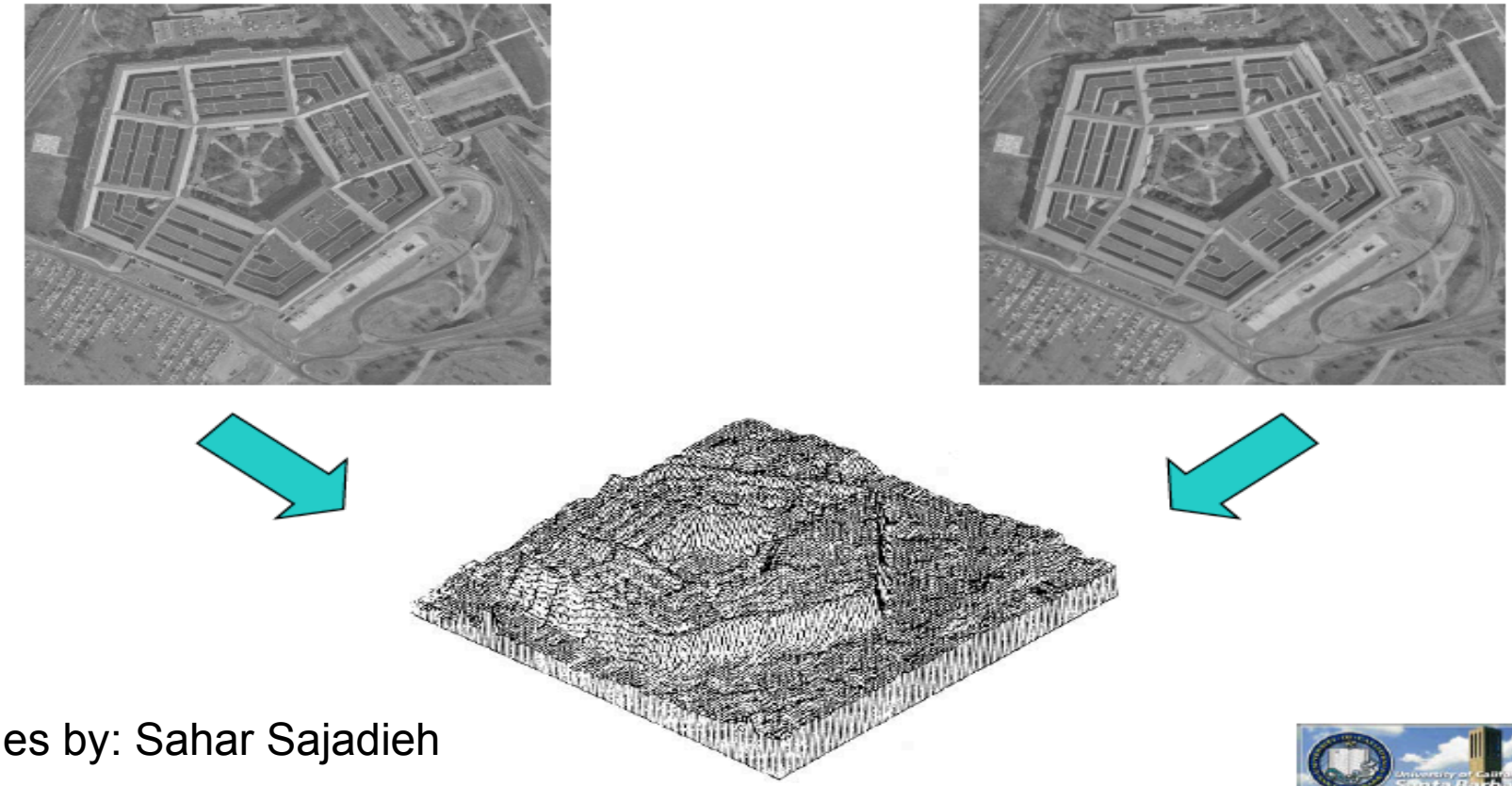
Stereo Analysis



Slides by: Sahar Sajadieh

Stereo Disparity

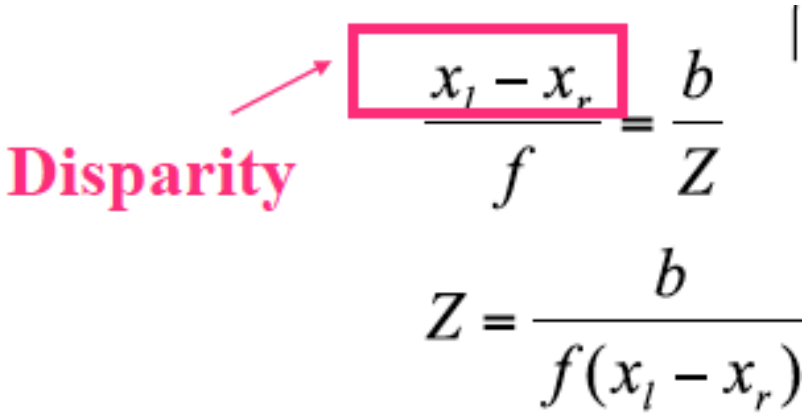
- HW4:

  For each pair of stereo images, you should construct a disparity map for the left