TP M3DA, semaine 1: éléments de géométrie projective et calibration de caméra

François Lepan

17 septembre 2013

Dans ce rapport nous allons voir comment calibrer une caméra via la méthode de Zhang. Cette méthode ce découpe en plusieurs étapes dont nous allons voir le fonctionnement.

Comparaison zhang.sce et rapport technique de zhang

Tout d'abord comparons le code scilab fournie avec une publication de Zhang afin de connaîtres les étapes de cette méthode. Cette méthode suppose :

- de connaitre plusieurs points de la scène ;
- ces points doivent être dans le meme plan ;
- on possède au moins deux images de ces points dont on connait les positions et les projections ;
- les différentes images ne doivent pas résulter d'une translation pure de la caméra. Elle doivent posséder des orientations et positions différentes.

Voici les étapes de cette méthode :

Pour chaque image on détermine l'homographie entre les points projetés et les points de la scène

C'est à dire que l'on va chercher comment passer d'un plan projectif à un autre via une transformation linéaire.

Rapport de Zhang section 2.2 : *Homography between the model plane and its image.*

Dans zhang.sce (ligne 41-42) :

```
// Estimer l'homographie entre la mire et l' image
H(:,:,i) = ZhangHomography(M(sansZ,:), m(:,:,i));
```

Avec les 2 contraintes sur les paramètres intrinsèques de la caméra on obtient 2n équations.

Les deux contraintes sur les paramètres intrinsèques viennent du fait qu'une homographie possède 8 dégrées de liberté et elle a 6 paramètres extrinsèques : 3 pour la rotation et 3 pour la translation.

Rapport de Zhang section 2.3 : *Constraints on the intrinsic parameters.*

Dans zhang.sce (ligne 43-44) :

```
// Ajouter deux lignes de contraintes dans V
V = [V; ZhangConstraints(H(:,:,i))];
```

On détermine les paramètres intrinsèques de la caméra.

Ces paramètres sont :
- la distance focale.
- les deux facteurs d'agrandissement.
- les deux coordonnées de la projection du centre optique de la caméra sur le plan image.
- la non-orthogonalité potentielle des lignes et des colonnes du capteur de la caméra.

Rapport de Zhang section B : *Extraction of the Intrinsic Parameters from Matrix B.*

Dans zhang.sce (ligne 48-49) :

```
// Estimation de la matrice intrinseque
A = IntrinsicMatrix(b);
```

On détermine les paramètres extrinsèques de la caméra pour chacune des images.

Rapport de Zhang section 3.1 : *Closed-form solution.*

Dans zhang.sce (ligne 51-56) :

```
// Estimations des matrices extrinseques
E = zeros(3, 4, ni);
for i = 1:ni
        E(:,:,i) = ExtrinsicMatrix(iA, H(:,:,i));
        disp(E(:,:,i))
end
```

Maintenant que l'on connaît les étapes de la méthode de Zhang on va voir comment elles sont implémenté en scilab.

Implémentation

Dans cette section nous allons voir 3 méthodes :

- `ZhangConstraintTerm` qui construit un vecteur ligne contenant les contraintes d'une homographie.
- `IntrinsicMatrix` qui contruit une matrice 3x3 contenant les paramètres intrinsèques de la caméra.
- `ExtrinsicMatrix` qui construit une matrice 3x4 contenant les paramètres extrinsèques de la caméra.

ZhangConstraintTerm

```
function v = ZhangConstraintTerm(H, i, j)
    v = zeros(6);
    v(1) = H(i,1) * H(j,1);
    v(2) = H(i,1) * H(j,2) + H(i,2) * H(j,1);
    v(3) = H(i,2) * H(j,2);
    v(4) = H(i,3) * H(j,1) + H(i,1) * H(j,3);
    v(5) = H(i,3) * H(j,2) + H(i,2) * H(j,3);
    v(6) = H(i,3) * H(j,3);
    v = v';
endfunction
```

IntrinsicMatrix

```
function A = IntrinsicMatrix(b)

        v0     = (b(2)*b(4) - b(1)*b(5)) / (b(1)*b(3) - b(2)^2);
        lambda = b(6) - ( b(4)^2 + v0*( b(2)*b(4) - b(1)*b(5) ) )/b(1);
        alpha  = sqrt( lambda / b(1) );
        beta_  = sqrt( lambda * b(1) / ( b(1)*b(3) - b(2)^2 ) );
        gama   = -b(2) * (alpha^2) * beta_ / lambda;
        u0     = (gama * v0 / beta_) - ( b(4) * (alpha^2)/lambda );

        A = [alpha, gama,  u0;
                0 , beta_, v0;
                0 , 0    , 1];
endfunction
```

ExtrinsicMatrix

```
function E = ExtrinsicMatrix(iA, H)

        lambda = 1 / norm(iA*H(:,1));
        r1 = lambda*iA*H(:,1);
        r2 = lambda*iA*H(:,2);
        r3 = r1.*r2;
        t = lambda*iA*H(:,3);

        E = [   r1', t(1);
                r2', t(2);
                r3', t(3)];
endfunction
```