AA 274: Principles of Robotic Autonomy | Winter 2018

Lecture 5: Camera Models and Camera Calibration

Scribes: Nathan Usevitch, Vik Pattabi, Huajian Huang, Zaifeng Zheng, Lucy Zhuang

5.1 Introduction

This lecture is on camera models and camera calibration. More specifically, we want to learn:

- What a camera model is, and how to use it.
- How to calibrate a camera.
- Other challenges in image processing.

5.2 Perspective Projection

Our goal is **to determine how points in the world map to points in our camera image**. We start with an assumption of the pinhole camera model (note that all following results also hold when the thin lens model is assumed if we set the camera to be focused at ∞).

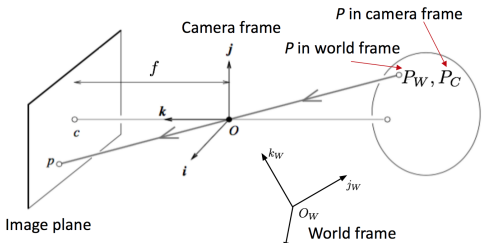

Figure 5.1: Recall the pinhole camera model discussed in the previous lecture. Our goal is transform coordinates in the world frame O_w to coordinates in the Image Plane.

Step 1

First, we aim to map our point P_c in the camera frame to a point $p = (x, y)$. Recall from the previous lecture:

$$x = f\frac{X_C}{X_C}$$
$$y = f\frac{Y_C}{X_C} \tag{5.1}$$

Step 2.a

Now, we must recognize that the actual origin of a camera coordinate system is not centered on the image plane, but rather usually in the lower right or lower left corner. This is typically done so as to avoid the need for negative coordinates (and consequently, negative array indices) when accessing a specific point on the image plane, which is typically represented as a grid or series of arrays in software.

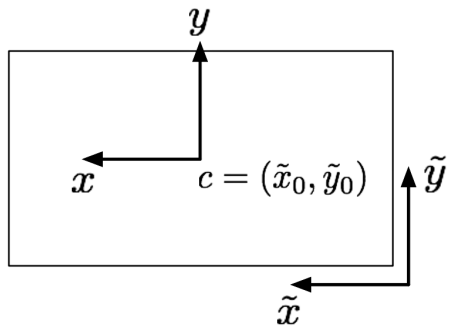

Figure 5.2: Here, we place our origin in the lower right corner of the image plane.

Given this coordinate translation, it's evident that our old point can be represented in the new axes as:

$$\begin{aligned} \tilde{x} &= f\frac{X_C}{Z_C} + \tilde{x}_0 \\ \tilde{y} &= f\frac{Y_C}{Z_C} + \tilde{y}_0 \end{aligned} \tag{5.2}$$

Step 2.b

Another issue we must account for is that the ratio of image coordinates to actual pixels is likely not one-to-one. In other words, a shift of one unit in our image plane may correspond to a shift of multiple pixels. Consequently, we introduce k_x and k_y which represent the number of pixels per unit distance in the x and y direction respectively. Although it's certainly possible that $k_x = k_y$, typically this will not be the case if the camera pixels are not square. From this intuition, it follows that...

$$\begin{aligned} u &= k_x\tilde{x} = k_x f\frac{X_C}{Z_C} + k_x\tilde{x}_0 \\ v &= k_y\tilde{y} = k_y f\frac{Y_C}{Z_C} + k_y\tilde{y}_0 \end{aligned} \tag{5.3}$$

We can simplify our result by renaming the following expressions...

$$k_x f = \alpha$$

$$k_y f = \beta$$

$$k_x \tilde{x}_0 = u_0$$

$$k_y \tilde{y}_0 = v_0$$

Giving us a final result of ...

$$\begin{aligned} u &= \alpha \frac{X_C}{Z_C} + u_0 \\ v &= \beta \frac{Y_C}{Z_C} + v_0 \end{aligned} \tag{5.4}$$

Homogeneous Coordinates

We now note that our transformations from camera frame to pixel value are, unfortunately, not linear. If we can represent these transformations as a linear mapping, we will be able to greatly simplify our math later on. Thus, we introduce the idea of homogeneous coordinates. Broadly defined, the homogeneous coordinates (x_1, x_2, x_3) for a point (x, y) are any three points such that $\frac{x_1}{x_3} = x$ and $\frac{x_2}{x_3} = y$.