image, indicating the disparity to the right image.

Stereo Disparity

- Why Disparity Matrix?

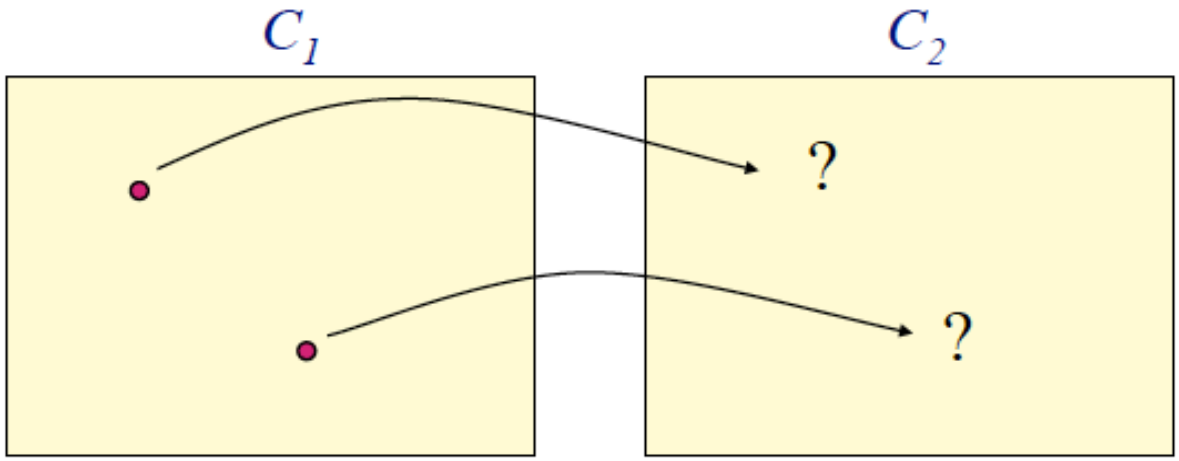
  Because using the disparity matrix, we can easily get the depth map for 3d reconstruction from the 2d images:



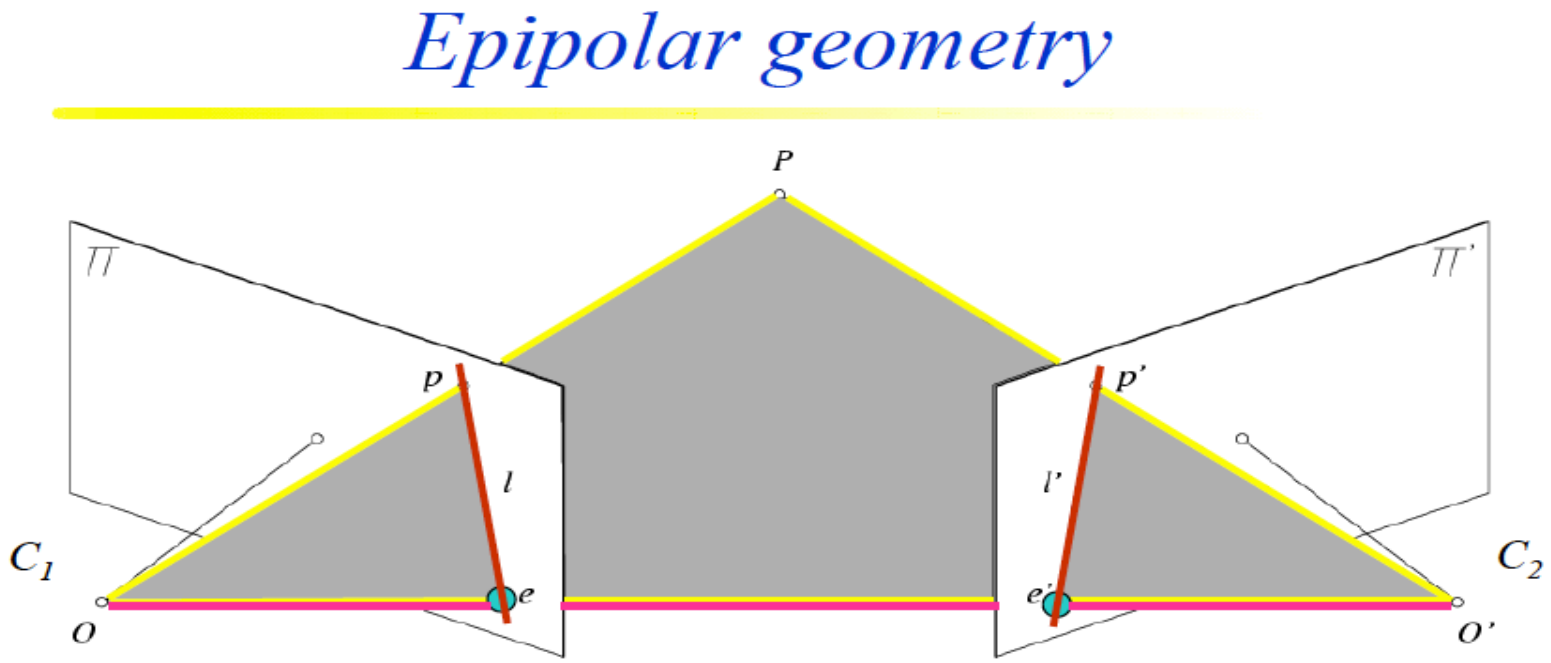
Stereo Matching

"Stereo matching" is the correspondence problem

- For a point in Image #1, where is the corresponding point in Image #2?



Epipolar Lines



- Epipolar Plane
- Epipolar Lines
- Epipoles
- Baseline

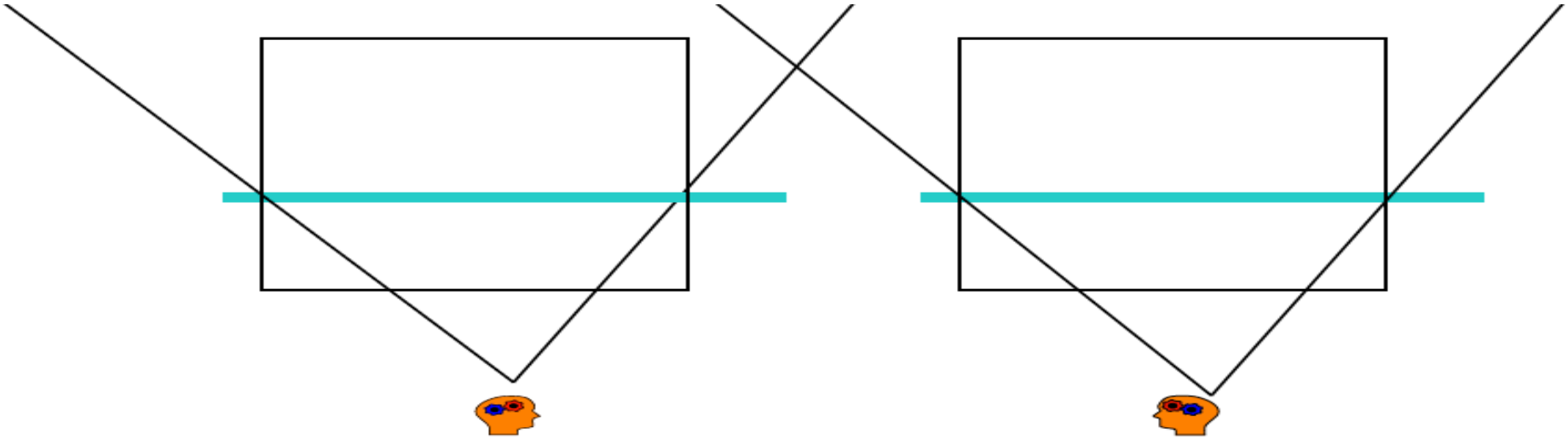
Good News!

