières images pour débuter le $SLAM$. Dans une approche plus générale où le plan serait détecté au cours du $SLAM$, il est plus difficile de contraindre les données déjà reconstruites sur une hypothèse de plan approximative. C'est pourquoi, nous proposons d'inclure les paramètres liés à la contrainte planaire dans le processus d'optimisation afin que l'entité plan soit remise en question au même titre que les poses caméras ou les points 3D. Après avoir introduit la méthode de $SLAM$, nous présenterons l'initialisation en ligne et l'optimisation d'une contrainte planaire dans un ajustement de faisceaux incrémental. La dernière partie sera consacrée aux résultats de notre approche comparativement aux $CSLAM$.

2 Algorithme de reconstruction incrémentale

La reconstruction incrémentale consiste à cartographier l'environnement à partir d'un flux vidéo à travers un processus itératif de sélection d'images clés et d'optimisation. Ce processus doit offrir un compromis satisfaisant entre la précision du résultat et le temps nécessaire pour effectuer les calculs. On considère le processus temps réel si le temps de mise à jour de la carte est inférieur au temps d'acquisition entre deux images clés. On utilise une caméra calibrée dans un environnement structuré et on suppose que le mouvement et la vitesse d'acquisition de la caméra permettent la mise en correspondance d'amers visuels entre deux images successives. La détection des points d'intérêt sur les images est réalisée avec le détecteur de Harris *et al.* [2], et leur appariement entre les images utilise le descripteur ZNCC *Zero Mean Normalized Cross Correlation* (qui a l'avantage d'être invariant à la luminosité).
Cette partie présente les différentes étapes de l'algorithme de $SLAM$:
– initialisation de la carte 3D,
– méthode de localisation à partir de la carte existante,
– ajout d'images clés,
– optimisation des paramètres du problème.
Un effort particulier a été fait sur les critères de sélection des données afin que celles-ci soient à la fois nécessaires et suffisantes. Les contributions sont apportées sur les critères de sélection :
– des images de l'initialisation,
– des images clés,
– des paramètres à optimiser.

2.1 Initialisation

L'initialisation de la trajectoire se fait par sélection de 3 images $I = \{I_1, I_2, I_3\}$ dans le flux vidéo. Cette sélection est telle que le déplacement de la caméra entre chacune des images permette à la fois l'appariement de points d'intérêt mais également le calcul de la géométrie des poses. Ce triplet vérifie que 75% des points d'intérêt de I_1 soient appariés avec I_2 et 50% soient appariés avec I_3. Pour estimer les paramètres des poses et des points 3D à partir des appariements entre les images de I, on applique l'algorithme des 5 points dans 3 vues présenté par Nister *et al.* [8]. L'ensemble des paramètres est ensuite optimisé dans un ajustement de faisceaux global.

2.2 Calcul de pose

Le calcul de pose consiste à estimer, au temps k, les paramètres de la pose courante C_k de la caméra à partir de l'image I_k. Pour cela, on effectue une mise en correspondance de points d'intérêt entre I_k et l'image I_{ref} correspondante à la pose clé C_{ref} la plus proche de la dernière pose connue C_{k-1} de la caméra. Royer *et al.* [11] préconisent d'utiliser une mise en correspondance avec prédiction des zones de recherche de points d'intérêt dans l'image. Ainsi, on apparie les points d'intérêt de I_k avec la projection des points 3D de I_{ref} dans I_k selon le critère de corrélation ZNCC et la proximité dans l'image. Les N correspondances 3D-2D obtenues, qui associent à chaque point 3D P_{p_n} un point d'intérêt p_n, sont ensuite utilisées pour estimer C_k avec une méthode itérative. Celle-ci consiste en une optimisation robuste basée sur la médiane des erreurs de reprojection. Lorsque l'erreur médiane devient suffisamment faible (< 2 pixels), il est possible d'utiliser plus de la moitié des observations. Les paramètres de la pose C_k sont initialisés avec la dernière position connue C_{k-1} puis optimisés par l'algorithme de Levenberg-Marquardt (voir section 2.5). La méthode minimise la somme des distances entre les N points d'intérêts p_n et les projetés dans l'image des points 3D P_{p_n} :

$$\hat{C}_k = \arg\min_{C_k} \sum_{n=0}^{N} \| \frac{p_n - \pi(C_k, P_{p_n})}{\sigma_H} \|^2 \qquad (1)$$

avec π la fonction de projection et σ_H l'incertitude de détection issue du détecteur de Harris (σ_H fixé à 2 pour les expérimentations).

