Longueur de pas algébrique pour l'ajustement de faisceaux

J. Michot[1] A. Bartoli[2] F. Gaspard[1] J.M. Lavest[2]

[1] CEA, LIST, Boîte Courrier 94, Gif-sur-Yvette, F-91191 France ;

[2] LASMEA, UMR 6602 CNRS/UBP, Clermont-Ferrand

julien.michot@cea.fr

Résumé

Nous proposons une nouvelle technique de recherche de la longueur optimale de pas (Line Search) pour l'ajustement de faisceaux. L'idée principale est de déterminer la longueur idéale du pas de déplacement par une approximation de l'erreur de reprojection, en substituant la distance euclidienne par la distance algébrique. Notre méthode est comparée avec plusieurs algorithmes de minimisation non-linéaires dans différentes configurations, sur des données synthétiques et réelles. Elle améliore la convergence de la minimisation de manière efficace et rapide.

Mots clefs

Line Search, moindres carrés non-linéaires, ajustement de faisceaux.

1 Introduction

Depuis quelques décennies les travaux de recherche dans le domaine de la vision par ordinateur se sont fortement orientés vers les problèmes de reconstruction 3D. Dans de nombreux problèmes liés à l'image ou à la vidéo, le calcul de la structure 3D de la scène filmée ainsi que de la localisation de la caméra est essentiel. Ce calcul, effectué à partir des images 2D, est connu sous le nom de *Structure-from-Motion* (SfM). Une bonne introduction au problème de la reconstruction 3D à partir d'une séquence d'images est donné par Pollefeys *et al.*, voir [1].

L'ajustement de faisceaux est une optimisation non-linéaire couramment employée dans les problèmes de reconstruction pour raffiner un modèle initial. Il s'agit d'optimiser directement un ensemble de paramètres modélisant la structure 3D d'une scène (primitives 3D), les positions et orientations des caméras (paramètres extrinsèques), et éventuellement les paramètres intrinsèques des caméras (focale, etc.). Le but est de minimiser la distance entre les éléments détectés dans les images et leurs reprojections théoriques définies par le modèle, voir [2]. Les algorithmes de minimisation non-linéaires sont habituellement utilisés dans ce contexte, par exemple Levenberg-Marquardt [3, 4]. Depuis quelques années des stratégies ont été développées afin de rendre l'ajustement de faisceaux local. Le raffinement n'est ainsi effectué que sur les paramètres les plus récents de la scène. De cette manière, il est aujourd'hui possible de faire appel à cet ajustement de faisceaux adapté dans des applications temps-réel [5]. Il devient donc primordial de développer des stratégies de minimisation rapides et efficaces.

L'un des inconvénients majeurs des méthodes de minimisation itératives est qu'elles ne fournissent qu'une direction de déplacement dans l'espace des paramètres, et non la longueur optimale du pas à réaliser. L'objectif des techniques de *Line Search* est donc de déterminer la longueur de pas la plus efficace, à chaque itération de l'algorithme. De nombreux travaux ont été effectués sur cette recherche (entre autres [6, 7, 8, 9, 10]). La plupart de ces méthodes sont itératives et sont particulièrement coûteuses en temps de calcul.

Nous proposons dans cet article une nouvelle technique de Recherche de la longueur de pas non itérative, pouvant être rapidement implantée dans un ajustement de faisceaux traditionnel. En approximant l'erreur de reprojection par une distance algébrique, nous sommes en mesure de calculer une longueur efficace du pas de déplacement. Nos expérimentations montrent que cette longueur, optimale en distance algébrique, est de plus performante en distance euclidienne.

Organisation de l'article. La section 2 définie les problèmes de *Structure-from-Motion* et d'ajustement de faisceaux. Nous décrivons ensuite notre approche, la longueur de pas algébrique. Enfin, la dernière section présente les résultats de la méthode sur des données réelles et simulées.

Notations. Nous adoptons les notations suivantes : les scalaires sont en italique, *e.g.* x, les vecteur en gras, *e.g.* $\mathbf{p}$ $\mathbf{P}$ et les matrices en caractères sans sérif *e.g.* M. I_3 est la matrice identité 3×3.