Consequently, we use the following rules to transform from inhomogeneous coordinates to homogeneous ones.

$$\begin{pmatrix} x \\ y \end{pmatrix} \Rightarrow \lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \text{ and } \begin{pmatrix} x \\ y \\ z \end{pmatrix} \Rightarrow \lambda \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

And we use these rules to transform back from homogeneous coordinates to inhomogeneous ones.

$$\lambda \begin{pmatrix} x \\ y \\ w \end{pmatrix} \Rightarrow \begin{pmatrix} x/w \\ y/w \end{pmatrix} \text{ and } \lambda \begin{pmatrix} x \\ y \\ z \\ w \end{pmatrix} \Rightarrow \begin{pmatrix} x/w \\ y/w \\ z/w \end{pmatrix}$$

Note that as we continue to use homogeneous coordinates, we might signal that a specific coordinate matrix is in homogeneous form by using a superscript h (e.g. P^h)

5.3 Perspective Projection In Homogeneous Coordinates

When we shift to representing our points in homogeneous coordinates, we arrive at the following result; note that this is equivalent to the set of equations we derived earlier which translated P_C to the Image

Plane, but now P_C is homogeneous.

$$\begin{bmatrix} \alpha & 0 & u_0 & 0 \\ 0 & \beta & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha X_c + u_0 Z_c \\ \beta Y_c + v_0 Z_c \\ Z_c \end{pmatrix} \tag{5.5}$$

$\begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix}$ is our pixel P_C in homogeneous coordinates; we will ultimately map these coordinates to P_W or world coordinates. The preceding 3x4 matrix is called K, and is alternatively referred to as the camera matrix or matrix of intrinsic parameters. K contains the fundamental characteristics of our given pinhole camera. Although we may have access to these values from the manufacturer specs, we often have to re-calibrate the camera to account for differences in tolerances or noise.

The K depicted in the above equation is typically standard. However, in some rare cases, the Image Plane coordinate axes may not be perpendicular, forcing us to introduce another variable, γ, representing this skew parameter Typically, we treat γ as 0.

$$K = \begin{bmatrix} \alpha & \gamma & u_0 & 0 \\ 0 & \beta & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{5.6}$$

Now, our next step is mapping between the world frame and digital camera frame.

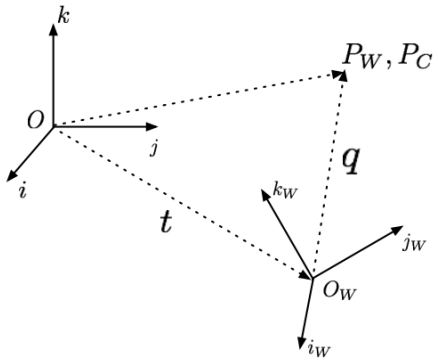

Figure 5.3: Here, we depict our point (P_W or P_C) relative to the origins of the camera frame and world frame (O and O_W respectively).

From this depiction, we have the following:

$$P_C = t + q \tag{5.7}$$

We express P in the camera reference through vector addition. t is the vector from the origin of camera frame to the origin of the world frame. q is the vector representing the position of P with respect to the world frame.

How do we determine q in relation to the camera frame instead of the world frame? We say:

$$q = RP_w \tag{5.8}$$

where R is the rotation matrix relating camera frame to world frame. Now, we must determine R. We start by re-expressing our vector q, as seen below:

$$q = q_{x,c} \cdot \hat{i} + q_{y,c} \cdot \hat{j} + q_{z,c} \cdot \hat{k} \tag{5.9}$$

The above equation expresses q with relation to the camera frame. At the same time, q can also be expressed with respect to the world frame:

$$q = q_{x,w} \cdot \hat{i_w} + q_{y,w} \cdot \hat{j_w} + q_{z,w} \cdot \hat{k_w} \tag{5.10}$$

We know these two expressions refer to the same q, so:

$$q_{x,w} \cdot \hat{i_w} + q_{y,w} \cdot \hat{j_w} + q_{z,w} \cdot \hat{k_w} = q_{x,c} \cdot \hat{i} + q_{y,c} \cdot \hat{j} + q_{z,c} \cdot \hat{k} \tag{5.11}$$