2.3 Sélection des images clés

Pour sélectionner les images clés, nous définissons un critère tel qu'une image I_k correctement localisée soit ajoutée à la carte 3D comme image clé si l'image I_{k+1} possède moins de 20% des points d'intérêt correctement associés à des points 3D déjà observés 3 fois. Ce critère arbitraire garantit d'avoir des points 3D fiables car observés dans au moins 3 images clés. Il offre également un réglage du

taux de propagation souhaité entre les points des nouvelles images et la carte.

2.4 Ajustement de faisceaux incrémental

L'ajout d'une image clé apporte de nouvelles informations sur les points 3D remettant en cause les paramètres constituant la carte 3D. Afin d'effectuer un minimum de calculs, on limite les paramètres à optimiser à un sous-ensemble relativement restreint correspondant aux paramètres qui sont directement remis en cause. Pour sélectionner les paramètres à optimiser, Klein *et al.* [4] définissent l'ensemble I_{opt} composé de la dernière image clé ainsi que des 4 images clés les plus proches. Ils définissent également l'ensemble P_{opt} contenant l'ensemble des points 3D vus dans I_{opt}. Ils optimisent alors les poses clés de I_{opt} ainsi que les points 3D de P_{opt} en tenant compte des projections des points 3D de P_{opt} dans toutes les images où ils apparaissent. Nous proposons de redéfinir I_{opt} comme l'ensemble des images clés ayant des points 3D en commun avec la dernière image clé. Cela permet de sélectionner les poses les plus pertinentes car directement remises en question par les nouvelles observations. Toutefois, on limite le nombre d'images clés de I_{opt} à 10 pour conserver une résolution en temps réel du problème. La fonction de coût $\mathcal{E}$ à minimiser correspond alors à la somme des K erreurs de reprojection entre les points d'intérêt p_k et les projetés des points 3D P_{p_k} dans les images I_{opt} avec C_{p_k} les poses caméras associées :

$$\mathcal{E} = \sum_{k=0}^{K} \| \frac{p_k - \pi(C_{p_k}, P_{p_k})}{\sigma_H} \|^2 \tag{2}$$

2.5 Résolution des équations normales

L'ensemble X des paramètres de poses et points 3D sont optimisés en utilisant l'algorithme de moindre carré non-linéaire Levenberg-Marquardt [3]. A chaque itération i, les paramètres sont mis à jour :

$$X_{i+1} = X_i - (J_i^T J_i + I\lambda_i)^{-1} J_i^T \epsilon_i \tag{3}$$

avec J la jacobienne de la fonction de coût, ϵ l'ensemble des résidus et λ un paramètre d'amortissement. Le calcul de l'incrément Δ à appliquer aux paramètres X pour diminuer l'erreur de la fonction de coût revient à résoudre le système linéaire (ou équations normales) suivant :

$$(J_i^T J_i + I\lambda_i)\Delta_i = J_i^T \epsilon_i \tag{4}$$

La résolution de ce système est détaillé dans [3] pour l'optimisation globale des paramètres et dans [7] pour le cas incrémental.

2.6 Performances de l'implémentation

L'implémentation (en langage C++) de l'algorithme de $SLAM$ présentée ici offre des performances temps réel. Le temps d'exécution est d'environ 10 ms pour ajouter une image clé, sur un ordinateur de bureau équipé d'un processeur i7 3.4Ghz en utilisant un seul fil d'exécution (processus système) et 500 points d'intérêt sont détectés sur chacune des images d'une résolution de 752×480. Les temps relatifs des étapes nécessaires à la reconstruction incrémentale sont répartis comme suit :

– 40% pour la détection de points d'intérêt,
– 15% pour les appariements,
– 10% pour le calcul de pose,
– 35% pour l'ajustement de faisceaux incrémental.