2 Contexte et travaux antérieurs

2.1 *Structure from Motion*

Dans l'espace projectif, les matrices 3×4 de projection des caméras sont notées P_i, avec $i = 1 \dots n$. Elles peuvent être décomposées en métrique en $\mathsf{P}_i^M = \mathsf{P}_i\mathsf{H} = \mathsf{K}_i\mathsf{R}_i(\mathsf{I}_3|-\mathbf{t}_i)$, où K_i contient les paramètres intrinsèques de la caméra i et R_i et $\mathbf{t}_i$ représentent l'orientation et la position des caméras dans le repère global. H est une homographie permettant de

passer d'un espace projectif à un espace métrique.
Le problème de reconstruction consiste à déterminer de manière précise toutes les positions et orientations des caméras, modélisées dans P_i^M - ou dans P_i en projectif - ainsi que l'ensemble des points 3D $\mathbf{Q}j$ de la scène. Une première solution est généralement déterminée à l'aide du calcul des matrices fondamentales [11], puis raffinée par un ajustement de faisceaux. Dans cet article nous considérons les cas projectif et métrique. Pour ce dernier, nous supposons que les matrices de paramètres intrinsèques des caméras sont constantes, connues et identiques : $\mathsf{K}_i = \mathsf{K}$.

Ajustement de faisceaux. A partir d'une première estimation de la structure de la scène et des positions et orientations des caméras, l'ajustement de faisceaux tente de raffiner le modèle en faisant appel à un algorithme d'optimisation non-linéaire, par exemple la méthode Levenberg-Marquardt. Le modèle à optimiser (vecteur $\mathbf{x}$ des p paramètres de l'optimisation) est habituellement composé des matrices P_i ou P_i^M et des points 3D $\mathbf{Q}_j$. Il peut toutefois contenir les paramètres intrinsèques de K et l'on parle alors d'auto-calibrage. Lorsque l'on réalise un ajustement de faisceaux projectif, l'optimisation du modèle des caméras ne s'effectue qu'à une homographie près. Il faut donc ensuite déterminer l'homographie rectificative H pour revenir à un modèle métrique.

Fonctions de coût. Dans l'ajustement de faisceaux, la fonction de coût $f(\mathbf{x})$ associée à l'algorithme d'optimisation calcule l'erreur de reprojection. Il s'agit d'une minimisation au sens des moindres carrés de la distance entre les observations 2D (mesures sur les images) et les reprojections théoriques. La caractérisation de la distance peut varier. La fonction classique modélisant l'erreur géométrique est la suivante :

$$\varepsilon_{euc} = f(\mathbf{x}) = \sum_{i,j} \left\| \mathbf{q}_{ij} - \Psi\left(\mathsf{P}_i \mathbf{Q}_j\right) \right\|^2 \qquad (1)$$

où $\mathbf{q}_{ij}$ est l'observation du point 3D $\mathbf{Q}_j$ sur l'image (caméra) P_i et $\Psi(\mathbf{q}) = \frac{1}{q_3}\begin{pmatrix} q_1 \\ q_2 \end{pmatrix}$
Une autre fonction de coût proposé dans la littérature [12] est basée sur la distance algébrique suivante :

$$d_{alg} = \left\| \mathsf{S}[\mathbf{q}_{ij}]_\times \mathsf{P}_i \mathbf{Q}_j \right\|^2 \qquad (2)$$

avec $\mathsf{S} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$ et $[\mathbf{q}_{ij}]_\times$ est la matrice associée au produit vectoriel.
Il est convenu [13] que sous une normalisation appropriée des données, les fonctions de coût algébriques donnent des résultats satisfaisants, eu égard au fait que cette distance ne possède pas réellement de signification géométrique ou statistique. La fonction de coût algébrique s'écrit alors :

$$\varepsilon_{alg} = \sum_{i,j} d_{alg}(\mathbf{q}_{ij}, \mathsf{P}_i \mathbf{Q}_j)^2 = \sum_{i,j} \left\| \mathsf{S}[\mathbf{q}_{ij}]_\times \mathsf{P}_i \mathbf{Q}_j \right\|^2 \qquad (3)$$

2.2 Techniques d'optimisation non-linéaire

Optimisation non-contrainte. Dans la suite de ce document, nous considérons le problème suivant :

$$\min_{\mathbf{x} \in \mathbb{R}^p} f(\mathbf{x}). \qquad (4)$$

où $f : \mathbb{R}^p \to \mathbb{R}$ est une fonction non-convexe, continue et dérivable deux fois. On recherche alors $\mathbf{x}^*$, l'optimum de f, tel que :

$$\nabla f(\mathbf{x}^*) = 0. \qquad (5)$$

En pratique, cette équation possède plusieurs solutions. Le problème est habituellement résolu par un algorithme itératif de minimisation qui génère une série de déplacements $\mathbf{x}_0, \mathbf{x}_1, \ldots \mathbf{x}_k$ pour lequel $f(\mathbf{x}_k) \xrightarrow[k\to\infty]{} f(\mathbf{x}^*)$. La séquence $\mathbf{x}_k$ est déterminée par :

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \boldsymbol{\delta}_k, \qquad (6)$$

où $\boldsymbol{\delta}_k$ est la direction proposée par l'optimisation non contrainte à l'itération k et α_k la longueur du pas du déplacement à entreprendre dans cette direction.
Plusieurs stratégies ont été proposées afin de minimiser une fonction de coût non-linéaire (méthodes de (quasi-)Newton [7], de descente de gradient, Levenberg-Marquardt [13, 3, 14, 5], des régions de confiance de Powell [15, 9, 16], ...)

