TIMG2: Vision par ordinateur
Projet de reconstruction 3D incrémentale

Maxime Hurtubise
& Hector Taler-Fraisse

Encadrant :
Guillaume Bourmaud : *gbourmaud@bordeaux-inp.fr*

Summary

1 Introduction

Ce projet s'inscrit dans un ensemble d'étapes de traitement permettant la reconstruction d'une scène 3D à partir d'une vidéo. Les trois étapes principales de la chaine de traîtement nécéssaire sont :

1. La détection des points d'intérêts
2. La mise en correspondance des points d'intérêt entre les images et création de chemin (tracks)
3. L'estimation des points 3D et des poses des caméras

Notre travail se concentre sur l'implémentation de la troisième étape à partir de données préalablement traitées selon les deux premières étapes.
Le processus de cette étape d'estimation de poses des caméras et des coordonnées des points 3D est dit "incrémental" car il débute par une initialisation sur deux images pour ensuite intégrer les vues suivantes une par une.

2 Présentation du principe de l'algorithme

Après une phase d'initialisation spécifique aux deux premières vues, chaque nouvelle image est intégrée au système en trois temps : estimation de sa pose de la nouvelle caméra (Localisation), ajout des nouveaux points dans le modèle de la scène 3D (Triangulation) et correction globale des positions de caméras et des coordonnées des points 3D (Bundle Adjustment) en suivant le schéma suivant.

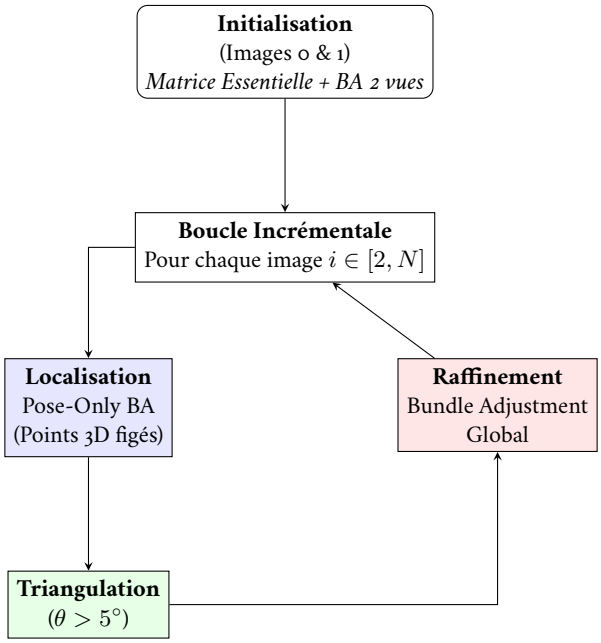


Figure 1: Workflow de la reconstruction incrémentale implémentée.

- **L'initialisation** évalue la matrice essentielle permettant de passer du référentiel de la première caméra à la seconde caméra grâce aux techniques en géométrie épipolaire.
- **La Localisation** remplace l'utilisation de la géométrie épipolaire pour fair une première estimation de la matrice de passage du référentiel monde à celui de la nouvelle caméra.
- **La Triangulation** est effectuée après la localisation ayant fournit une première estimation des matrice de passages. Nous proposons une approche "temporelle" permettant de chercher la meilleure des caméras précédentes avec laquelle effectuer la triangulation pour minimiser les erreure.

- **Le Bundle Adjustment Global** intervient en fin de cycle pour rectifier l'erreur de reprojection sur l'ensemble des paramètres de toutes les étapes précédentes (poses et points). Cela permet d'éviter l'accumulation d'erreurs au fur et à mesure que le nombre d'images augmente.

3 Structure des données

Pour gérer la reconstruction, le script s'appuie sur plusieurs structures de données Python essentielles qui permettent de relier :

- L'indice du point 2D de l'image j (`p2D_id`$_j$) à ses coordonnées 2D dans l'image j.
- L'indice du point 3D (`p3D_id`) de la scène à ses coordonnées.
- Les indices des points 2D à leurs indices de point 3D correspondant.

