Camera calibration

To estimate the geometry of 3D scenes, it helps to know the camera parameters, both external and internal. The problem of finding all these parameters is typically called *camera calibration.*

Recall at the end of lecture 3, we had a transformation $\mathbf{P}$ that takes a point $(X, Y, Z, 1)$ in world coordinates to a pixel position (wx, wy, w):

$$\mathbf{P} = \mathbf{KR}[\ \mathbf{I}\ |\ -\mathbf{c}]$$

where $\mathbf{K}$, $\mathbf{R}$, and $\mathbf{I}$ are 3×3 matrices, $\mathbf{c}$ is a 3×1 vector, and so $\mathbf{P}$ is 3×4. $\mathbf{P}$ is called a *finite projective camera.* Recall that it assumes a pinhole only.

Estimating P using least squares

How could we estimate $\mathbf{P}$? The standard method assumes we have a set of known points (X_i, Y_i, Z_i) in the scene. For example, we might have a cube of known size and whose faces have a checkerboard pattern on them (also of known size). The corners of the squares on the checkerboard would define 3D points in a world coordinate system defined by a corner of the cube.

If we take a photograph of the cube, we could then (by hand) find the pixel coordinates of the known points on the checkerboard, then we would have a set of constraints relating the pixel positions (x_i, y_i) and the scene positions (X_i, Y_i, Z_i)

$$x_i = \frac{wx_i}{w} = \frac{P_{11}X_i + P_{12}Y_i + P_{13}Z_i + P_{14}}{P_{31}X_i + P_{32}Y_i + P_{33}Z_i + P_{34}}$$

$$y_i = \frac{wy_i}{w} = \frac{P_{21}X_i + P_{22}Y_i + P_{23}Z_i + P_{24}}{P_{31}X_i + P_{32}Y_i + P_{33}Z_i + P_{34}}$$

and so

$$x_i(P_{31}X_i + P_{32}Y_i + P_{33}Z_i + P_{34}) = P_{11}X_i + P_{12}Y_i + P_{13}Z_i + P_{14}$$

$$y_i(P_{31}X_i + P_{32}Y_i + P_{33}Z_i + P_{34}) = P_{21}X_i + P_{22}Y_i + P_{23}Z_i + P_{24}$$

We arrange these equations using a $2N \times 12$ matrix such that

$$\begin{bmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -x_1X_1 & -x_1Y_1 & -x_1Z_1 & -x_1 \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -y_1X_1 & -y_1Y_1 & -y_1Z_1 & -y_1 \\ \vdots \\ \vdots \\ X_N & Y_N & Z_N & 1 & 0 & 0 & 0 & 0 & -x_NX_N & -x_NY_N & -x_NZ_N & -x_N \\ 0 & 0 & 0 & 0 & X_N & Y_N & Z_N & 1 & -y_NX_N & -y_NY_N & -y_NZ_N & -y_N \end{bmatrix} \begin{bmatrix} P_{11} \\ P_{12} \\ P_{13} \\ P_{14} \\ \vdots \\ P_{31} \\ P_{32} \\ P_{33} \\ P_{34} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$\mathbf{P}$ has 12 parameters, so if we want to solve for $\mathbf{P}$ then we need $N \geq 6$ corresponding point pairs i.e. we need 12 or more data values. In practice the values of $(x_i, y_i, X_i, Y_i, Z_i)$ will contain some measurement error and the projection model will only be an approximation to the actual physical situation, one typically tries to find a least squares solution and uses $N \gg 6$ points.

Recall that the matrix $\mathbf{P}$ is undefined up to scale, since it maps homogeneous points to homogeneous points. So we enforce a solution that $\parallel \mathbf{P} \parallel = 1$, where

$$\parallel \mathbf{P} \parallel = \sqrt{\sum_{ij} {P_{ij}}^2}$$

which is called the *Frobenius norm.* Notice that we just have a (version 1) least squares problem – see lecture 18. The matrix $\mathbf{A}$ is the $2N \times 12$ matrix above, and so $\mathbf{A}^T\mathbf{A}$ is 12×12. We would like the eigenvector $\mathbf{A}^T\mathbf{A}$ with smallest eigenvalue. To get it, we either compute it directly, or we take the SVD of $\mathbf{A}$ and take the column of $\mathbf{V}$ with the smallest singular value.