Levenberg-Marquardt. La technique de minimisation Levenberg-Marquardt (LM) est une combinaison des méthodes de Descente de Gradient et de Gauss-Newton. L'algorithme détermine le déplacement $\boldsymbol{\delta}_k$ en résolvant l'équation suivante :

$$\left(\mathsf{J}_k^\top \mathsf{J}_k + \lambda_k \mathrm{I}_3\right) \boldsymbol{\delta}_k = -\mathbf{g}_k \qquad (7)$$

où $\mathsf{J}_k = \frac{\partial f}{\partial \mathbf{x}_k}(\mathbf{x}_k)$, $\mathbf{g}_k = \mathsf{J}_k^\top f(\mathbf{x}_k)$ et λ_k est le paramètre de *damping*. C'est ce dernier paramètre qui autorise l'algorithme LM à choisir le comportement de minimisation le plus adapté, suivant le contexte de la solution courante, entre la Descente de Gradient et la méthode de Gauss-Newton.
Plusieurs heuristiques de mise à jour de ce paramètre existent. Lors de nos expérimentations, nous avons choisi les règles décrites par Hartley et Zisserman dans [13] ainsi que les méthodes proposées par Nielsen dans [14].

Recherche de la longueur de pas (*Line Search*). Puisque la minimisation de Levenberg-Marquardt ne s'attache pas à calculer la distance optimale du déplacement à réaliser suivant la direction $\boldsymbol{\delta}_k$, les techniques de *Line Search* tentent de déterminer cette longueur de pas minimisant le plus efficacement possible la fonction objective. Il existe deux types de *Line Search*, les méthodes exactes et inexactes.
Les méthodes exactes sont définies par :

$$\alpha_k = \arg\min_{\alpha_k > 0} f(\mathbf{x}_k + \alpha_k \boldsymbol{\delta}_k). \qquad (8)$$

Malheureusement la majorité des problèmes de vision possède des fonctions objectives f non linéaires et l'équation

(8) devient alors difficile à résoudre. De ce cas, il est beaucoup plus intéressant de faire appel aux algorithmes de recherche inexacte de la longueur de pas.

Pour trouver la meilleure distance de déplacement suivant une direction définie par un algorithme de minimisation non-linéaire, la plupart des méthodes de *Line Search* inexactes s'initialisent sur une première valeur du pas et raffinent ensuite cette proposition itérativement à l'aide de différents critères, comme par exemple les critères de Goldstein ou de Wolfe.

Un des algorithmes les plus courants est l'algorithme de Backtracking. Initialisé sur une première valeur de pas, il génère et évalue ensuite à chaque itération une proposition, et fournit en fin d'execution la meilleure proposition minimisant l'erreur finale.

De nombreuses techniques de *Line Search*, comme notamment [6, 8, 9] ou [10], dérivent de la technique dite du Backtracking. Après avoir réduit un intervalle de recherche, elles se proposent de chercher la meilleur distance de déplacement à l'aide de différentes heuristiques. Frandsen *et al.* [10] proposent une technique de *Line Search* itérative grâce à une interpolation quadratique de la fonction objective. Liu et Nocedal [7] font appel aux critères de Wolfe dans le cadre d'une minimisation *quasi-Newtonienne* limitée (*L-BFGS*), Hager et Zhang dans [17] étudient la convergence de la méthode de gradient conjugué munie d'un *Line Search* basé sur les conditions de Wolfe. Nous renvoyons le lecteur au chapitre de Nocedal et Wright [18] sur les méthodes classiques de *Line Search* couramment employées en optimisation non-linéaire.

Ces différentes techniques de recherche de la longueur de pas sont itératives et, puisque pour chaque proposition la fonction de coût est évaluée, elles ne peuvent être employées dans des algorithmes à contraintes temps-réel. La complexité de la fonction à minimiser dans l'ajustement de faisceaux dépend du nombre de points 3D de la scène ainsi que des caméras du modèle à raffiner. En pratique, leur nombre est important, ces méthodes sont par conséquent peu utilisées dans les ajustement de faisceaux temps-réel.

3 *Line Search* Algébrique (ALS)

Dans cette partie nous présentons notre contribution : utiliser la distance algébrique dans une technique de Recherche de la longueur de pas.