- You don't need to worry about calculating epipolar lines, because the images are rectified

Epipolar lines are scanlines(horizontal) lines

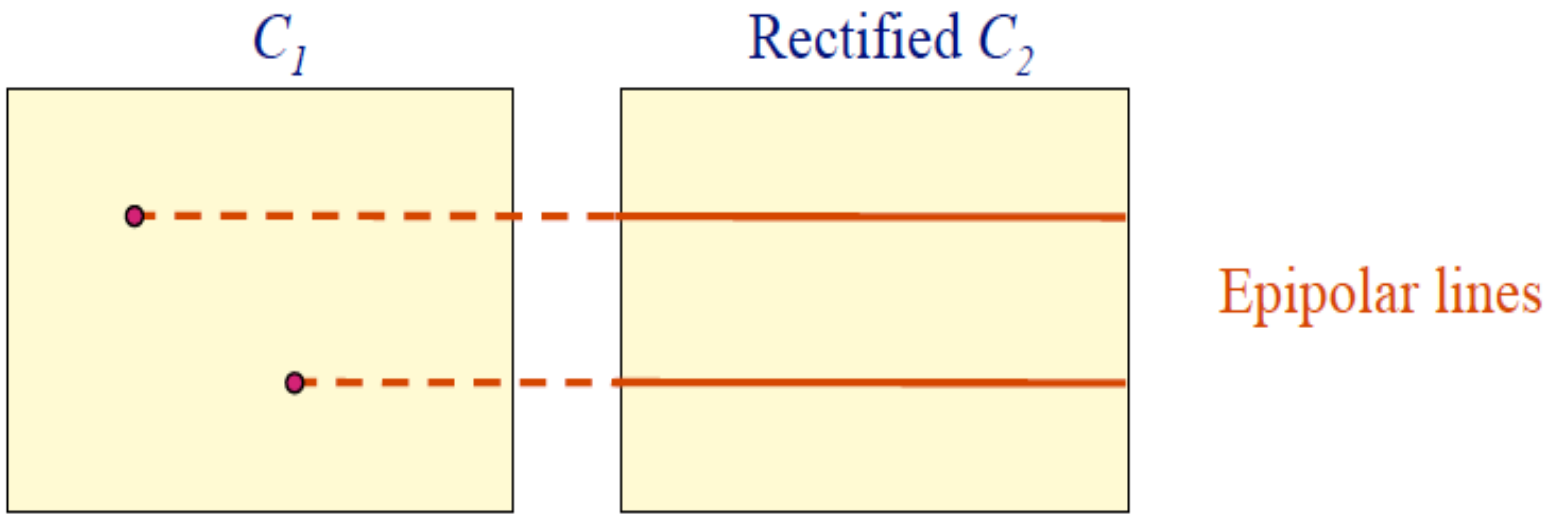
Rectified Image

- What is a rectified stereo image pair?
  - Images where the epipolarlines are the scanlines (horizontal lines)
  - Images where the ith row in the left image corresponds to the ith row in the right image → much simpler calculations
  - The 2d search problem becomes a 1d problem