3 SLAM et plan contraints

Cette partie détaille les modifications apportées au $SLAM$ pour ajouter une contrainte planaire ainsi que l'optimisation du plan Π par l'ajustement de faisceaux. Les différentes sections sont :

– la détection automatique du plan dans le processus de reconstruction,
– la définition de la contrainte planaire,
– l'optimisation des paramètres du problème,
– les performances avec un résultat d'application.

Les contributions apportées sont :

– une contrainte planaire invariante au facteur d'échelle,
– un ajustement incrémental de faisceaux et de plan.

3.1 Initialisation du plan

Le plan est initialisé au cours du $SLAM$ à partir des mises en correspondance de points d'intérêt issus des deux dernières images clés. Comme le montrent Pritchett *et al.* [10], les points d'intérêt appartenant à un même plan de l'espace définissent une même homographie entre deux images. C'est pourquoi, trouver le plan associé au plus grand nombre de points d'intérêt revient à trouver l'homographie correspondant à un maximum d'appariements entre deux images. Ceci se fait par un processus RANSAC [1]. Puis, à l'aide des points 3D issus du $SLAM$, et associés au plus grand nombre d'appariements satisfaisant une même homographie, on calcule l'équation du plan avec une décomposition en valeurs singulières [9]. De plus, en faisant l'hypothèse que le plan obtenu correspond à un réel objet planaire de la scène, on peut supposer que la taille de celui-ci est finie. C'est pourquoi, on choisit de limiter la dimension du plan au disque englobant les projections orthogonales des points 3D sur le plan, comme illustré sur la figure 2. Le diamètre du disque ainsi calculé n'est pas remis en cause par la suite. On suppose ici que la surface du plan initial est assez proche de l'optimal donc que celle-ci ne peut pas varier de manière significative. Si la détection de plan échoue, une nouvelle tentative est faite au prochain ajout d'image clé.

3.2 Contrainte planaire

L'objectif d'une contrainte planaire est de réduire la distance 3D entre un plan estimé et les points 3D supposés appartenir à ce plan. Ceci implique que le processus d'optimisation combine des erreurs pixelliques (erreurs de reprojection) et des erreurs métriques (distances 3D point-plan). L'homogénéisation de ces données nécessite la pondération des erreurs 3D par un coefficient σ_Π. Toutefois, il est nécessaire

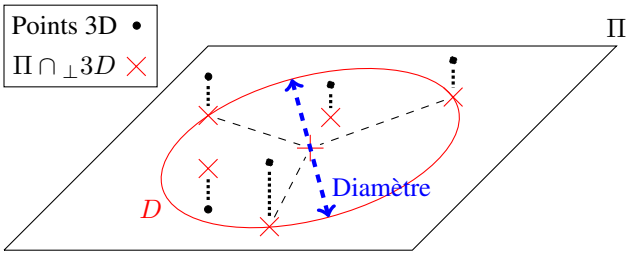


FIGURE 1 – D correspond au disque englobant les projections orthogonales des points 3D sur le plan Π.

que σ_Π soit invariant au facteur d'échelle de la reconstruction donc estimé en fonction d'un objet 3D présent dans la scène. Ainsi, σ_Π est choisi en fonction du diamètre du disque englobant les projections orthogonales des points 3D (voir section 3.1) :

$$\sigma_\Pi = D_\Pi \times \lambda_\Pi \tag{5}$$

avec D_Π le diamètre du disque. Le paramètre λ_Π correspond à la force de la contrainte que l'on souhaite appliquer selon l'application visée. En pratique, on prend $\lambda_\Pi = 2\%$, ce qui signifie que la contrainte tolère un écart au plan correspondant à 2% de la taille de celui-ci. Si la contrainte planaire devait s'appliquer à une surface sur laquelle des objets non planaires seraient présents, on utiliserait une valeur de λ_Π plus élevée afin d'augmenter la tolérance aux points 3D distants du plan.