Now, for example, we might take the dot product of both sides with the unit vector $\hat{i}$. Because the dot product of orthogonal vectors is 0, we are able to eliminate some terms and are left with the following:

$$q_{x,c} = q_{x,w}\hat{i_w} \cdot \hat{i} + q_{y,w}\hat{j_w} \cdot \hat{i} + q_{z,w}\hat{k_w} \cdot \hat{i} \tag{5.12}$$

The derivation of the the other coordinates between world and camera frame is similar, and we get the following rotation matrix:

$$R = \begin{bmatrix} \hat{i_w} \cdot \hat{i} & \hat{j_w} \cdot \hat{i} & \hat{k_w} \cdot \hat{i} \\ \hat{i_w} \cdot \hat{j} & \hat{j_w} \cdot \hat{j} & \hat{k_w} \cdot \hat{j} \\ \hat{i_w} \cdot \hat{k} & \hat{j_w} \cdot \hat{k} & \hat{k_w} \cdot \hat{k} \end{bmatrix} \tag{5.13}$$

These sorts of matrices are common in image processing and camera-related tasks.

5.4 Finalizing the Transformation

We are able to express the rigid transformation containing both rotation and translation between a point in the world frame to a point in the camera frame by using homogeneous coordinates as follows:

$$\begin{bmatrix} P_c \\ 1 \end{bmatrix} = \begin{bmatrix} R & t \\ 0_{3x0} & 1 \end{bmatrix} \begin{bmatrix} P_w \\ 1 \end{bmatrix} \tag{5.14}$$

We can combine the transform between a point in the world frame to a point in the camera frame and the transform from a point in the camera frame to a point in the image frame to generate the transform from the the world frame directly to the image frame:

$$p^h = K \begin{bmatrix} I_{3x3} & 0_{3x1} \end{bmatrix} \begin{bmatrix} R & t \\ 0_{3x0} & 1 \end{bmatrix} P_w^h \tag{5.15}$$

$$p^h = K[R \quad t]P_W^h. \tag{5.16}$$

Here the matrix K is the matrix of intrinsic parameters, and the $[R \quad t]$ matrix contains the extrinsic parameters, which describe the cameras relationship to the points in the photograph. The superscript h means the point is expressed in homogeneous coordinates, meaning that p^h is the $3x1$ homogeneous coordinates matrix of the point in the image frame, and P_w^h is the $4x1$ homogeneous coordinates matrix of the point in the world frame.

The total degrees of freedom for the combined matrices are 5 for K, 3 for the rotation matrix R and 3 for the translation t, leading to a total of 11 parameters. Note that the rotation matrix R has 3 DOF because rotation of a coordinate frame takes place in three linearly independent planes. If we assume that our skewness γ is 0, K has 4 parameters and the total number of degrees of freedom is 10 instead.

This perspective projection naturally causes several consequences which can be mathematically derived. For example, parallel lines in images appear to meet at the horizon, and objects that are far away appear smaller than their closer counterparts. Both effects are shown in the following image of train tracks.

Figure 5.4: An image of train tracks. Note that the parallel tracks appear to meet at the horizon. The wooden railroad ties (perpendicular to the tracks) are all the same size, but those further from the camera appear smaller than the closer railroad ties.

5.4.1 Direct Linear Calibration

The problem of direct linear calibration is the following: given known correspondences between known points in the camera frame (p_i) and known points in the world frame $P_{W,i}$, determine the intrinsic parameters (K) and the extrinsic parameters (R, t) of the camera.

A common method of determining these known correspondences is to use images of a grid with a chessboard like pattern. Such a pattern enables simple extraction of the corners, and identification of relationships between objects in the world frame.

5.5 Solving Calibration Problems

The major idea in camera calibration is that, given some known positions in the image frame, and the corresponding world frame positions, we back out the components of the matrices in the previous sections. These matrices include (1) the intrinsic parameter matrix, (2) the rotation matrix from image frame to world frame, and (3) the vector connecting the world frame and image frame.

