

## Visual Perception

### Calibrating simulated camera

#### 1. Introduce

In this course work, a simulated camera has been constructed by using some given parameters. The parameters given are based on the method of Faugeras. 2D projection points can be gotten from 3D points through the Faugeras camera model. Using 3D points and their 2D projection, Hall transformation matrix can be built. By increasing the number of points, a more accurate camera model can be built even there are some affections from the noise.

#### 2. Procedures

##### Part 1

##### Step 1 and Step 2

The parameters are given based on the method of Faugeras. From the method of Faugeras the intrinsic parameters matrix is:

$$\begin{pmatrix} \alpha_u & 0 & u_0 & 0 \\ 0 & \alpha_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

The extrinsic parameters matrix is:

$${}^c K_W = \begin{pmatrix} {}^c R_{W3 \times 3} & {}^c T_{W3 \times 1} \\ \mathbf{0}_{1 \times 3} & 1 \end{pmatrix}$$

$${}^c R_W = Rot(X, \alpha) \cdot Rot(Y, \beta) \cdot Rot(X_1, \alpha_1)$$

$${}^c R_W = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha & -\sin \alpha \\ 0 & \sin \alpha & \cos \alpha \end{pmatrix} \begin{pmatrix} \cos \beta & 0 & \sin \beta \\ 0 & 1 & 0 \\ -\sin \beta & 0 & \cos \beta \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha_1 & -\sin \alpha_1 \\ 0 & \sin \alpha_1 & \cos \alpha_1 \end{pmatrix}$$

$${}^c T_W = \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix}$$

Actualization by MATLAB is as follows:

```
%step 1
```

```
au=557.0943; av=712.9824; u0=326.3819; v0=298.6679;
```

```
f=80;
```

```
Tx=100; Ty=0; Tz=1500;
```

```
Phix=0.8*pi/2; Phiy=-1.8*pi/2; Phix1=pi/5;
```

```
image=zeros(480,640);
```

```
%step 2
```

```
intrinsics=[au 0 u0 0; 0 av v0 0; 0 0 1 0];
```

```
rx=[1 0 0; 0 cos(Phix) -sin(Phix); 0 sin(Phix) cos(Phix)];
```

```
ry=[cos(Phiy) 0 sin(Phiy); 0 1 0; -sin(Phiy) 0 cos(Phiy)];
```

```

rx1=[1 0 0; 0 cos(Phix1) -sin(Phix1); 0 sin(Phix1) cos(Phix1)];
T=[Tx; Ty; Tz];
extrinsecs=[rx*ry*rx1 T;0 0 0 1];

```

### Step 3

At least 6 points need to be defined without linear and coplanar. In this step just 6 points are enough, but after step 8 more points are needed.

The defined of 6 points is as follows:

```

Pw1=[100;100;0;1];Pw2=[400;400;0;1];
Pw3=[100;0;100;1];Pw4=[400;0;400;1];
Pw5=[0;100;100;1];Pw6=[0;400;400;1];

```

More points defined are as follows:

```

N=50;
a=-480;
b=480;
Pw=a+(b-a)*rand(N,3);% define 3D points
I=ones(N,1);
Pw=[Pw, I];

```

3D points are generated by function “random”, because points are a lot; it is easy to be non-linear and non-coplanar.

### Step 4

Using the transformation equation as follows:

$$\begin{pmatrix} s^I X_u \\ s^I Y_u \\ s \end{pmatrix} = \begin{pmatrix} \alpha_u & 0 & u_0 & 0 \\ 0 & \alpha_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} {}^C R_{W^{3 \times 3}} & {}^C T_{W^{3 \times 1}} \\ \mathbf{0}_{1 \times 3} & 1 \end{pmatrix} \begin{pmatrix} {}^W X_w \\ {}^W Y_w \\ {}^W Z_w \\ 1 \end{pmatrix}$$

The actualization by MATLAB is as follows:

```

A_Faugeras=intrinsics*extrinsecs;%Faugeras transformation matrix
for i=1:N
    Pu_o(i,:)=(A_Faugeras*Pw(i,:))';%compute 2D projection points for
3D points
    Pu_o(i,:)=Pu_o(i,:)/Pu_o(i,3);%normalization
end

