

# 第四次作业

## 第一题：图像去畸变

### 代码

```
// 先归一化
double image_x=(u-cx)/fx;
double image_y=(v-cy)/fy;
double r = sqrt((image_x*image_x)+(image_y*image_y));
//径向畸变
double step1_x=image_x*(1+k1*r+k2*r*r*r);
double step1_y=image_y*(1+k1*r+k2*r*r*r);
//切向畸变+径向畸变
double step2_x=2*p1*image_x*image_y+p2*
(r*r+2*image_x*image_x)+step1_x;
double step2_y=p1*
(r*r+2*image_y*image_y)+2*p2*image_x*image_y+step1_y;
//赋值
u_distorted=step2_x*fx+cx;
v_distorted=step2_y*fy+cy;
```

### 结果图





鱼 镜头的畸变校正.

1.  $\vec{x}_w$  为三维空间中一点

$$\vec{x}_c = T_{cw} \cdot \vec{x}_w \quad \vec{x}_c = \begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix}$$

2. 归一化到  $z=1$  平面.

$$a = \frac{x_c}{z_c} \quad b = \frac{y_c}{z_c}$$

3. 记  $r^2 = a^2 + b^2$ ,  $\theta = \arctan(b/a)$

4. 畸变.

$$\theta_d = \theta (1 + k_1 \theta^2 + k_2 \theta^4 + k_3 \theta^6 + k_4 \theta^8)$$

5. distorted point are  $(x', y')$

$$x' = \frac{r_d}{r} a = \frac{\theta_d}{\theta} \cdot a$$

$$y' = \frac{r_d}{r} b = \frac{\theta_d}{\theta} \cdot b$$

6. 归一化平面  $\rightarrow$  像素平面.

