Stereo

CS 763

Ajit Rajwade

Contents

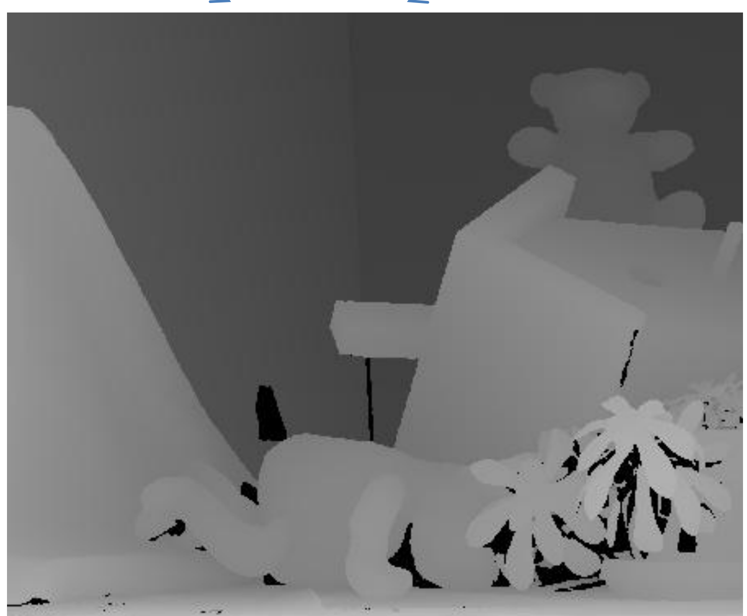
- Introduction – stereo in the human eye
- Stereo vision – simplest case
- Epipolar geometry
- Uncalibrated stereo
- Correspondence problem and how to “solve” it

What is (geometric, binocular) stereo?

- A technique to reconstruct the 3D scene underlying two images taken from two different (usually very close) viewpoints.
- Biological motivation: Our brain infers the 3D structure of the scene from the *difference* between the images formed by the left and right eyes.
- Of course, the brain makes use of other cues for inferring depth, but stereo is the most basic one.

Stereo vision: human eye

- Hold your index finger an arm’s length away.
- Look at it through the left eye keeping the right eye closed.
- Now look at it through the right eye keeping the left one closed.
- You will perceive a shift - this is called as **stereo disparity** and the brain uses it heavily to infer depth!



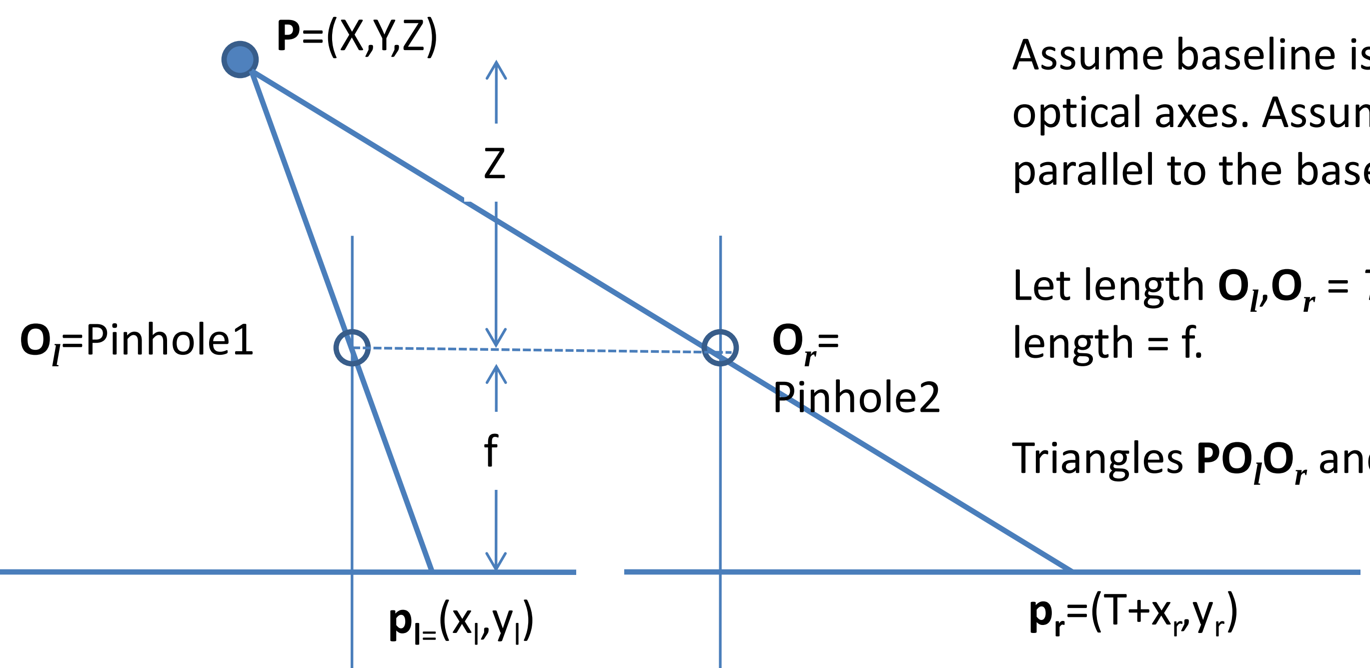
Aim: reconstruct 3D shape given two images captured by cameras in two different positions

Simplest case: stereo

- To perform 3D reconstruction, we must know point correspondences – i.e. given a point in the left image, which is the corresponding point in the right image?
- Let’s make some assumptions about the camera positions!

Simplest case: stereo

- Assume that the pinhole positions of the two cameras are known and that their optical axes are perfectly aligned (parallel).



Line $\mathbf{O}_l\mathbf{O}_r$ = *baseline*.

Assume baseline is perpendicular to the optical axes. Assume camera X-axis is parallel to the baseline.

Let length $\mathbf{O}_l,\mathbf{O}_r$ = T = baseline length. Focal length = f.

Triangles $\mathbf{PO}_l\mathbf{O}_r$ and $\mathbf{Pp}_l\mathbf{p}_r$ are similar.

Simplest case: stereo

- From similarity of triangles, we have:

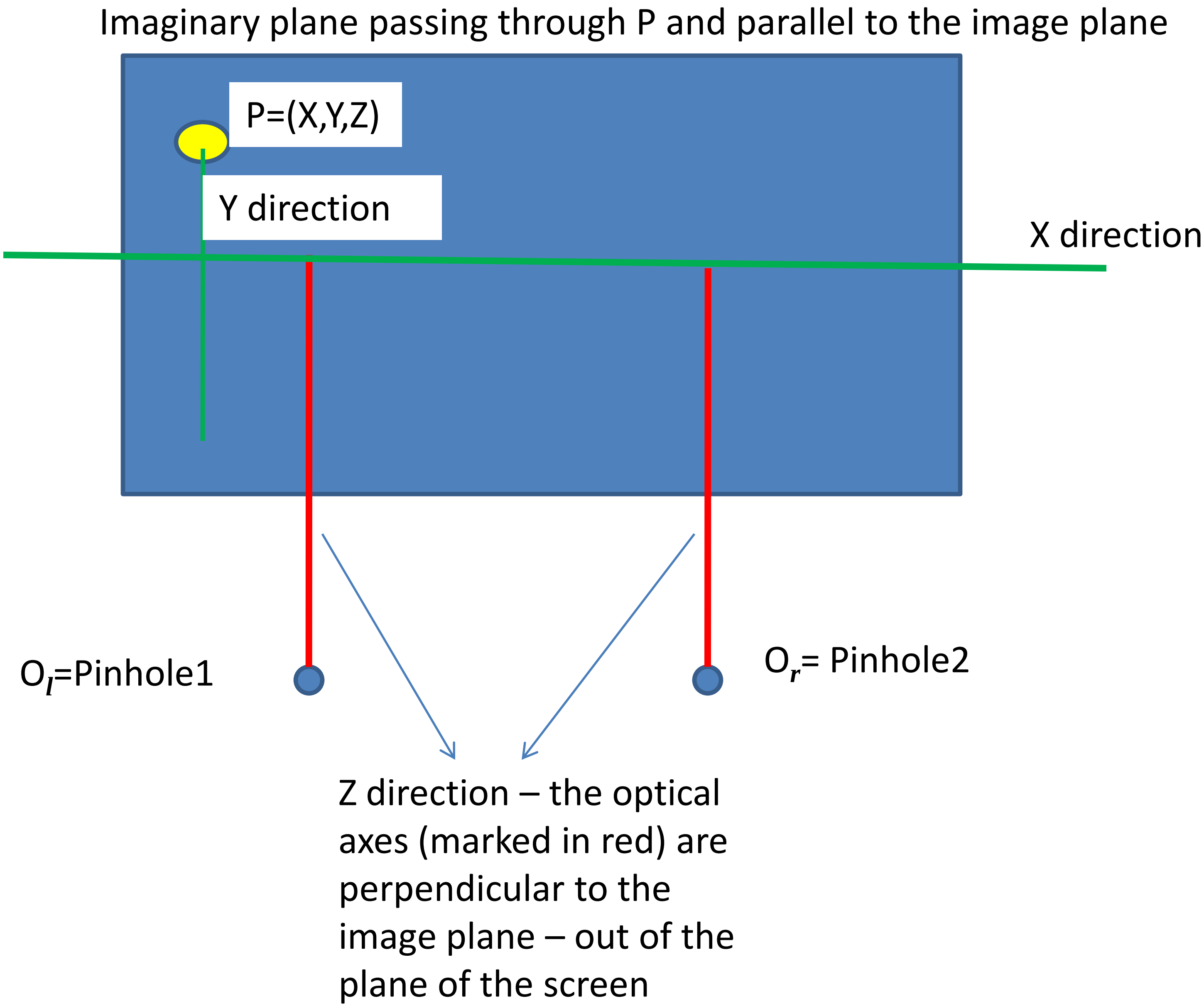
$$\frac{T}{Z} = \frac{T + x_r - x_l}{Z + f}$$

$$Z = \frac{fT}{x_r - x_l} = \frac{fT}{d}$$

Disparity – a spatially varying quantity. At each point *(x,y)* in the left image, we have disparity *d(x,y)* and *x+d(x,y)* is the x-coordinate of its corresponding point in the second image.

$$y_l = y_r = f\frac{Y}{Z}$$

Note that y-coordinates are equal because of our assumption that the X axis of the cameras is parallel to the baseline.


