Lecture 12: Stereo and Surface Estimation

When camera positions have been determined, using structure from motion, we would like to compute a dense surface model of the scene. In this lecture we will study the so called Stereo Problem, where the goal is to estimate the depth of each pixel in an image. This requires computing a match for each pixel in the image even if the texture is ambiguous.

Figure 1: Left and Middle: Two images used for computing depth estimates for every pixel in the left image. Right: The depth estimate color coded.

1 Rectified Cameras, Disparity vs. Depth

Dense depth estimation requires matching of each pixel to a corresponding pixel in neighboring images. Because of ambiguous texture this problem is difficult to solve. Since cameras are known we can however use the epipolar lines to limit the search.

We will start by assuming that we have a pair of so called rectified cameras, $P_1 = K[I \quad 0]$ and

$$P_2 = K \begin{bmatrix} I & \begin{pmatrix} b \\ 0 \\ 0 \end{pmatrix} \end{bmatrix}. \tag{1}$$

Geometrically this means that both cameras have the same orientation and that the second camera position is a translation in the x-direction of the first camera, see Figure 2. In general image pairs taken with regular cameras do not fulfill these assumptions, however they can be modified to do so. The the line segment joining the two camera centers is called the baseline.

There are several ways of rectifying two cameras. A simple approach is to first select the orientation of axes of the new camera coordinate system in one of the cameras and then rotate both the cameras to this new coordinate system. The new x-axis should be parallel to the baseline and the new y and z-axes should be perpendicular to it. To keep the changes that occur when transforming the image small, we should select the new coordinate system to be as similar to the old one as possible. We can select the new x-axis by projecting the old x-axis onto the baseline and normalizing it. When the x-axis has been determined we can chose the new-z axis as the projection of the old z-axis onto the plane going through the camera center with the new x-axis as normal. When both the x and z axes are determined we can find the y axis by using the cross product. Once we known the new camera orientations we rotate the cameras. Since the rectified cameras and old cameras are related through pure rotations the rectified images are obtained by transforming the images using a homography.

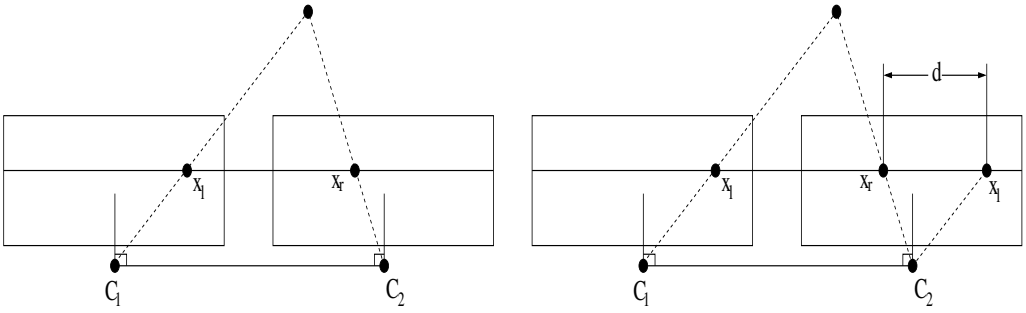


Figure 2: Stereo with rectified cameras. The cameras have the same orientation and the image plane normals are perpendicular to the baseline. Left: The epipolar lines are parallel to the x-axis. Right: The disparity.

Next we will show that for this setup the epipolar lines are parallel to the x-axis of the image. Suppose that

$$K = \begin{pmatrix} \gamma f & sf & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{2}$$

Then the projection of a scene point $X = (X, Y, Z, 1)$ in the cameras P_1 and P_2 is given by

$$x_l = K \begin{bmatrix} I & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} \gamma f X + sfY + x_0 Z \\ fY + y_0 Z \\ Z \end{pmatrix} \tag{3}$$

$$x_R = K \begin{bmatrix} I & \begin{pmatrix} b \\ 0 \\ 0 \end{pmatrix} \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} \gamma f(X+b) + sfY + x_0 Z \\ fY + y_0 Z \\ Z \end{pmatrix} \tag{4}$$

In regular coordinates this gives us the projections

$$(x_L, y_L) = \left(\tfrac{fX + sfY + x_0 Z}{Z}, \tfrac{fY + y_0 Z}{Z} \right) \tag{5}$$

$$(x_R, y_R) = \left(\tfrac{f(X+b) + sfY + x_0 Z}{Z}, \tfrac{fY + y_0 Z}{Z} \right). \tag{6}$$

Therefore $y_l = y_R$ for any two corresponding points, which means that the epipolar lines are horizontal. The difference between the x-coordinates

$$d = x_R - x_L = \frac{\gamma f b}{Z} \tag{7}$$

is called the disparity. The matching problem can be seen as the problem of assigning a disparity to each point in the image. The disparity is inversely proportional to the depth Z, see Figure 3. If we assume that disparities can only be determined up to integer values (no-subpixel accuracy), then it can be seen from Figure 3 that accurate (high resolution) depth estimation can only be achieved when Z is relatively small (with respect to the baseline b).