3.3 Ajustement incrémental de faisceaux et de plan

Pour intégrer la contrainte planaire à l'ajustement de faisceaux, on ajoute à la fonction d'erreur du $SLAM$ vue dans la section 2.4 un terme correspondant à la distance orthogonale $d_\perp(\Pi, P_q^\Pi)$ entre le plan Π et l'ensemble P^Π des points 3D associés à Π. Un point 3D P_q^Π est associé au plan Π si tous les faisceaux issus des points d'intérêt de P_q^Π intersectent le disque du plan. On suppose ici que le plan est opaque, et qu'aucun objet ne se trouve entre la caméra et le plan. La fonction à minimiser dépend donc des K erreurs de reprojection et des Q erreurs 3D de P^Π :

$$\mathcal{E} = \sum_{k=0}^{K} \| \frac{p_k - \pi(C_{p_k}, P_{p_k})}{\sigma_H} \|^2 + \sum_{q=0}^{Q} \| \frac{d_\perp(\Pi, P_q^\Pi)}{\sigma_\Pi} \|^2 \tag{6}$$

avec σ_Π la pondération sur l'erreur 3D définie dans la section 3.2. La contrainte planaire est paramétrée par les trois composantes du vecteur normal unitaire du plan. La matrice jacobienne de la fonction d'erreur $\mathcal{E}$ s'écrit :

$$J = \begin{bmatrix} J_c & 0 & J_p \\ 0 & J_\Pi & J_{p\Pi} \end{bmatrix} \tag{7}$$

avec respectivement J_c et J_p les jacobiennes des erreurs de reprojection des poses caméras et des points 3D, et J_Π et $J_{p\Pi}$ les jacobiennes de l'erreur 3D entre le plan et les points 3D subissant la contrainte planaire. L'ajustement de faisceaux est classiquement utilisé avec deux familles de paramètres : les poses et les points 3D. Dans notre application, nous disposons d'une nouvelle famille de paramètres, constituée des paramètres du plan. L'équation de Levenberg-Marquardt vue dans la section 2.5 reste valable. Toutefois, une implémentation efficace du problème nécessite d'exploiter les *a priori* liés à la structure éparse de la jacobienne. C'est pourquoi, on utilise un ajustement de faisceaux spécifique à trois familles de paramètres développé par Lébraly *et al.* [5].

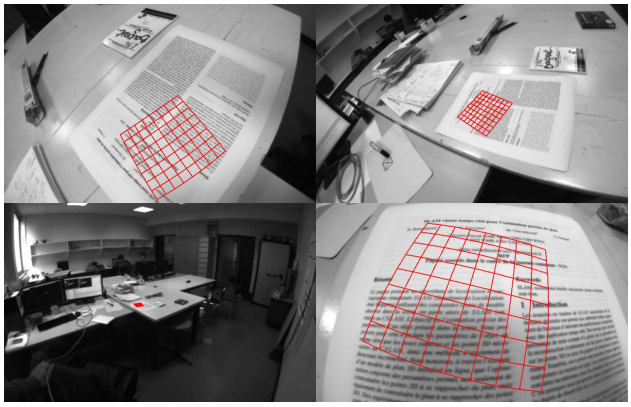


FIGURE 2 – Illustration du $CSLAM$: le quadrillage rouge correspond au plan optimisé et projeté dans l'image.

3.4 Performances de l'implémentation

La figure 3 illustre le résultat de notre approche avec le plan projeté dans l'image sous la forme d'un quadrillage rouge. Le temps d'exécution de l'ajustement de faisceaux avec optimisation du plan est approximativement supérieur de 20% à l'ajustement de faisceaux sans contrainte. Cette augmentation est due au calcul des résidus et des dérivées de la contrainte planaire, ainsi qu'à l'augmentation du nombre de paramètres du problème.