Imaginary plane passing through P and parallel to the image plane
P=(X,Y,Z)
Y direction
X direction
O_l=Pinhole1
O_r= Pinhole2
Z direction – the optical axes (marked in red) are perpendicular to the image plane – out of the plane of the screen

Comments

- The search for a point corresponding to one in the left image is restricted to a line parallel to the X axis, as the y-coordinates are the same! This is called the **epipolar line**.
- A point in the left image may not have a counterpart in the right image (shadows, specularities, occlusions, difference in field of view between the cameras), but if it does, it **must** lie on the epipolar line.

$$Z = \frac{fT}{x_r - x_l} = \frac{fT}{d}$$

$$\frac{\partial Z}{\partial d} = \frac{-fT}{d^2}$$

Comments

- Disparity and depth (i.e. distance from camera image plane) are inversely proportional. So distance to faraway objects can be measured less accurately than to nearby ones.
- Disparity is directly proportional to focal length (as you increase focal length, magnification increases).
- Disparity is directly proportional to baseline length – but a large baseline is a problem (due to missing correspondences as the fields of view will be very different!)

Two notes of caution

- In most practical stereo systems, it is unreasonable to assume that the optical axes of the two cameras are parallel. We will deal with the case of unaligned cameras on the next bunch of slides.
- Even with parallel optical axes, the correspondence problem is not at all easy! We will deal with this problem later.

Parameters of a stereo system

- **Intrinsic parameters** – focal lengths, optical centers, camera resolutions
- **Extrinsic parameters** – rotation and translation to align the coordinate systems of the two cameras.
- The intrinsic or extrinsic parameters or both are often unknown. Stereo reconstruction is essentially a calibration problem!

Epipolar Geometry

- Let’s now study the case where the optical axes of the cameras were not aligned.
- But we will assume full knowledge of camera parameters (intrinsic and extrinsic).
- This is called as **fully calibrated stereo.**

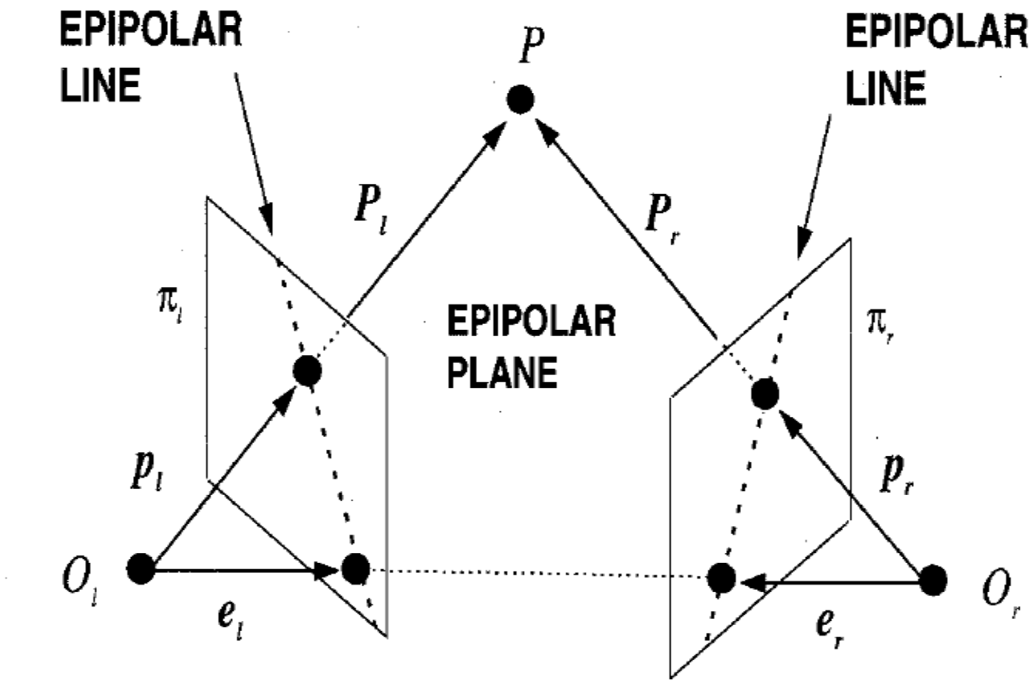


Figure 7.6 The epipolar geometry.

Rotation matrix

Translation vector

Camera reference frames are related as follows: $\mathbf{P_r} = \mathbf{R}(\mathbf{P_l} - \mathbf{T})$

where $\mathbf{P_r}$ and $\mathbf{P}_l$ are coordinates of point **P** in the reference frame of the left and right cameras. The image of **P** in the two image planes has coordinates $\mathbf{p}_l$ and $\mathbf{p_r}$.

•The line joining $\mathbf{O}_l$ and $\mathbf{O_r}$ intersects the image planes at point $\mathbf{e}_l$ and $\mathbf{e_r}$ – called as the **(left/right) epipoles**. The left epipole is the image of $\mathbf{O_r}$ and right epipole is the image of $\mathbf{O}_l$.

•The points **P**, $\mathbf{O}_l$ and $\mathbf{O_r}$ form the **epipolar plane** for point **P**. The epipolar plane intersects each image plane in the **(left/right) epipolar line** for point **P**.

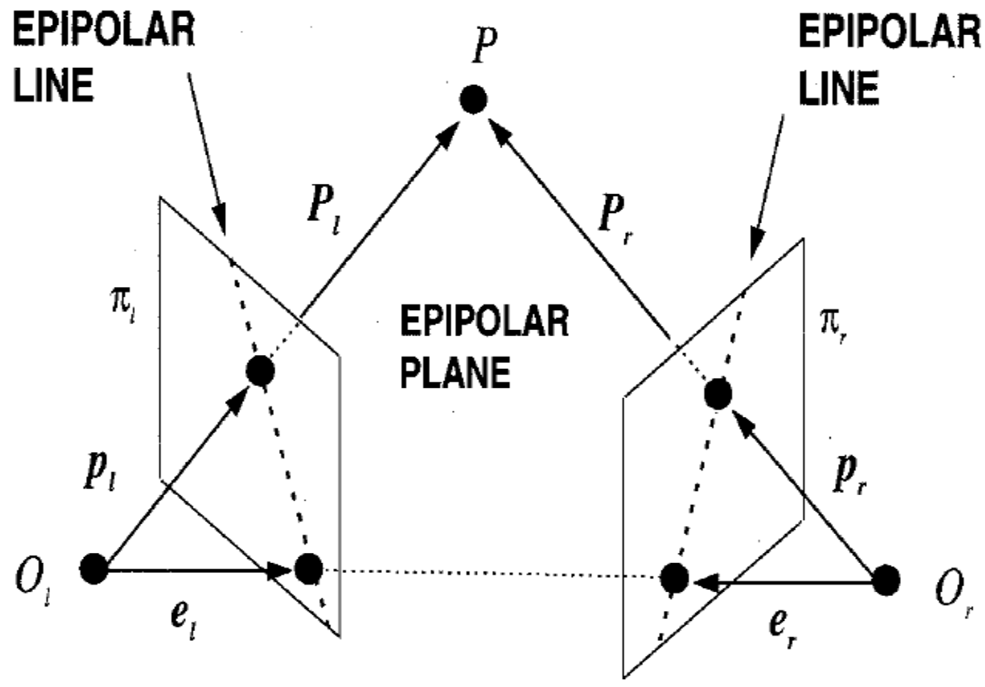


Figure 7.6 The epipolar geometry.

Epipolar Constraint:

Given **p_l**, the point **P** can lie at any point on the line from **O_l** to **p_l**. *The image of ray **$O_l p_l$** on the right image plane is contained in the right epipolar line* (Why? Because **O_l**, **p_l** and **P** are collinear – hence their images under perspective projection on the right image plane must also be collinear).

This is called the **epipolar constraint**. What this means is that the point on the right image plane corresponding to **p_l** (i.e. point **p_r**) **is restricted to lie on a single line which happens to be the right epipolar line**. All epipolar lines pass through the respective epipoles.