Will this least squares problem give us a good solution? Not necessarily. Why not?

- (not discussed in class) It turns out that if the 3D data all lie on one plane, then the $\mathbf{A}$ matrix may have multiple zero eigenvalues. In this case, we have a multidimensional subspace of solutions for $\mathbf{P}$.

- The error we are minimizing in the above least squares solution is algebraically convenient, but it is not obvious how to interpret it in terms of the geometry of the situation. In class, I discussed how to use the above estimate of $\mathbf{P}$ as an initial estimate for a *non-linear* least squares formulation (see last lecture), namely we re-estimate $\mathbf{P}$ by trying to minimize the sum of squared errors:
$$\sum_i \{(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2\}$$
where (see page 1)
$$\hat{x}_i = \frac{P_{11}X_i + P_{12}Y_i + P_{13}Z_i + P_{14}}{P_{31}X_i + P_{32}Y_i + P_{33}Z_i + P_{34}}$$
$$\hat{y}_i = \frac{P_{21}X_i + P_{22}Y_i + P_{23}Z_i + P_{24}}{P_{31}X_i + P_{32}Y_i + P_{33}Z_i + P_{34}}.$$
The idea is to iteratively adjust the $\mathbf{P}_{ij}$ values to reduce the sum of squared errors. This least squares method is more geometrically meaningful, since the errors are the errors in pixel position.

Factoring P into internal and external camera parameters

We would now like to decompose $\mathbf{P}$ into the product of a 3×3 upper triangular matrix $\mathbf{K}$ and a 3×4 matrix $\mathbf{R}[\mathbf{I}| - \mathbf{c}]$. As shown in the following Appendix, this can be done using the *RQ decomposition*, which is a technique from linear algebra. (In the RQ decomposition, R refers to a right upper triangular matrix (everything below the diagonal is zero) and Q refers to an orthonormal matrix. Matlab doesn't implement the RQ decomposition. It only implements the QR decomposition, which is more standard.)

You will not be examined on this Appendix, but the method is very nice so I encourage you to read through it to get the idea.

Appendix

Step 1: normalize P

We note that the elements $(P_{31} P_{32} P_{33})$ are not all zero. (Recall from Exercise 1 Question 10d that this 3-vector is normal to the projection plane. It cannot be all 0's if we indeed have a projection matrix.) The first step is to divide each element of $\mathbf{P}$ by $\| (P_{31}, P_{32}, P_{33}) \|$ so that this vector is a unit normal.

For the next steps, we consider only the left 3×3 submatrix of $\mathbf{P}$ whose 3rd row is now of length 1. Let $\tilde{\mathbf{P}}$ be the 3×3 matrix which is the first three columns of $\mathbf{P}$. We decompose $\tilde{\mathbf{P}}$ into $\mathbf{KR}$. To to this, we want to find an orthonormal matrix $\mathbf{R}$ such that $\tilde{\mathbf{P}}\mathbf{R}^T = \mathbf{K}$.

Step 2: find R

We will construct a rotation matrix $\mathbf{R}$ in three steps. First, we find a rotation matrix $\mathbf{R}_{Z,\theta}$ such that $\tilde{\mathbf{P}}\mathbf{R}_{Z,\theta}$ has its $(3,1)$ element equal to zero. Define

$$\mathbf{R}_{Z,\theta} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

and choose θ such that

$$P_{31}\cos\theta - P_{32}\sin\theta = 0$$

and so we need

$$\theta = \arctan(P_{31}/P_{32}).$$

Note that a Z-rotation does not affect the 3rd column of $\tilde{\mathbf{P}}$.

