Estimation de Matrice Fondamental-RANSAC

Jahmal BAPTISTE
Bunthet SAY

Janvier 2020

Contents

1 Introduction

Dans ce TP, nous travaillons sur l'estimation de la matrice fondamentale encodant la géométrie épipolaire entre deux images prises par une caméra en mouvement.

Pour ce faire, nous utilisons d'abord l'algorithme KAZE de OpenCV pour détecter et encoder la description (features) des éléments de chaque image. A partir de ces descriptions, nous utilisons ensuite la méthode FLANN (Fast Library for Approximate Nearest Neighbors) pour trouver les éléments correspondants entre les deux images. Enfin, la méthode RANSAC de OpenCV nous permets d'estimer la matrice fondamental.

Pour utiliser la méthode KAZE, il y a certains paramètres à modifier. Dans la première question, nous allons étudier l'influence de ces paramètres pour avoir une bonne performance de la détection et description des images données.

Dans la deuxième question, nous allons implémenter à la main l'algorithme de RANSAC et régler certains paramètres pour avoir des résultats les plus proches possibles de la fonction utilisé dans la question 1.

Notre travail se trouve dans ce repositoire de GitHub.

2 Question 1: Influence des détection de points d'intérêt

Les descripteurs locaux des pixels des images que nous utilisons ici sont la librairie CV2 et plus partivulièrement de Kaze. Il est important de comprendre les paramètres pour bien réaliser la mise en correspondance des points d'intérêt. Le principe de Kaze est d'attribuer à chaque pixel plusieurs descripteurs locaux, chacun correspondant à un descripteur dans une version filtrée de l'image originale, les filtres ne différant que de l'échelle de filtrage. Faire la correspondance entre deux pixels revient à faire la correspondance entre les multiples descripteurs de ces pixels.

Les paramètres sont :

- `threshold`
  seuil de correspondance minimum pour établir le lien;
- `nOctaves`
  octave maximal d'agrandissement d'échelle des filtres (par rapport à l'échelle minimale);
- `nOctaveLayers`
  nombre de sous-niveau par octave;

- `diffusivity`
  méthode de filtrage pour la création des images.

Les paramètres qui nous semblent être les plus importants pour calibrer l'algorithme de correspondance sont `threshold` et **nOctaves**. En effet `threshold` permet de s'assurer de ne pas faire la correspondance trop facilement et `nOctaves` permet de maîtriser l'echelle de filtrage des images qui seront prises en compte pour le mise en correspondance des points.
Intéressons-nous à l'échelle maximale de filtrage. Si elle est plus grande que l'échelle caractéristique de motifs présents sur l'image originale ces motifs disparaîtront sur l'image filtrée. Les pixels sur les zones de ces motifs se ressembleront alors sur l'image filtrée et la correspondance sera donc faussée.

En ce qui concerne les images du TP, nous avons choisi le jeu de valeur suivant :

- **POP**
  `threshold` = 0.005
  `nOctaves` = 2
  `nOctaveLayers` = 2
  `diffusivity` = KAZE_DIFF_PM_G2

- **DeathStar**
  `threshold` = 0.011
  `nOctaves` = 4
  `nOctaveLayers` = 2
  `diffusivity` = KAZE_DIFF_PM_G2

Ce jeu de valeurs a donné les correspondances décrites par les images en FIGURE 1 pour les images POP01.jpg et POP02.jpg et FIGURE 2 pour les images DeathStar1.jpg et DeathStar2.jpg.

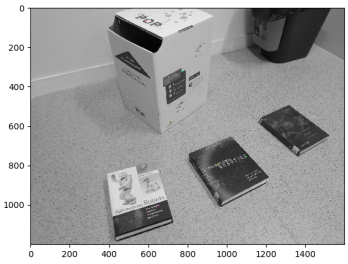
Le choix des autres paramètres s'est fait à tâton.

Les résultats sont les suivants :

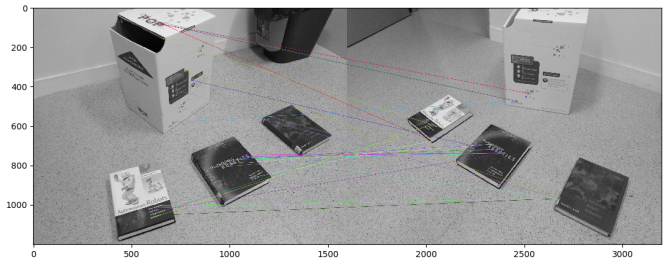
- **POP**
  9 inliers pour 24 correspondances, soit une proportion de **37% d'inliers**;

- **DeathStar**
  13 inliers pour 44 correspondances, soit une proportion de **29% d'inliers**.