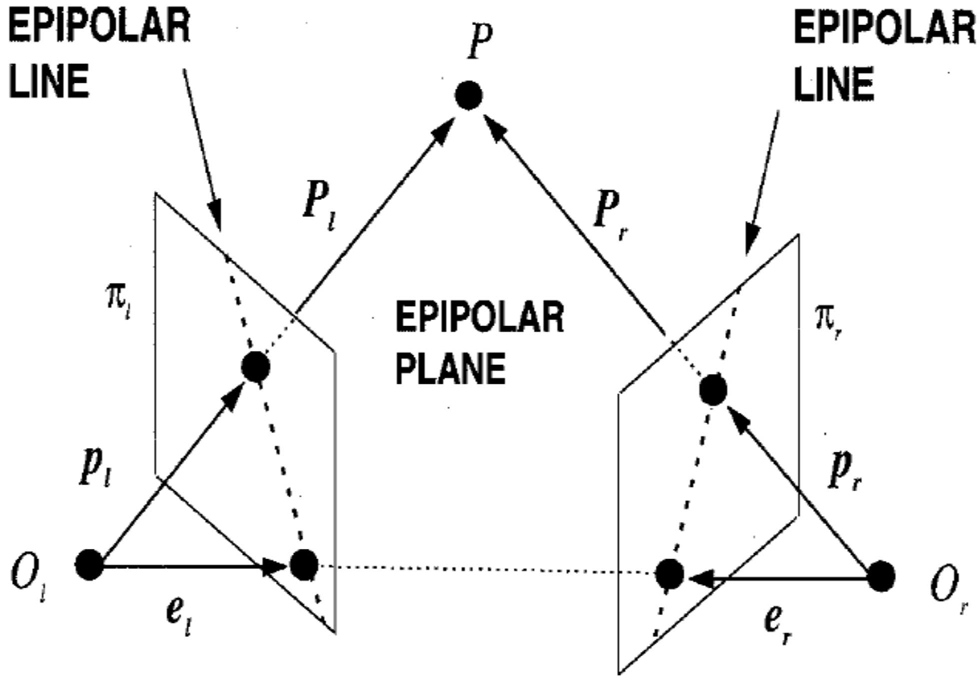


Figure 7.6 The epipolar geometry.

The points **P**, $\mathbf{O}_l$ and $\mathbf{O}_r$ form the **epipolar plane** for point **P**. Hence vectors $\mathbf{P}_l$, $\mathbf{O}_r\mathbf{O}_l$ (which equals **T**, the translation vector) and $\mathbf{P}_r$ are coplanar. Now $\mathbf{P}_r = \mathbf{R}(\mathbf{P}_l - \mathbf{T})$. Hence we can write:

$$(\mathbf{P}_l - \mathbf{T})^t (\mathbf{T} \times \mathbf{P}_l) = 0$$

$$\therefore (\mathbf{R}^T \mathbf{P}_r)^t (\mathbf{T} \times \mathbf{P}_l) = 0$$

$$\mathbf{T} \times \mathbf{P}_l = \begin{pmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{pmatrix} \mathbf{P}_l = \mathbf{S}\mathbf{P}_l$$

$$\therefore \mathbf{P}_r{}^t (\mathbf{RS}) \mathbf{P}_l = 0$$

$$\therefore \mathbf{P}_r{}^t (\mathbf{E}) \mathbf{P}_l = 0$$

$\mathbf{E}$ has rank 2.

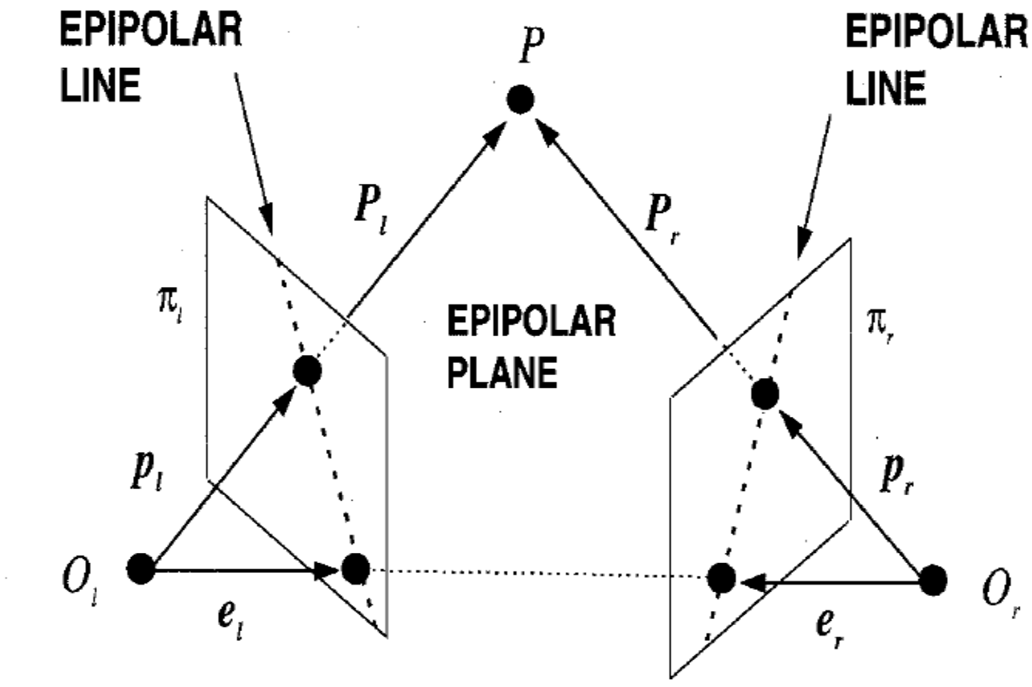


Figure 7.6 The epipolar geometry.

$$\mathbf{P}_r^{\,t}(\mathbf{RS})\mathbf{P}_l = 0$$

$$\mathbf{P}_r^{\,t}(\mathbf{E})\mathbf{P}_l = 0$$

$\mathbf{E}$ has rank 2.

E is the essential matrix. It gives an explicit relationship between the epipolar lines and the extrinsic parameters of the stereo system. What's more – given a set of corresponding points (in camera coordinate system), one can recover the essential matrix!

$$\mathbf{p}_r = \frac{f_r}{Z_r}\mathbf{P}_r, \mathbf{p}_l = \frac{f_l}{Z_l}\mathbf{P}_l$$

As $\mathbf{P}_r^{\mathbf{t}}\mathbf{E}\mathbf{P}_l = 0$, we have

$$\frac{f_r}{Z_r}\mathbf{P}_r^{\mathbf{t}}\mathbf{E}\frac{f_l}{Z_l}\mathbf{P}_l = 0$$

$$\therefore \mathbf{p}_r^{\mathbf{t}}\mathbf{E}\mathbf{p}_l = 0$$

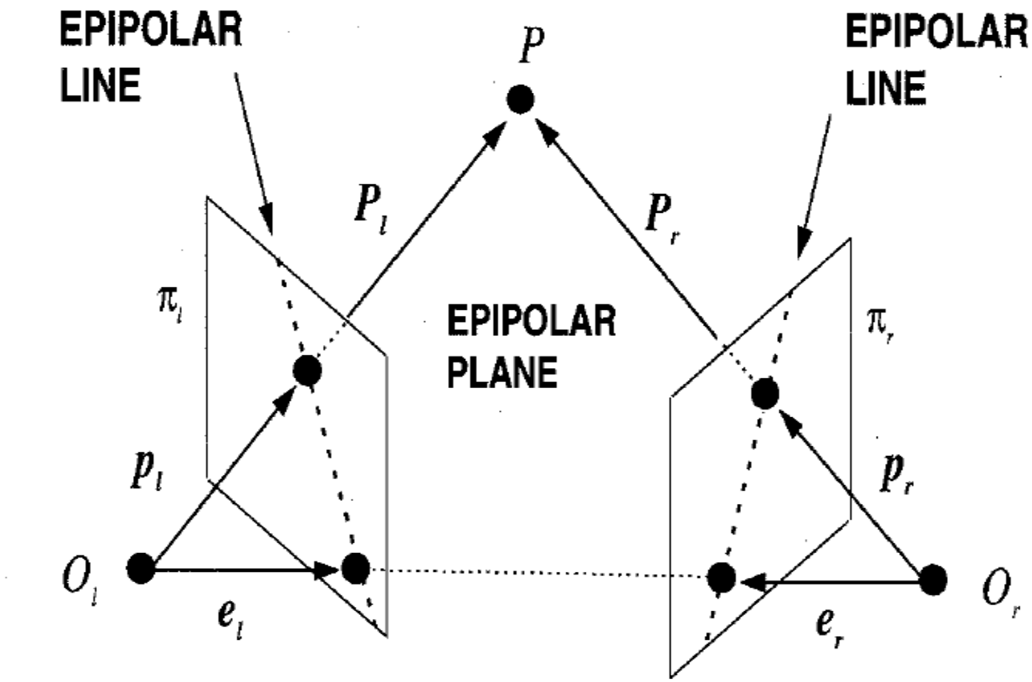


Figure 7.6 The epipolar geometry.

$$\mathbf{p}_r^{\mathrm{t}}\mathbf{E}\mathbf{p}_l = 0$$

$$\mathbf{p}_l = \mathbf{M}_l^{-1}\tilde{\mathbf{p}}_l$$

$$\mathbf{p}_r = \mathbf{M}_r^{-1}\tilde{\mathbf{p}}_r$$

$$\therefore \tilde{\mathbf{p}}_r^t[(\mathbf{M}_r^{-1})^T\mathbf{E}\mathbf{M}_l^{-1}]\tilde{\mathbf{p}}_l = 0$$

$$\therefore \tilde{\mathbf{p}}_r^t[\mathbf{F}]\tilde{\mathbf{p}}_l = 0$$

Intrinsic parameter matrices for left and right cameras

The **essential matrix E** gives the relationship between the corresponding points measured in **camera coordinates**. The **fundamental matrix F** gives the relationship between the corresponding points measured in **homogeneous coordinates** with the x and y components measured in the **pixel coordinate system**. **F** also has rank 2.

Essential and fundamental matrix

- Consider $\mathbf{p}_r^{\mathbf{t}}\mathbf{E}\mathbf{p}_l = 0, \widetilde{\mathbf{p}}_r^t[\mathbf{F}]\widetilde{\mathbf{p}}_l = 0$.

- These equations tell you that given a fixed point $\mathbf{p}_l$ in the left image, the corresponding point in the right image (i.e. $\mathbf{p}_r$) lies on a line (what’s the equation of the line?).