Table 1: Description des structures de données principales du projet

Variable	Type / Format	Objectif et contenu
`p`	Liste de tableaux *numpy*	Stocke l'ensemble des coordonnées 2D (x, y) des points d'intérêt détectés pour chaque image.
`tracks_full`	Liste de dictionnaires	Contient l'intégralité des correspondances pré-calculées : lie chaque point d'intérêt 2D (`p2D_ids`) à sa clé de trace 3D unique (`p3D_keys`).
`tracks`	Liste de dictionnaires	Objet dynamique de structure identique à tracks_full mais regroupant uniquement les points 2D et leurs clés 3D pour les points **déjà reconstruits** dans chaque vue.
`Uw`	Matrice $N \times 3$	Représente le nuage des N points 3D de la scène (coordonnées X, Y, Z dans le référentiel monde).
`Mwc`	Liste de matrices 4×4	Stocke la pose de chaque caméra composée de la matrice de rotation et du vecteur de translation du référentiel mode vers le référentiel caméra .
`p3D_keys_to_ids`	Vecteur *numpy* d'indexation	Relie la clé `p3D_keys` présente dans les tracks à son indice dans Uw donnant les coordonnées 3D.
`p3D_keys_reconstructed`	Vecteur *numpy*	Liste des clés `p3D_keys` des points 3D ayant été validés par triangulation (ou l'initialisation).

4 Initialisation

4.1 Objectif

La phase d'initialisation constitue le point de départ de la reconstruction 3D. Son objectif est d'estimer la pose relative entre les deux premières caméras ainsi que une première structure 3D à partir de correspondances 2D–2D entre les deux images initiales. Toute erreur à ce stade est propagée dans les phases suivantes de localisation, triangulation et bundle adjustment, ce qui la rend cruciale.

4.2 Explication du code fourni

Les deux premières images sélectionnées sont les images d'indices 0 et 1. À partir des pistes de suivi (*tracks_full*) sont extraits les points 2D correspondant à des clés 3D communes observées dans les deux images. Ces correspondances partagées entre les deux vues sont conservées et constituent l'ensemble des observations nécessaires à l'estimation de la géométrie épipolaire initiale.

Estimation de la pose relative par géométrie épipolaire

La matrice essentielle E_{AB} est estimée à l'aide de l'algorithme des cinq points et à partir des correspondances 2D–2D. La décomposition de cette matrice permet d'obtenir :

- la rotation relative R_{BA}
- la translation relative t_{BA}

Et permet de définir la matrice de projection M_{BA}, qui est telle que :

$$M_{BA} = \begin{pmatrix} R_{BA}(1,1) & R_{BA}(1,2) & R_{BA}(1,3) & t_{BA}(1) \\ R_{BA}(2,1) & R_{BA}(2,2) & R_{BA}(2,3) & t_{BA}(2) \\ R_{BA}(3,1) & R_{BA}(3,2) & R_{BA}(3,3) & t_{BA}(3) \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Triangulation initiale des points 3D

Une fois les matrices de projection des deux caméras définies, les points correspondants sont triangulés à l'aide de la méthode linéaire de triangulation de la bibliothèque **opencv**.
Une première structure 3D est calculée et sera utilisée comme base pour les étapes suivantes de la reconstruction.

Bundle Adjustment sur deux vues

Un bundle adjustment global sur deux vues est appliqué à l'aide de la classe BA_LM_two_views_schur, afin de raffiner à la fois les poses des deux caméras et les positions des points 3D.

Cette étape permet de corriger les imprécisions dues à l'estimation initiale de la matrice essentielle et à la triangulation linéaire. En effet, les erreurs de reprojection chutent significativement sur les deux images initiales après l'optimisation. Les projections des points 3D s'alignent précisément avec les observations 2D (les croix bleus -projections- sont dans les cercles rouges -observations-), validant la cohérence de la géométrie estimée.

C'est cette reconstruction qui sert de point d'ancrage fiable pour mener à bien la suite du projet.

5 Localisation

5.1 Principe

Pour chaque nouvelle image, l'objectif de la phase de localisation est d'estimer précisément la pose de la caméra associée, à partir de la structure 3D du monde qui est déjà partiellement connue.

