CHAPTER 12

Registration

Computing a transformation that aligns an image or a depth map or a set of images with another such is generally known as *registration*. One approach to registration is to abstract the image (etc.) as a set of points, often referred to as *point clouds*. Generally, we will write $\mathcal{P}$ for a point cloud whose i'th point is $\mathbf{p}_i$ and so on. Now assume we have two point clouds, $\mathcal{X}$ and $\mathcal{Y}$. Each is obtained by starting with a set of reference points, dropping some of them at random, transforming the remaining points, then adding noise to the points and also including some pure noise points. This is a reasonable model of what a depth camera or a LIDAR sensor might produce when it views an object. The model means that there is transformation that maps from one to the other, though it may not place every point in one on top of some point in the other. We want to determine the transformation between the two point clouds.

12.1 PRELIMINARIES

Registration occurs in a wide range of practical applications. As we shall see, calibrating a camera involves solving a version of this problem (Section 35.2). Determining where you are in a known map very often involves solving a version of this problem. Imagine, for example, a camera looking directly downwards from an aircraft flying at fixed height. The image in the camera translates and rotates as the aircraft moves. If we can compute the transformation from image i to image $i+1$, we can tell how the aircraft has moved. Another useful case occurs when we have a depth map of a known object and want to compute the *pose* of the object (its position and orientation in the frame of the depth sensor). We could do so by reducing the object model to a point cloud, then computing the transformation from the object model to the point cloud from the depth sensor.

How one approaches this class of problem depends on three important factors.

- **Correspondence:** if it is known *which* observation corresponds to *which* reference point, the problem is relatively straightforward to solve (unless there are unusual noise effects). If correspondence is not known, computing the transformation becomes rather harder.

- **Transformation:** there are closed form solutions for known correspondence and some kinds of transformation. When a closed form solution is known, you should use it. In other cases, we are forced to use some kind of optimization procedure.

- **Robustness:** computing a transformation can become very hard if many of the observations do not come from reference points, if many of the reference points are dropped, or if some observations are subject to very large noise effects.

12.1.1 Types of Transformation

Generally, we are going to transform points in $\mathcal{Y}$ to lie on $\mathcal{X}$. These points could be in either 3D or in 2D (or others – very little of what we will do depends on dimension). We will consider a variety of types of affine transformation and of projective transformations.

Affine transformations transform $\mathbf{y}$ to $\mathbf{x} = \mathcal{M}\mathbf{y} + \mathbf{t}$, where $\mathcal{M}$ has non-zero determinant. Some kinds of affine transform have specialized names:

- **Translation:** when $\mathcal{M}$ is the identity.
- **Rotation:** when $\mathbf{t} = \mathbf{0}$; $\mathcal{M}^T\mathcal{M}$ is the identity and $\mathcal{M}$ has positive determinant.
- **Homogenous scaling:** when $\mathcal{M}$ is σ times the identity, and $\sigma \neq 0$.
- **Scaling:** when $\mathcal{M}$ is diagonal.
- **Euclidean or rigid body:** when $\mathcal{M}^T\mathcal{M}$ is the identity and $\mathcal{M}$ has positive determinant.

Projective transformations are a new class. We will see much more of the geometry underlying projective transformations later. A projective transformation in N dimensions is given by an $N+1 \times N+1$ dimensional matrix $\mathcal{P}$ with non-zero determinant. There are a number of different ways of representing the effect of a projective transformation. For now, we will write an N D projective transformation as

$$\begin{pmatrix} \mathcal{M} & \mathbf{v} \\ \mathbf{m}_{N+1}^T & v_{N+1} \end{pmatrix} = \begin{pmatrix} \mathbf{m}_1^T & v_1 \\ \dots & \\ \mathbf{m}_N^T & v_N \\ \mathbf{m}_{N+1}^T & v_{N+1} \end{pmatrix}$$

where $\mathcal{M}$ is $N \times N$, and the vectors are $N \times 1$. This transformation takes $\mathbf{y}$ to

$$\mathbf{x} = \begin{bmatrix} \frac{\mathbf{m}_1^T\mathbf{y}+m_1}{\mathbf{m}_{N+1}^T\mathbf{y}+m_{N+1}} \\ \frac{\mathbf{m}_2^T\mathbf{y}+m_2}{\mathbf{m}_{N+1}^T\mathbf{y}+m_{N+1}} \\ \dots \\ \frac{\mathbf{m}_N^T\mathbf{y}+m_N}{\mathbf{m}_{N+1}^T\mathbf{y}+m_{N+1}} \end{bmatrix}.$$

