1 问题描述

在已经有了相机模型和已知圆的3个特征点的世界坐标及其像素坐标的情况下，我们可以用数值求解的方式求解出相机的位姿。

假设世界坐标系的圆点为圆心

2 约束条件

已知相机模型

$$Z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{f}{\mu_x} & 0 & c_x \\ 0 & \frac{f}{\mu_y} & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & T \\ \overrightarrow{0} & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{1}$$

其中

$$S = \begin{bmatrix} \frac{f}{\mu_x} & 0 & c_x \\ 0 & \frac{f}{\mu_y} & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

是相机的内参矩阵，是一个固定的矩阵，因此可以放到等式左边，化简方程

$$Z_c S^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & T \\ \overrightarrow{0} & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{3}$$

其中

$$S^{-1} = \begin{bmatrix} \frac{\mu}{f} & 0 & -\frac{C_x \mu}{f} \\ 0 & \frac{\mu}{f} & -\frac{C_y \mu}{f} \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

（4）中的RT矩阵用四元数的表达方式可以写成

$$\begin{bmatrix} -2y^2-2z^2+1 & -2wz+2xy & 2wy+2xz & -T_x\left(-2y^2-2z^2+1\right)-T_y\left(-2wz+2xy\right)-T_z\left(2wy+2xz\right) \\ 2wz+2xy & -2x^2-2z^2+1 & -2wx+2yz & -T_x\left(2wz+2xy\right)-T_y\left(-2x^2-2z^2+1\right)-T_z\left(-2wx+2yz\right) \\ -2wy+2xz & 2wx+2yz & -2x^2-2y^2+1 & -T_x\left(-2wy+2xz\right)-T_y\left(2wx+2yz\right)-T_z\left(-2x^2-2y^2+1\right) \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

四元数的各个数值约束为

$$w^2 + x^2 + y^2 + z^2 = 1 \tag{13}$$

2.1 圆心特征点

考虑世界坐标系圆心坐标 $P(0,0,0)$，像素坐标p_0

对应缩放

$$Zc = -T_x\left(-2wy + 2xz\right) - T_y\left(2wx + 2yz\right) - T_z\left(-2x^2 - 2y^2 + 1\right) \tag{7}$$

像素坐标

$$Z_c S^{-1} p_0 = \begin{bmatrix} -T_x\left(-2y^2 - 2z^2 + 1\right) - T_y(-2wz + 2xy) - T_z(2wy + 2xz) \\ -T_x(2wz + 2xy) - T_y\left(-2x^2 - 2z^2 + 1\right) - T_z(-2wx + 2yz) \end{bmatrix} \tag{8}$$

2.2 X方向直径端点

考虑世界坐标系X方向直径端点 $P(L,0,0)$，像素坐标p_x

对应缩放

$$Z_c = L\left(-2wy + 2xz\right) - T_x\left(-2wy + 2xz\right) - T_y\left(2wx + 2yz\right) - T_z\left(-2x^2 - 2y^2 + 1\right) \tag{9}$$

像素坐标

$$Z_c S^{-1} p_x = \begin{bmatrix} L\left(-2y^2 - 2z^2 + 1\right) - T_x\left(-2y^2 - 2z^2 + 1\right) - T_y(-2wz + 2xy) - T_z(2wy + 2xz) \\ L(2wz + 2xy) - T_x(2wz + 2xy) - T_y\left(-2x^2 - 2z^2 + 1\right) - T_z(-2wx + 2yz) \end{bmatrix} \tag{10}$$

2.3 Y方向直径端点

考虑世界坐标系X方向直径端点$P(0,L,0)$，像素坐标p_y

对应缩放

$$Z_c = L\left(2wx + 2yz\right) - T_x\left(-2wy + 2xz\right) - T_y\left(2wx + 2yz\right) - T_z\left(-2x^2 - 2y^2 + 1\right) \tag{11}$$

像素坐标

$$Z_c S^{-1} p_y = \begin{bmatrix} L(-2wz + 2xy) - T_x\left(-2y^2 - 2z^2 + 1\right) - T_y(-2wz + 2xy) - T_z(2wy + 2xz) \\ L\left(-2x^2 - 2z^2 + 1\right) - T_x(2wz + 2xy) - T_y\left(-2x^2 - 2z^2 + 1\right) - T_z(-2wx + 2yz) \end{bmatrix} \tag{12}$$