Determining fundamental and essential matrix

- We now look at an algorithm to determine the fundamental matrix given 8 or more pairs of corresponding points (in pixel coordinates) from the left and right images.
- The algorithm is called **Eight-Point Algorithm**.
- There is a very similar algorithm for determining the essential matrix (given points in camera coordinates) from 8 points.
- As **E** has only 5 DOF (why?), there exist algorithms that require just 5 correspondences, but those are a lot more complicated.

Determining fundamental and essential matrix

- The fundamental matrix **F** has 7 DOF (the first two rows = 6 DOF + third row = linear combination of first two rows, giving 8 DOF – minus 1 since the scale factor is removed).
- There exist algorithms that need only 7 points, but they are not as simple as the 8-point algorithm.
- Note: these 8 pairs can be obtained from manual input or using SIFT.

Eight point algorithm

$\forall i, 1 \le i \le N, \tilde{\mathbf{p}}^t_{\boldsymbol{r,i}}[\mathbf{F}]\tilde{\mathbf{p}}_{\boldsymbol{l,i}} = 0$

Let $\tilde{\mathbf{p}}_{\boldsymbol{r,i}} = (x_{r,i}, y_{r,i}, 1), \tilde{\mathbf{p}}_{\boldsymbol{l,i}} = (x_{l,i}, y_{l,i}, 1)$

$\therefore$ we have :

$$\begin{pmatrix} x_{r,1}x_{l,1} & x_{r,1}y_{l,1} & x_{r,1} & y_{r,1}x_{l,1} & y_{r,1}y_{l,1} & y_{r,1} & x_{l,1} & y_{l,1} & 1 \\ x_{r,2}x_{l,2} & x_{r,2}y_{l,2} & x_{r,2} & y_{r,2}x_{l,2} & y_{r,2}y_{l,2} & y_{r,2} & x_{l,2} & y_{l,2} & 1 \\ . & & & & & & & & . \\ . & & & & & & & & . \\ . & & & & & & & & . \\ . & & & & & & & & . \\ . & & & & & & & & . \\ . & & & & & & & & . \\ x_{r,N}x_{l,N} & x_{r,N}y_{l,N} & x_{r,N} & y_{r,N}x_{l,N} & y_{r,N}y_{l,N} & y_{r,N} & x_{l,N} & y_{l,N} & 1 \end{pmatrix} \begin{pmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$\therefore \mathbf{Af} = \mathbf{0}$

Eight point algorithm

- We solve for **f** (which contains the 9 entries of **F**) by computing the SVD of **A** (size N by 9, $N \geq 8$) and taking the column vector from **V** corresponding to the least singular value.
- The solution is obtained up to an arbitrary sign and scaling constant.
- Ideally **A** has rank 8 (proof out of scope) but in practice **A** has rank 9 (due to errors in measurement of point coordinates).

Eight point algorithm

- Rearrange elements of **f** to give **F** (up to a scaling constant and sign).
- **F** has size 3 by 3, but it should have rank 2, i.e. it should be rank-deficient. The previous step does not guarantee rank-deficiency.
- So we need another step. Compute SVD of **F** and nullify its smallest singular value. This gives us the final **F**.

$$\mathbf{U_F S_F V_F^T} = \mathbf{F}$$

$$\text{Let } \mathbf{S_F} = diag(a,b,c), a \geq b \geq c$$

$$\mathbf{F}_{final} = \mathbf{U_F}\, diag(a,b,0)\mathbf{V_F^T}$$

Find the nearest rank-2 matrix! Use SVD again (Eckhart-Young theorem)

Eight point algorithm

In practice, the stability of the estimates can be improved by performing some pre- and post-processing steps:

$$* \bar{x}_r = \frac{\sum_{i=1}^{N} x_{r,i}}{N}, \bar{y}_r = \frac{\sum_{i=1}^{N} y_{r,i}}{N}, \bar{x}_l = \frac{\sum_{i=1}^{N} x_{l,i}}{N}, \bar{y}_l = \frac{\sum_{i=1}^{N} y_{l,i}}{N}$$

$$* \sigma_r = \frac{\sum_{i=1}^{N} \sqrt{(x_{r,i} - \bar{x}_r)^2 + (y_{r,i} - \bar{y}_r)^2}}{N}, \sigma_l = \frac{\sum_{i=1}^{N} \sqrt{(x_{l,i} - \bar{x}_l)^2 + (y_{l,i} - \bar{y}_l)^2}}{N}$$

$$* x'_{r,i} \leftarrow \frac{x_{r,i} - \bar{x}_r}{\sigma_r}, y'_{r,i} \leftarrow \frac{y_{r,i} - \bar{y}_r}{\sigma_r}, x'_{l,i} \leftarrow \frac{x_{l,i} - \bar{x}_l}{\sigma_l}, y'_{l,i} \leftarrow \frac{y_{l,i} - \bar{y}_l}{\sigma_l}$$

$$* \text{Now estimate the fundamental matrix } \mathbf{F_1} \text{ from } \{(x'_{r,i}, y'_{r,i}), (x'_{l,i}, y'_{l,i})\}_{i=1}^{N}.$$

$$* \mathbf{F} \text{ can be estimated from } \mathbf{F_1}.$$

Estimating epipoles from **F**

- The left epipole lies on all epipolar lines in the left image. Hence we can write:

$$\tilde{\mathbf{p}}_{\mathbf{r}}^{\mathbf{t}}\mathbf{F}\tilde{\mathbf{e}}_{l} = 0$$

$$\therefore \mathbf{F}\tilde{\mathbf{e}}_{l} = \mathbf{0}$$

$$\therefore \tilde{\mathbf{e}}_{l} \text{ lies in the nullspace of } \mathbf{F}.$$

$$\text{Likewise, } \tilde{\mathbf{e}}_{\mathbf{r}} \text{ lies in the nullspace of } \mathbf{F}^{T}.$$

$$\mathbf{U_F S_F V_F^T} = \mathbf{F}$$

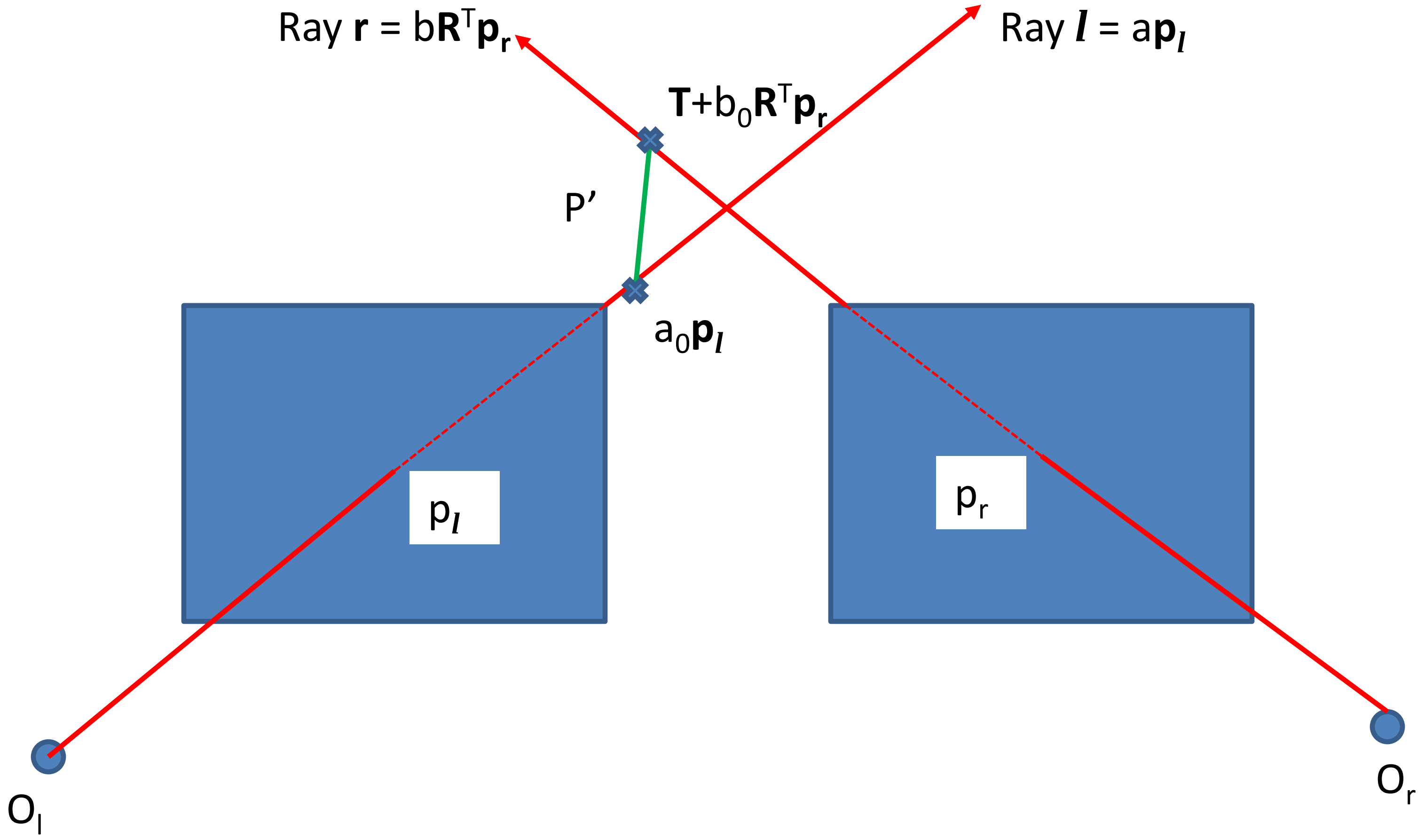
$$\tilde{\mathbf{e}}_{l} = \text{column of } \mathbf{V_F} \text{ corresponding to null singular value}$$

$$\tilde{\mathbf{e}}_{\mathbf{r}} = \text{column of } \mathbf{U_F} \text{ corresponding to null singular value}$$

More about using **E** or **F**

- We saw how **F** can be estimated from 8 pairs of corresponding points.
- Given **F**, we get the equation for the epipolar line for any point, which will restrict the search space for correspondences along this line (instead of the whole image).
- If the camera instrinsic parameters are known, we can also determine **E**, and use that to infer **R** and **T** (we will see how this inference is done later).