Notice that there could be a divide by zero issue here. For the moment, we will ignore this and assume it never happens. In fact, a great deal of interesting geometry follows from paying attention to this issue, as we shall see in Chapter 35.2.

Notice that every affine transformation is a projective transformation (set $\mathbf{m}_N = 0$ and $m_{N+1,N+1} = 1$). As we shall see, a projective transformation from 2D to 2D is a model of what happens when a plane is viewed in a camera; this model is sometimes referred to as a *homography.*

12.1.2 Correspondence

In the simplest case, correspondences are known (i.e. we know *which* point in $\mathcal{Y}$ comes from *which* point in $\mathcal{X}$). This case occurs in applications.

Image registration consists of finding a transformation that places a part of one image $\mathcal{A}$ on top of part of an image $\mathcal{B}$. We did the simplest cases in Chapter **??**), but we now have more sophisticated tools. We could register images by finding interest points, computing local coordinate systems and descriptors, then matching interest points in $\mathcal{A}$ to those in $\mathcal{B}$. The descriptor is a vector, and we have constructed the descriptor to have two important properties: the descriptor for the same interest point in different images should be similar; and different interest points should have different descriptors. We could then match by finding nearest neighbors. In particular, for each interest point in $\mathcal{A}$ (write the feature vector of the i'th as $\mathbf{a}_i$), find the interest point in $\mathcal{B}$ whose feature vector is most similar. Assume that this is the j'th point in $\mathcal{B}$, so that $(\mathbf{a}_i - \mathbf{b}_j)^T(\mathbf{a}_i - \mathbf{b}_j)$ is smaller than $(\mathbf{a}_i - \mathbf{b}_k)^T(\mathbf{a}_i - \mathbf{b}_k)$ for $k \neq j$. Now check the matching is *symmetric* – if $\mathbf{b}_j$ is closest to $\mathbf{a}_i$ in $\mathcal{B}$, then $\mathbf{a}_i$ should be the closest to $\mathbf{b}_j$ in $\mathcal{A}$. Now look at pairs where the matching is symmetric *and* the distance between matched feature vectors is small – these are likely to be corresponding points if the distance threshold is small enough.

Localization and mapping by registration is common in robotics. A *beacon* is an object that identifies itself (perhaps by wearing a barcode; by transmitting some code; by a characteristic pattern) and can be localized. Place beacons at various points in an environment, and record where those beacons are in a map. Now a robot observes those beacons, but in its own coordinate system because its sensors are attached to it. If this robot can compute the transformation that registers its observations to the map, it knows where it is. In fact, the robot can do more. Assume you lose the map, or fail to record beacon locations. If the robot explores and sees enough beacons in each observation, it can register the beacons to one another. Once it has done so, it has a map, and can now register itself to that map (and so determine where it is within that map).

12.2 REGISTRATION WITH KNOWN CORRESPONDENCE AND GAUSSIAN NOISE

12.2.1 Affine Transformations and Gaussian Noise

In the simplest case, the correspondence is known – perhaps $\mathcal{Y}$ consists of beacons and $\mathcal{X}$ of observations – and the only noise is Gaussian (so $N = M$). We will assume the noise is isotropic, which is by far the most usual case. Once you have followed this derivation, you will find it easy to incorporate a known covariance matrix. We have

$$\mathbf{x}_i = \mathcal{M}\mathbf{y}_i + \mathbf{t} + \xi_i \tag{12.1}$$

where ξ_i is the value of a normal random variable with mean $\mathbf{0}$ and covariance matrix $\Sigma = \sigma^2\mathcal{I}$. A natural procedure to estimate $\mathcal{M}$ and $\mathbf{t}$ is to maximize the likelihood of the noise. Because it will be useful later, we assume that there is a weight w_i for each pair, so the negative log-likelihood we must minimize is proportional to

$$\sum_i w_i \left(\mathbf{x}_i - \mathcal{M}\mathbf{y}_i - \mathbf{t}\right)^T \left(\mathbf{x}_i - \mathcal{M}\mathbf{y}_i - \mathbf{t}\right) \tag{12.2}$$

