Last lecture we discussed epipolar geometry and the fundamental matrix. Today we begin by discussing how to estimate the fundamental matrix. More generally we will discuss how to find corresponding points between the left and right images and how to infer depth.

Estimation of Fundamental matrix

How can we estimate the fundamental matrix that relates two images? Suppose we find a pair of matching points (x_1, y_1) and (x_2, y_2) in the first and second camera's image, respectively and write them in homogeneous coordinates with a 1 in the third position. Then from last lecture,

$$\tilde{\mathbf{x}}_2^T \mathbf{F} \tilde{\mathbf{x}}_1 = 0$$

where the $\tilde{\mathbf{x}}$ notation reminded us that we were dealing with pixel coordinates. *We'll drop that notation for the rest of the lecture today since we'll always be in pixel coordinates.*

Multiplying out the above equation gives us the following constraint on the nine $\mathbf{F}_{ij}$ elements.

$$(x_1x_2,\ y_1x_2,\ x_2,\ x_1y_2,\ y_1y_2,\ y_2,\ x_1,\ y_1,\ 1) \cdot (F_{11},\ F_{12},\ F_{13},\ F_{21},\ F_{22},\ F_{23},\ F_{31},\ F_{32},\ F_{33}) = 0$$

If we have N of these pairs, we can index them by i and stack the coefficients into an $N \times 9$ data matrix $\mathbf{A}$ as follows:

$$\begin{bmatrix} x_1^1x_2^1 & y_1^1x_2^1 & x_2^1 & x_1^1y_2^1 & y_1^1y_2^1 & y_2^1 & x_1^1 & y_1^1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ x_1^ix_2^i & y_1^ix_2^i & x_2^i & x_1^iy_2^i & y_1^iy_2^i & y_2^i & x_1^i & y_1^i & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ x_1^Nx_2^N & y_1^Nx_2^N & x_2^N & x_1^Ny_2^N & y_1^Ny_2^N & y_2^N & x_1^N & y_1^N & 1 \end{bmatrix} \begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

8 point algorithm vs. least squares

If $N = 8$, then we can find an exact solution to $\mathbf{Ax} = \mathbf{0}$ where $\mathbf{x}$ is the vector representation of the matrix $\mathbf{F}$. The exact solution is the right null space of $\mathbf{A}$. This solution method is called the *8 point algorithm* for estimating $\mathbf{F}$. Note that it actually requires eight points in *each* image, so 16 points in total. (So think of it as the "8 matching pairs algorithm".) One can solve this by adding on a row of 0's to give a 9×9 matrix and then taking the SVD $\mathbf{A} = \mathbf{U\Sigma V^T}$. The solution is the column of $\mathbf{V}$ with singular value 0. This will work provided that the eight rows are linearly independent.

As we have seen in earlier problems, the positions of any chosen eight matching pairs will typically be noisy, however. Therefore, to get a more accurate estimate, we would need to use $N \gg 8$ matching pairs of points. By inspection, the least squares problem is to find the $\mathbf{F}$ that minimizes:

$$\sum_{i=1}^{N} (\mathbf{x}^{i}{}_2^T \mathbf{F} \mathbf{x}_1^i)^2$$

subject to a constraint such as $\| \mathbf{F} \| = 1$. As usual, we take the SVD: $\mathbf{A} = \mathbf{U\Sigma V^T}$. The solution is the column of $\mathbf{V}$ with the smallest singular value.

 ©Michael Langer

There is problem with this solution, however, which we will describe below. More describing this problem, we mention one familiar detail which is again important in practice.

Data normalization

Just as we saw with calibration and estimation of homographies, one obtains better estimates if one normalizes the data, so that the mean position in each image is (0,0) and the standard deviations of pixel positions are 1.

To normalize, we subtract the means and divide by the standard deviations, similarly to what we defined last lecture:

$$\mathbf{M_1} : (x_1^i, y_1^i) \rightarrow (\frac{x_1^i - \bar{x_1}}{\sigma_1}, \frac{y_1^i - \bar{y_1}}{\sigma_1})$$

$$\mathbf{M_2} : (x_2^i, y_2^i) \rightarrow (\frac{x_2^i - \bar{x_2}}{\sigma_2}, \frac{y_2^i - \bar{y_2}}{\sigma_2})$$