3D reconstruction: known parameters



3D reconstruction: known parameters

- The rays **r** and $\boldsymbol{l}$ may not intersect in practice due to measurement errors.
- Instead we find a line segment **s** perpendicular to both **r** and $\boldsymbol{l}$, with one endpoint on **r** and another on $\boldsymbol{l}$.
- Thus we have **s** lying on the line $\mathbf{w} = \mathbf{p}_{\boldsymbol{l}} \times \mathbf{R}^{T}\mathbf{p}_{\mathbf{r}}$.
- We treat the midpoint of **s** as the point of intersection of rays **r** and $\boldsymbol{l}$. The midpoint is the point of minimum distance from rays **r** and $\boldsymbol{l}$.

3D reconstruction: known parameters

- The concerned segment starts at point $a_0\mathbf{p}_l$ on ray $\boldsymbol{l}$ and ends at point $\mathbf{T}+b_0\mathbf{R}^\mathbf{T}\mathbf{p}_\mathbf{r}$ on ray **r**.
- A point on segment **s** (note that segment **s** lies on line **w**) can be expressed as $a_0\mathbf{p}_l + c_0\mathbf{w} =$
  $a_0\mathbf{p}_l + c_0 (\mathbf{p}_l \times \mathbf{R}^T\mathbf{p}_\mathbf{r})$.
- Hence we have $\mathbf{T}+b_0\mathbf{R}^\mathbf{T}\mathbf{p}_\mathbf{r} = a_0 \mathbf{p}_l + c_0 (\mathbf{p}_l \times \mathbf{R}^T\mathbf{p}_\mathbf{r})$. Solve for the coefficients a_0,b_0,c_0.
- Moral of the story: With known camera parameters, 3D reconstruction is essentially unambiguous. Accuracy depends on noise level.

3D reconstruction: only intrinsic parameters are known.

- **Assumptions:** intrinsic parameters known, **N** = 8+ pairs of corresponding points are available.
- Essential matrix **E** (instead of fundamental matrix **F**) can be easily computed as pixel coordinates can be converted to camera coordinates.
- But 3D coordinates can be computed only up to an unknown scale factor since extrinsic parameters are unknown.
- The scale factor can be determined if you knew beforehand the exact distance between 2 points in the scene.

3D reconstruction: only intrinsic parameters are known.

$$\mathbf{E} = \mathbf{RS}$$

Remember: **E** is known only up to an unknown scale and sign!

$$\mathbf{E}^T\mathbf{E} = \mathbf{S}^T\mathbf{S}$$

$$\therefore \mathbf{E}^T\mathbf{E} = \begin{pmatrix} T_y^2 + T_z^2 & -T_xT_y & -T_xT_z \\ -T_xT_y & T_x^2 + T_z^2 & -T_yT_z \\ -T_xT_z & -T_yT_z & T_y^2 + T_x^2 \end{pmatrix}$$

$$\therefore \mathrm{trace}(\mathbf{E}^T\mathbf{E}) = 2(T_x^2 + T_y^2 + T_z^2) = 2\|\mathbf{T}\|^2$$

$$\therefore \|\mathbf{T}\| = \pm\sqrt{\mathrm{trace}(\mathbf{E}^T\mathbf{E})/2}$$

$$\hat{\mathbf{E}} = \mathbf{E}/\|\mathbf{T}\|$$

Normalized essential matrix

$$\therefore \hat{\mathbf{E}}^T\hat{\mathbf{E}} = \begin{pmatrix} 1-\hat{T}_x^2 & -\hat{T}_x\hat{T}_y & -\hat{T}_x\hat{T}_z \\ -\hat{T}_x\hat{T}_y & 1-\hat{T}_y^2 & -\hat{T}_y\hat{T}_z \\ -\hat{T}_x\hat{T}_z & -T_yT_z & 1-\hat{T}_z^2 \end{pmatrix}$$

Now estimate the components of **T** – but these can be recovered only up to an unknown common sign and scaling factor.

3D reconstruction: only intrinsic parameters are known.

You know **T** (up to a sign and scale), so you know **S** (up to the same sign and scale)

$$\hat{S} = \begin{pmatrix} 0 & -\hat{T}_z & \hat{T}_y \\ \hat{T}_z & 0 & -\hat{T}_x \\ -\hat{T}_y & \hat{T}_x & 0 \end{pmatrix}$$

$$\hat{\mathbf{E}} = \hat{\mathbf{R}}\hat{\mathbf{S}}$$

$$\hat{\mathbf{R}} = \begin{pmatrix} \hat{\mathbf{R}}_1 \\ \hat{\mathbf{R}}_2 \\ \hat{\mathbf{R}}_3 \end{pmatrix},$$

$$\hat{\mathbf{R}}_1 = \mathbf{W}_1 + \mathbf{W}_2 \times \mathbf{W}_3,$$

$$\hat{\mathbf{R}}_2 = \mathbf{W}_2 + \mathbf{W}_3 \times \mathbf{W}_1,$$

$$\hat{\mathbf{R}}_3 = \mathbf{W}_3 + \mathbf{W}_1 \times \mathbf{W}_2$$

$$\mathbf{W}_i = \hat{\mathbf{E}}_i \times \hat{\mathbf{T}}, i \in \{1,2,3\}$$

Row i of the normalized essential matrix

Method by Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections", Nature, 1981

3D reconstruction: only intrinsic parameters are known.

$$\mathbf{P_r} = \hat{\mathbf{R}}(\mathbf{P}_l - \hat{\mathbf{T}})$$

$$\therefore Z_r = \hat{\mathbf{R}}_3^T(\mathbf{P}_l - \hat{\mathbf{T}})$$

$$\therefore \mathbf{p}_r = \frac{f_r \hat{\mathbf{R}}^T(\mathbf{P}_l - \hat{\mathbf{T}})}{\hat{\mathbf{R}}_3^T(\mathbf{P}_l - \hat{\mathbf{T}})}$$

$$\text{But } \mathbf{p}_l = \frac{f_l \mathbf{P}_l}{Z_l}$$

$$\therefore \mathbf{P}_l = \frac{Z_l \mathbf{p}_l}{f_l}$$

Plug in the expression for **P**$_l$ into the expression for **p**$_r$ and re-arrange to get an expression for Z$_l$

As we know the translation direction only and not its magnitude

Solve for Z_l (uptoa scale) and hence Z_r (uptoa scale)

$$Z_l = f_l \frac{(f_r \hat{\mathbf{R}}_1 - x_r \hat{\mathbf{R}}_3)^T \hat{\mathbf{T}}}{(f_r \hat{\mathbf{R}}_1 - x_r \hat{\mathbf{R}}_3)^T \mathbf{p}_l}, \mathbf{P_r} = \hat{\mathbf{R}}(\mathbf{P}_l - \hat{\mathbf{T}})$$

$$Z_r = \hat{\mathbf{R}}_3^T\left(\frac{Z_l \mathbf{p}_l}{f_l} - \hat{\mathbf{T}}\right)$$

3D reconstruction: only intrinsic parameters are known.

Out of the four solutions of $(\hat{\mathbf{E}}, \hat{\mathbf{T}})$, only one of them is valid, i.e. yields positive values of Z_l and Z_r for all points.

1. Estimate $\hat{\mathbf{E}}$ (up to unknown sign)

2. Estimate $\hat{\mathbf{T}}$ (up to unknown sign)

3. Estimate $\hat{\mathbf{R}}$

4. Estimate Z_l and Z_r for all points

5a. If the values of Z_l and Z_r are both negative for some point, then change the sign of $\hat{\mathbf{T}}$ and go to step 4

5b. If the values of Z_l and Z_r are both positive for all points, then exit.

5c. If either Z_l or Z_r (exactly one) is negative, then change the sign of all entries in $\mathbf{E}$ and go to step 3

3D reconstruction: only intrinsic parameters are known.

- **To summarize:**

✓ Our input was a set of **N** = 8+ corresponding points from two images taken with cameras of known intrinsic parameters. The extrinsic parameters of the stereo system (i.e. rotation and translation between the optical axes of the two cameras) are unknown.

✓ In such a case, you can estimate only the direction of the baseline vector (i.e. translation direction **T**) and not its magnitude.

✓ You can estimate the 3D coordinates of the points only up to an unknown scale.

✓ I will once again remind you: we assume correspondences were available or were manually marked. Automated correspondences is not an easy problem, and we will study it soon.

3D reconstruction: intrinsic and extrinsic parameters are unknown

- Consider equations for a corresponding pair of points:

$$\begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix} = \mathbf{P_1} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}, \begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} = \mathbf{P_2} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}, \mathbf{P_1} \text{ and } \mathbf{P_2} \text{ are projection matrices of size } 3 \text{ x } 4$$

- Now consider:

$$\begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix} = (\mathbf{P_1 A})\mathbf{A}^{-1} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}, \begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} = (\mathbf{P_2 A})\mathbf{A}^{-1} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}, \mathbf{A} \text{ is an arbitrary invertible matrix}$$

$$\text{of size } 4 \text{ x } 4$$

3D reconstruction: intrinsic and extrinsic parameters are unknown

- This means that for any invertible matrix **A** (size 4 by 4), exactly the same pair of images would be produced by cameras with projection matrices $\mathbf{P_1 A}$ and $\mathbf{P_2 A}$, and 3D points whose coordinates are given by

$\{\mathbf{A}^{-1}(X_i|Y_i|Z_i|1)^t\}$.