(the constant of proportionality is σ^2, which doesn't affect the optimization problem). The gradient of this cost with respect to $\mathbf{t}$ is

$$-2\sum_i w_i \left(\mathbf{x}_i - \mathcal{M}\mathbf{y}_i - \mathbf{t}\right) \tag{12.3}$$

which vanishes at the solution. In turn, if $\sum_i w_i\mathbf{x}_i = \sum_i w_i\mathcal{M}\mathbf{y}_i$, $\mathbf{t} = \mathbf{0}$. One straightforward way to achieve this is to ensure that both the observations and the reference points have a center of gravity at the origins. Write

$$\mathbf{c}_x = \frac{\sum_i w_i\mathbf{x}_i}{\sum_i w_i} \tag{12.4}$$

for the center of gravity of the observations (etc.) Now form

$$\mathbf{u}_i = \mathbf{x}_i - \mathbf{c}_x \text{ and } \mathbf{v}_i = \mathbf{y}_i - \mathbf{c}_y \tag{12.5}$$

and if we use $\mathcal{U}$ and $\mathcal{V}$, then the translation will be zero. In turn, the translation from the original reference points to the original observations is $\mathbf{c}_x - \mathbf{c}_y$.

We obtain $\mathcal{M}$ by minimizing

$$\sum_i w_i \left(\mathbf{u}_i - \mathcal{M}\mathbf{v}_i\right)^T \left(\mathbf{u}_i - \mathcal{M}\mathbf{v}_i\right). \tag{12.6}$$

Now write $\mathcal{W} = \mathsf{diag}\left([w_1, \ldots, w_N]\right)$, $\mathcal{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_N]$ (and so on). You should check that the objective can be rewritten as

$$\mathsf{Tr}\left((\mathcal{U} - \mathcal{M}\mathcal{V})^T\mathcal{W}(\mathcal{U} - \mathcal{M}\mathcal{V})\right). \tag{12.7}$$

Now the trace is linear; $\mathcal{U}^T\mathcal{U}$ is constant; and $\mathsf{Tr}\left(\mathcal{ABC}\right) = \mathsf{Tr}\left(\mathcal{BCA}\right) = \mathsf{Tr}\left(\mathcal{CAB}\right)$ (check this by writing it out, and remember it; it's occasionally useful; more in Section 35.2). This means the cost is equivalent to

$$\mathsf{Tr}\left(-2\mathcal{M}\mathcal{V}\mathcal{W}\mathcal{U}^T + \mathcal{M}^T\mathcal{M}\mathcal{V}\mathcal{W}\mathcal{V}^T\right) \tag{12.8}$$

which will be minimized when

$$\mathcal{M}\mathcal{V}\mathcal{W}\mathcal{V}^T = \mathcal{V}\mathcal{W}\mathcal{U}^T \tag{12.9}$$

(which you should check). Many readers will recognize a least squares solution here. The trace isn't necessary here, but it's helpful to see an example using the trace, because it will be important in the next case.

12.2.2 Euclidean Motion and Gaussian Noise

One encounters affine transformations relatively seldom in practice, though they do occur. Much more interesting is the case where the transformation is Euclidean. The least squares solution above isn't good enough, because the $\mathcal{M}$ obtained that way won't be a rotation matrix. But we can obtain a least squares solution with a rotation matrix, using a neat trick. We adopt the notation of the previous section, and change coordinates from $\mathbf{x}_i$ to $\mathbf{u}_i$ as above to remove the need to estimate translation.

