Mise en correspondance stéréoscopique

Bruno Ordozgoiti

October 2, 2013

La première part s'agit simplement de initialiser la matrice que permet calculer aisément le produit vectoriel.

Calcul de la matrice fondamentale

$$p^{\times} = \begin{vmatrix} 0 & -p_z & p_y \\ p_z & 0 & -p_x \\ -p_y & p_x & 0 \end{vmatrix}$$

Ça nous permet calculer la matrice fondamental F, qui nous donnera la possibilité d'obtenir les droites epipolaires.

$$F = (P_2 O_1)^{\times} P_2 P_1^{+}$$

La matrice obtenue est la suivante.

$$\begin{vmatrix} -4.55191e-15 & -1.87385e+01 & 4.49723e+03 \\ -1.87384e+01 & 2.06585e-13 & 3.05726e+05 \\ 4.49722e+03 & -2.93733e+05 & -2.87822e+06 \end{vmatrix}$$

La pseudo-inverse a été calculée en utilisant la méthode p1.inv(DECOMP_SVD)

Soit

$$A_1 = \begin{vmatrix} \alpha_1 & \gamma_1 & u_1 \\ 0 & \beta_1 & v_1 \\ 0 & 0 & 1 \end{vmatrix}$$

la matrice des paramètres intrinsèques de la caméra située à gauche. Alors l'equation de la droite épipolaire de l'image droite associée au centre de l'image de gauche est la suivante:

$$d = F \begin{vmatrix} u_1 \\ v_1 \\ 1 \end{vmatrix}$$

Soit

$$A_2 = \begin{vmatrix} \alpha_2 & \gamma_2 & u_2 \\ 0 & \beta_2 & v_2 \\ 0 & 0 & 1 \end{vmatrix}$$

la matrice des paramètres intrinsèques de la caméra située à droite. Alors la droite épipolaire de l'image gauche associée au centre de l'image de droite est la suivante:

$$d = F^t \begin{vmatrix} u_2 \\ v_2 \\ 1 \end{vmatrix}$$

Soit

$$A_3 = \begin{vmatrix} \alpha_3 & \gamma_3 & u_3 \\ 0 & \beta_3 & v_3 \\ 0 & 0 & 1 \end{vmatrix}$$

la matrice des paramètres intrinsèques de la caméra située à gauche. Alors la droite épipolaire de l'image droite associée au point situé au centre du côté haut de l'image de gauche.

$$d = F \begin{vmatrix} u_3 \\ 0 \\ 1 \end{vmatrix}$$

Extraction des coins

Après avoir utilisé la fonction goodFeaturesToTrack, on transform les vecteurs en coordonnées homogènes en ajoutant une troisième composante avec la valeur 1.
Avec la valeur MAX_CORNERS présent dans le code (32), la mèthode ne trouve pas toutes les coins qu'on voit dans la sphere. En l'augmentant jusqu'à 64, le resultat est satisfaisant. Néanmoins, cela provoque l'apparition d'un plus grand nombre de droites epipolaires, ce qui peut faire la mise en correspondance plus difficile.

Calcule des distances
Pour la part suivante du processus il faut calculer la matrice de distances. Chaque valeur a_{ij} de cette matrice sera calculée de la façon suivante.

$$a_{ij} = dist(m1_i, d1_j) + dist(m2_j, d2_i)$$

où $m1_i$ est un point dans le plan π_1,
$m2_j$ est un point dans le plan π_2,
$d1_j$ est la droite epipolaire du plan π_1 associée au point $m2_j$ et
$d2_i$ est la droite epipolaire du plan π_2 associée au point $m1_i$.

Mise en correspondance
Ayant la matrice des distances, on peut rechercher des homologues. Chaque ligne répresent un coin dans l'image droite. Chaque colonne répresent un pixel dans l'image gauche.
Pour retrouver quel pixel de l'image gauche est l'homologue de chaque pixel de l'image droite il suffit rechercher la valeur la plus petite de chaque ligne, sachant que pour répresenter en fait un pair, elles doit être plus petite que le seuil indiqué.
Sachant que la mise en correspondance est faite en calculant que la distance entre les coins et les droites épipolaires, il peut y avoir des correspondences

completement faux. On peut ajouter une autre contrainte pour minimiser le nombre de erreurs.
Etant donné que l'image droite a été obtenue avec une faible rotation de la caméra autour de l'axe Y (vertical), les homologues droites des pixels gauches doit avoir une valeur x plus petit. On peut alors enlever tous les homologues dont cette contrainte n'est pas confirmée. On pourrait aussi géneraliser cette optimisation en prenant en compte l'angle de rotation de la caméra.
En utilisant, comme indiqué avant, un plus grande valeur de la variable MAX_CORNERS, le nombre de correspondences faux augmente vite.
Avec les resultats obtenus, la tâche de compter les points occultés sur chaque image semble être ambigu. En fait, il ne semble y avoir qu'un coin de l'image gauche qui n'a pas été trouvé dans l'image droite, mais prenant en compte que la mèthode a trouvé plusiers coins dans chaque intersection, on pourrait considérer qu'il y en a plus.
Le nombre de correspondences trouvés est 21, dont 10 sont faux.