$$\frac{u - c_x}{\sigma_x} = x'$$

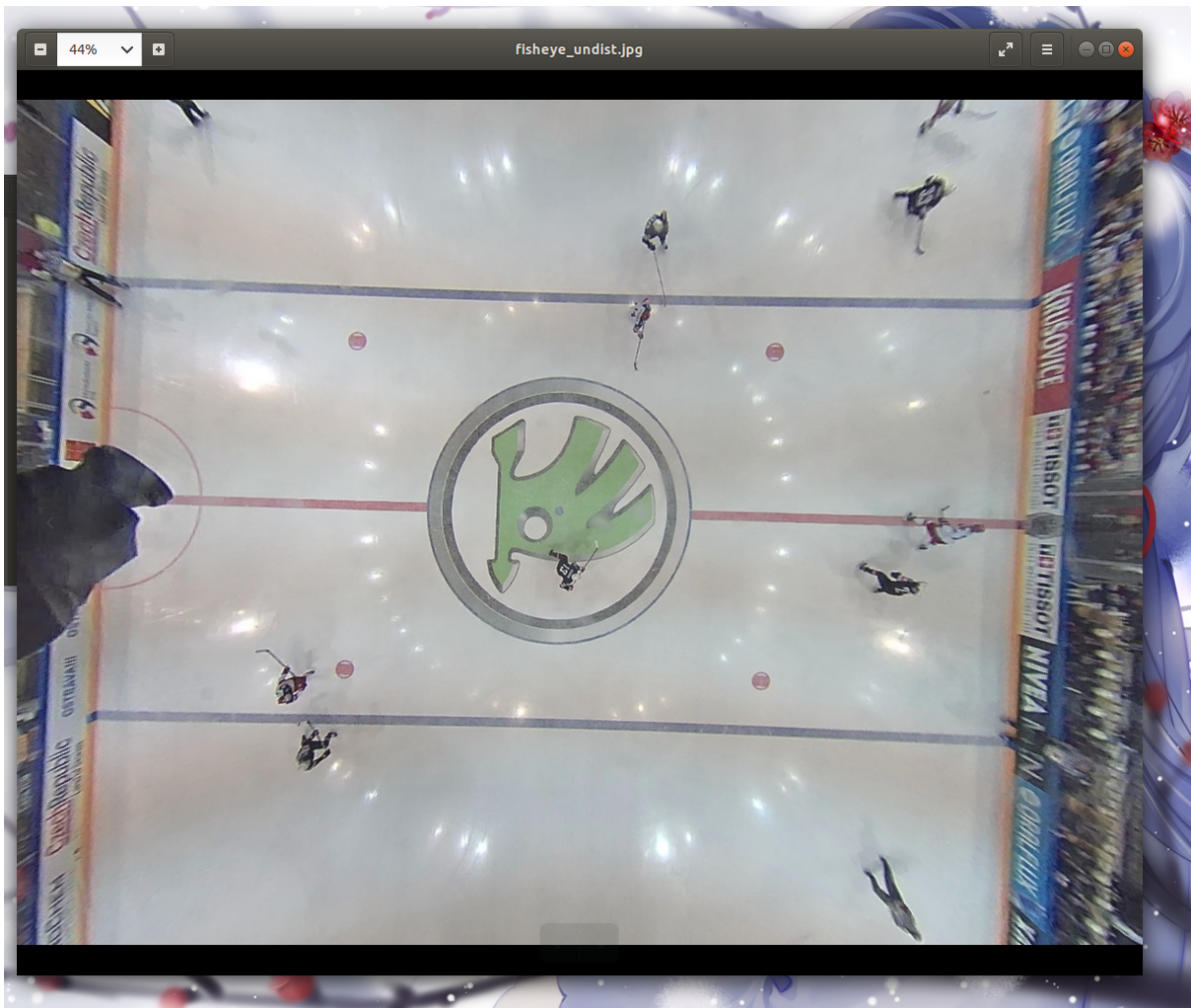
$$\frac{v - c_y}{\sigma_y} = y'$$

$$a = \frac{r}{\theta_d} x'$$

$$b = \frac{r}{\theta_d} y'$$

## 2.3 图片矫正

```
double x=(u-cx)/fx;
double y=(v-cy)/fy;
double ang_d= atan(sqrt(x*x+y*y));
double ang=
ang_d/(1+k1*ang_d*ang_d+k2*ang_d*ang_d*ang_d*ang_d+k3*ang_d*ang_d*ang_d*ang_d*ang_d*ang_d+k4*ang_d*ang_d*ang_d*ang_d*ang_d*ang_d*ang_d);
double r= tan(ang);
double x_distorted=ang/sqrt(x*x+y*y)*x;
double y_distorted=ang/sqrt(x*x+y*y)*y;
u_distorted=fx*(x_distorted+0.01*y_distorted)+cx;
v_distorted=fy*y_distorted+cy;
```



## 2.4 畸变参数为零去畸变的原理

鱼眼相机的畸变模型为多阶泰勒展开， $K_1 \dots K_4$ 取零相当于只取了第一项，只会影响精度不会影响功能。

## 2.5 图片损失

鱼眼图一般为圆形，边缘的信息被压缩的很密，经过去除畸变后原图中间的部分会被保留的很好，而边缘位置一般都会被拉伸的很严重、视觉效果差，所以通常会进行切除，因此肯定会带来图像内容的损失。可以通过增大去畸变时图像的尺寸，或者使用单目相机和鱼眼相机图像进行融合，补全丢失的信息。

## 第三题：双目视差的使用

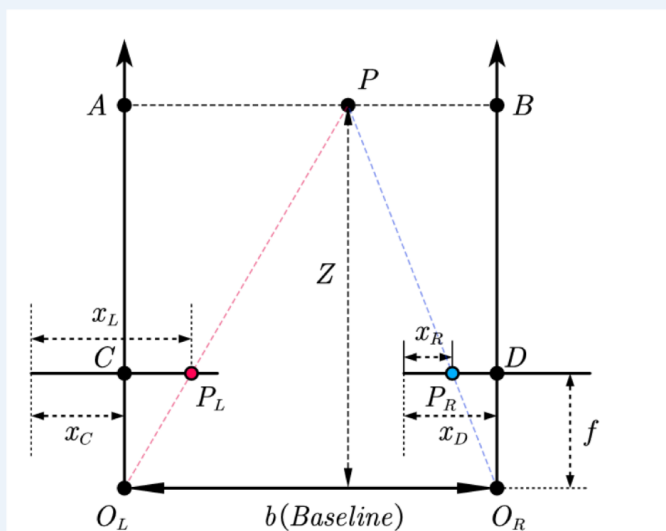
### 3.1 视差原理的证明

1. 假设：

1. 相机内参相同，即焦距、分辨率一致。
2. 两相机光轴平行。
3. 成像平面位于同一水平线上。

直观

几何法



单位(m)

$O_L, O_R$  : 左右相机光心  
 $b$  : 两相机基线长度  
 $P$  : 空间中的点  
 $P_L, P_R$  : 左、右相机成像平面上的像点  
 $f$  : 相机的焦距

$$\triangle O_L A P \sim \triangle O_L C P_L \quad \text{且} \quad \triangle O_R B P \sim \triangle O_R D P_R$$