4 Résultats et comparatifs

4.1 Méthodologie

A travers des expérimentations sur données de synthèse et réelles, nous voulons montrer que l'ajustement de faisceaux contraint avec optimisation du plan, que l'on notera AF_Π, est plus précis et offre une meilleure vitesse de convergence que l'ajustement de faisceaux contraint sans optimisation du plan $AF_{\overline{\Pi}}$. C'est pourquoi, un comparatif est effectué sur des critères de précision et de temps d'exécution entre les approches AF_Π et $AF_{\overline{\Pi}}$. Les tests sont réalisés avec la configuration matérielle décrite dans la section 2.6.

4.2 Comparaison sur données de synthèse

Différence de comportement avec et sans optimisation du plan. La figure 4 se compose de 4 visualisations 3D de données de synthèse constituées de 10 poses caméras

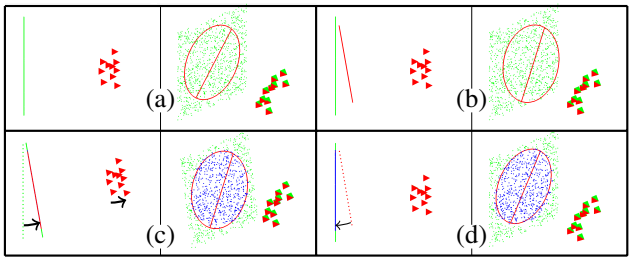
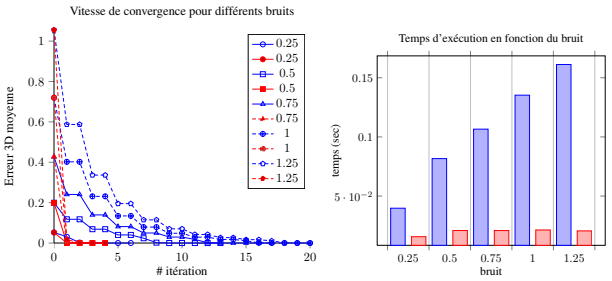


FIGURE 3 – Visualisation du comportement de l'ajustement de faisceaux : (a) le jeu de données original, (b) le jeu de données bruité, (c) le résultat de l'ajustement de faisceaux contraint sans optimisation du plan sur le jeu de donnée de (b), et (d) le résultat de l'ajustement de faisceaux avec optimisation du plan sur le jeu de donnée de (b). Les quatre images se compose d'une vue de profil à gauche et une vue en plongée à droite.



(a) (b)

FIGURE 4 – (a) présente la variation de l'erreur 3D avec la méthode AF_{Π} en rouge et $AF_{\overline{\Pi}}$ en bleu et (b) correspond aux temps d'exécution des tests de (a).

(blocs pyramidales rouges et verts), 1000 points 3D (points verts), et un plan dont le disque et le diamètre sont représentés par le cercle et le segment rouge. Chacune des 4 images de la figure visualise le jeu de données selon une vue de profil à gauche et une vue en plongée à droite. L'image 4(a) correspond aux données de synthèse générées aléatoirement, telles que les caméras observent l'ensemble des points 3D (on a donc 10k points d'intérêt), et que ceux-ci appartiennent au même plan. On limite la taille du disque associé au plan de telle sorte que la moitié des points 3D subissent la contrainte planaire. L'image 4(b) représente le même jeu de données sur lequel on a appliqué un bruit aléatoire sur les paramètres du plan. On observe, sur la vue de profil, que le plan ne coïncide plus avec le nuage de points 3D. L'image 4(c) représente le résultat de la méthode $AF_{\overline{\Pi}}$ sur le jeu de données de 4(b). On remarque que l'ensemble des paramètres de points 3D et de poses ont été modifiés et que le nuage de points coïncide de nouveau avec le plan. On interprète ce résultat de la manière suivante : pour minimiser l'erreur 3D induite par la contrainte planaire, l'optimisation doit réduire l'espace séparant les points 3D associés au disque et le plan en modifiant les paramètres des points 3D. Ceci a pour effet d'augmenter les erreurs de reprojections car les points 3D sont également liés aux poses caméras. Pour minimiser les erreurs de reprojection, l'optimisation modifie les paramètres des points 3D et des poses. Sachant qu'une partie des points 3D subit la contrainte planaire, la solution pour minimiser les erreurs de reprojection, et donc le total des erreurs, est de déplacer les poses caméras. On obtient alors le résultat 4(c) où les points 3D et les poses ont été contraints à se déplacer pour satisfaire la contrainte planaire, alors que seuls les paramètres du plan étaient bruités. L'image 4(d) représente le résultat de la méthode AF_{Π} sur le jeu de données de 4(b). Dans ce cas là, on observe que le plan coïncide de nouveau avec le nuage de points 3D. Ce dernier ne s'est déplacé de manière significative (*idem* pour les poses caméras). Dans cette configuration, la somme des erreurs 2D et 3D diminue plus fortement lorsque l'on modifie les paramètres du plan que lorsque ce sont l'ensemble des paramètres de poses et points 3D qui sont modifiés. Ce sont donc les paramètres du plan qui sont d'abord remis en cause par l'optimisation.