Then we solve for the fundamental matrix $\mathbf{F}_{normalized}$ by minimizing the sum of squares over i using the normalized data points:

$$\sum_i \{ \begin{bmatrix} \tilde{x}_2^i & \tilde{y}_2^i & 1 \end{bmatrix} \mathbf{F}_{normalized} \begin{bmatrix} \tilde{x}_1^i \\ \tilde{y}_1^i \\ 1 \end{bmatrix} \}^2$$

where the $(\tilde{x}_1^i, \tilde{y}_1^i)$ and $(\tilde{x}_2^i, \tilde{y}_2^i)$ are the normalized values. The fundamental matrix is then:

$$\mathbf{F} = (\mathbf{M}_2)^T \mathbf{F}_{normalized} \mathbf{M}_1 \ .$$

Rank 2 approximation

If there is any noise in the positions of the points then the estimated $\mathbf{F}$ will be of rank 3 rather than rank 2. If we use this estimated $\mathbf{F}$, then the epipoles (the left and right null spaces) will not be defined. This is a problem! We cannot talk about epipolar lines intersecting at the epipoles if the epipoles are not defined. What should we do? The recommended approach is to find a rank 2 approximation of $\mathbf{F}$.

Note that since the normalization matrices $\mathbf{M}_1$ and $\mathbf{M}_2$ are invertible, $\mathbf{F}_{normalized}$ and $\mathbf{F}$ have the same rank. So one could alternatively find the best rank 2 approximation of $\mathbf{F}_{normalized}$. Indeed that is what is recommended. So we take the SVD,

$$\mathbf{F}_{normalized} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \ .$$

To find a rank 2 approximation of $\mathbf{F}_{normalized}$, we set the third singular value to 0. That is, we change $\mathbf{\Sigma}$ to $\mathbf{\Sigma}'$ by setting Σ_{33} to 0, and then take

$$\mathbf{F}_{normalized} \equiv \mathbf{U}\mathbf{\Sigma}'\mathbf{V}^T$$

as the solution instead. This forces the $F_{normalized}$ matrix to be rank 2, which defines the epipoles and epipolar lines. Of course, since we are making a least squares approximation and enforcing a rank 2 constraint somewhat arbitrarily, we are not guarenteed that the true corresponding points will lie on corresponding modelled epipolar lines. But it is the best we can do.

 ©Michael Langer

RANSAC

To find corresponding points, we can use a method like RANSAC. This protects us against having incorrect matches. The idea is similar to what we did in fitting a homography to matching pairs in two images. In the case of a fundamental matrix, however, we need to use 8 matching pairs. This sounds like a lot to ask for, but if we use features like SIFT that are distinctive, then finding matching pairs can be done reliably.

Note that RANSAC has two fitting stages. In one stage, one uses the minimal number of points (say 8) and solves the problem exactly (using the 8 point algorithm). In this case, data normalization is not helpful because we are seeking an exact solution. Data normalization only is needed in the least squares stage, namely once a consensus set has been found and one fits the matching pairs in the consensus set.

Rectification (based on F)

Suppose we have estimated a fundamental matrix that relates two images by matching keypoints. We would next like to find *all* matching (also called "corresponding") points in the two images, not just the keypoints. For each point in the left image, the fundamental matrix gives us the line in the right image where the corresponding point must lie. Similarly, for each point in the right image, the fundamental matrix gives us the line in the left image where the corresponding point must lie. Thus, finding the corresponding points really only involves a 1D search, rather than a 2D search.

To solve the matching problem for all points (called *dense correspondence or dense matching*), we don't want to be burdened with the particular details of the epipolar geometry. Therefore one typically rectifies the images by applying a homography to each image. This transforms the epipolar lines to corresponding horizontal lines in the image.

Assuming we have $\mathbf{F}$ and thus we know $\mathbf{e}_1$ and $\mathbf{e}_2$, we compute homographies $\mathbf{H}_1$ and $\mathbf{H}_2$ which map the respective epipoles to the point(s) at infinity, $(\pm 1, 0, 0)$, and also ensure that corresponding points lie in the same row i.e. have the same y value. So, for example, let $\mathbf{e}_1 = (e_u, e_v, 1)$. Then we could choose

$$\mathbf{H}_1 = \begin{bmatrix} 1 & 0 & 0 \\ -\frac{e_v}{e_u} & 1 & 0 \\ -\frac{1}{e_u} & 0 & 1 \end{bmatrix}$$

so that $\mathbf{H_1 e_1} = (e_u, 0, 0)$ which indeed maps the epipole in camera 1 to a point at infinity in the direction of the x axis.