Notre idée **n'est pas** d'utiliser la distance algébrique en tant que fonction de coût de l'ajustement de faisceaux, nous gardons pour cela la distance géométrique. La distance algébrique est employée seulement pour trouver une longueur de pas efficace.

3.1 ALS à une dimension

Une fois que l'algorithme d'optimisation nous fournit une direction ($\boldsymbol{\delta} = \left\{\boldsymbol{\delta}_{\mathsf{P}_i}, \boldsymbol{\delta}_{\mathbf{Q}_j}\right\}$), il reste à définir la distance de déplacement α le long de cette direction. En considérant un nouveau déplacement (équation (6)), l'équation (3) devient

$$\varepsilon_{alg}(\alpha) = \sum_{i,j} \left\| \mathsf{S}[\mathbf{q}_{ij}]_\times (\mathsf{P}_i + \alpha\boldsymbol{\delta}_{\mathsf{P}_i})(\mathbf{Q}_j + \alpha\boldsymbol{\delta}_{\mathbf{Q}_j}) \right\|^2 \quad (9)$$

Puisque nous cherchons le pas α dans la direction $\boldsymbol{\delta}$ qui minimise $\varepsilon_{alg}(\alpha)$, les valeurs optimales sont données par les racines de l'équation :

$$\begin{aligned} \tfrac{\partial \varepsilon_{alg}}{\partial \alpha} = \; & \textstyle\sum_{i,j} \left(\mathsf{S}\left[\mathbf{q}_{ij}\right]_\times \mathsf{P}_i\mathbf{Q}_j\right)^\top \mathsf{S}\left[\mathbf{q}_{ij}\right]_\times (\boldsymbol{\delta}_{\mathsf{P}_i}\mathbf{Q}_j + \mathsf{P}_i\boldsymbol{\delta}_{\mathbf{Q}_j}) \\ & + \Big(\left(\mathsf{S}\left[\mathbf{q}_{ij}\right]_\times \mathsf{P}_i\mathbf{Q}_j\right)^\top \mathsf{S}\left[\mathbf{q}_{ij}\right]_\times \boldsymbol{\delta}_{\mathsf{P}_i}\boldsymbol{\delta}_{\mathbf{Q}_j} \\ & + (\mathsf{S}\left[\mathbf{q}_{ij}\right]_\times \left(\boldsymbol{\delta}_{\mathsf{P}_i}\mathbf{Q}_j + \mathsf{P}_i\boldsymbol{\delta}_{\mathbf{Q}_j}\right))^2 \Big) \alpha \\ & + 3\left(\mathsf{S}\left[\mathbf{q}_{ij}\right]_\times (\boldsymbol{\delta}_{\mathsf{P}_i}\mathbf{Q}_j + \mathsf{P}_i\boldsymbol{\delta}_{\mathbf{Q}_j})\right)^\top \mathsf{S}\left[\mathbf{q}_{ij}\right]_\times \boldsymbol{\delta}_{\mathsf{P}_i}\boldsymbol{\delta}_{\mathbf{Q}_j}\, \alpha^2 \\ & + 2(\mathsf{S}\left[\mathbf{q}_{ij}\right]_\times \boldsymbol{\delta}_{\mathsf{P}_i}\boldsymbol{\delta}_{\mathbf{Q}_j})^\top (\mathsf{S}\left[\mathbf{q}_{ij}\right]_\times \boldsymbol{\delta}_{\mathsf{P}_i}\boldsymbol{\delta}_{\mathbf{Q}_j})\, \alpha^3 \end{aligned} \quad (10)$$

En approximant ainsi la fonction classique de reprojection par une distance algébrique (approximation quadratique) nous simplifions le problème et le rendons résolvable littéralement. La résolution du polynôme (10) de degré 3 en α nous fournit donc une ou trois solutions et, dans la majorité des cas, nous obtenons une seule solution réelle.

Cette méthode de calcul du déplacement - optimale en distance algébrique - est simple, rapide et approxime relativement bien la fonction de coût géométrique. La majorité du temps de calcul de la méthode se situe lors du calcul des coefficients de l'équation (10). Contrairement aux méthodes classiques de *Line Search*, aucune itération n'est nécessaire, ce qui rend cette technique rapide.