2.4 总结

综上，(6)(8)(10)(12)中共有7个方程

已知的是$L, f, \mu, p_0, p_x, p_y, C_x, C_y$

未知数$T_x, T_y, T_z, w, x, y, z$

共有7个方程，7个未知数，因此可以求解

3 代码实现

主要是用了scipy中的fsolve函数进行数值求解

```
def func(xx):
    T_x, T_y, T_z,  x, y, z, w = xx
    p0x = -T_x*(-2*y**2 - 2*z**2 + 1) - T_y * \
        (-2*w*z + 2*x*y) - T_z*(2*w*y + 2*x*z)
    p0y = -T_x*(2*w*z + 2*x*y) - T_y*(-2*x**2 -
                                      2*z**2 + 1) - T_z*(-2*w*x + 2*y*z)
    Z_p0 = -T_x*(-2*w*y + 2*x*z) - T_y*(2*w*x + 2*y*z) - \
        T_z*(-2*x**2 - 2*y**2 + 1)
    pxx = -T_x*(-2*y**2 - 2*z**2 + 1) - T_y*(-2*w*z + 2*x*y) - \
        T_z*(2*w*y + 2*x*z) - 2*y**2 - 2*z**2 + 1
    pxy = -T_x*(2*w*z + 2*x*y) - T_y*(-2*x**2 - 2*z**2 + 1) - \
        T_z*(-2*w*x + 2*y*z) + 2*w*z + 2*x*y
    Z_px = -T_x*(-2*w*y + 2*x*z) - T_y*(2*w*x + 2*y*z) - \
        T_z*(-2*x**2 - 2*y**2 + 1) - 2*w*y + 2*x*z
    pyx = -T_x*(-2*y**2 - 2*z**2 + 1) - T_y*(-2*w*z + 2*x*y) - \
        T_z*(2*w*y + 2*x*z) - 2*w*z + 2*x*y
    pyy = -T_x*(2*w*z + 2*x*y) - T_y*(-2*x**2 - 2*z**2 + 1) - \
        T_z*(-2*w*x + 2*y*z) - 2*x**2 - 2*z**2 + 1
    Z_py = -T_x*(-2*w*y + 2*x*z) - T_y*(2*w*x + 2*y*z) - \
        T_z*(-2*x**2 - 2*y**2 + 1) + 2*w*x + 2*y*z
    return [p0x/Z_p0-pixels[0, 0],
            p0y/Z_p0-pixels[0, 1],
            pxx/Z_px-pixels[1, 0],
            pxy/Z_px-pixels[1, 1],
            pyx/Z_py-pixels[2, 0],
            pyy/Z_py-pixels[2, 1],
            w**2+x**2+y**2+z**2-1]

from scipy.optimize import fsolve
guess = [0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1]
ans = fsolve(func, guess)
```

4 实验结果

fsolve的参数包括一个函数和一个初始值，因为是数值求解，所以只能求出一个解，但实际上有多组解（其中只有一组解是真的）

每次计算在2ms左右(个人电脑)

这里设置多个初始值的（x,y），达到近似遍历的效果，在实际可能的范围内将答案求解

```
solution 1 : [-69.55  16.3  -12.8  -60.1  -12.86 -77.73]
solution 2 : [  -4.     -2.81   -0.41 -149.55  -69.47   31.06]
solution 3 : [  -4.     -2.81    0.41 -149.55  -69.47 -148.94]
solution 4 : [  1.32   1.17   0.01  -0.14  -0.58 179.93]
solution 5 : [   5.2    3.3   -0.5   30.    60.  -144. ]
solution 6 : [ 5.2  3.3  0.5 30.  60.  36. ]
solution 7 : [  1.1    7.43   0.1   -0.07  -2.1  179.27]
solution 8 : [   2.5     0.      0.5    42.27   41.36 -165.22]
solution 9 : [  0.57   0.58  -0.    59.9  -36.1  158.04]
solution 10 : [  4.01   4.76   0.03   0.1   -0.07 179.93]
Cost 0.04948520660400390 6 seconds, 25 guesses
The true ans is [ 5.2  3.3  0.5 30.  60.  36. ]
```

可以看到，解中包含了真实答案，但也有许多其他答案，需要后续进行继续分析排除