```

### Step 5

MATLAB actualizes the plot of 2D projection as follows:

```

figure(1)
hold on
stem(Pu_o(i,1),Pu_o(i,2),'LineStyle','none');
title('2D projection points');
hold off

```

### Step 6

In order to obtain the transformation matrix using the method of Hall, at least 6 3D points and their 2D projection points are needed. From the 5 steps above the points pairs can be gotten. The transformation matrix is the solution system of following linear equation system:

$$QA = B$$

A is the transformation matrix; B is the 2D projection points' matrix:

$$B_{2i-1} = \begin{pmatrix} {}^I X_{ui} \end{pmatrix}$$

$$B_{2i} = \begin{pmatrix} {}^I Y_{ui} \end{pmatrix}$$

i is the ith 2D projection points,  ${}^I X_{ui}$  is the x-axis coordinate,  ${}^I Y_{ui}$  is the y-axis coordinate.

Q is the coefficient matrix made by the coordinates of 3D points and their 2D projection points:

$$Q_{2i-1} = \begin{pmatrix} {}^W X_{wi} & {}^W Y_{wi} & {}^W Z_{wi} & 1 & 0 & 0 & 0 & 0 & -{}^I X_{ui} {}^W X_{wi} & -{}^I X_{ui} {}^W Y_{wi} & -{}^I X_{ui} {}^W Z_{wi} \end{pmatrix}$$

$$Q_{2i} = \begin{pmatrix} 0 & 0 & 0 & 0 & {}^W X_{wi} & {}^W Y_{wi} & {}^W Z_{wi} & 1 & -{}^I Y_{ui} {}^W X_{wi} & -{}^I Y_{ui} {}^W Y_{wi} & -{}^I Y_{ui} {}^W Z_{wi} \end{pmatrix}$$

Pseudo inverse for the linear equation system leads to a unique solution:

$$A = Q^{-1}B \Rightarrow A = (Q^T Q)^{-1} Q^T B$$

The actualization by MATLAB is as following:

```
Q(2*i-1,:)=[Pw(i,1) Pw(i,2) Pw(i,3) 1 0 0 0 0 -Pu(i,1)*Pw(i,1)
-Pu(i,1)*Pw(i,2) -Pu(i,1)*Pw(i,3)];
Q(2*i,:)=[0 0 0 0 Pw(i,1) Pw(i,2) Pw(i,3) 1 -Pu(i,2)*Pw(i,1)
-Pu(i,2)*Pw(i,2) -Pu(i,2)*Pw(i,3)];
B(2*i-1)=Pu(i,1);
B(2*i)=Pu(i,2);
```

```
A_Hall=inv(Q'*Q)*Q'*B;
```

### Step 7

This is the transformation matrix gotten by the method of Hall:

```
-0.332440977389438  0.0623705255264864  -0.266219007970257  363.521520000000
-0.120680015546083  0.490343478823258  0.102838669991825  298.6679000000005
6.36610018746190e-05  0.000397783421925086  -0.000531187415632573  1
```

This is the transformation matrix gotten by the method of Faugeras:

```
-0.332440977389386  0.0623705255264979  -0.266219007970222  363.521520000000
-0.120680015546123  0.490343478823154  0.102838669991847  298.6679000000000
6.36610018750175e-05  0.000397783421925034  -0.000531187415632491  1
```

From the two matrixes we can see that they are almost the same except very low digital.

### Step 8

In order to add some Gaussian noise to the 2d points producing discrepancies between the range [-1, +1] pixels for the 95% of points, we need to know the value of mean and the standard deviation. The CDF of normal distribution is as follows:

$$\Phi(x; \mu, \sigma^2) = \int_{-\infty}^x \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(t-\mu)^2}{2\sigma^2}\right] dt, X \in \mathfrak{R}$$

The mean value can be set 0, Using the standard normal distribution table we can get the value for standard deviation. The standard normal distribution table is evaluated with mean  $\mu = 0$  and standard deviation  $\sigma = 1$ ,

$$X \sim N(0,1) : \Phi(x) = \int_{-\infty}^x \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}} dt, X \in \mathfrak{R}$$

So what we need is:

$$\Phi\left(\frac{1-\mu}{\sigma}\right) - \Phi\left(\frac{-1-\mu}{\sigma}\right) = P = 95\%$$

As above setting mean is zero, checking the standard normal distribution table with property  $\Phi(-x) = 1 - \Phi(x)$ , we can get  $\sigma = 0.5102$ .