Correspondence problem

- Several methods:

✓ Correlations/squared difference based methods

✓ Optimization method for inferring the disparity map

✓ Feature-based methods/ Constrained methods – based on dynamic programming

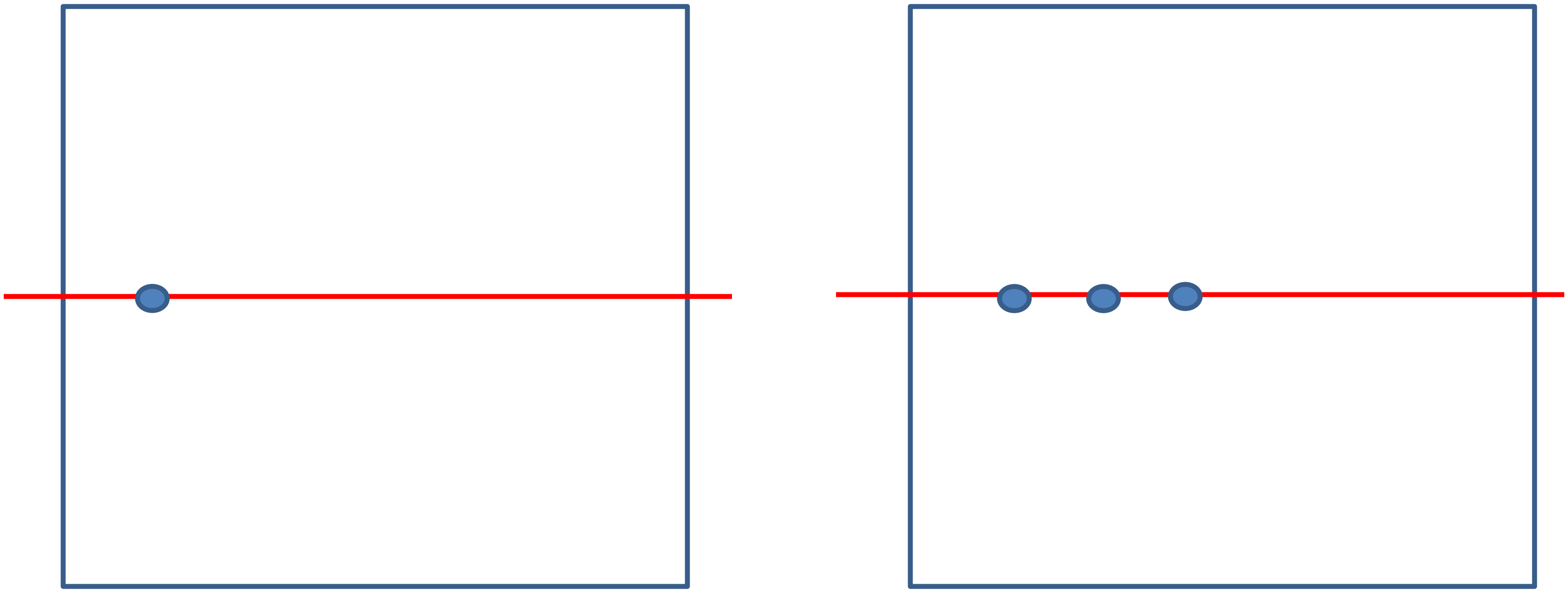
Assumptions

- We will assume the case of coordinate systems of the two cameras being parallel (only a simplification – the method is applicable to the more general case), and their X axes being parallel to the baseline.
- Consider $\mathbf{p}_l = (x_l, y_l)$ and $\mathbf{p}_r = (x_r, y_r)$ are images of a given point (X,Y,Z) in the two cameras.
- Assume that the gray-levels of corresponding points in the two images are equal.
- So, $I_l(x_l, y_l) = I_r(x_r, y_r)$.

Assumptions

- Is this brightness constancy assumption valid here?
- Yes, if object is Lambertian.
- Violations: noise, specularity, shadows, occlusion, non-Lambertian surfaces

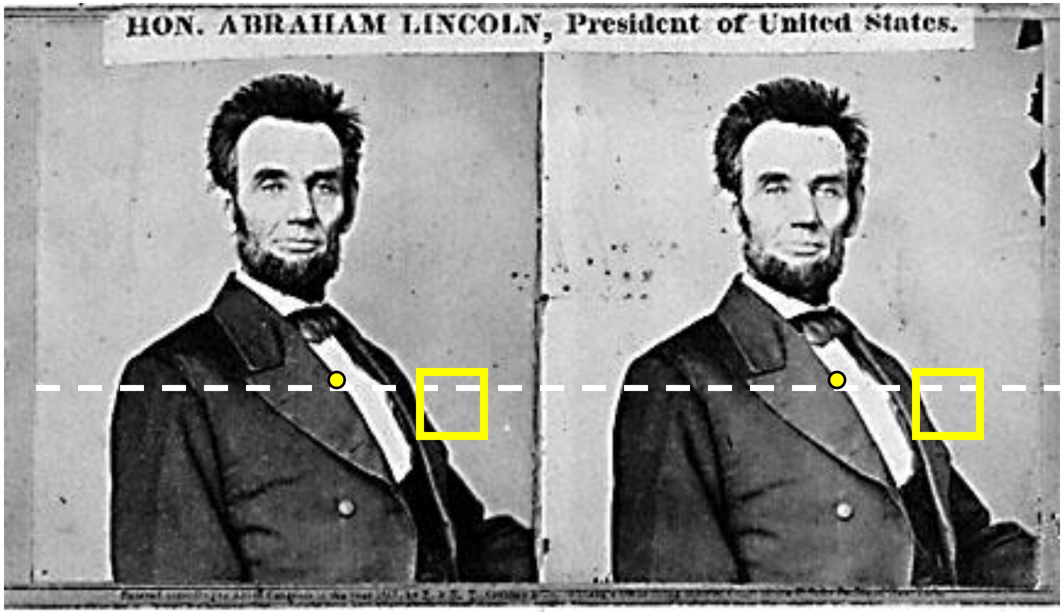
Remember: epipolar constraint!



But ambiguity remains!

Method 1: Comparing patches using correlation or squared differences

Slide taken from a University of Washington course on computer vision – Steve Seitz



For each epipolar line

For each pixel in the left image

- compare with every pixel on same epipolar line in right image
- pick pixel with minimum match cost

This leaves too much ambiguity, so:

Improvement: match ***patches (also called windows)***

(Seitz)

Method 1: Correlation or squared difference

- Assume most scene points are visible from both cameras (perfectly reasonable)
- Corresponding image ***regions*** are similar.
- Define image region as a square-shaped patch of size ($2K$+1) x ($2K$+1).

Method 1: Correlation or squared difference

- For each pixel (x_l,y_l) in $\mathbf{I}_l$, and every possible displacement $(d^{(x)},0)$, find coordinates $(x_r,y_r)=(x_l,y_l)+(d^{(x)},0)$ in $\mathbf{I}_r$ such that the SSD is minimized or Correlation is maximized:

$$SSD(d) = \sum_{i=-K}^{K}\sum_{j=-K}^{K}(I_l(x_l + j, y_l + i) - I_r(x_l + d^{(x)} + j, y_l + i))^2$$