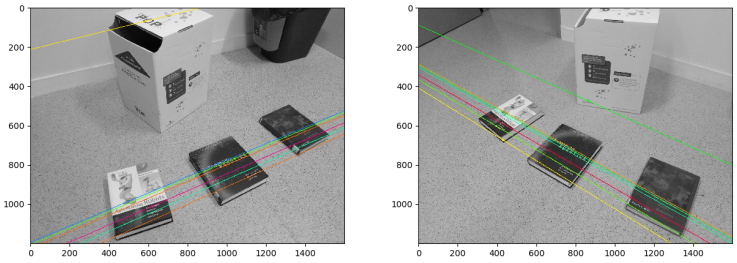
Nous n'avons pas réussi à obtenir un meilleur ratio avec d'autres configurations (en expolrant dans l'espoir de trouver une meilleure combinaison).



(a) Points de référence dans la première image

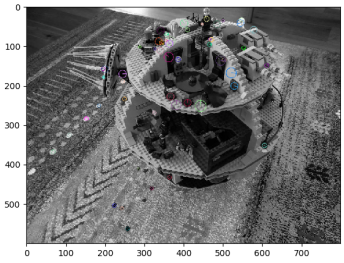


(b) Correspondances dans la deuxième image

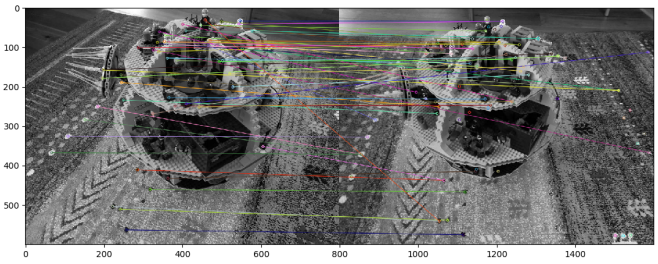


(c) Lignes épipolaires des inliners

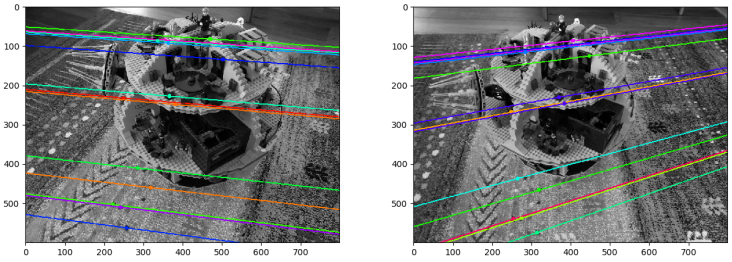
Figure 1: Couple d'images POP01.jpg et POP02.jpg



(a) Points de référence dans la première image



(b) Correspondances dans la deuxième image



(c) Lignes épipolaires des inliners

Figure 2: Couple d'images DeathStar1.jpg et DeathStar2.jpg

3 Question 2: Implémentation de méthode RANSAC

La méthode RANSAC que nous avons implémentée est sous la forme d'une fonction qui renvoie la matrice fondamentale calculée par l'algorithme et l'ensemble des points qui ont donné cette matrice.
Le seul paramètre que nous pensons pouvoir manipuler dans cet algorithme pour améliorer la recherche de matrice fondamentale est le nombre de tours de boucle. L'algorithme prend aussi en compte l'exigence en nombre de pixels de la définition des inliers.

Les indications en nombre de tours de boucle pour une proportion d'inliers initiale aussi faible que celle que nous avions, nous avons décidé de faire 60 tours de boucles. En ce qui concerne le nombre de pixels pour l'écart maximal avec les lignes épipolaires nous l'avons fixé à 1.

Avec ces arguments nous obtenons les résultats suivants (en lançant l'algorithme 10 fois pour chaque jeu d'images) :

- **POP**
  13 inliers pour 24 correspondances, soit une proportion de **54% d'inliers**

- **DeathStar**
  10 inliers pour 44 correspondances, soit une proportion de **23% d'inliers**.

Nous remarquons ici que la méthode RANSAC que nous avons codé permet d'obtenir de meilleurs résultats que celle qu'offre OpenCV pour les images POP, passant d'une proportion de 37% à 54% d'inliers. Elle ne permet en revanche pas d'obtenir des résultats aussi bons sur les images DeathStar, passant de 29% à 23% d'inliers.

4 Conclusion

L'obtention de la matrice fondamentale, bien que facilitée par des librairies spécialisées, n'en demeure pas moins complexe car les paramètres importants pour de bonnes correspondances ne sont pas universels.
En ce qui concerne la méthode de détermination de la matrice à partir des correspondances, cette étape peut être plus ou moins efficace selon l'algorithme (voir sa variante) utilisé. Dans notre cas nous avons implémenté la méthode RANSAC pour nous apercevoir qu'elle était différente de celle déjà disponible via OpenCV - il est prudent de penser que c'est dû à des différences entre les algorithmes mais nous ne savons pas pù elles peuvent être.