3.2 ALS à deux dimensions

Étant donné la nature différente des deux types de données du modèle (les caméras et les points 3D de la scène), il semble intéressant de séparer la recherche de la longueur optimale du pas. Nous cherchons dès lors deux déplacements α_{cam} et α_{str}, les pas optimaux pour respectivement les déplacements des caméras et de la scène (points 3D). Cela revient à déterminer le minimum $\{\alpha^*_{cam}, \alpha^*_{str}\}$ de la fonction objective algébrique :

$$\begin{aligned} & \varepsilon_{alg}(\{\alpha^*_{cam}, \alpha^*_{str}\}) = \\ & \textstyle\sum_{i,j} \left\| \mathsf{S}\left[\mathbf{q}_{ij}\right]_\times (\mathsf{P}_i + \alpha^*_{cam}\boldsymbol{\delta}_{\mathsf{P}_i}) \left(\mathbf{Q}_j + \alpha^*_{str}\boldsymbol{\delta}_{\mathbf{Q}_j}\right) \right\|^2 \end{aligned} \quad (11)$$

La recherche du minimum $\{\alpha^*_{cam}, \alpha^*_{str}\}$ peut être effectuée par un algorithme de minimisation, ou éventuellement de manière analytique. Les résultats présentés dans la section suivante ont été effectués à l'aide de la fonction *fminsearch* de Matlab. Le temps de calcul de la méthode pourrait donc être diminué par la résolution analytique du système.

Notes sur les ajustements métriques. Lors des ajustements métriques, les paramètres des caméras ne sont plus définis par la matrice P_i, mais par des informations d'orientation et de position. Nous utilisons une paramétrisation en petits angles d'Euler $\boldsymbol{\delta}r_i$ (seulement) pour les changements d'orientation des caméras. Dans ce contexte, et afin de conserver la forme de l'équation (9), nous définissons $\boldsymbol{\delta}^k_{\mathsf{P}_i}$ tel que $\boldsymbol{\delta}^k_{\mathsf{P}_i} = \hat{\mathsf{P}}^{k+1}_i - \mathsf{P}^k_i$ avec

$$\hat{\mathsf{P}}^{k+1}_i = \mathsf{K}\mathsf{R}^k_i[\boldsymbol{\delta}\mathbf{r}^k_i]\left(\mathrm{I}_3| - (\mathbf{t}^k_i + \boldsymbol{\delta}\mathbf{t}i^k)\right) \quad (12)$$

est la matrice de la pose i à l'itération k pour une longueur de pas unitaire. Et $[\mathbf{R}] = [(\alpha\ \beta\ \gamma)^\top]$ transforme les angles d'Euler en une matrice de rotation.
Cela signifie que pour les ajustements de faisceaux métriques, nous considérons l'approximation suivante :

$$\mathsf{P}_i^{k+1} \approx (1-\alpha)\mathsf{P}_i^k + \alpha\hat{\mathsf{P}}_i^{k+1} \tag{13}$$

4 Résultats Expérimentaux

Dans cette partie nous présentons les résultats de notre méthode obtenus sur des données réelles et simulées.

4.1 Contexte de l'évaluation

L'ensemble de nos essais ont été réalisés en langage MATLAB. Une normalisation isotropique ([11]) a été appliquée sur chaque observation, puis rectifiée pour l'affichage des résultats (*RMS*).

Heuristiques. Il est important de noter que la distance algébrique (qui n'est pas réellement une distance) n'est qu'une approximation de la distance euclidienne puisque :

$$d_{euc}(\mathbf{X}, \widetilde{\mathbf{X}}) = \frac{d_{alg}(\mathbf{X}, \widetilde{\mathbf{X}})}{z\widetilde{z}} \tag{14}$$

avec $\mathbf{X}^\top = (x, y, z)$ et $\widetilde{\mathbf{X}}^\top = (\widetilde{x}, \widetilde{y}, \widetilde{z})$.
En conséquence, les minima des fonctions de coût associées ne sont pas obligatoirement identiques. L'expérience montre cependant qu'avec une normalisation adaptée, les minima sont toutefois proches. Nous avons donc expérimenté plusieurs heuristiques employant la distance algébrique.
Dans les deux heuristiques mentionnées ci-après (ALS_{1D} et ALS_{2D}), nous procédons à plusieurs tests de vérification. D'une part, nous limitons le domaine de définition de la longueur du pas afin d'éviter les déplacements trop grands ou trop faibles : $\alpha = \min\left(\max\left(\alpha, \alpha_{min}\right), \alpha_{max}\right)$ avec $\alpha_{min} > 0$.
Et d'autre part, nous vérifions que la simple distance de déplacement unité n'est pas meilleure que la longueur de pas définie par l'heuristique.
$\underline{ALS_{1D}} : \alpha = \arg\min_\alpha\left(\varepsilon_{euc}(\alpha = 1), \varepsilon_{euc}(\alpha = \alpha_{ALS})\right)$
$\underline{ALS_{2D}} : \alpha = \arg\min_\alpha(\varepsilon_{euc}(\alpha = 1), \varepsilon_{euc}(\alpha = \{\alpha^*_{cam}, \alpha^*_{str}\}))$ Cette dernière heuristique reprend l'idée d'ALS en deux dimensions de l'équation (11).

