Unité 5I-IG4 – Vision et apprentissage
TP4 – homographies, vision binoculaire

Xavier Hilaire
x.hilaire@esiee.fr

19 octobre 2022

1 Préparation

Téléchargez et extrayez quelque part l'archive qui se trouve ici : `https://perso.esiee.fr/~hilairex/5I-IG4/TP4.tgz`

Le Makefile suppose que vous travaillez sur les machines ESIEE, donc que OpenCV se trouve dans `/nfs/opt/bdr/OpenCV`. Rappelez-vous également que la variable d'environnement `LD_LIBRARY_PATH` doit valoir `/nfs/opt/bdr/OpenCV/lib`

2 Homographie

Considérez les images `panneau.jpg` et `trump.jpg`. On voudrait produire une image dans laquelle le portrait de D.Trump soit affiché en lieu et place de la publicité du panneau, comme sur la figure 1.

FIGURE 1 –

Des deux fonctions OpenCV getAffineTransform() et findHomography(), laquelle est la plus appropriée au problème, et pourquoi ?

Complétez le code de `trump.cpp` pour qu'il produise l'image voulue. La fonction dont vous aurez besoin pour « plaquer » `trump.jpg` sur `panneau.jpg` est soit warpAffine(), soit warpPerspective(), selon votre réponse à la question précédente. Il se peut également que vous ayez à recopier les pixels de la nouvelle image créée vers l'image `panneau`.

3 Géométrie épipolaire

Le répertoire `images` contient plusieurs fichiers d'images, qui ont toutes été acquises avec 2 caméras : les images en `-L` par la caméra « gauche », et celles en `-R` par la caméra « droite ». Les séries n'ont pas d'importance dans ce TP, prenez simplement garde à bien apparier des images de même préfixe.

3.1 Calibrage stéréo

Le calibrage d'un système de stéréovision est l'opération qui consiste à estimer sa matrice fondamentale. La méthode de OpenCV qui réalise cela est findFundamentalMat(). Elle prend en paramètres :

- Deux tableaux de points, `points1` dans l'image gauche, et `points2` dans la droite.
- La méthode à utiliser pour estimer la matrice. Dans le cas particulier où celle-ci est RANSAC (présentée en cours), on peut y ajouter :
  - L'erreur consensus admissible pour le modèle. Il s'agit de la plus grande distance entre le modèle prédit et les données effectivement observées. Dans le cas de la géométrie épipolaire, la distance en question est celle d'un point observée à sa ligne épipolaire la plus proche.
  - La probabilité avec laquelle on espère que RANSAC réussisse. Elle correspond à τ dans le cours.

La fonction findFundamentalMat() distingue également si elle doit utiliser l'algorithme à 7 ou à 8 points évoqués en cours. Pour $N > 8$, le problème est surcontraint, et les méthodes d'estimation tentent de minimiser l'erreur de reprojection totale en assurant $\det(F) = 0$ soit exactement, soit le mieux possible(l'algorithme à 7 points peut ne pas calculer de solution optimale s'il n'a pas assez de points).

Les points stockés dans `points1` et `points2` sont censés correspondre un à un : `points2[i]` est l'homologue droit de `point1[i]` pour tout i. Pour cette raison, on utilise souvent des échiquiers, car les points sont faciles à repérer et sauf disposition inhabituelle des caméras, le sens de lecture est en prnicipe le même d'une image à l'autre.

Dans son état actuel, le programme `calibrate3d.cpp` se contente de charger l'image passée en unique argument, d'y chercher un échiquier, d'affiner la position des points trouvés, et le résultat.

Ajoutez-lui le code nécessaire pour qu'il utilise la paire d'images qu'il reçoit en arguments pour calibrer le système stéréo. Vous utiliserez RANSAC comme algorithme de validation. Affichez la matrice fondamentale qui en résulte.

3.2 Tracé des lignes épipolaires

Ecrivez une fonction `void epilignes(u, F, im)`, qui, étant donné un ensemble de points **u** censés provenir d'une caméra, et une matrice fondamentale **F**, trace sur l'image `im`, censée être acquise l'autre caméra, les lignes épipolaires par lesquelles doivent passer les homologues de ces points. Les fonctions computeCorrespondEpilines() et line() peuvent vous être utiles.

Testez votre fonction, en affichant les lignes épipolaires gauches et droites de la paire d'images passée en arguments. Les lignes semblent-elles bien converger (au moins approximativement) en un épipole dans chaque image ?