5.2 Solution Algorithmique

Pour la résolution algorithmique, l'hypothèse fondamentale adoptée est que deux caméras consécutives sont spatialement proches. Cette hypothèse est réaliste dans un contexte d'acquisition séquentielle (vidéo ou déplacement continu de la caméra) et permet d'obtenir une initialisation fiable de la pose de la nouvelle caméra à partir de celle de la caméra précédente.
Pour une nouvelle image d'indice *new_im_id*, on commence par identifier les points 3D déjà reconstruits et visibles dans cette image. Cette étape repose sur une intersection entre :

- les clés 3D déjà reconstruites : *p3D_keys_reconstructed*
- les clés observées dans la nouvelle image : *tracks_full[new_im_id]['p3D_keys']*

```
common_keys, idsReconstruct, idsNew = np.intersect1d(p3D_keys_reconstructed,
tracks_full[new_im_id]['p3D_keys'], assume_unique=False, return_indices=True)
```

Les points 3D correspondants sont ensuite projetés dans le repère caméra initial estimé. Afin de garantir la validité géométrique du problème, seuls les points ayant une profondeur strictement positive dans le repère caméra sont conservés.

```
    assert(np.all(U_c[:,2] > 0.))
```

Cette étape permet d'éliminer les correspondances dégénérées qui pourraient perturber l'optimisation.

En accord avec l'hypothèse fondamentale de proximité spatiale des caméras successives énoncée précedemment, la pose initiale de la nouvelle caméra est directement héritée de la caméra précédente :

```
    Mwc_guess = Mwc[-1]
```

Cette initialisation fournit une approximation grossière mais cohérente de la position et de l'orientation de la caméra. Avant optimisation, l'erreur de reprojection associée à cette pose initiale est calculée et visualisée. On observe fréquemment des erreurs élevées, ce qui confirme la nécessité d'un raffinement précis de la pose.
Afin d'affiner la pose de la caméra, un bundle adjustment local est appliqué à l'aide de la classe BA_LM_localization, que nous avons implémenté comme un cas particulier de BA_LM_schur. Contrairement au bundle adjustment global, cette étape n'optimise que les paramètres de la caméra, à savoir la rotation –R_{wc}– et la translation –t_{wc}–. Nous expliquerons le principe de l'algorithme de Bundle Adjustment dans la partie dédiée dans la suite du rapport.

Les points 3D sont volontairement figés afin d'éviter une dérive de la structure déjà reconstruite en optimisant les points à partir d'une pose de caméra qui est fausse. On peut s'assurer que les points 3D n'ont pas été optimisés grâce aux lignes de code suivantes :

```
    Uw_new = BA_loc.getPointCloud()
    assert(np.allclose(Xw, Uw_new))
```

où *Xw* et *Uw_new* sont les nuages de points 3D avant et après le bundle adjustment, respectivement.

Après convergence du bundle adjustment de localisation, la pose optimisée est utilisée pour recalculer les projections des points 3D dans l'image courante. L'erreur de reprojection moyenne est alors recalculée et comparée à celle obtenue avant optimisation.

5.3 Résultats et Conclusion sur la phase de localisation

Les résultats observés sont particulièrement concluants. En effet, l'erreur de reprojection moyenne par image passe de valeurs parfois élevées –de l'ordre de plusieurs dizaines de pixels– à quelques pixels seulement (Moyenne de l'erreur de reprojection moyenne sur l'ensemble de la reconstruction : 4.63 pix) :

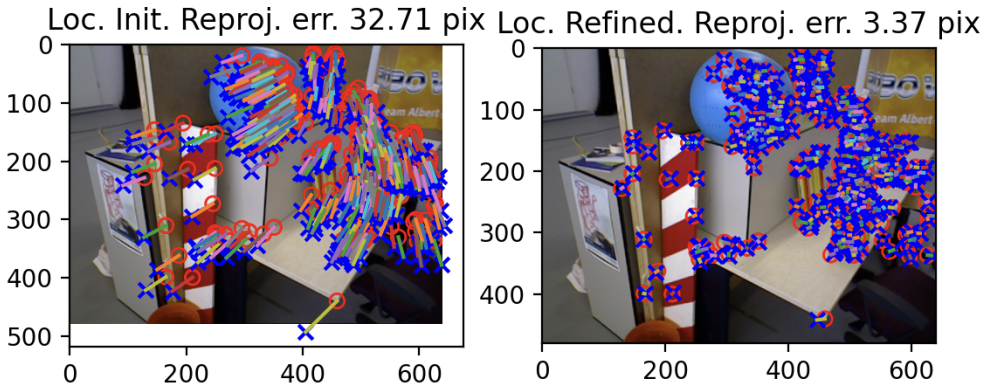


Figure 2: Exemple de l'erreur de reprojection avant (gauche) et après localisation (droite).