Second, we find a rotation matrix

$$\mathbf{R}_{X,\beta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\beta & \sin\beta \\ 0 & -\sin\beta & \cos\beta \end{bmatrix}$$

such that $\mathbf{P}\mathbf{R}_{z,\theta}\mathbf{R}_{X,\beta}$ has its (3,1) and (3,2) elements equal to 0. Using the same method as above, take

$$\beta = \arctan\ (\ \frac{(\tilde{\mathbf{P}}\mathbf{R}_{Z,\theta})_{3,2}}{(\tilde{\mathbf{P}}\mathbf{R}_{Z,\theta})_{33}}\ ).$$

Note that this rotation does not affect the first column, and so the last row of $\tilde{\mathbf{P}}\mathbf{R}_{z,\theta}\mathbf{R}_{X,\beta}$ must be $(0, 0, \pm 1)$. Also, recall that we began by normalizing such that the last row of $\tilde{\mathbf{P}}$ was of unit length. The two rotations above will not change this property since rotations preserve length.

Third, we find a rotation matrix $\mathbf{R}_{Z,\gamma}$ which sets element $(2,1)$ to 0. Again this is done by a suitable choice of rotation angle, namely

$$\gamma = \arctan\ (\ \frac{(\tilde{\mathbf{P}}\mathbf{R}_{Z,\theta}\mathbf{R}_{X,\beta})_{2,1}}{(\tilde{\mathbf{P}}\mathbf{R}_{Z,\theta}\mathbf{R}_{X,\beta})_{2,2}}\ ).$$

Note that this X rotation doesn't affect the third row since its first two elements are 0.

We now have that $\tilde{\mathbf{P}}\mathbf{R}_{Z,\theta}\ \mathbf{R}_{X,\beta}\ \mathbf{R}_{Z,\gamma}$. is an upper triangular 3×3 matrix. Since the 3rd row this matrix has length 1, and since elements $(3,1)$ and $(3,2)$ are 0, element $(3,3)$ must be either 1 or -1.

Finally, we would like to transform this into the **K** matrix, and so we need the diagonal elements (1,1), (2,2) and (3,3) to be non-negative. If any of these diagonal elements of $\tilde{\mathbf{P}}\mathbf{R}_{Z,\theta}\ \mathbf{R}_{X,\beta}\ \mathbf{R}_{Z,\gamma}$ are negative, then we need to reflect about the X, Y, Z axis, and so we have

$$\mathbf{K} = \tilde{\mathbf{P}}\mathbf{R}_{Z,\theta}\ \mathbf{R}_{X,\beta}\ \mathbf{R}_{Z,\gamma} \begin{bmatrix} \pm 1 & 0 & 0 \\ 0 & \pm 1 & 0 \\ 0 & 0 & \pm 1 \end{bmatrix}.$$

This gives us $\mathbf{K} = \tilde{\mathbf{P}}\mathbf{Q}$ where $\mathbf{Q}$ is orthonormal, and so $\tilde{\mathbf{P}} = \mathbf{K}\mathbf{Q}^T = \mathbf{K}\mathbf{R}$.

Step 3: obtain camera position c

Since **K R (-c)** is the fourth column of **P**, we can obtain **-c** by multiplying

$$\mathbf{R}^T\mathbf{K}^{-1} \begin{bmatrix} P_{14} \\ P_{24} \\ P_{34} \end{bmatrix} = -\mathbf{c}$$

i.e. $\mathbf{R}^T$ is a rotation matrix, so $\mathbf{R}^T = \mathbf{R}^{-1}$.

We next to a related topic: homographies.

Homographies

Let's now look at another image transformation, called a *homography*, which arises very often in real scenes, both manmade and natural.

Case 1: scene plane to image pixels

Suppose we have a planar surface in the world (e.g. a wall, a ground plane) and we view it with a camera with projection matrix $\mathbf{P}$. The planar surface is 2D and so we can give it a coordinate system (s,t). Points on the plane are situated in the 3D world and their 3D positions can be expressed in XYZ *world coordinates*. The transformation from (s,t) coordinates to world coordinates XYZ can be obtained by multiplying $(s,t,1)$ by a 4×3 matrix, as follows:

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} a_x & b_x & X_0 \\ a_y & b_y & Y_0 \\ a_z & b_z & Z_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s \\ t \\ 1 \end{bmatrix} \tag{1}$$

The first two columns can be interpreted as direction vectors corresponding to the coordinate system's basis vectors, namely where $(1,0,0)$ and $(0,1,0)$ are mapped to. The third column is the 3D world coordinate position of the *origin* of the plane, i.e. $(s,t) = (0,0)$. Note that this mapping takes the origin $(s,t) = (0,0)$ to (X_0, Y_0, Z_0); it takes the corner $(s,t) = (0,1)$ to $(X_0+b_x, Y_0+b_y, Z_0+b_z)$, and it takes the corner $(s,t) = (1,0)$ to $(X_0 + a_x, Y_0 + a_y, Z_0 + a_z)$, etc.