得:  $\frac{Z}{f} = \frac{PA}{x_L - x_L} = \frac{PB}{x_R - x_R} = \frac{PA + PB}{x_L - x_R + (x_R - x_L)} = \frac{b}{d + (x_R - x_L)}$  得.

$$Z = \frac{bf}{d + (x_R - x_L)}, \quad \text{由于两相机参数相等,}$$

因此  $x_R - x_L = d$ , 故而有:  $Z = \frac{bf}{d}$

$$\text{视差 } x_L - x_R = d.$$

## 2. 相机参数推导法.

1) 由假设得  $K_L = K_R = K = \begin{pmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{pmatrix}$

2) 相机外参 (以右相机到左相机为例):  $K_{R \rightarrow L} = E$ ,  $t_{R \rightarrow L} = \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix} = \begin{pmatrix} b \\ 0 \\ 0 \end{pmatrix}$

3) 设  $P_{0L}, P_{0R}, P_L, P_R$  的坐标分别为

$$P_{0L} = \begin{pmatrix} x_L \\ y_L \\ z_L \end{pmatrix}; P_{0R} = \begin{pmatrix} x_R \\ y_R \\ z_R \end{pmatrix}; P_L = \begin{pmatrix} u_L \\ v_L \\ 1 \end{pmatrix}; P_R = \begin{pmatrix} u_R \\ v_R \\ 1 \end{pmatrix}$$

$P$  在左、右相机坐标系下

左、右图像坐标系.

由小孔成像原理, 有:

$$P_L = \begin{pmatrix} u_L \\ v_L \\ 1 \end{pmatrix} = K_L \frac{1}{z_L} P_{0L}, \quad P_R = \begin{pmatrix} u_R \\ v_R \\ 1 \end{pmatrix} = K_R \frac{1}{z_R} P_{0R}$$

$$P_L = R_{R \rightarrow L} P_R + t_{R \rightarrow L} = E R_R + t_{R \rightarrow L} = P_R + \begin{pmatrix} b \\ 0 \\ 0 \end{pmatrix}$$

联立可得,  $P_L - P_R = \begin{pmatrix} u_L - u_R \\ v_L - v_R \\ 0 \end{pmatrix} = K \frac{1}{z} \begin{pmatrix} b \\ 0 \\ 0 \end{pmatrix}$

又, 左右相机位于同一水平线. 且内参相等, 故  $z_L = z_R = z$

得  $\begin{pmatrix} u_L - u_R \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{pmatrix} \frac{1}{z} \begin{pmatrix} b \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{b f_x}{z} \\ 0 \\ 0 \end{pmatrix}$  设  $d = u_L - u_R$ , 则  $z = \frac{b f_x}{d}$

## 3.2 代码实现

```
double depth = fx * d / (disparity.at<char>(v,u));
double x = (u - cx)/fx * depth;
double y = (v - cy)/fy * depth;
point[0] = x;
point[1] = y;
point[2] = z;
```

//pangolin 一生之敌了, 属于是 目前猜测是因为ros的原因 准备找台电脑 从零开始。。。

## 第四题：矩阵运算微分

4.1 矩阵  $A \in R^{N \times N}$ , 那么  $d(A_X)/d_x$  是什么?

与矩阵运算规则

1. 设矩阵  $A \in R^{N \times N}$ , 那么  $d(A^T)/d_x$  是什么?

$$\frac{d(A^T)}{d_x} = A^T$$

4.2 矩阵  $A \in R^{N \times N}$ , 那么  $d(X^T A_X)/d_x$  是什么?

2). 矩阵  $A \in R^{N \times N}$ , 那么  $d(X^T A_X)/d_x$  是什么?

$$\frac{d(X^T A_X)}{d_x} = \frac{(dx)^T A_X + X^T A}{dx} = \frac{dx^T A_X + dX^T A}{dx} = (A + A^T)X$$

4.3 证明

$$3). \quad X^T A X = \text{tr}(A X X^T)$$

证明:

$$X^T A X = (x_1, x_2, \dots, x_n) \cdot \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

$$= x_1 \sum_{i=1}^n a_{i1} x_i + x_2 \sum_{i=1}^n a_{i2} x_i + \dots + x_n \sum_{i=1}^n a_{in} x_i$$

$$\text{tr}(A X X^T) = \text{tr} \left( \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \cdot (x_1, x_2, \dots, x_n) \right)$$

$$= \text{tr} \left( \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix} \cdot \begin{pmatrix} x_1 x_1 & x_1 x_2 & \dots & x_1 x_n \\ x_2 x_1 & x_2 x_2 & \dots & x_2 x_n \\ \vdots & \vdots & \ddots & \vdots \\ x_n x_1 & x_n x_2 & \dots & x_n x_n \end{pmatrix} \right)$$

$$= x_1 \sum_{i=1}^n a_{i1} x_i + x_2 \sum_{i=1}^n a_{i2} x_i + \dots + x_n \sum_{i=1}^n a_{in} x_i$$