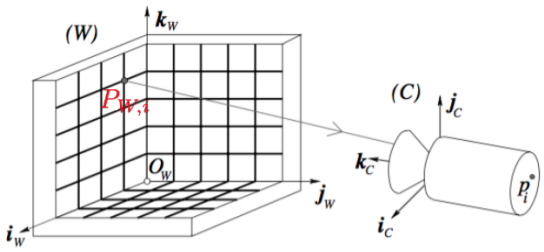

Figure 5.5: An example of camera calibration. Here, we know the relationship between the red point $P_{W,i}$ in the world and p_i in the image plane.

First of all, we have the fundamental relationships, relating p_i^h, the projected point on the image plane in homogeneous coordinates, and $P_{W,i}^h$ the corresponding point in the world frame in homogeneous coordinates:

$$p_i^h = MP_{W,i}^h \,, \quad i = 1, ...n \tag{5.17}$$

where

$$M = K[R \;\; t] = \begin{bmatrix} m_1 \\ m_2 \\ m_3 \end{bmatrix} \tag{5.18}$$

The process involves feeding in n corresponding pairs of image frame positions and world frame positions, where i is the index of the current pair. The M matrix is size 3 by 4, and each m_i is size 1 by 4. Note that we are working with homogeneous coordinates. We have:

$$p_i^h = [\lambda u \;\; \lambda v \;\; \lambda]^T \tag{5.19}$$

$$P_{W,i}^h = [x \;\; y \;\; z \;\; 1]^T \tag{5.20}$$

which conforms with the matrix multiplication rules given M is 3 by 4.

If we write out the relationships between a pair of p_i^h and $P_{W,i}^h$, we can obtain the following:

$$\lambda = m_3 \cdot P_{W,i}^h \tag{5.21}$$

$$u_i = \frac{\lambda u_i}{\lambda} = \frac{m_1 \cdot P_{W,i}^h}{m_3 \cdot P_{W,i}^h} \tag{5.22}$$

$$v_i = \frac{\lambda v_i}{\lambda} = \frac{m_2 \cdot P^h_{W,i}}{m_3 \cdot P^h_{W,i}} \tag{5.23}$$

which can be transformed into the form of two constraint equations:

$$u_i(m_3 \cdot P^h_{W,i}) - m_1 \cdot P^h_{W,i} = 0 \tag{5.24}$$

$$v_i(m_3 \cdot P^h_{W,i}) - m_2 \cdot P^h_{W,i} = 0 \tag{5.25}$$

Now if we stack all of these (n pairs of these u_i and v_i equations) together, we can obtain a system of equations, and to write them compactly, we can use matrix multiplication:

$$\tilde{P}m = 0, \qquad m = \begin{bmatrix} m_1^T \\ m_2^T \\ m_3^T \end{bmatrix} \tag{5.26}$$

Let's write out a few rows of $\tilde{P}$ that correspond to the u_i and v_i constraints above to make things more concrete:

$$\tilde{P} = \begin{bmatrix} -(P^h_{W,1})^T & 0_{1x4} & u_1(P^h_{W,1})^T \\ 0_{1x4} & -(P^h_{W,1})^T & v_1(P^h_{W,1})^T \\ -(P^h_{W,2})^T & 0_{1x4} & u_2(P^h_{W,2})^T \\ \vdots & \vdots & \vdots \end{bmatrix} \tag{5.27}$$

The size of $\tilde{P}$ is 2n by 12, and the size of m is 12 by 1.

Now, we ideally want to directly solve this system of equations and obtain the unique set of m_i values, which should allow us to obtain all parameters of the camera. We have 12 parameters to determine (including the scaling parameter for the projection). Potentially, we might want to feed in more pairs of corresponding positions to make this process more robust; however, that would result in more equations than coefficients to solve for, making the problem "over-constrained".

