Computer Vision Exercise 2

Vezeteu Eugeniu - 886240

September 2020

1 Pinhole camera

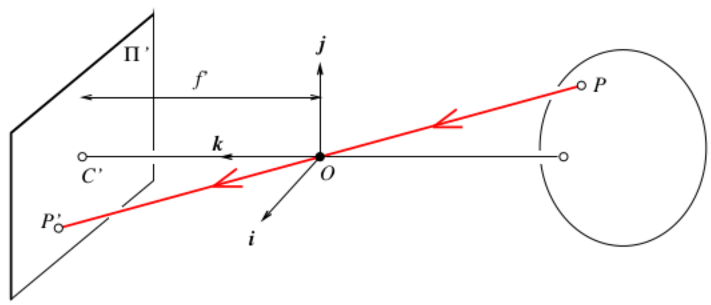

Figure 1: Pinhole camera [2]

We have point P from Figure 1, P = $[x, y, z]$, whose camera coordinate frames are $[x_c, y_c, z_c]$
P' is its image point, P'=$[x_p, y_p, z_p]$, projected on image plane. We have $z_p = f$, because P' lies in the image plane. Also, from Figure 1 we see that points P', P and O are collinear, and therefore $\overrightarrow{P'O} = \alpha * \overrightarrow{PO}$

we have that:

$$\begin{cases} x_p = \alpha * x_c \\ y_p = \alpha * y_c \\ f = \alpha * z_c \end{cases} => \alpha = \frac{x_p}{x_c} = \frac{y_p}{y_c} = \frac{z_p}{z_c} = \frac{f}{z_c}$$

from the equation above results:

$$\begin{cases} x_p = f * \frac{x_c}{z_c} \\ y_p = f * \frac{y_c}{z_c} \end{cases}$$

2 Pixel coordinate frame

- u and v axis are parallel to x and y axis

$$\begin{cases} u = x_p * m_u + u_0 \\ v = y_p * m_v + v_0 \end{cases}$$

- u axis is parallel to x axis and the angle between u and v axis is θ

$$\begin{cases} u = x_p * m_u + cos\theta * y_p * m_v + u_0 \\ v = sin\theta * y_p * m_v + v_0 \end{cases}$$

3 Intrinsic camera calibration matrix

$$\hat{p} = K * X_c = \begin{bmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} f * x_c + u_0 * z_c \\ f * y_c + v_0 * z_c \\ z_c \end{bmatrix} \xRightarrow[\text{by z}]{\text{divide}} \begin{bmatrix} \frac{f*x_c+u_0*z_c}{z_c} \\ \frac{f*y_c+v_0*z_c}{z_c} \\ 1 \end{bmatrix} = \begin{bmatrix} f * \frac{x_c}{z_c} \\ f * \frac{y_c}{z_c} \\ 1 \end{bmatrix} + \begin{bmatrix} u_0 \\ v_0 \\ 0 \end{bmatrix}$$

4 Camera projection matrix

We have world point $\hat{X} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \Rightarrow$ image point $\hat{x} = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f * \frac{X}{Z} \\ f * \frac{Y}{Z} \end{bmatrix}$

In homogeneous coordinates it will be:

world point $\hat{X} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \Rightarrow$ image point $\hat{x} = \begin{bmatrix} f * X \\ f * Y \\ Z \end{bmatrix} = \overbrace{\begin{bmatrix} f & & 0 & \\ & f & 0 & \\ & & 1 & 0 \end{bmatrix}}^{P} * X$

For pinhole cam $P = \begin{bmatrix} f & 0 & u_0 & 0 \\ 0 & f & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

$P = \begin{bmatrix} K & 0 \end{bmatrix}$

Let's define a transformation from world to camera , T

$T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}$

We get the point coordinates transform with the inverse $X_c = T^{-1} * X_w$

$T^{-1} = \begin{bmatrix} R^T & -R^T * t \\ 0 & 1 \end{bmatrix}$

Therefore we have $x = P * X$, $P = K \circ T = \begin{bmatrix} K * R^T & -K * R^T * t \end{bmatrix} = K[R|t]$

5 Rotation matrix

$RX = cos(\theta)X + sin(\theta)uxX + (1 - cos(\theta))(u * X)u$

a) geometric derivation

Let user Figure 1 for our coordinate system, with origin O, where P is an arbitrary point (in our case P = x), Q = point obtained after rotating x with θ, (in our case Q is x'), And O' is a projected coordinates of P on the axis n (in our case n is u).

For ease I will keep the notation from Figure 1

Let e'_x - unit vector parallel to $\overrightarrow{O'P}$

$$e'_x = \frac{\overrightarrow{O'P}}{||\overrightarrow{O'P}||} \tag{1}$$

and e'_y is orthogonal to e'_x and axis n

$$e'_y = n \times e'_x \tag{2}$$

e'_x, e'_y, n is an orthonormal basis if 3D space. We have θ a rotation between $\overrightarrow{O'Q}$ and e'_x, and therefore we can represent $\overrightarrow{O'Q}$ as:

$\overrightarrow{O'Q} = ||\overrightarrow{O'Q}|| * cos(\theta) * e'_x + ||\overrightarrow{O'Q}|| * sin(\theta) * e'_y$

The following $\overrightarrow{O'Q} = ||\overrightarrow{O'P}|| * cos(\theta) * e'_x + ||\overrightarrow{O'P}|| * sin(\theta) * e'_y$ because $\overrightarrow{O'Q}$ is obtained from rotation of $\overrightarrow{O'P}$ aroung axis n.