Techniques comparées. Pour chaque jeu de données nous comparons notre technique de *Line Search* avec différents algorithmes de minimisation non-linéaires.

1. `BA-Classic` est une implémentation de l'ajustement de faisceaux de [13] utilisant Levenberg-Marquardt
2. `BA-Nielsen` est la technique proposée par Nielsen dans [14]. Celui-ci utilise aussi Levenberg-Marquardt mais avec notamment une règle distincte de mise à jour du paramètre de *damping*.
3. `BA-Powel-Lourakis` est un ajustement de faisceaux basé sur les régions de confiance [16].
4. LS_{FJNT} est l'algorithme de *Line Search* de [10] décrit en section 2.2.

4.2 Données simulées

Nous générons aléatoirement 500 points 3D dans un cube de 6 mètres, et 30 caméras positionnées autour à une distance de 20 mètres par rapport au centre du cube. Les caméras possèdent des paramètres intrinsèques identiques et connus (focale : 1000 pixels, au centre de l'image, les pixels sont carrés), nous appliquons ensuite les projections 2D sur des images (640×480). L'ajout d'un bruit gaussien sur les points 3D et sur les poses des caméras nous fournit notre vecteur initial $\mathbf{x}_0$ de l'ajustement de faisceaux. La Figure 1(a) illustre la configuration de la scène simulée.
Les résultats (Figure 1) montrent que sur des données synthétiques, notre *Line Search* Algébrique choisit une longueur de pas de déplacement efficace puisqu'à chaque itération de l'algorithme de minimisation, l'erreur de reprojection est moins importante que les autres algorithmes. Comme prévu, la technique LS_{FJNT} itérative est beaucoup plus performante que les autres méthodes, mais elle est aussi beaucoup plus lente.

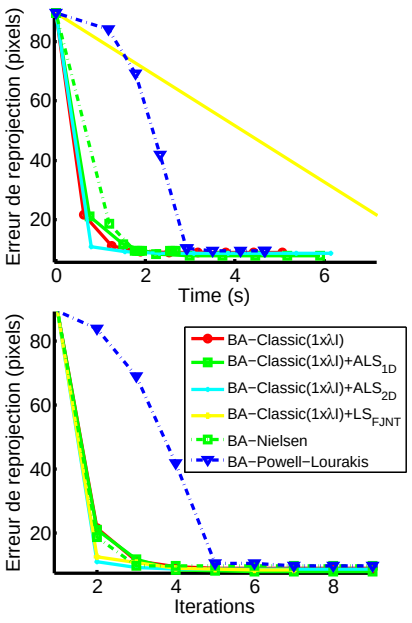
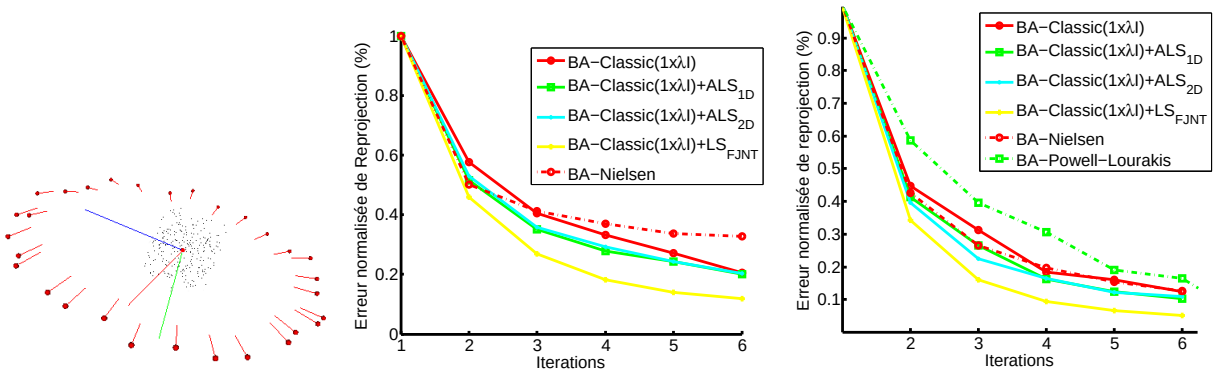


Figure 2 – *RMS (pixels) de reprojection suivant le temps et les itérations, pour un ajustement projectif.*