Ces résultats confirment l'efficacité du bundle adjustment de localisation pour estimer précisément la pose de la caméra.

6 Triangulation

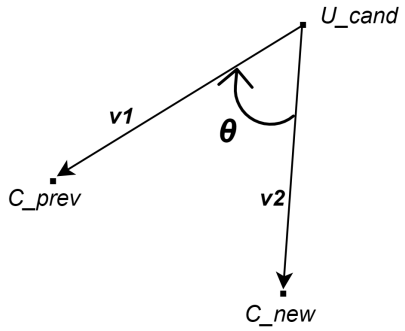
La position de la nouvelle caméra ainsi estimée, elle peut être utilisée pour ajouter de nouveaux points 3D à la scène par triangulation.

6.1 Sélection de la caméra

Pour chaque point d'intérêt non encore reconstruit (points "orphelins"), nous recherchons parmi les caméras précédentes celles qui ont également observé ce point : ce sont les caméras candidates. Puis, parmis ces caméras candidates nous sélectionons celle qui présente le plus grand angle de parallaxe. Finalement, pour garantir la précision de la reconstruction, la triangulation avec cette caméra n'est validée que si l'angle est supérieur à un seuil critique (fixé à $5°$ dans notre implémentation).
Si cette approche permet d'éviter la construction de points de profondeur instable, elle implique aussi que lors d'une itération de triangulation, certains points "orphelins" ne sont pas triangulés ni ajoutés à la scène 3D. Ce comportement ne présente pas de problème puisqu'ils seront triangulés lors de prochaines itérations présentant au moins une caméra voyant ce point 3D et dont l'angle de parallaxe sera assez élevé.

6.2 Calcul de l'angle de parallaxe entre les caméras

Afin de calculer un angle de parallaxe entre deux caméras pour filtrer par précision de triangulation, il nous faut effectuer une triangulation temporaire (donnant *U_cand*) pour obtenir l'angle entre $\vec{v}_1$ et $\vec{v}_2$ comme présenté ci-dessous.



$$\theta = \arccos\left(\frac{\vec{v}_1 \cdot \vec{v}_2}{\|\vec{v}_1\| \, \|\vec{v}_2\|}\right) > 5°$$

7 Bundle Adjustment

7.1 Principe de l'algorithme

L'étape d'ajustement de faisceaux (Bundle Adjustement) permet d'optimiser les poses de toutes les caméras et coordonnées de points 3D estimées dans les itérations précédentes.
Le problème est formulé comme une minimisation de l'erreur de reprojection :

$$L(R_{1w}, t_{1w}, R_{2w}, t_{2w}, \{U_i^w\}_{i=1,\dots,N}) = \sum_{i=1}^{N} \|p_{1,i} - K\,\Pi(R_{1w}U_i^w + t_{1w})\|_2^2 + \|p_{2,i} - K\,\Pi(R_{2w}U_i^w + t_{2w})\|_2^2$$

Avec :

- K : matrice des paramètres intrinsèques de la caméra permettant de passer du plan focal au plan image.
- $\Pi(\cdot)$: fonction de projection perspective d'un point 3D de coordonnées caméra vers le plan image normalisé.
- R_{1w}, R_{2w} : matrices de rotation par rapport au repère monde.
- t_{1w}, t_{2w} : vecteurs de translation par rapport au repère monde.
- U_i^w : coordonnées du i-ème point 3D dans le repère monde.
- $p_{1,i}, p_{2,i}$: coordonnées observées du point i dans les images 1 et 2.

Chaque terme de la somme correspond donc à l'erreur de reprojection du point 3D U_i^w dans l'image associée (ici, images 1 et 2 respectivement).
Cette fonction de coût non linéaire est linéarisée de telle sorte que l'on puisse la ramener à un problème de moindre carré linéaire et appliquer l'algorithme de Gauss-Newton. L'approximation linéaire implique que cet algorithme seul ne garantit pas une baisse la fonction de coût à toutes les itérations. L'algorithme de Levenberg-Marquardt est alors utilisé afin de valider la mise à jours des paramètres uniquement lorsque celle-ci permet faire baisser la fonction de coût sur l'erreur de reprojection citée précédemment.

