Projet de reconstruction 3D incrémentale

1 Description du projet

À partir de plusieurs images d'une même scène, une méthode de reconstruction 3D vise à obtenir un nuage de points 3D de la scène ainsi que la pose (rotation et translation) de chaque caméra (Fig. 1). Une telle méthode comporte en général trois étapes : une étape de détection de points d'intérêt, une étape de mise en correspondance et de création de *chemins* et finalement une étape d'estimation du nuage de points 3D et des poses.

Dans ce projet, nous considérons le cas où les deux premières étapes ont été effectuées. De plus nous supposerons que ces correspondances **ne contiennent pas de correspondances aberrantes**. L'objectif du projet consiste donc à réaliser la 3ème étape, c'est-à-dire à estimer le nuage de points 3D et les poses des caméras.

(a) (b)

FIG. 1 – Reconstruction 3D : (1a) Exemples d'images en entrée de l'algorithme de reconstruction, (1b) Nuage de points 3D et poses des caméras estimés

2 Reconstruction 3D incrémentale

Une manière d'obtenir une reconstruction 3D à partir de *chemins* consiste à effectuer une reconstruction pour deux images, puis à ajouter les images les unes après les autres. On parle alors de reconstruction 3D incrémentale. Les étapes d'une telle approche sont les suivantes :

1. **INITIALISATION**
   À partir de deux images $\mathtt{I}_1$ et $\mathtt{I}_2$, estimer les poses $\mathtt{R}_{w1}$, $\mathbf{t}_{w1}$, $\mathtt{R}_{w2}$, $\mathbf{t}_{w2}$ ainsi que les coordonnées des points 3D vus dans ces deux images.

Remarque : Dans le code fourni, cette étape est déjà implémentée de la façon suivante - a) estimation de la matrice essentielle et décomposition en pose relative, b) triangulation des points 3D, c) ajustement de faisceaux.

2. Choisir une troisième image $\mathtt{I}_3$ (par exemple l'image suivante dans la liste car, dans notre cas, les images sont issues d'une vidéo, il y a donc du recouvrement entre les images consécutives).
3. **LOCALISATION**
   Estimer $\mathtt{R}_{w3}$, $\mathbf{t}_{w3}$ en minimisant l'erreur de reprojection de cette 3ème caméra par rapport aux points 3D déjà reconstruits.
   Remarque : Les points 3D sont figés durant cette étape. Il s'agit donc ici d'appliquer un algorithme d'ajustement de faisceaux simplifié car uniquement $\mathtt{R}_{w3}$ et $\mathbf{t}_{w3}$ sont optimisés. Les paramètres $\mathtt{R}_{w3}$ et $\mathbf{t}_{w3}$ seront initialisés avec les valeurs de $\mathtt{R}_{w2}$ et $\mathbf{t}_{w2}$.
   Conseil : Avant de passer à l'implémentation de Levenberg-Marquardt pour cette étape de localisation, écrire la fonction de coût que vous souhaitez minimiser.
4. **TRIANGULATION**
   Trianguler les points 3D qui peuvent l'être grâce à l'ajout de cette troisième image.
   Remarque : Vous pourrez vous inspirer de l'implémentation de la partie triangulation présente dans l'étape INITIALISATION.
5. **RAFFINEMENT**
   Appliquer l'algorithme d'ajustement de faisceaux aux poses des 3 caméras ($\{\mathtt{R}_{wi}, \mathbf{t}_{wi}\}_{i=1...3}$) ainsi qu'aux points 3D.
6. Choisir une quatrième image $\mathtt{I}_4$ (par exemple l'image suivante dans la liste car les images sont issues d'une vidéo, il y a donc du recouvrement entre les images consécutives).
7. **LOCALISATION**
   Estimer $\mathtt{R}_{w4}$, $\mathbf{t}_{w4}$ en minimisant l'erreur de reprojection de cette 4ème caméra par rapport aux points 3D déjà reconstruits (les points 3D sont figés durant cette étape).
8. **TRIANGULATION**
   Trianguler les points 3D qui peuvent l'être grâce à l'ajout de cette quatrième image.
9. **RAFFINEMENT**
   Appliquer l'algorithme d'ajustement de faisceaux aux poses des 4 caméras ($\{\mathtt{R}_{wi}, \mathbf{t}_{wi}\}_{i=1...4}$) ainsi qu'aux points 3D.
10. Continuer à ajouter les images une par une en suivant la même logique (Localisation, Triangulation, Raffinement) jusqu'à ce qu'il n'y ait plus d'image à ajouter.

3 Travail à effectuer

L'objectif du projet est de réaliser une reconstruction incrémentale des 127 images fournies, en implémentant les étapes décrites précédemment.
Remarques

— Un code Python vous est fourni. Ce code contient un script principal **où l'étape INITIALISATION est déjà effectuée. Il est fortement conseillé de prendre le temps de bien comprendre ce code car l'implémentation des autres étapes du projet pourra fortement s'en inspirer.**

— Une implémentation fonctionnelle de l'algorithme d'ajustement de faisceaux pour un nombre arbitraire de caméras ('BA_LM_Schur.py') vous est fournie **et ne nécessite pas de modification.**
— Pour mener à bien un tel projet, il est indispensable :
  — de réaliser de nombreux affichages ('print' dans la console et figure matplotlib),
  — et d'employer la fonctionnalité 'assert' de Python.
— Lorsque vous aurez fini d'implémenter et de tester chaque étape, la reconstruction 3D incrémentale des 127 images devrait très fortement ressembler à la figure (1b). Vous pouvez obtenir cette visualisation en exécutant de script 'main_show_final_reconstruction.py'. Cette reconstruction prend plusieurs minutes, une façon de l'accélérer consiste à effectuer l'étape de RAFFINEMENT moins souvent, par exemple uniquement lorsque l'erreur de reprojection à l'issue de la LOCALISATION devient grande.