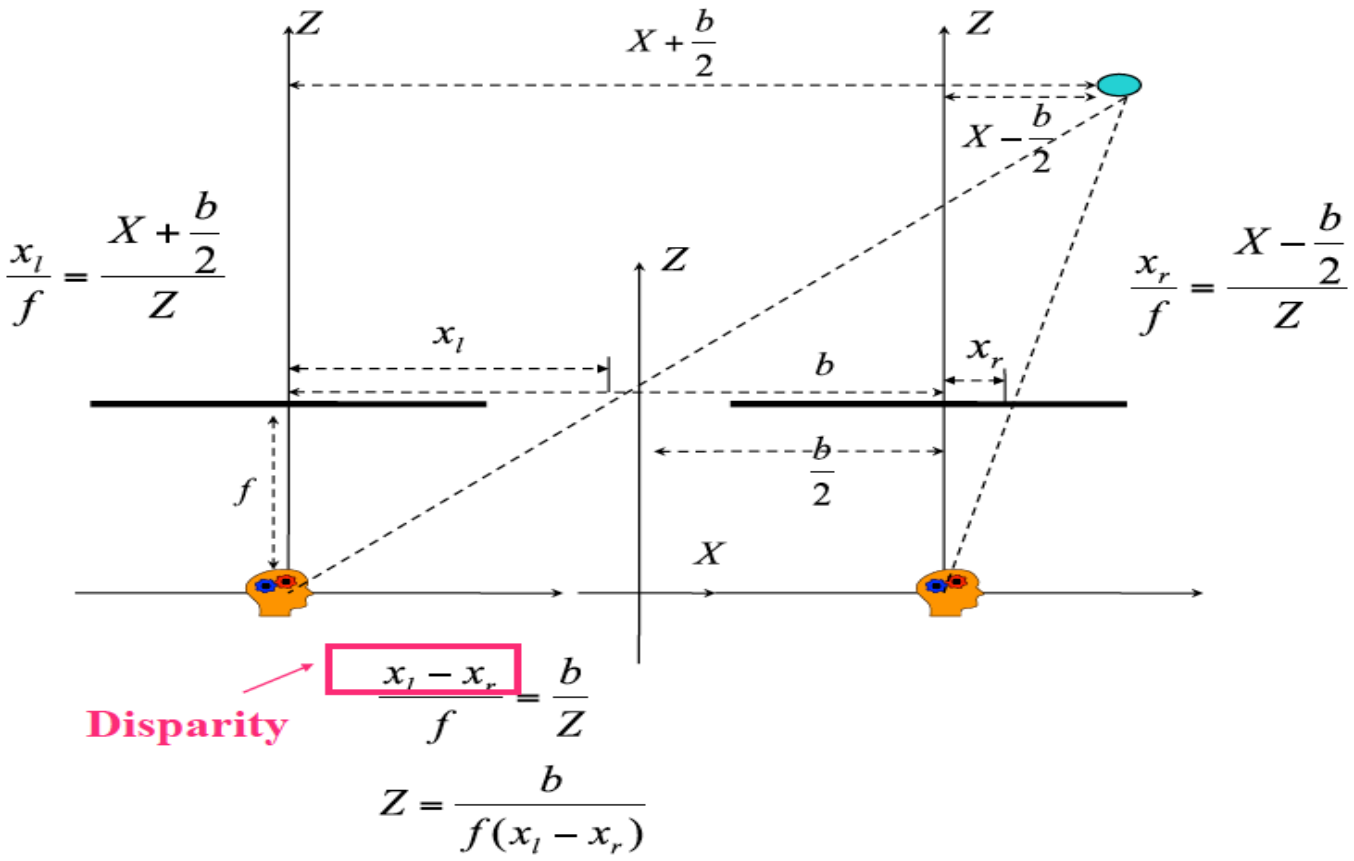
Rectified Image

For a point in Image #1, where is the corresponding point in Image #2?