We substitute equation 1 in the above equation we get:

$\overrightarrow{O'Q} = cos(\theta) * \overrightarrow{O'P} + ||\overrightarrow{O'P}|| * sin(\theta) * e'_y$

Since,

$$\overrightarrow{OO'} + \overrightarrow{O'P} = \overrightarrow{OP} \tag{3}$$

we can write $\overrightarrow{O'Q} = cos(\theta)(\overrightarrow{OP} - \overrightarrow{OO'}) + ||\overrightarrow{O'P}||sin(\theta)e'_y$ As was stated, O' is a projection of P on the axis n =¿

$$\overrightarrow{OO'} = (\overrightarrow{OP}, n)n \tag{4}$$

We can write: $\overrightarrow{O'Q} = cos(\theta)(\overrightarrow{OP} - (\overrightarrow{OP}, n)n) + ||\overrightarrow{O'P}|| * sin(\theta)(n \times e'_x)$

By substituing equation 2 into above equation, we get:

$$\overrightarrow{O'Q} = cos(\theta)(\overrightarrow{OP} - (\overrightarrow{OP}, n)n) + ||\overrightarrow{O'P}|| * sin(\theta)(n \times e'_x)$$

By substituing equation 1 into above equation, we get:

$$\overrightarrow{O'Q} = cos(\theta)(\overrightarrow{OP} - (\overrightarrow{OP}, n)n) + sin(\theta)(n \times \overrightarrow{O'P})$$

By substituing equation 4 into above equation, we get:

$$\overrightarrow{O'Q} = cos(\theta)(\overrightarrow{OP} - (\overrightarrow{OP}, n)n) + sin(\theta)\{n \times (\overrightarrow{OP} - \overrightarrow{OO'}\})$$

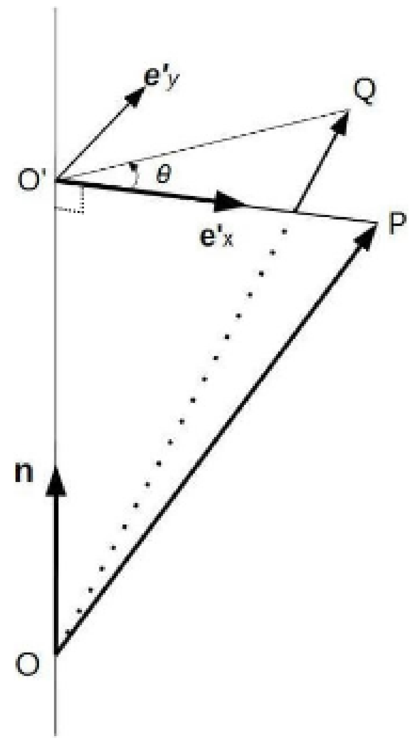

Figure 2: Rodrigues formula derivations, from [1]

where $n \times \overrightarrow{OO'} = 0$ because $\overrightarrow{OO'}$ is parallel with n. Therefore,

$$\overrightarrow{O'Q} = cos(\theta)(\overrightarrow{OP} - (\overrightarrow{OP}, n)n) + sin(\theta)\{n \times \overrightarrow{OP}\})$$

By substituing equation 4 into above equation, we get:

$$\overrightarrow{OQ} = (\overrightarrow{OP}, n)n + cos(\theta)(\overrightarrow{OP} - (\overrightarrow{OP}, n)n) + sin(\theta)(n \times \overrightarrow{OP} = cos(\theta)\overrightarrow{OP} + (1 - cos(\theta))(\overrightarrow{OP}, n)n + sin(\theta)(n \times \overrightarrow{OP})$$

If we rewrite $\overrightarrow{OP}$ as x, and $\overrightarrow{OQ}$ as x', and n as u, we get:

$$r' = cos(\theta)x + (1 - cos(\theta))(x, u)u + sin(\theta)(u \times x)$$

b) $R = \begin{bmatrix} cos\theta + u_1^2(1 - cos\theta) & u_1u_2(1 - cos\theta) - u_3 sin\theta & u_1u_3(1 - cos\theta) + u_2 sin\theta \\ u_2u_1(1 - cos\theta) + u_3 sin\theta & cos\theta + u_2^2(1 - cos\theta) & u_2u_3(1 - cos\theta) - u_1 sin\theta \\ u_3u_1(1 - cos\theta) - u_2 sin\theta & u_3u_2(1 - cos\theta) + u_1 sin\theta & cos\theta + u_2^2(1 - cos\theta) \end{bmatrix}$

References

[1] Rodrigues' rotation formula. https://semath.info/src/rodrigues-rotation.html. Accessed: 2020-09-12.

[2] Forsyth Ponce. *Computer vision*.