The image pixel (x,y) corresponding to a point (s,t) in the scene plane is obtained by

$$\begin{bmatrix} wx \\ wy \\ w \end{bmatrix} = \mathbf{P} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{P} \begin{bmatrix} a_x & b_x & X_0 \\ a_y & b_y & Y_0 \\ a_z & b_z & Z_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s \\ t \\ 1 \end{bmatrix}$$

The mappings from $(s,t,1)'$ to $(wx,wy,w)'$ defines a 3×3 matrix,

$$\mathbf{H} == \mathbf{P} \begin{bmatrix} a_x & b_x & X_0 \\ a_y & b_y & Y_0 \\ a_z & b_z & Z_0 \\ 0 & 0 & 1 \end{bmatrix}.$$

In general, a 3×3 matrix $\mathbf{H}$ that is invertible is called a *homography*.

Note that the inverse of this mapping is:

$$w' \begin{bmatrix} s \\ t \\ 1 \end{bmatrix} = \mathbf{H}^{-1} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

The inverse is well-defined as long as the camera does not lie in the plane. That is a geometric description of the invertibility condition.

ASIDE: on the invertibility of H

There was a bit of confusion in class about how to define this "invertibility" *algebraically*. The reason for the confusion is that I neglected the fact that Eq. (1) above defines XYZ in world coordinates, i.e. independent of the camera. But to define invertibility, we need to say something about the camera and so we need to use matrix $\mathbf{P}$. Here is how we do it.

Recall $\mathbf{P} = \mathbf{K}[\mathbf{I}| - \mathbf{c}]$. Since $\mathbf{K}$ is automatically invertible, the matrix $\mathbf{H}$ will fail to be invertible exactly when

$$[\,\mathbf{I}\mid -\mathbf{c}]\begin{bmatrix} a_x & b_x & X_0 \\ a_y & b_y & Y_0 \\ a_z & b_z & Z_0 \\ 0 & 0 & 1 \end{bmatrix}$$

fails to be invertible, or equivalently that the 3×3 matrix

$$\begin{bmatrix} a_x & b_x & X_0 - c_x \\ a_y & b_y & Y_0 - c_y \\ a_z & b_z & Z_0 - c_y \end{bmatrix}$$

is not invertible, which means that the columns of that last matrix are linear dependent, which means in particular that the vector from the camera to the 3D origin of the plane, i.e. corresponding to $(s,t) = (0,0)$, lies in the plane spanned by vectors $\mathbf{a}$ and $\mathbf{b}$. Phew!

Homography 2 (two cameras, one scene plane)

Suppose the same scene plane is viewed by a second camera, which would have a different $\mathbf{P}$ matrix. We would now have two homographies $\mathbf{H}_1$ and $\mathbf{H}_2$, defined by the two cameras. This implies that the composite mapping $\mathbf{H}_2\mathbf{H}_1^{-1}$ maps pixels in the first camera to pixels in the second camera. That is, each camera defines a homography of the form

$$\mathbf{H}^{-1}\begin{bmatrix} wx \\ wy \\ w \end{bmatrix} = \begin{bmatrix} s \\ t \\ 1 \end{bmatrix}$$

but the right side is the same for both (since it is independent of the camera), so we just equate the left sides for the two cameras.

$$\mathbf{H}_2^{-1}\begin{bmatrix} w_2x_2 \\ w_2y_2 \\ w_2 \end{bmatrix} = \mathbf{H}_1^{-1}\begin{bmatrix} w_1x_1 \\ w_1y_1 \\ w_1 \end{bmatrix}$$

and so

$$\begin{bmatrix} w_2x_2 \\ w_2y_2 \\ w_2 \end{bmatrix} = \mathbf{H}_2\mathbf{H}_1^{-1}\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

Note that this construction relies critically on the scene being a planar surface.