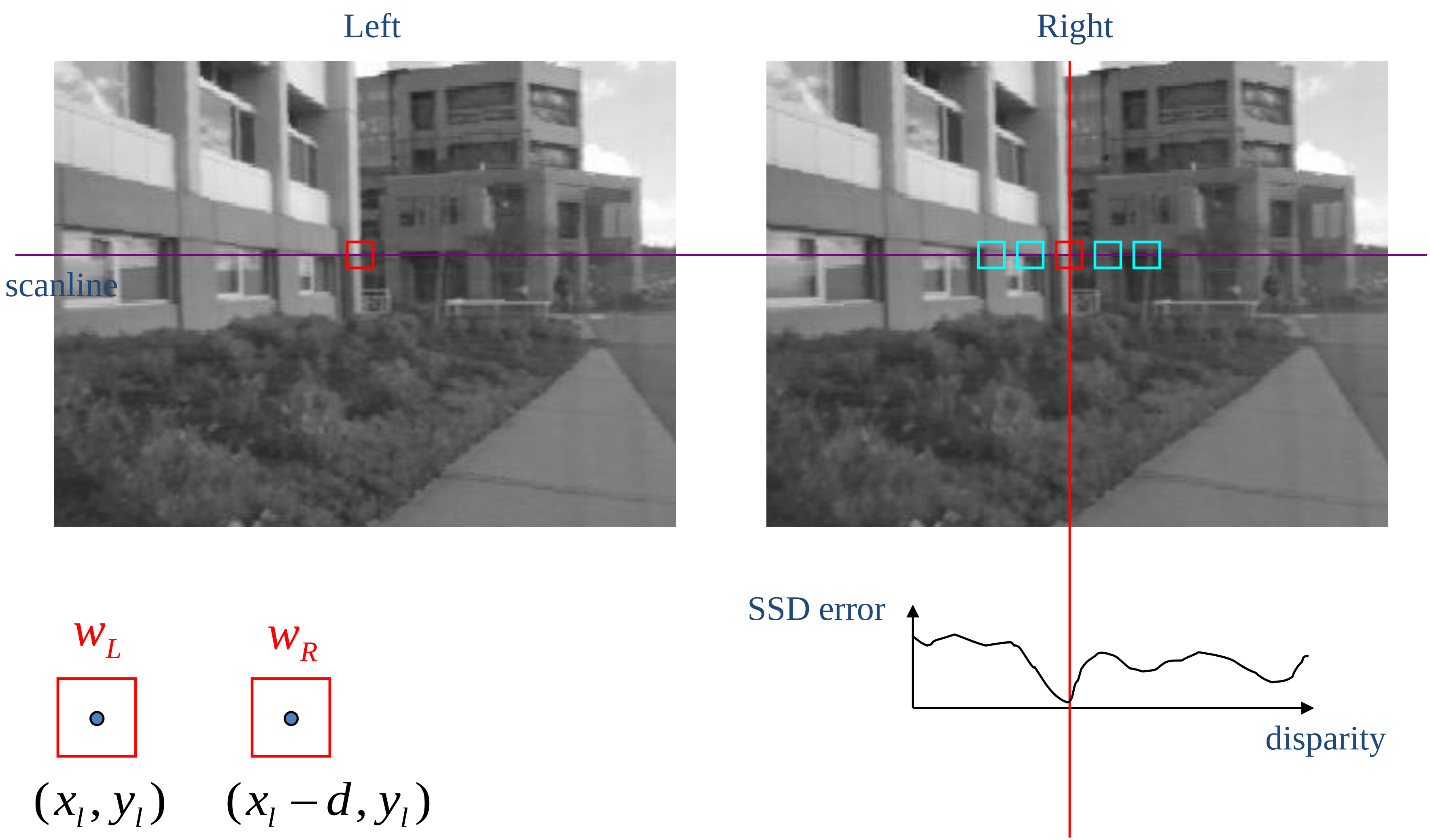
$$d^* = \min_{d \in R(p_l)} SSD(d)$$

R(p_l) – the search window – chosen to be small to avoid very faraway similar patches from being selected

$$Corr(d) = \sum_{i=-K}^{K}\sum_{j=-K}^{K} I_l(x_l + j, y_l + i) I_r(x_l + d^{(x)} + j, y_l + i)$$

$$d^* = \max_{d \in R(p_l)} Corr(d)$$

Slide taken from a University of Washington course on computer vision – Steve Seitz



Method 1: Correlation or squared difference

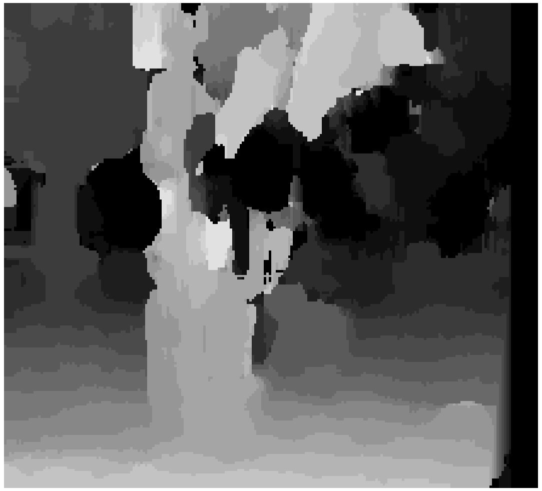
- If there is illumination difference between the two images, you can maximize normalized cross-correlation instead

$$NCorr(d) = \frac{\left| \sum_{i=-K}^{K} \sum_{j=-K}^{K} (q_l(x_l + j, y_l + i) - \bar{q}_l)(q_r(x_l + d^{(x)} + j, y_l + d^{(y)} + i) - \bar{q}_r) \right|}{\sqrt{\sum_{i=-K}^{K} \sum_{j=-K}^{K} (q_l(x_l + j, y_l + i) - \bar{q}_l)^2} \sqrt{\sum_{i=-K}^{K} \sum_{j=-K}^{K} (q_r(x_l + d^{(x)} + j, y_l + d^{(y)} + i) - \bar{q}_r)^2}}$$

$$d^* = \max_{d \in R(p_l)} Corr(d)$$

Slide taken from a University of Washington course on computer vision – Steve Seitz

W = 3



W = 20

- Effect of window size
- Some approaches have been developed to use an adaptive window size (try multiple sizes and select best match)

(Seitz)

Method 2: Feature-based methods

- Instead of computing SSD over intensity, compute it over features such as some combination of

  (i) image gradient magnitude/orientation

  (ii) average/variance of intensity values in a window
- The latter may make the search faster.

Method 3: Optimization method to infer disparity map

From book by B K P Horn

$$I_l(x_l, y) = I_r(x_r, y)$$
$$\therefore I_l(x, y) = I_r(x + d(x, y), y)$$
$$\therefore d^* = \min_d \iint_{\Omega} (I_l(x, y) - I_r(x + d(x, y), y))^2 \, dxdy$$

Severely underconstrained – need to introduce smoothness terms

Method 3: Variational method to infer disparity map

$$d^* = \min_d \iint_\Omega \left[(I_l(x,y) - I_r(x + d(x,y), y))^2 + \lambda(d_x^2 + d_y^2) \right] dxdy$$

Taking derivatives w.r.t. d(x,y):

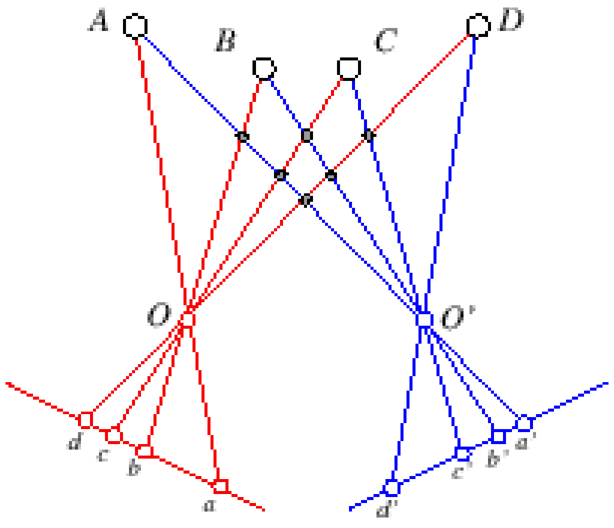
$$\therefore (I_r(x + d(x,y), y) - I_l(x,y)) \frac{\partial I_r(x + d(x,y), y)}{\partial d(x,y)} =$$

$$\lambda(4d(x,y) - d(x+1,y) - d(x,y+1) - d(x-1,y) - d(x,y-1))$$

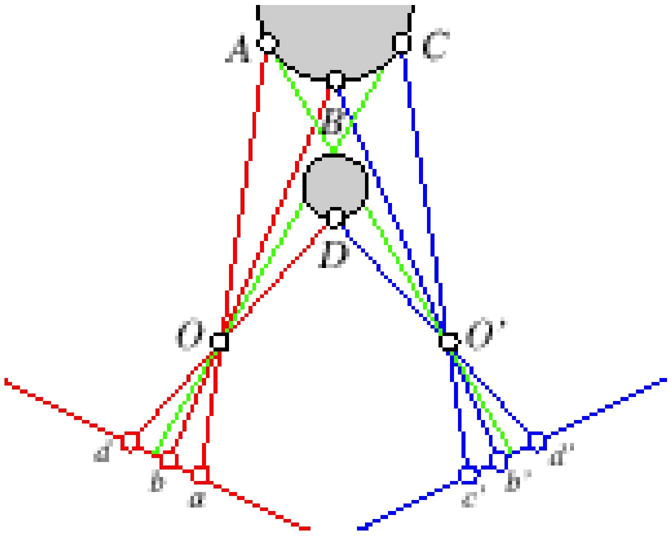
We can solve for d(x,y) at all locations iteratively using methods such as Jacobi.

Method 4: Dynamic programming

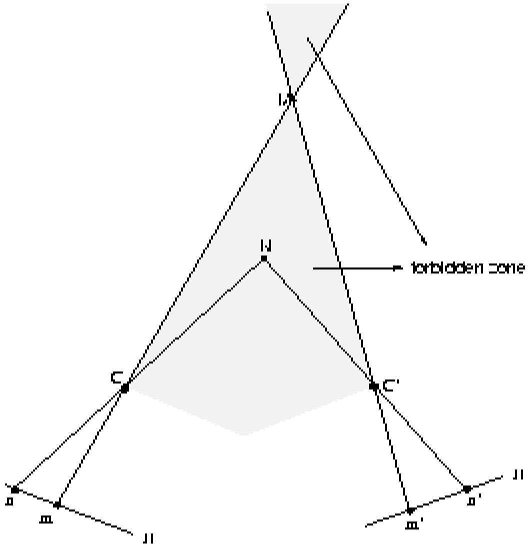
- There is one important constraint we didn't impose so far! **Ordering constraint.**



Ordering constraint…



…and its failure



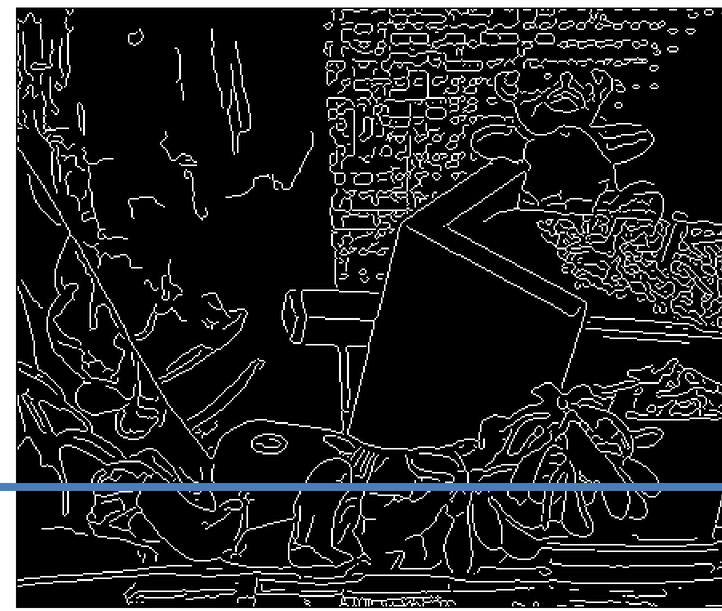
The ordering constraint fails if a given 3-D point (*N* here) falls onto the forbidden zone of another 3-D point (*M*). In the left image (Π), *m* is to the right of *n*, but in the right image (Π'), this ordering is reversed.

http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT11/node5.html

Method 4: Dynamic programming

- Step 1: Run an edge detection algorithm on both images.
- Remember: As we assumed parallel optical axes along Z direction with X-direction baseline, the epipolar lines are horizontal.
- Step 2: For each scanline $\mathbf{L}_l$ (epipolar line) in the left image, form a list of edge points. Form a similar list of edge points in the right image on the same scanline (denoted $\mathbf{L}_r$).
- The number of points in these lists may be unequal – let's denote it as M and N respectively.