Furthermore, due to the uncertainties which arise in obtaining any of these positions data, it's very likely that this system of equations is not perfectly satisfied. Therefore, we will frame this as a minimization problem where we try to satisfy this system of equations as well as possible, with a non-trivial m coefficient vector. Note that, without the constraint of square norm of m equals 1, the minimization would result in trivial coefficients (all m_i being zeros).

$$min_{m \in R^{12}} \quad \|\tilde{P}m\|^2 \tag{5.28}$$

subject to

$$\|m\|^2 = 1 \tag{5.29}$$

This choice of the norm of m relies on the fact the a scalar multiple of the solution is still a solution of the original system of equations. Now, let's construct the Lagrangian first, by adding the cost and the Lagrange multiplier times the constraint:

$$L = m^T \tilde{P}^T \tilde{P} m + \lambda(1 - m^T m) \tag{5.30}$$

We take the gradient and set it to zero as the necessary condition of optimality:

$$\nabla L = 2\tilde{P}^T\tilde{P}m - 2\lambda m = 0 \rightarrow \tilde{P}^T\tilde{P}m = \lambda m \tag{5.31}$$

Now, it's obvious that the m we are looking for is an eigenvector of the $\tilde{P}^T\tilde{P}$. Furthermore, it is the eigenvector that has the smallest eigenvalue that would minimize the square norm.

After obtaining the M matrix upon solving the minimization problem, we will factorize M as

$$M = [KR \;\; Kt] \tag{5.32}$$

The first 3 columns of the matrix are the product of the intrinsic parameter matrix, which is upper triangular, and the rotation matrix, which is orthogonal. This calls for the efficient RQ factorization method, and we can obtain the parameters from the corresponding entries.

5.6 Radial Distortion

So far, there's another problem not being considered: radial distortion. For a real lens, the linear assumption will not hold any more. Either barrel distortion or pincushion distortion will exist and affect the real pixel coordinates.

Figure 5.6: Different kinds of distortions

There are many algorithms that address image distortion. A simple and efficient way is to set up relations between the ideal pixel position (u, v) and the distorted pixel position (u_d, v_d)

$$\begin{bmatrix} u_d \\ v_d \end{bmatrix} = \begin{bmatrix} u_d \\ v_d \end{bmatrix} (1 + kr^2) \begin{bmatrix} u - u_{cd} \\ v - v_{cd} \end{bmatrix} + \begin{bmatrix} u_{cd} \\ v_{cd} \end{bmatrix} \tag{5.33}$$

where k is the radial distortion factor that differs in different cameras and needs to be pre-calculated. $r^2 = (u - u_{cd})^2 + (v - v_{cd})^2$ is the square of the distance between the ideal pixel location and the center of distortion. (u_{cd}, v_{cd}) are the coordinates of the image center.

Although this algorithm is efficient, there are many more sophisticated tools which we will not address.

5.7 Measuring Depth

Now, once we know our parameters K, R, t, can we find out the corresponding coordinates in the real world if p on our image plane is known?

Unfortunately, the answer is no. That's because the equation $p^h = K[Rt]P_W^h$ is not revertible. We still need to decide an extra variable in the 3D world - the depth of P, Z_W. The only thing we are sure now is that P lies along the line connecting O and p.

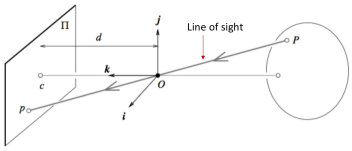

Figure 5.7: The depth of the real object is not known yet.

5.8 Recovering Structure

The previous discussion tells us that we can not simply determine the depth/structure from the pixel coordinates and camera parameters. We see this effect when we look at the picture below. We only know the man is not really stepping on the tower because of our own contextual clues or experience, not from the picture itself.

Figure 5.8: In the picture, the man seems to be stepping on the tower.

However, we can achieve the goal using other tools. This process is usually called structure recovering - we aim to reconstruct a 3D scene only with access to 2D images. Some common methods are listed below.

- Through recognition of landmarks (e.g., orthogonal walls)
- Depth from focus: determines distance to one point by taking multiple images with better and better focus
- Stereo vision: processes two distinct images taken at the same time and assumes that the relative pose between the two cameras is known
- Structure from motion: processes two images taken with the same or different cameras at different times and from different unknown positions

5.9 References

Relevant readings related to or referenced in the lecture include:

- SNS: 4.2.3
- D.A.Forsyth and J. Ponce[FP]. Computer Vision: A Modern Approach (2nd Edition). Prentice Hall, 2011. Chapter 1.
- R. Hartley and A. Zisserman [HZ]. Multiple View Geometry in Computer Vision. Academic Press, 2002. Chapter 6.1.
- Z. Zhang. A Flexible New Technique for Camera Calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2000.