We must choose $\mathcal{R}$ to minimize

$$\sum_i w_i(\mathbf{u}_i - \mathcal{R}\mathbf{v}_i)^T(\mathbf{u}_i - \mathcal{R}\mathbf{v}_i). \tag{12.10}$$

This can be done in closed form (a fact you should memorize). Equivalently, we must minimize

$$\begin{aligned}\sum_i w_i(\mathbf{u}_i - \mathcal{R}\mathbf{v}_i)^T(\mathbf{u}_i - \mathcal{R}\mathbf{v}_i) &= \mathsf{Tr}\left(\mathcal{W}(\mathcal{U} - \mathcal{R}\mathcal{V})(\mathcal{U} - \mathcal{R}\mathcal{V})^T\right) \\ &= \mathsf{Tr}\left(-2\mathcal{U}\mathcal{W}\mathcal{V}^T\mathcal{R}^T\right) + K \\ &\quad \text{(because } \mathcal{R}^T\mathcal{R} = \mathcal{I}\text{)} \\ &= -2\mathsf{Tr}\left(\mathcal{R}\mathcal{U}\mathcal{W}\mathcal{V}^T\right)\end{aligned}$$

Now we compute an SVD of $\mathcal{U}\mathcal{V}^T$ to obtain $\mathcal{U}\mathcal{W}\mathcal{V}^T = \mathcal{A}\mathcal{S}\mathcal{B}^T$ (where $\mathcal{A}$, $\mathcal{B}$ are orthonormal, and $\mathcal{S}$ is diagonal – Section 35.2 if you're not sure). Now $\mathcal{B}^T\mathcal{R}\mathcal{A}$ is orthonormal, and we must maximize $\mathsf{Tr}\left(\mathcal{B}^T\mathcal{R}\mathcal{A}\mathcal{S}\right)$, meaning $\mathcal{B}^T\mathcal{R}\mathcal{A} = \mathcal{I}$ (check this if you're not certain), and so $\mathcal{R} = \mathcal{B}\mathcal{A}^T$.

Procedure: 12.1 *Weighted Least Squares for Euclidean Transformations*

We have N reference points $\mathbf{x}_i$ whose location is measured in the agent's coordinate system. Each corresponds to a point in the world coordinate system with known coordinates $\mathbf{y}_i$, and the change of coordinates is a Euclidean transformation (rotation $\mathcal{R}$, translation $\mathbf{t}$). For each $(\mathbf{x}_i, \mathbf{y}_i)$ pair, we have a weight w_i. We wish to minimize

$$\sum_i w_i(\mathbf{x}_i - \mathcal{R}\mathbf{y}_i - \mathbf{t})^T(\mathbf{x}_i - \mathcal{R}\mathbf{y}_i - \mathbf{t}) \tag{12.11}$$

Write

$$\begin{aligned}
\mathbf{c}_x &= \frac{\sum_i w_i \mathbf{x}_i}{\sum_i w_i} \\
\mathbf{c}_y &= \frac{\sum_i w_i \mathbf{y}_i}{\sum_i w_i} \\
\mathbf{u}_i &= \mathbf{x}_i - \mathbf{c}_x \\
\mathbf{v}_i &= \mathbf{y}_i - \mathbf{c}_y
\end{aligned}$$

Then the least squares estimate $\hat{\mathbf{t}}$ of $\mathbf{t}$ is

$$\hat{\mathbf{t}} = \mathbf{c}_x - \mathbf{c}_y \tag{12.12}$$

Write $\mathcal{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_N]$ (etc); $\mathcal{W} = \text{diag}(w_1, \ldots, w_N)$; and $\text{SVD}(\mathcal{USV}) = \mathcal{A}\Sigma\mathcal{B}^T$. The least squares estimate $\hat{\mathcal{R}}$ is

$$\hat{\mathcal{R}} = \mathcal{B}\mathcal{A}^T \tag{12.13}$$

12.2.3 Homographies and Gaussian Noise

We now work with points on the plane, and allow the transformation to be a homography. Solving for a homography requires solving an optimization problem, but estimating a homography from data is useful, and relatively easy to do. We can't recover the translation component from centers of gravity (exercises **TODO:** homography exercise). Write m_{ij} for the i, j'th element of matrix $\mathcal{M}$. In affine coordinates, a homography $\mathcal{M}$ will map $\mathbf{y}_i = (y_{i,x}, y_{i,y})$ to $\mathbf{x}_i = (x_{i,x}, x_{i,y})$ where

$$x_{i,x} = \frac{m_{11}y_{i,x} + m_{12}y_{i,y} + m_{13}}{m_{31}y_{i,x} + m_{32}y_{i,y} + m_{33}} \text{ and } x_{i,y} = \frac{m_{21}y_{i,x} + m_{22}y_{i,y} + m_{23}}{m_{31}y_{i,x} + m_{32}y_{i,y} + m_{33}} \tag{12.14}$$