第五题：高斯牛顿法的曲线拟合实验

```

double error = 0;    // 第i个数据点的计算误差
Vector3d J; // 雅可比矩阵
J[0] = -xi*xi*exp(ae * xi * xi + be*xi + ce); // de/da
J[1] = -xi* exp(ae * xi * xi + be*xi + ce); // de/db
J[2] = -exp(ae * xi * xi + be*xi + ce); // de/dc
H += J * J.transpose(); // GN近似的H
b += -error * J;

```

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

-- Configuring done
-- Generating done
-- Build files have been written to: /home/cp/learn_slam/ch4/test4/build
cp@CP:~/learn_slam/ch4/test4/build$ make
Scanning dependencies of target gaussnewton
[ 50%] Building CXX object CMakeFiles/gaussnewton.dir/gaussnewton.cpp.o
[100%] Linking CXX executable gaussnewton
[100%] Built target gaussnewton
cp@CP:~/learn_slam/ch4/test4/build$ ./gaussnewton
total cost: 3.19575e+06
total cost: 376785
total cost: 35673.6
total cost: 2195.01
total cost: 174.853
total cost: 102.78
total cost: 101.937
total cost: 101.937
total cost: 101.937
cost: 101.937, last cost: 101.937
estimated abc = 0.890912, 2.1719, 0.943629
cp@CP:~/learn_slam/ch4/test4/build$

```

## 第六题：批量最大似然估计



$$e = z - Hx$$

$$x = (x_0, x_1, x_2, x_3)^T$$

$$z = (v_1, v_2, v_3, y_1, y_2, y_3)^T$$

$$\tilde{x}_k = \tilde{x}_{k-1} + V_k + W_k$$

$$y_k = x_k + \eta_k$$

$$V_k = x_k - x_{k-1} - W_k$$

$Hx$  为  $6 \times 1$  的列向量,  $x$  为  $4 \times 1$

且  $H$  为  $6 \times 4$ , ( $W_k, \eta_k$  均忽略).

$$R = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} x_1 - x_0 \\ x_2 - x_1 \\ x_3 - x_2 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix} \quad H \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} x_1 - x_0 \\ x_2 - x_1 \\ x_3 - x_2 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

$$H = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

2.  $W^{-1}$  为此问题的信息矩阵.

即高斯分布协方差矩阵之逆.

$$\Sigma = \text{diag}(Q_1, Q_2, Q_3, R_1, R_2, R_3).$$

3. 存在唯一解.

$$\text{目标函数简化为: } x_{MLE}^* = \arg\max (e^T \Sigma^{-1} e)$$

$$\text{唯一解: } x_{MLE}^* = (H^T \Sigma^{-1} H)^{-1} H^T \Sigma^{-1} y.$$

## 参考链接

[https://blog.csdn.net/u011852872/article/details/117340713?spm=1001.2101.3001.6650.5&utm\\_medium=distribute.pc\\_relevant.none-task-blog-2%7Edefault%7EOPENSEARCH%7ERate-6-117340713-blog-122014468.pc\\_relevant\\_multi\\_platform\\_whitelistv4&depth\\_1-utm\\_source=distribute.pc\\_relevant.none-task-blog-2%7Edefault%7EOPENSEARCH%7ERate-6-117340713-blog-122014468.pc\\_relevant\\_multi\\_platform\\_whitelistv4&utm\\_relevant\\_index=10](https://blog.csdn.net/u011852872/article/details/117340713?spm=1001.2101.3001.6650.5&utm_medium=distribute.pc_relevant.none-task-blog-2%7Edefault%7EOPENSEARCH%7ERate-6-117340713-blog-122014468.pc_relevant_multi_platform_whitelistv4&depth_1-utm_source=distribute.pc_relevant.none-task-blog-2%7Edefault%7EOPENSEARCH%7ERate-6-117340713-blog-122014468.pc_relevant_multi_platform_whitelistv4&utm_relevant_index=10)

3.1

[https://blog.csdn.net/qg\\_40918859/article/details/123984329](https://blog.csdn.net/qg_40918859/article/details/123984329)

## 结语

---