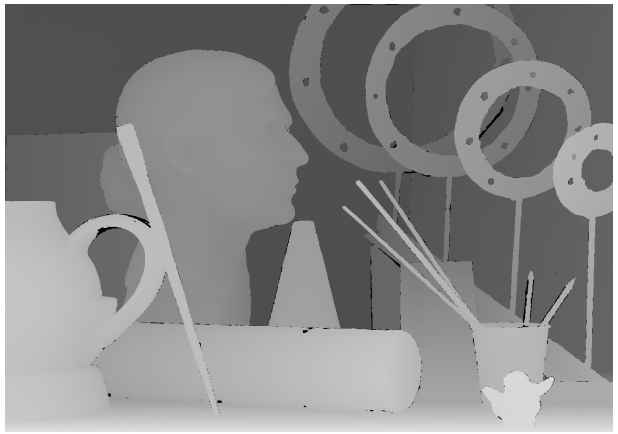


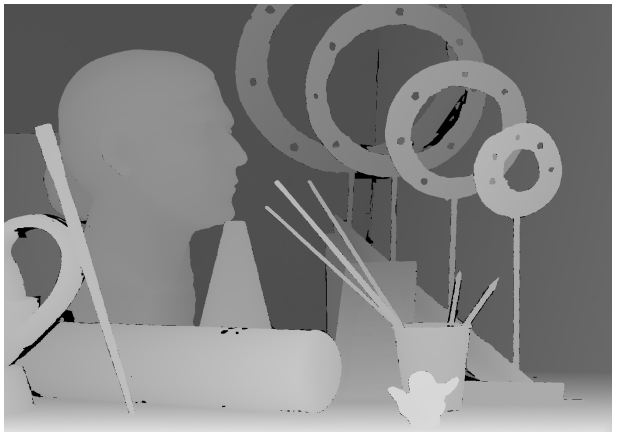
Rectified Image

Basic Stereo Configuration: rectified images



How to Do Stereo Disparity?





Stereo Matching with Dynamic Programming

- What is stereo matching with dynamic programming?

  An algorithm that finds a “path” through an image which provides the best (with minimal cost) match.

Stereo Matching with Dynamic Programming

- Assumption:

  Every pixel in one image can either match one pixel in another image, or be marked as occluded.

Stereo Matching with Dynamic Programming

Pseudo-code describing how to calculate the optimum match:

```
For (i=1; i<=N; i++) { D(i, 1)= i * occlusion}
For (i=1; i<=M; i++) { D(1, i)= i * occlusion}
For (i=2; i<=N; i++){
        For (j=2; j<=M; j++){
                min1 = D(i-1, j-1) + d(i , j)
                min2 = D(i-1, j) + occlusion
                min3 = D(i, j-1) + occlusion
                min4 = D(i-1, j-1) + 2* occlusion
                D(i, j)= Dmin= min (min1, min2, min3, min4)
                If(min1=Dmin) A(i,j)=1
                If(min2=Dmin) A(i,j)=2
                If(min3=Dmin) A(i,j)=3
                If(min4=Dmin) A(i,j)=4
        }
}
```

Variables:

occlusion: penalty for a pixel being occluded. Experiment with the values to find a value which makes your output as close as possible to the ground truth image on the Middlebury website.

N: number of pixels in the left scanline

M: number of pixels in the right scanline

D(i, j): the cost of matching the first i pixels in the left image with the first j pixels in the right image

d(i, j): the cost of matching the ith pixel in the left image with the jth pixel in the right image

A(i, j): used as a flag for the next part of the algorithm

Stereo Matching with Dynamic Programming

- There are many different matching pixel algorithms. In each, we use a different method for calculating the cost of matching a pixel in the left image with a pixel in the right image(d(i,j))
- For this HW, you can use this cost function:

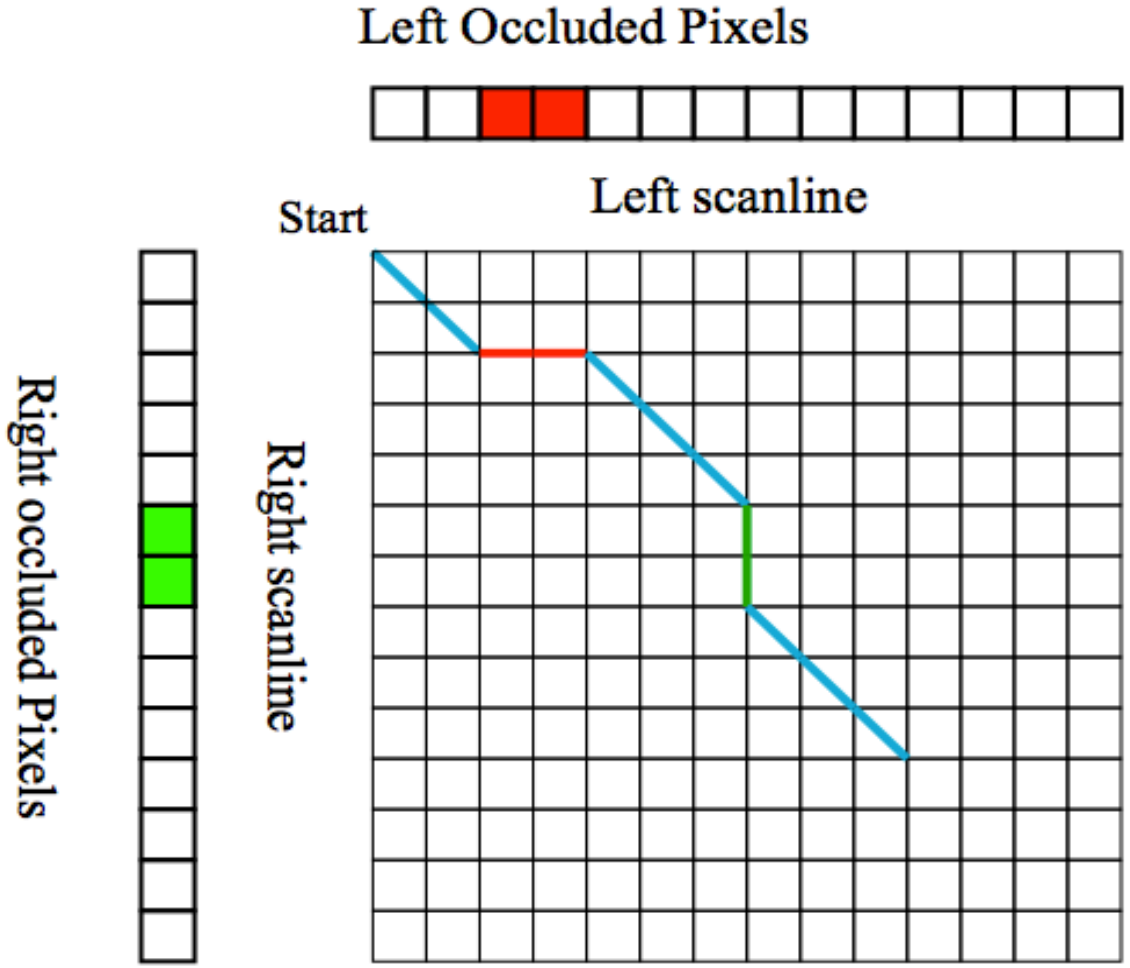
$$d(i,j) = sqr(leftrow(i) - rightrow(j))$$

Stereo Matching with Dynamic Programming

Pseudo-code describing how to reconstruct the optimum match:

```
P=N
q=M
while(p!=0 && q!=0){
        switch(A(p,q)){
                case 1:
                        p matches q
                        p--, q--
                        break
                case 2:
                        p is unmatched
                        p--
                        break
                case 3:
                        q is unmatched
                        q--
                        break
                case 4:
                        p, q are both unmatched
                        p--, q--
                        break
}       }
```