Write $\mathcal{M}(\mathbf{y})$ for the result of applying the homography to $\mathbf{y}$ as above. In most cases of interest, the coordinates of the points are not measured precisely, so we observe $\mathbf{x}_i = \mathcal{M}(\mathbf{y}_i) + \xi_i$, where ξ_i is some noise vector drawn from an isotropic normal distribution with mean $\mathbf{0}$ and covariance Σ. Again, assume that the noise is isotropic, and so that $\Sigma = \sigma^2\mathcal{I}$. The homography can be estimated by minimizing the negative log-likelihood of the noise, so we must minimize

$$\sum_i w_i \xi_i^T \xi_i \tag{12.15}$$

where

$$\xi_i = \left[\begin{array}{c} x_{i,x} - \frac{m_{11}y_{i,x}+m_{12}y_{i,y}+m_{13}}{m_{31}y_{i,x}+m_{32}y_{i,y}+m_{33}} \\ x_{i,y} - \frac{m_{21}y_{i,x}+m_{22}y_{i,y}+m_{23}}{m_{31}y_{i,x}+m_{32}y_{i,y}+m_{33}} \end{array} \right] \tag{12.16}$$

using standard methods (Levenberg-Marquardt is favored; Chapter 35.2). This approach is sometimes known as *maximum likelihood* . Experience teaches that this optimization is not well behaved without a strong start point.

There is an easy construction for a good start point. Notice that the equations for the homography mean that

$$x_{i,x}(m_{31}y_{i,x} + m_{32}y_{i,y} + m_{33}) - m_{11}y_{i,x} + m_{12}y_{i,y} + m_{13} = 0 \tag{12.17}$$

and

$$x_{i,y}(m_{31}y_{i,x} + m_{32}y_{i,y} + m_{33}) - m_{21}y_{i,x} + m_{22}y_{i,y} + m_{23} = 0 \tag{12.18}$$

so each corresponding pair of points $\mathbf{x}_i$, $\mathbf{y}_i$ yields two *homogeneous* linear equations in the coefficients of the homography. They are homogeneous because scaling $\mathcal{M}$ doesn't change what it does to points (check this if you're uncertain). If we obtain sufficient points, we can solve the resulting system of homogeneous linear equations. Four point correspondences yields an unambiguous solution; more than four – which is better – can be dealt with by least squares (exercises **TODO:** fourpoint homography). The resulting estimate of $\mathcal{M}$ has a good reputation as a start point for a full optimization.

Procedure: 12.2 *Estimating a Homography from Data*

Given N known source points in 2D (write $\mathbf{y}_i = (y_{i,x}, y_{i,y})$) and N corresponding target points $\mathbf{x}_i$ with measured locations $(x_{i,x}, x_{i,y})$ and where measurement noise has zero mean and covariance $\Sigma = \sigma^2\mathcal{I}$, estimate the homography $\mathcal{M}$ with i, j'th element m_{ij} by minimizing:

$$\sum_i \xi_i^T \xi_i \tag{12.19}$$

where

$$\xi = \left[\begin{array}{c} x_{i,x} - \frac{m_{11}y_{i,x}+m_{12}y_{i,y}+m_{13}}{m_{31}y_{i,x}+m_{32}y_{i,y}+m_{33}} \\ x_{i,y} - \frac{m_{21}y_{i,x}+m_{22}y_{i,y}+m_{23}}{m_{31}y_{i,x}+m_{32}y_{i,y}+m_{33}} \end{array} \right] \tag{12.20}$$