2 Matching Criteria

Given a pixel and a candidate depth/disparity we need to have a criteria for evaluating if this is a good match or not. Previously in the course we have used SIFT features. However, this type of features have various invariance properties that we do not want here, since the camera positions are already known. A simple commonly used criteria is the normalized cross correlation. The cross correlation compares a patch around the pixel to a patch around the potential match. If I_1 and I_2 are the two patches then their correlation is

$$NCC(I_1, I_2) = \frac{1}{n-1} \sum_{i=1}^{n} \frac{(I_1(x_i) - \bar{I}_1)(I_2(x_i) - \bar{I}_2)}{\sigma(I_1)\sigma(I_2)}, \tag{8}$$

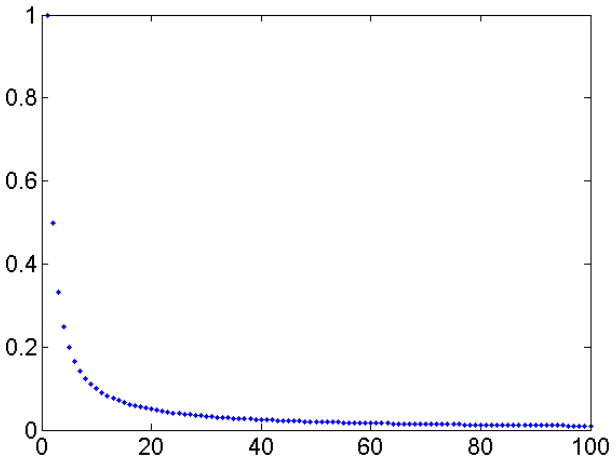


Figure 3: Depth as a function of disparity ($\gamma fb = 1$). Large disparities gives higher depth resolution.

where the sequence x_i are the pixels being compared, $\bar{I}_1$, $\bar{I}_2$, $\sigma(I_1)$ and $\sigma(I_2)$ are the mean values and standard deviations of of the two patches. The result is a number between -1 and 1 where 1 means that the patches similar.

The normalized cross correlation is invariant to translation and rescaling of the image intensities, which is very useful if the two images are captured under different lighting conditions. Figure 4 shows the evaluation of normalized cross correlation along an epipolar line.

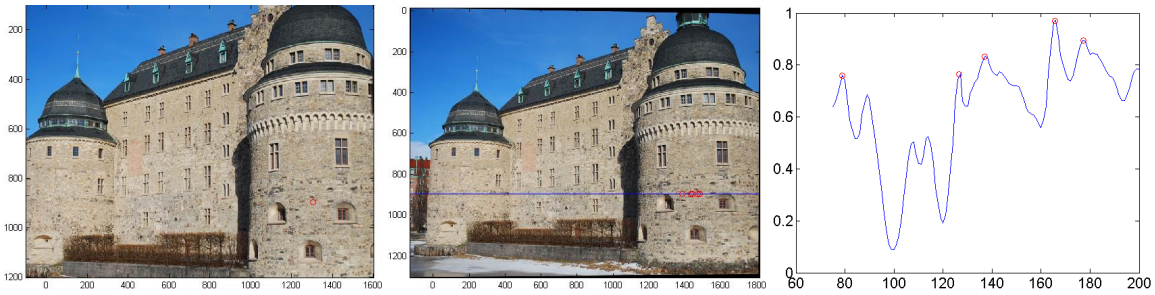


Figure 4: Evaluation of the normalized cross correlation along an epipolar line. Left: Left image and the image point of interest. Middle: Corresponding epipolar line and a few local maxima of the NCC. Right: NCC for a range of disparities. Red rings are the same local maxima as in the middle image.