La Figure 2 présente une comparaison des différents algorithmes de minimisation avec et sans *Line Search* sur un ajustement projectif. La technique de l'ALS en deux dimensions (ALS_{2D}) s'avère être très efficace puisque dès la première itération la minimisation est fortement améliorée. Le gain dans cet exemple est de plus supérieur aux résultats de la méthode LS_{FNJT}. Dans la majorité des tests ef-



(a) (b) (c)

Figure 1 – *Résultats sur les données synthétiques (a). Les figures (b) et (c) montrent l'évolution des RMS moyens normalisés (%) sur 20 simulations, pour des ajustements projectifs (b) et métriques (c).*

fectués sur des données simulées, nos heuristiques de *Line Search* ALS_{1D} et ALS_{2D} ont montré de bons comportements. La technique ALS_{2D} semble cependant être plus efficace que l'ALS_{1D} en projectif. En métrique, les deux méthodes semblent être équivalentes.

4.3 Données réelles

Les jeux de données réelles ont été obtenus à l'aide une caméra de type *pinhole* placée sur un véhicule se déplaçant dans une ville (Figure 4(a)) sur 500 mètres. Un exemple de reconstruction 3D est présenté en Figure 3.

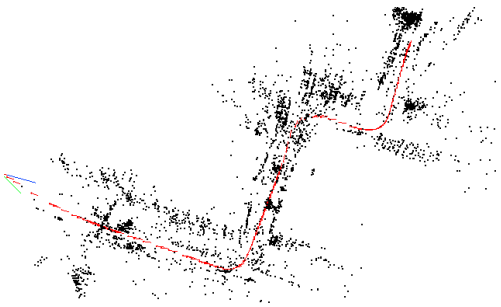


Figure 3 – *Reconstruction 3D de la ville et localisation du véhicule.*

Le calcul initial (précédant l'optimisation) des poses des caméras et de la structure 3D de la scène a été effectué par la méthode du SLAM incrémental temps réel de Mouragnon-Lhuiller [5]. Nous appliquons ensuite un ajustement de faisceaux global, c'est-à-dire sur toutes les données (150 caméras et 8000 points 3D), ou incrémental, i.e. seulement sur les dernières données (les $N = 15$ plus récentes caméras, avec les $M = 300$ derniers points 3D).

Avec des ajustements de faisceaux globaux, il est généralement difficile de trouver une longueur de pas de déplacement satisfaisant l'ensemble des paramètres à optimiser. La distance optimale est en effet souvent proche de l'unité. Nos tests ont cependant montré que nos heuristiques améliorent cependant la convergence de l'optimisation et restent équivalents à un *Line Search* LS_{FJNT} dans le cadre d'ajustements globaux.

Lorsque l'on considère des ajustements hiérarchiques (résultats présentés en Figure 4) le *Line Search* Algébrique apporte réellement un gain à l'optimisation, la minimisation converge plus rapidement.

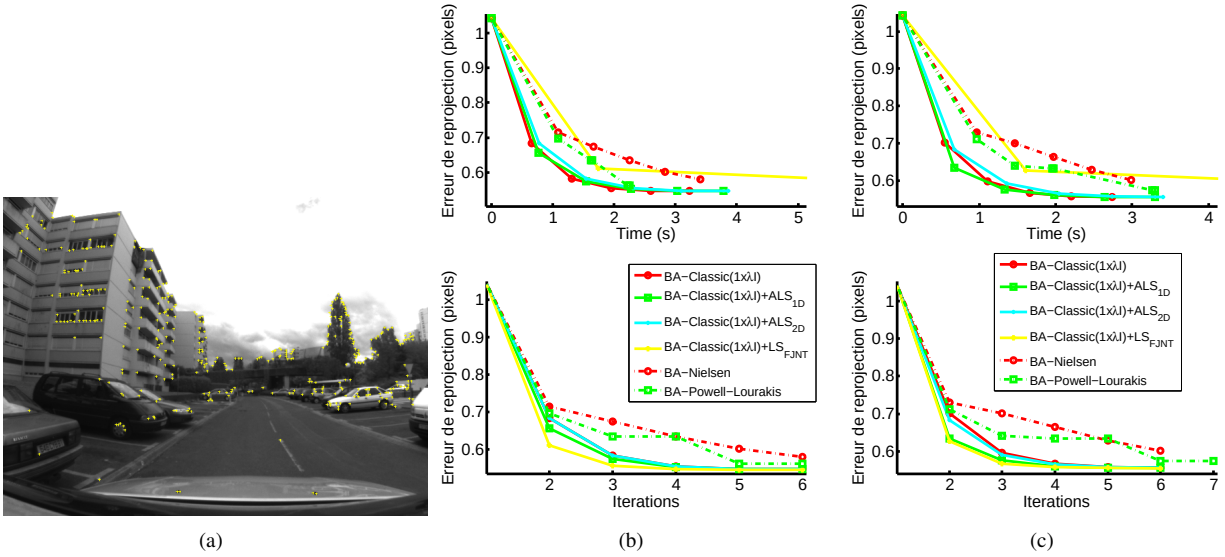
Les heuristiques ALS_{1D} et ALS_{2D} définissent une longueur de pas relativement efficace et minimisent ainsi d'autant plus l'erreur de reprojection. Elles ont de plus le même comportement que le LS_{FJNT} dans les ajustement métriques, mais sont bien plus rapides ! Enfin, l'ALS_{1D} est plus rapide que l'ALS_{2D} comme attendu devant la complexité des méthodes.