Vitesse de convergence. La figure 5 présente le comparatif chiffré du test vu à la section 4.2. La figure 5(a) illustre la variation de l'erreur 3D moyenne entre les points 3D du disque et le plan en fonction de bruits appliqués aux paramètres du plan. Les perturbations sont générées uniformément entre 0.25 et 1.25 par pas de 0.25. L'optimisation s'arrête lorsque l'erreur totale est inférieure à 10^{-10} ou que 25 itérations ont été effectuées. Les courbes rouges et bleues correspondent respectivement aux méthodes AF_{Π} et $AF_{\overline{\Pi}}$. On constate que l'approche AF_{Π} (en rouge) converge plus vite (moins de 5 itérations) que l'approche $AF_{\overline{\Pi}}$. On note également qu'avec un bruit faible, la différence entre les deux méthodes tend à être négligeable. La figure 5(b) présente le temps de calcul nécessaire aux optimisations selon les deux méthodes. On note une augmentation significative du temps de calcul de l'approche $AF_{\overline{\Pi}}$ lorsque le bruit croît, tandis que le temps de calcul de la méthode AF_{Π} n'augmente que très peu. Cette différence s'explique par une convergence plus rapide de l'approche AF_{Π} comme le montre la figure 5(a).

4.3 Comparaison sur données réelles

La figure 6 illustre la précision du $CSLAM$ à travers le recalage d'un objet virtuel sur un objet réel. L'objet réel est un motif planaire constitué de deux carrés concentriques et de même orientation. Des amers visuels sont répartis sur les côtés des carrés afin de disposer de points d'accroche. On dispose également d'un modèle d'objet virtuel correspondant au même objet double carré modélisé par 8 segments à un facteur d'échelle prêt. Au cours du $SLAM$, on définit une contrainte planaire initialisée approximativement sur l'objet réel contraignant alors dans un même plan les points 3D du motif. Puis, afin d'évaluer visuellement la précision de l'estimation du nuage de points 3D contraints, on recale le modèle d'objet virtuel correspondant au motif sur le nuage de points 3D (par décomposition en valeurs singulières pour que l'objet virtuel se confonde au mieux avec le nuage de points). On utilise alors la position cou-

