



## Problem Set 4 Stereo Vision

In this problem set you will implement and test some simple stereo algorithms discussed in class. In each case you will take two images  $I_l$  and  $I_r$  (a left and a right image) and compute the horizontal disparity (ie., shift) of pixels along each scanline. This is the so-called baseline stereo case, where the images are taken with a forward-facing camera, and the translation between cameras is along the horizontal axis.

### 1 Block Matching

To get the disparity value at each point in the left image, you will search over a range disparities, and compare the windows using two different metrics mentioned in class: Sum of Absolute Differences (SAD) and Sum of Squared Differences. Do this for windows of size  $w$  where  $w = 1, 5$  and  $9$ . The disparity,  $d$ , is restricted to be in range  $-D$  to  $D$ . For this assignment we will use  $D = 8$ . You should try at least 2 pairs of images and for each show for each window size and each metric disparity image.

### 2 Dynamic programming

Consider two scanlines  $I_l(i)$  and  $I_r(j)$ . Pixels in each scanline may be matched, or skipped (considered to be occluded in either the left or right image). Let  $d_{ij}$  be the cost associated with matching pixel  $I_l(i)$  with pixel  $I_r(j)$ . Here we consider a squared error measure between pixels given by:

$$d_{ij} = \frac{(I_l(i) - I_r(j))^2}{\sigma^2} \quad (1)$$

where  $\sigma$  is some measure of pixel noise. The cost of skipping a pixel (in either scanline) is given by a constant  $c_0$ . For the experiments here we will use  $\sigma = 2$  and  $c_0 = 1$ . Given these costs, we can compute the optimal (minimal cost) alignment of two scanlines recursively as follows:

1.  $D(1, 1) = d_{11}$
2.  $D(i, j) = \min(D(i-1, j-1) + d_{ij}, D(i-1, j) + c_0, D(i, j-1) + c_0)$

The intermediate values are stored in an N-by-N matrix, D. The total cost of matching two scanlines is  $D(N, N)$ . Note that this assumes the lines are matched at both ends (and hence have zero disparity there). This is a reasonable approximation provided the images are large relative to the disparity shift. Given D we find the optimal alignment by backtracking. In particular, starting at  $(i, j) = (N, N)$ , we choose the minimum value of D from  $(i-1, j-1), (i-1, j), (i, j-1)$ . Selecting  $(i-1, j)$  corresponds to skipping a pixel in  $I_l$  (a unit increase in disparity),



while selecting  $(i, j - 1)$  corresponds to skipping a pixel in  $I_r$  (a unit decrease in disparity). Selecting  $(i - 1, j - 1)$  matches pixels  $(i, j)$ , and therefore leaves disparity unchanged. Beginning with zero disparity, we can work backwards from  $(N, N)$ , tallying the disparity until we reach  $(1, 1)$ .

A good way to interpret your solution is to plot the alignment found for single scan line. To display the alignment plot a graph of  $I_l$  (horizontal) vs  $I_r$  (vertical). Begin at  $D(N, N)$  and work backwards to find the best path. If a pixel in  $I_l$  is skipped, draw a horizontal line. If a pixel in  $I_r$  is skipped, draw a vertical line. Otherwise, the pixels are matched, and you draw a diagonal line. The plot should end at  $(1, 1)$ .

## 2.1 Disparity computation

Repeat the process explained above for each row of the image and compute the disparity maps for image pairs.

## 3 Deliverables

You are required to deliver the following:

- Your code.
- Report including explanation of your code and the results of a number of test images.