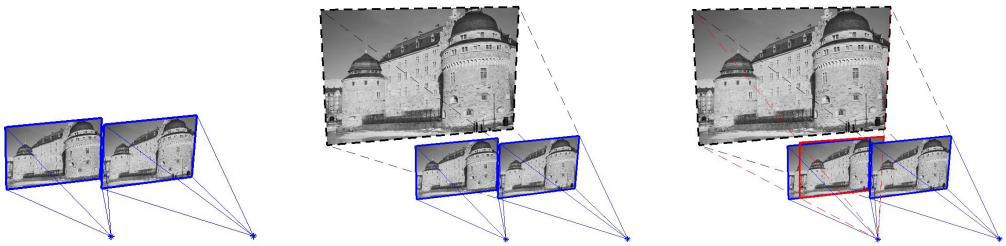
3 Plane Sweep Algorithms

When we use more than two cameras it might not be possible to rectify all the images simultaneously. In this case an alternative is to employ a plane sweep algorithm. The basic idea is that if all the pixels have the same depth then all the scene points are located on a plane in 3D. Since projections in two images from points on a plane are related by a homography, we can hypothesize a depth, transform the pixels of one camera into the second and compute cross correlation. Sweeping the plane through a series of hypotheses gives a cost for assigning pixels to the hypothesized depths.

If the cameras are $P_1 = K[I \quad 0]$ and $P_2 = K[R \quad t]$ then the plane Π with scene points at the depth Z is given by $\Pi = (0, 0, -1/Z, 1)$. The homography H_Z for the depth Z is then given by the formula

$$H_Z = K\left(R + t\begin{pmatrix}0 & 0 & \frac{1}{Z}\end{pmatrix}\right)K^{-1}. \tag{9}$$

(See Assignment 1, Exercise 6 for a derivation.)



(a) (b) (c)

Figure 5: Plane sweep algorithms. Given two cameras (a), hypothesize a depth for all the pixels (b), project into the second camera and compare pixel values (c). Sweeping the scene plane over different depths gives costs of assigning pixels to these hypothesized depths.

Compared to rectified cameras where we can simply compare patches along the epipolar line, transforming the image using the homography is more computationally expensive. On the other hand this approach is very convenient in that it works with general camera configurations. Furthermore, since the image is re-sampled during the transformation we do not need to limit ourselves to depths that are are inverse values of integer valued disparities. Therefore sup-pixel accuracy is naturally achieved with this approach.

4 Regularization/Energy Minimization

The result of the plane sweep algorithm is a function for each pixel that specifies a cost of assigning that pixel a certain depth. By selecting the smallest cost for each pixel we obtain a dense depth estimate. The resulting surface will often be noisy. This is because individual pixel estimates can be unreliable due to ambiguous texture and self occlusion.

To improve the estimated surface and reduce the noise a common approach is to seek an assignment where neighboring pixels have similar depths. Typically one minimizes an energy functional of the form

$$\sum_i \sum_{j \in \mathcal{N}(i)} E_{ij}(Z_i, Z_j) + \sum_i E_i(Z_i). \tag{10}$$

The second term $E_i(Z_i)$ is the cost of assigning the depth Z_i to pixel i. This term is typically referred to as the data term since it is based on image data. The first term $E_{ij}(Z_i, Z_j)$ penalizes differences in depth between pixel i and a pixel j that is in the neighborhood $\mathcal{N}(i)$ of pixel i. This term is usually called the smoothness term since it favors solutions with smooth surfaces.

Optimizing this type of energy is challenging due to the numerous local minima of the individual data terms $E_i(Z_i)$, see Figure 4. The most successful methods for doing this are based on discrete graph methods. If Z_i is binary, then the energy (10) can be seen as the minimum cut on a graph where the nodes correspond to the pixels and the edges correspond to the neighborhood structure. Finding the minimum cut in a graph can be solved efficiently (in polynomial time complexity) by computing the maximal flow on the same graph.

Let us first consider a two pixel case with the energy functional

$$E(Z_i, Z_j) = E_{ij}(Z_i, Z_j) + E_{ji}(Z_j, Z_i) + E_i(Z_i) + E_j(Z_j). \tag{11}$$

The corresponding graph has two nodes representing the pixels and two special nodes; the source s and the sink t. The graph has directed edges from s to i and j, from i and j to t and between i and j. In addition each edge has a (non-negative) capacity $c(i, j)$. A cut in this graph is a partitioning of the nodes into two disjoint sets S and T where S are the nodes connected to the source and T the nodes connected to the sink. The value of the cut is the sum of all the edge capacities of the edges going form S to T.