Obtain a start point by as a least-squares solution to the set of homogeneous linear equations

$$x_{i,x}(m_{31}y_{i,x} + m_{32}y_{i,y} + m_{33}) - m_{11}y_{i,x} + m_{12}y_{i,y} + m_{13} = 0 \tag{12.21}$$

and

$$x_{i,y}(m_{31}y_{i,x} + m_{32}y_{i,y} + m_{33}) - m_{21}y_{i,x} + m_{22}y_{i,y} + m_{23} = 0. \tag{12.22}$$

12.2.4 Projective Transformations and Gaussian Noise

A *projective transformation* is the analogue of a homography for higher dimensions. In affine coordinates, a projective transformation $\mathcal{M}$ will map $\mathbf{y}_i = (y_{i,1}, \ldots, y_{i,d})$ to $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,d})$ where

$$x_{i,1} = \frac{m_{11}y_{i,1} + \ldots + m_{1d}y_{i,d} + m_{1(d+1)}}{m_{(d+1)1}y_{i,1} + \ldots + m_{(d+1)d}y_{i,d} + m_{(d+1)(d+1)}} \tag{12.23}$$

and

$$x_{i,d} = \frac{m_{d1}y_{i,1} + \ldots + m_{dd}y_{i,d} + m_{d(d+1)}}{m_{(d+1)1}y_{i,1} + \ldots + m_{(d+1)d}y_{i,d} + m_{(d+1)(d+1)}} \tag{12.24}$$

Estimating this transformation follows the recipe for a homography, but there are now more parameters. I have put the result in a box, below.

Procedure: 12.3 *Estimating a Projective Transformation from Data*

Given N known source points $\mathbf{y}_i = (y_{i,1}, \ldots, y_{i,d})$ in affine coordinates and N corresponding target points $\mathbf{x}_i$ with measured locations $(x_{i,1}, \ldots, x_{i,d})$ and where measurement noise has zero mean and is isotropic, the homography $\mathcal{M}$ with i, j'th element m_{ij} by minimizing:

$$\sum_i \xi_i^T \Sigma^{-1} \xi_i \tag{12.25}$$

where

$$\xi_i = \begin{bmatrix} x_{i,1} - \frac{m_{11}y_{i,1}+\ldots+m_{1d}y_{i,d}+m_{1(d+1)}}{m_{(d+1)1}x_{i,1}+\ldots+m_{(d+1)d}x_{i,d}+m_{(d+1)(d+1)}} \\ \ldots \\ x_{i,d} - \frac{m_{d1}y_{i,1}+\ldots+m_{dd}y_{i,d}+m_{d(d+1)}}{m_{(d+1)1}x_{i,1}+\ldots+m_{(d+1)d}x_{i,d}+m_{(d+1)(d+1)}} \end{bmatrix} \tag{12.26}$$

Obtain a start point by as a least squares solution to the set of homogeneous linear equations

$$\begin{aligned} 0 &= x_{i,1}(m_{(d+1)1}y_{i,1} + \ldots + m_{(d+1)d}y_{i,d} + m_{(d+1)(d+1)}) - \\ &\quad m_{11}y_{i,1} + \ldots + m_{1d}y_{i,d} + m_{1(d+1)} \\ \ldots & \\ 0 &= x_{i,d}(m_{(d+1)1}y_{i,1} + \ldots + m_{(d+1)d}y_{i,d} + m_{(d+1)(d+1)}) - \\ &\quad m_{(d+1)1}x_{i,1} + \ldots + m_{(d+1)d}x_{i,d} + m_{(d+1)(d+1)} \end{aligned}$$

12.3 REGISTRATION AT LARGE SCALES

You are on vacation in some famous, but unfamiliar, city. A majestic landmark presents itself - but what is it called? There might be a plaque at the base, but there might not. If it's that majestic, someone else has (a) photographed it and (b) labelled the photograph, so you could resolve the question by taking a picture of the landmark, and registering that picture to a collection of labelled pictures. There are likely a lot of images in that collection – even in one city, you'd expect lots of different views each of a lot of majestic landmarks. This means that simply registering to each image in the collection one-by-one would not work.