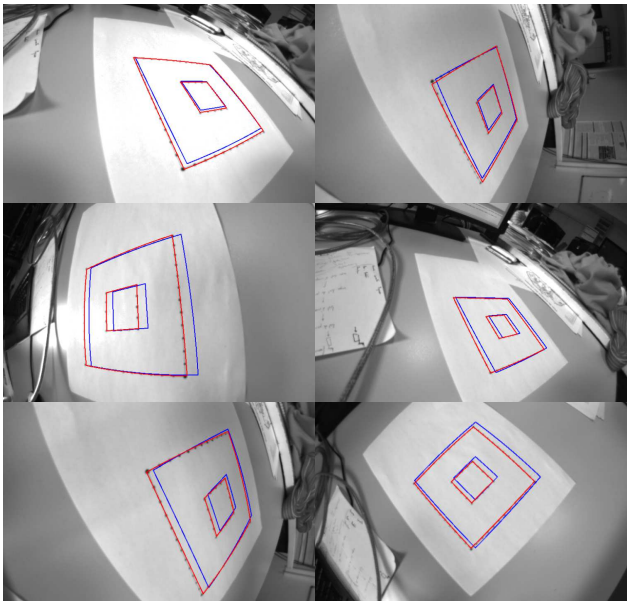


FIGURE 5 – Recalage d'un objet virtuel sur le nuage de points 3D du plan : les approches AF_{Π} et $AF_{\overline{\Pi}}$ sont respectivement en rouge et bleu.

rante de la caméra pour projeter l'objet virtuel dans l'image et évaluer visuellement sa proximité avec l'objet réel. La figure 6 illustre la différence entre les deux approches. On remarque que, suite à une mauvaise estimation du plan, le motif bleu issu de l'approche $AF_{\overline{\Pi}}$ est décalée par rapport au motif réel. Le motif rouge correspondant à l'approche AF_{Π} est quant à lui correctement recalé car la contrainte planaire a été optimisée pour correspondre aux points 3D du motif.

5 Conclusion

L'utilisation de contraintes dans l'ajustement de faisceaux permet d'améliorer la précision des méthodes de $SLAM$. Nous avons proposé un ajustement de faisceaux pour optimiser ces contraintes afin que l'ensemble des paramètres puissent converger vers une meilleure solution. Les tests de synthèse ont montré une convergence plus rapide et un temps d'exécution plus faible lorsque les paramètres du plan étaient remis en cause dans le processus d'optimisation. L'apport de la méthode a également été illustré sur des données réelles. L'optimisation conjointe des paramètres offre un résultat visuel précis pour des applications de recalage d'objets avec une contrainte planaire estimée automatiquement au cours du $SLAM$.

Remerciement

Ce travail est cofinancé par l'Union européenne. L'europe s'engage en Auvergne avec le Fonds européens de développement régional.

Références

[1] M. A. Fischler and R. C. Bolles. Random sample consensus : a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 1981.

[2] C. Harris and M. Stephens. A combined corner and edge detector. In *Alvey vision conference*. Manchester, UK, 1988.

[3] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*, volume 2. Cambridge Univ Press, 2000.

[4] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *Mixed and Augmented Reality, ISMAR*. IEEE, 2007.

[5] P. Lébraly, E. Royer, O. Ait-Aider, C. Deymier, and M. Dhome. Fast calibration of embedded non-overlapping cameras. In *Robotics and Automation (ICRA)*. IEEE, 2011.

[6] G. H. Lee, F. Fraundorfer, and M. Pollefeys. Mav visual slam with plane constraint. In *Robotics and Automation (ICRA)*. IEEE, 2011.

[7] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Real time localization and 3d reconstruction. In *Computer Vision and Pattern Recognition*. IEEE, 2006.

[8] D. Nistér. An efficient solution to the five-point relative pose problem. *Pattern Analysis and Machine Intelligence*, 26(6) :756–770, 2004.

[9] W. H. Press. *Numerical recipes in Fortran 77 : the art of scientific computing*, volume 1. Cambridge university press, 1992.

[10] P. Pritchett and A. Zisserman. Wide baseline stereo matching. In *ICCV. Sixth International Conference on*, pages 754–760. IEEE, 1998.

[11] E. Royer, M. Lhuillier, M. Dhome, and J.-M. Lavest. Monocular vision for mobile robot localization and autonomous navigation. *International Journal of Computer Vision*, 74(3) :237–260, 2007.

[12] M. Tamaazousti, V. Gay-Bellile, S. Collette, S. Bourgeois, and M. Dhome. Nonlinear refinement of structure from motion reconstruction by taking advantage of a partial knowledge of the environment. In *CVPR*. IEEE, 2011.