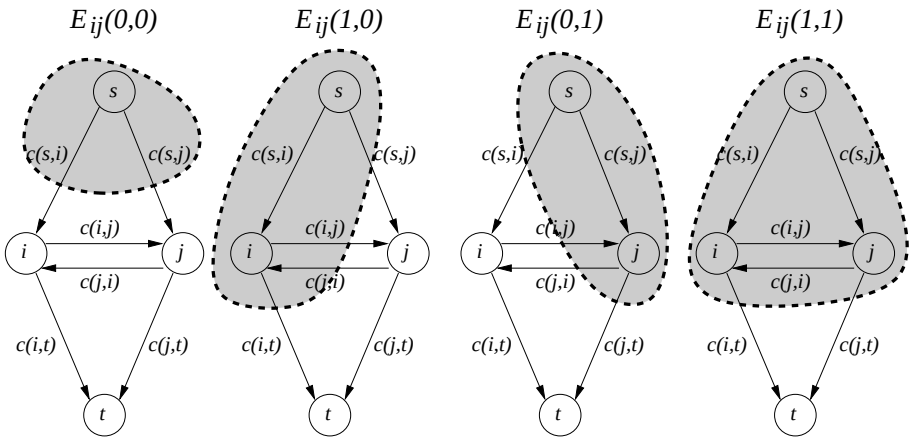


Figure 6: The four cuts corresponding to the values of the energy term in (12)-(15).

The energy (11) can be solved as a minimal cut problem on this graph under certain conditions. First of all we see that $E(Z_i, Z_j) \geq 0$ since graph edges have non-negative capacities. Note that if the energy is not positive we can always add a constant to $E(Z_i, Z_j)$ without changing the minimizer and thereby obtain an equivalent positive energy. If we let $i \in S$ mean that $Z_i = 1$ then from Figure 6 we see that

$$E(0,0) = c(s,i) + c(s,j) \tag{12}$$
$$E(1,0) = c(i,j) + c(s,j) + c(i,t) \tag{13}$$
$$E(0,1) = c(j,i) + c(s,i) + c(j,t) \tag{14}$$
$$E(1,1) = c(i,t) + c(j,t). \tag{15}$$

Since the capacities are all positive we see from these equations that in order to be able to represent this energy with a graph, then

$$E(0,0) + E(1,1) \leq E(1,0) + E(0,1) \tag{16}$$

must hold. Energies with pairwise smoothing terms that fulfill (16) are called submodular. It can be shown that the condition (16) actually ensures that the energy can be represented by a graph. Furthermore, it is easy to see that the sum of submodular energies is also submodular. This means that if the terms E_{ij} of the energy (10) are all submodular then we can minimize this energy by solving a max-flow/min-cut problem.

When Z_i is not binary (as in stereo) we typically employ move making algorithms. The goal of these algorithms is to modify as many pixels as possible simultaneously in order to avoid getting stuck in poor local minima. One of the most popular approaches is that of α-expansion. Suppose that we have a current assignment of depths. Then each pixel is given the option to switch from the current depth to a new depth α. Since each pixel has two choices (move to α or retain the old value) this can be formulated as a binary problem. If this new binary problem is submodular it can be solved efficiently using max-flow/min-cut algorithms.

Suppose that in the current assignment $Z_i = \beta$ and $Z_j = \gamma$ and consider the term $E_{ij}(Z_i, Z_j)$ when we let the pixels switch to α. We define a new binary energy term $E_B(x_i, x_j)$ by letting $x_i = 0$ when Z_i retains its current assignment, $x_i = 1$ when Z_i switches to α (and x_j similarly). Then

$$E_B(0,0) = E_{ij}(\beta, \gamma) \tag{17}$$
$$E_B(0,1) = E_{ij}(\beta, \alpha) \tag{18}$$
$$E_B(1,0) = E_{ij}(\alpha, \gamma) \tag{19}$$
$$E_B(1,1) = E_{ij}(\alpha, \alpha). \tag{20}$$

Therefore, if the smoothnes terms of the energy (10) fulfills

$$E_{ij}(\beta, \gamma) + E_{ij}(\alpha, \alpha) \leq E_{ij}(\beta, \alpha) + E_{ij}(\alpha, \gamma) \tag{21}$$

we can solve the α-expansion efficiently using max-flow/min-cut algorithms.

In many cases we can assume that the energy is non-negative, symmetric $E_{i,j}(\beta, \gamma) = E(\gamma, \beta)$ and that $E(\alpha, \alpha) = 0$. In these cases it can be seen that E_{ij} has to be a metric distance since (21) reduces to the triangle inequality. A commonly used smoothness term that fulfills the metric conditions is the truncated absolute distance

$$E_{ij}(\beta, \gamma) = \min(|\beta - \gamma|, t). \tag{22}$$

This term favors assignments where neighboring depths have small disparity differences. In real images we should however also allow for discontinuous depth assignments due to object boundaries (transitions between smooth surfaces). Therefore the threshold t is added to the energy to prevent over-smoothing at discontinuities.