5 Conclusion

Cet article propose une nouvelle approche au problème de *Line Search* pour l'ajustement de faisceaux. L'idée principale consiste à utiliser une fonction objective basée sur la distance algébrique seulement lors de la recherche de la longueur du pas. Nos expérimentations, conduites sur des données réelles et synthétiques, montrent qu'à chaque itération de l'algorithme de minimisation, notre proposition fournit une distance de déplacement efficace pour un faible temps de calcul. Cette méthode peut être aisément greffée à des ajustements de faisceaux métriques ou projectifs, mais possède de meilleurs résultats pour ces derniers. Les futurs travaux s'attacheront à entreprendre une résolution exacte de l'ALS en deux dimensions et à expérimenter cette approche en temps réel pour des applications de type SLAM.

Références

[1] M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, et R. Koch. Visual mode-



(a) (b) (c)

Figure 4 – *Résultats sur des données réelles (a). Les figures (b) et (c) montrent l'évolution du RMS (pixels) par rapport au temps de calcul (en haut) et par rapport aux itérations (en bas), pour des ajustements locaux projectifs (b) et métriques (c)*

ling with a hand-held camera. *Int. J. Comput. Vision*, 59(3) :207–232, 2004.

[2] B. Triggs, P. F. Mclauchlan, R. Hartley, et A. W. Fitzgibbon. Bundle adjustment – a modern synthesis. *Lecture Notes in Computer Science*, 1883 :298+, January 2000.

[3] K. Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, Jul. 1944.

[4] D. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal on Applied Mathematics*, 11 :431, 441, 1963.

[5] E. Mouragnon, M. Lhuiller, M. Dhome, F. Dekeyser, et P. Sayd. Generic and real-time structure from motion. Dans *BMVC*, 2007.

[6] M. Al-Baali et R. Fletcher. An efficient line search for nonlinear least squares. *J. Optim. Theory Appl.*, 48(3) :359–377, 1986.

[7] D. C. Liu et J. Nocedal. On the limited memory BFGS method for large scale optimization. *Math. Program.*, 45(3) :503–528, 1989.

[8] J. Moré et D. Thuente. Line search algorithms with guaranteed sufficient decrease. *ACM Trans. Math. Softw.*, 20(3) :286–307, 1994.

[9] J. Nocedal et Y. Yuan. Combining trust region and line search techniques. Rapport technique, Advances in Nonlinear Programming, (Kluwer), 1992.

[10] P. E. Frandsen, K. Jonasson, H. B. Nielsen, et O. Tingleff. *Unconstrained Optimization*. 1999.

[11] R. Hartley. In defence of the 8-point algorithm. Dans *ICCV '95 : Proceedings of the Fifth International Conference on Computer Vision*, page 1064, Washington, DC, USA, 1995. IEEE Computer Society.

[12] R. Hartley. Minimizing algebraic error. Dans *ICCV '98 : Proceedings of the Sixth International Conference on Computer Vision*, page 469, Washington, DC, USA, 1998. IEEE Computer Society.

[13] R. Hartley et A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, Cambridge, UK, second édition, 2003.

[14] H. B. Nielsen. Damping parameter in marquardt's method. Rapport technique, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, apr 1999.

[15] M. J. D. Powell. A hybrid method for nonlinear equations. *Numerical Methods for Nonlinear Algebraic Equations, P. Rabinowitz (Ed.)*, pages 87–114, 1970.

[16] M. Lourakis et A. Argyros. Is levenberg-marquardt the most efficient optimization algorithm for implementing bundle adjustment ? *ICCV*, 2005.

[17] W. Hager et H. Zhang. Algorithm 851 : CG DESCENT, a conjugate gradient method with guaranteed descent. *ACM Trans. Math. Softw.*, 32(1) :113–137, 2006.

[18] J. Nocedal et S.J. Wright. *Numerical Optimization*. Springer Series in Operations Research, New york, 1999.