In MATLAB, we can use function 'normrnd' to generate noise with  $\mu = \text{mu}$  and  $\sigma = \text{sigma}$ :

```
Pu(i,:) = Pu_o(i,:) + [normrnd(mu, sigma) normrnd(mu, sigma) 0]; % add  
Gaussian noise to the point
```

In this step there are only 6 points, so the affection by the noise is obvious:

The hall transformation matrix without noise:

$$\begin{array}{cccc} -0.3324 & 0.0623 & -0.2662 & 363.5215 \\ -0.1206 & 0.4903 & 0.1028 & 298.6679 \\ 6.3661\text{e-}05 & 0.0003977 & -0.0005311 & 1 \end{array}$$

The hall transformation matrix with noise:

$$\begin{array}{cccc} -0.3419 & 0.06935 & -0.2556 & 364.3856 \\ -0.1367 & 0.5069 & 0.1269 & 297.7176 \\ 3.6202\text{e-}05 & 0.0004223 & -0.0004879 & 1 \end{array}$$

Because the noise is random, so the error in the hall transformation matrix with noise will be different every time, but the matrix is always different from the one without noise.

## Step 9

When the number of point is 10, the hall transformation matrix with noise:

0.333401774412375	0.0585275923051620	0.265921790613736	363.643057463555
0.120911828222105	0.487961390335204	0.101770591057943	298.572451056323
6.28276891507509e05	0.000388573578007667	0.000530243222820151	1

When the number of point is 50, the hall transformation matrix with noise:

0.330854745979193	0.0597982816386456	0.266717051032841	363.519679488837
0.120091973109105	0.489112714413392	0.102798138043009	298.485023467790
6.72859846272826e05	0.000392262535766809	0.000532466407036633	1

When the number of point is 100, the hall transformation matrix with noise:

0.332449911568769	0.0623853253041157	0.267761624787159	363.439239614644
0.120903719895725	0.490532558173192	0.101751311229659	298.614083163518
6.31026416270586e05	0.000398869031557931	0.000534010738079502	1

It is easy to find out that as the number of points increase, the error of the hall transformation matrix with noise decrease. When the number reaches 100, the error becomes very small compare with the one without noise.

## Part 2

### Step 9

The linear transformation matrix equation system using method of Faugeras is as follows:

$$QX = B$$

$$X = (T_1 \ T_2 \ T_3 \ C_1 \ C_2)^T$$

$$Q = \begin{pmatrix} {}^w P_{wi}^t & -{}^I X_{ui} {}^w P_{wi}^t & \dots & 0_{1 \times 3} & 1 & 0 \\ 0_{1 \times 3} & -{}^I Y_{ui} {}^w P_{wi}^t & {}^w P_{wi}^t & \dots & 0 & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots \end{pmatrix}$$

$$B = \begin{pmatrix} \dots \\ {}^I X_{ui} \\ {}^I Y_{ui} \\ \dots \end{pmatrix}$$

The solution by using least-squares (LS) is as follows:

$$X = (Q^T Q)^{-1} Q^T B$$

There are two ways to solve the linear equation system by using Singular Value Decomposition (SVD).

First is use:  $X = VS^{-1}U^T B$

U and V is the left and right singular vector matrix, S is singular value matrix.

Second is use:  $\bar{x} = \sum_{i=1}^r \frac{u_i^T b}{\sigma_i} v_i$

In this coursework we use the first SVD way to solve the linear equation system. In this case, the result will be the same as LS.

The parameters extracted from X are the same at the ones defined in Step 1.

The actualization by MATLAB is as follows:

```
X_LS=inv(Q'*Q)*Q'*B;
```

```
[U,S,V] = svd(Q,0);
```

```
X_SVD=V*inv(S)*U'*B;
```

Step 11

Part 3

Step 12

This step mainly based on some figure function in MATLAB, such as 'line' and 'plot3'. Using these figure function, a 3D model has been constructed. From the 3D model it can be found that all the optical rays cross at the focal point and 2D points lie on the optical rays defined by the 3D points.