An alternative homography would apply a rotation that brings the epipole to the x axis direction, and then maps it to a point at infinity in the x axis direction:

$$\mathbf{H}_1' = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -\frac{1}{e_u^2+e_v^2} & 0 & 1 \end{bmatrix} \begin{bmatrix} e_u & e_v & 0 \\ -e_v & e_u & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

namley $(e_u^2 + e_v^2, 0, 0)^T = \mathbf{H}_1' \mathbf{e}_1^T$. Note that in general, any homography whose second and third row are both perpendicular to $(e_u, e_v, 1)$ will satisfy the property that it maps the epipole to a point at infinity in the direction of the x axis.

A similar homography needs to be applied to the second image. The challenging is choosing the two homographies such that the the rows in the corresponding points will be aligned (corresponding

 ©Michael Langer

epipolar lines will map to corresponding rows). Several algorithm exist for doing this. e.g. see "Projective rectification from the fundamental matrix" by Mallon and Whelan. Image and Vision Computing (2005). Details omitted.

Projective Reconstruction Theorem

It is important to appreciate that if we don't know the camera parameters (internals and externals), then there are fundamental limitations on what we can say about the true geometry of scene points, *even if we can find correspondences between all points in the left image and all points in the right image*. Here is an example of these limitations.

Suppose the two camera matrices are $\mathbf{P}_1$ and $\mathbf{P}_2$. Then for any 3D point in the scene $\mathbf{X} = (X, Y, Z)$, which is written in the scene coordinate system, we get image pixel positions:

$$\begin{bmatrix} w_1 x_1 \\ w_1 y_1 \\ w_1 \end{bmatrix} = \mathbf{P}_1 \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} w_2 x_2 \\ w_2 y_2 \\ w_2 \end{bmatrix} = \mathbf{P}_2 \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}.$$

But notice that, for any 4×4 invertible matrix $\mathbf{M}$, exactly the same image position would be produced by cameras $\mathbf{P_1 M}$ and $\mathbf{P_2 M}$, and by 3D points $\mathbf{M}^{-1}(X, Y, Z, 1)^T$ since

$$\begin{bmatrix} w_i x_i \\ w_i y_i \\ w_i \end{bmatrix} = (\mathbf{P}_i \mathbf{M})(\mathbf{M}^{-1} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix})$$

for $i = 1, 2$. Since there are infinitely many invertible 4×4 matrices $\mathbf{M}$ there are infinitely many possible cameras and scenes that could produce the same images. This ambiguity is often called the *projective reconstruction theorem*.

Typically one *does* know something about the camera, since most image formats do store information about the camera internals. Moreover, sometimes once can use information in the image such as vanishing points to constrain the solution.

Binocular disparity and Depth – a simple example

Recall the setup at the start of lecture 12 when we began discussing 3D geometry and perspective. Suppose we have a 3D point and now suppose we have two cameras rather than one. The second camera is translated by a distance T relative to the first, namely it is located at position $(T, 0, 0)$ in camera 1's coordinate system. Then any scene point that is located at position (X_1, Y_1, Z_1) in camera 1's coordinates will be at position $(X_2, Y_2, Z_2) = (X_1 - T, Y_1, Z_1)$ in camera 2's coordinates.

 ©Michael Langer

Assuming the projection plane is at $Z = f$ for both cameras, this 3D point will project to image positions in the two cameras:

$$(x_1, y_1) = (\frac{fX_1}{Z_1}, f\frac{Y_1}{Z_1})$$

$$(x_2, y_2) = (\frac{f(X_1 - T)}{Z_1}, \frac{Y_1}{Z_1}).$$

Define the *binocular disparity* to be

$$d = x_1 - x_2 = \frac{fT}{Z}.$$

Note that the disparity is defined only in the x direction. The y positions are the same in the two images. Also, for simplicity, the disparity here is measured in the projection plane so the units would be physical (m) rather than pixel units. One can convert from physical to pixel coordinates with a scaling and translation, namely using internal parameters of the camera specified by matrix **K** (if this matrix is known).

The main observation here is not the units, but rather the fact that the binocular disparity depends *inversely* on the distance Z to the point. (We have seen this inverse depth behavior before – recall lecture 12.) Transforming the x_1 and x_2 values from physical units (mm) to pixel units doesn't change this basic fact. In particular, scaling the x_1 and x_2 values by the same scale factor just scales the disparity values, and translating the x_1 and x_2 values by the same amount does not change the disparity at all. *These informal observations motivate the problem of estimating the disparity for rectified images – namely that the disparity is related inversely to depth. In particular, by estimating the disparities of corresponding/matching points in the left and right images, we can say something about the relative depths of these points*

 ©Michael Langer