7.2 Mise en place de l'algorithme

Cette dernière étape de raffinement est triviale mais nécessaire. En effet, il s'agit ici d'appliquer l'algorithme d'ajustement des faisceaux aux poses des caméras ($\{R_{wi}, t_{wi}\}_{i=1\dots new\_im\_id}$) ainsi qu'aux points 3D reconstruits pour les optimiser.

Dans une optique de réduire le temps de calcul de la reconstruction, nous avons décidé de réaliser cette étape de raffinement une fois sur quatre, sinon lorsque l'erreur moyenne de reprojection à l'issue de la localisation est supérieure à un seuil arbitraire (ici 6 pix) :

```
    if(reproj_err_list[-1]>6.0 or new_im_id%4==0):
        BA_global = BA_LM_schur(
            ...
        )
        #...
```

Ce qui nous permet de reconstruire l'ensemble des images en environ 15 minutes (sur un Macde 2022 avec puce M2 Pro).

8 Résultats

Afin d'évaluer qualitativement les résultats de la reconstruction, une visualisation finale a été réalisée à l'aide de la bibliothèque open3d, en exploitant les données sauvegardées à l'issue du pipeline (poses caméra et nuage de points 3D). Cette visualisation est générée par le script *main_show_final_reconstruction.py*.

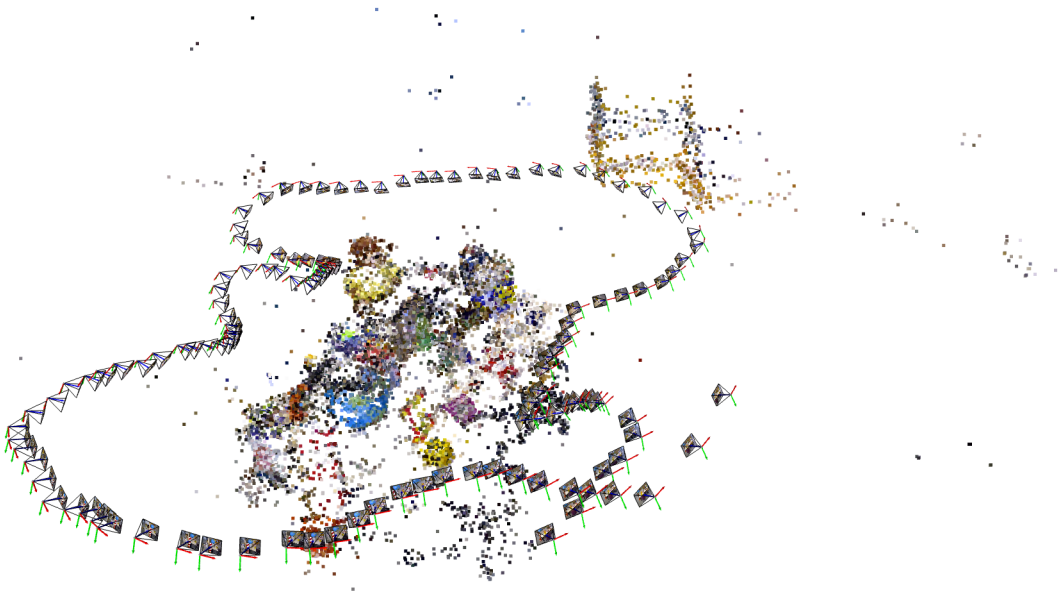


Figure 3: Reconstruction spatiale : nuage de points 3D et poses des caméras estimés

La reconstruction met en évidence une structure 3D cohérente. Les points reconstruits présentent une organisation spatiale conforme à la scène observée et qui correspond à la reconstruction attendue dans le sujet.

9 Conclusion

Dans ce projet, nous avons mis en œuvre et implémenté les principales techniques fondamentales de la vision par ordinateur liées à la reconstruction 3D incrémentale. À partir de correspondances 2D pré-calculées, nous avons estimé les poses des caméras et reconstruit progressivement une structure 3D cohérente, en combinant géométrie épipolaire, triangulation et bundle adjustment. Les résultats obtenus confirment la pertinence de l'approche et la bonne compréhension des concepts abordés dans ce module.