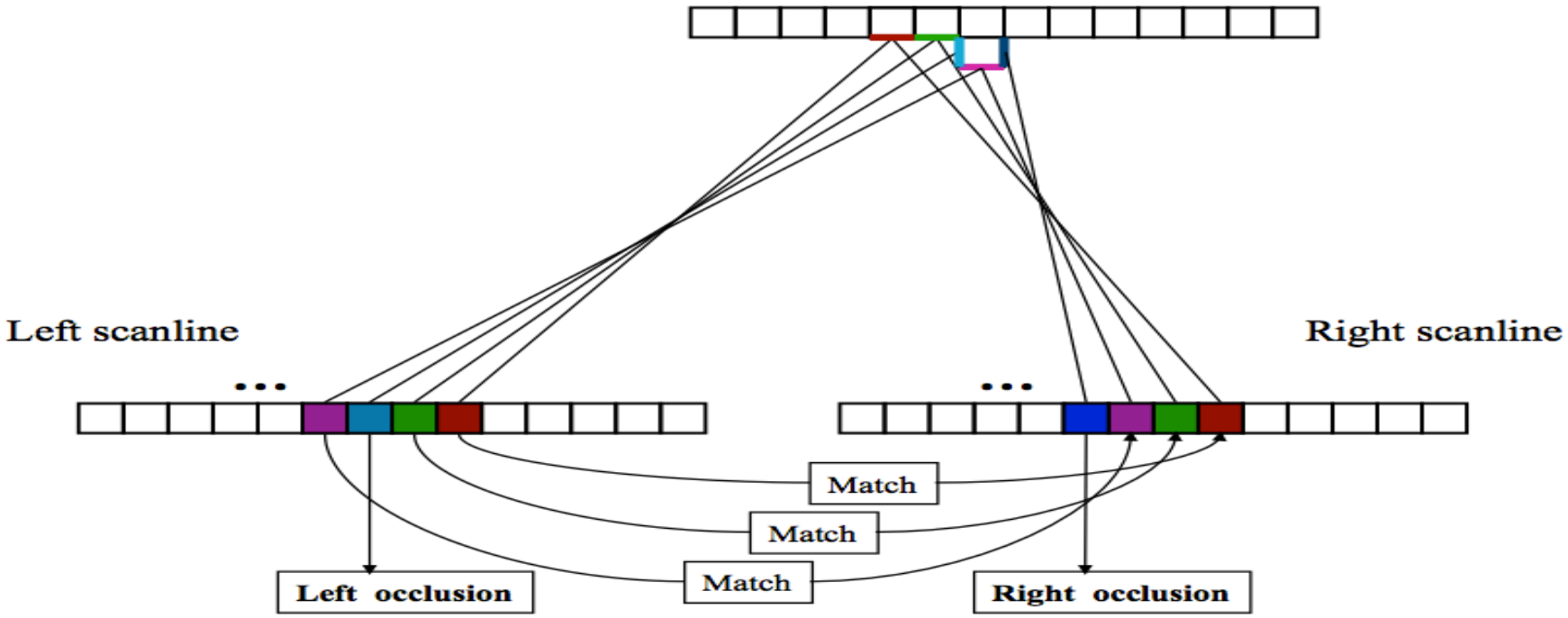
Stereo Matching with Dynamic Programming



Dynamic programming yields the optimal path through grid. This is the best set of matches that satisfy the ordering constraint

Stereo Matching with Dynamic Programming

Correspondences



Stereo Matching with Dynamic Programming

- To display the disparity map for the left image:
  - If case 1: Disp(p)=p-q
  - If case 2: Disp(p)=0
  - If case 3: Do Nothing! Why?
  - If case 4: Disp(p)=0

Stereo Matching with Dynamic Programming

- To display the disparity map for the left image:
  - If case 1: Disp(p)=p-q
  - If case 2: Disp(p)=0
  - If case 3: Do Nothing! Why?
  - If case 4: Disp(p)=0
- How do we display the disparity map for the right image?

Stereo Matching with Dynamic Programming

- Please note:
  - The previous steps were for finding one row in the disparity image→ Repeat it for all the rows in the left & right image.
  - You should scale your Disparity matrix with the scalefactor, (your program's argument), before displaying → disparity image would become more similar to the ground truth.
  - You can see the scalefactor for every set of images in the online image database.
  - Output image should be in grey scale.
  - You should modify occulation parameter and any other parameters to get as close as possible to the ground truth image on the website.
  - In the database, the stereo images in one set have the same amount of exposure (brightness). Otherwise you had to modify the pixel values of one image w.r.t the average of the pixels in the 2nd image.

Stereo Matching with Dynamic Programming

- Your program should work by running the following command.

  disparity(left.png, right.png, scalefactor, output.png);

- You can imagine we will put a stereo pair, named left.png and right.png under the main directory of matlab. And we will enter proper scalefactor for the input images. The name of your output image should be output.png.

- Your disparity matrix should be for the left image.

Questions?