Method 4: Dynamic programming

- We want to assign nodes from the left list to nodes in the **right** one.
- The **ordering constraint** must be obeyed – if point $\mathbf{a}_l$ is located before $\mathbf{b}_l$ on $\mathbf{L}_l$, then $\mathbf{a}_\mathbf{r}$ (the node to which $\mathbf{a}_l$ is assigned) must be located before $\mathbf{b}_\mathbf{r}$ (the node to which $\mathbf{b}_l$ is assigned) on $\mathbf{L}_r$.
- The assignment of correspondences can be framed as a problem of finding a path in a bounded 2D grid with top-left corner at (0,0) and bottom-right corner at (M,N) (see next slide).

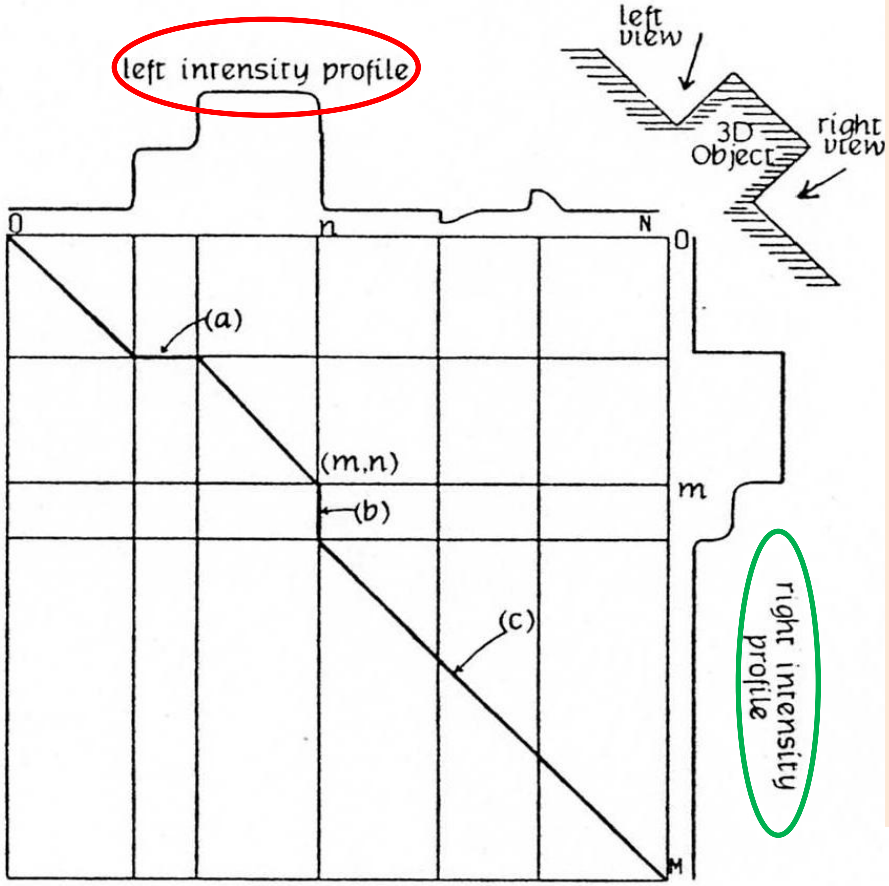


Fig. 3. 2D search plane for intra-scanline search. Intensity profiles are shown along each axis. The horizontal axis corresponds to the left scanline and the vertical one corresponds to the right scanline. Vertical and horizontal lines are the edge positions, and path selection is done at their intersections.

- ✓ Edge points on left scanline – vertical lines
- ✓ Edge points on right scanline – horizontal lines
- ✓ Find a legal path through this grid from grid-point (0,0) to grid-point (M,N) having least cost. A legal path moves from top-left to right-bottom corner of the grid monotonically, i.e. without moving backwards.
- ✓ A path contains a list of grid-points. Grid-point **q** = (*m*,*n*) is part of a path if edge point *m* in $\mathbf{L}_l$ is assigned to edge point *n* in $\mathbf{L}_r$.

Vertical lines: edges on the left scanline
Horizontal lines: edges on the right scanline
Grid-points = points of intersection of the horizontal and vertical lines

Source of figure: Ohta and Kanade, “Stereo by Intra- and Inter- scanline search using dynamic programming”, IEEE TPAMI, 1985

Method 4: Dynamic programming

- While searching for correspondence between a pair of edge points, one on $\mathbf{L}_l$ (say point $\mathbf{p}_l$) and one on $\mathbf{L}_r$ (say point $\mathbf{p}_r$), the edge points on the left of $\mathbf{p}_l$ and $\mathbf{p}_r$ (on $\mathbf{L}_l$ and $\mathbf{L}_r$ respectively) should already be processed!

- Start-point and end-point of $\mathbf{L}_l$ and $\mathbf{L}_r$ are both treated as edge-points for convenience.

Method 4: Dynamic programming

- We will denote the cost of a path from grid-point **k** to grid-point **m** as D(**m**,**k**). If **k** = (0,0) (i.e. top-left corner of the grid), then we simply denote the cost as D(**m**).
- The cost of a path is the sum total of the costs of its constituent *primitive paths*. A primitive path between grid-points **k** and **m** is a path that consists of a *single straight line segment.*
- The cost of the primitive path between **m** and **k** is denoted as d(**m**,**k**).

Method 4: Dynamic programming

Now, $\bar{D}(\boldsymbol{m})$ can be defined recursively as

$$D(\boldsymbol{m}) = \min_{\{\boldsymbol{i}\}} \{d(\boldsymbol{m}, \boldsymbol{m} - \boldsymbol{i}) + D(\boldsymbol{m} - \boldsymbol{i})\}$$

$$D(O) = 0$$

where $\boldsymbol{m} = (m, n), \boldsymbol{i} = (i, j), 0 \leqslant i \leqslant m,$
$0 \leqslant j \leqslant n, i + j \neq 0, \boldsymbol{o} = (0, 0).$

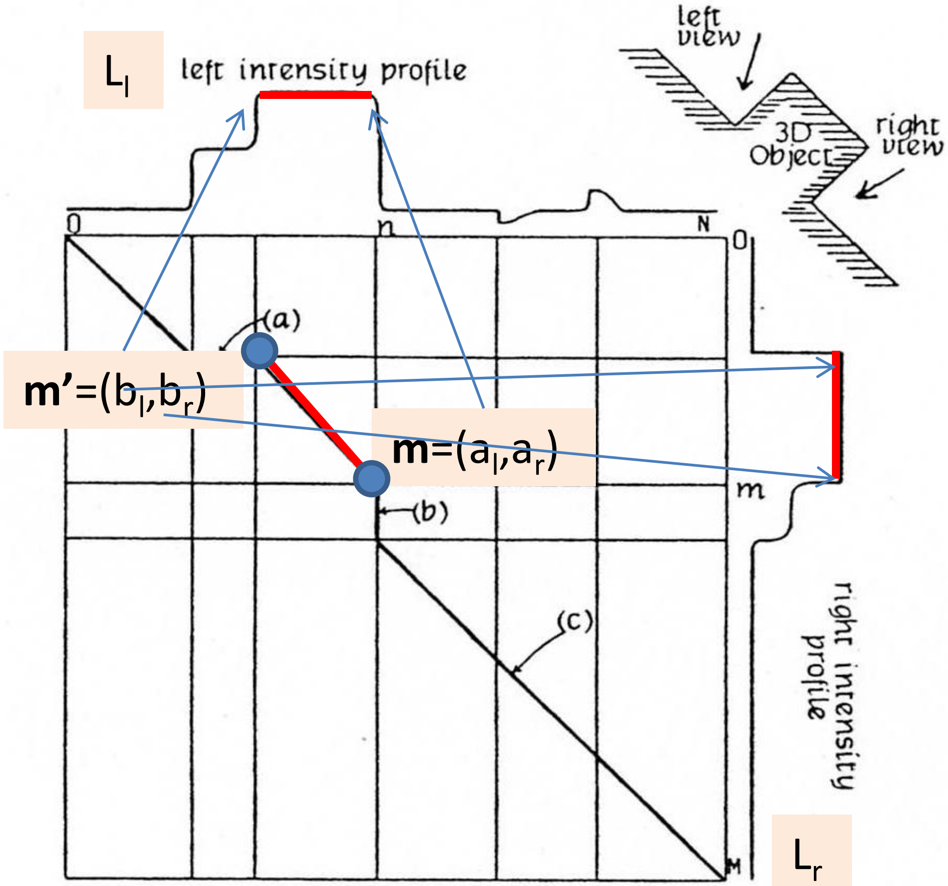


Fig. 3. 2D search plane for intra-scanline search. Intensity profiles are shown along each axis. The horizontal axis corresponds to the left scanline and the vertical one corresponds to the right scanline. Vertical and horizontal lines are the edge positions, and path selection is done at their intersections.

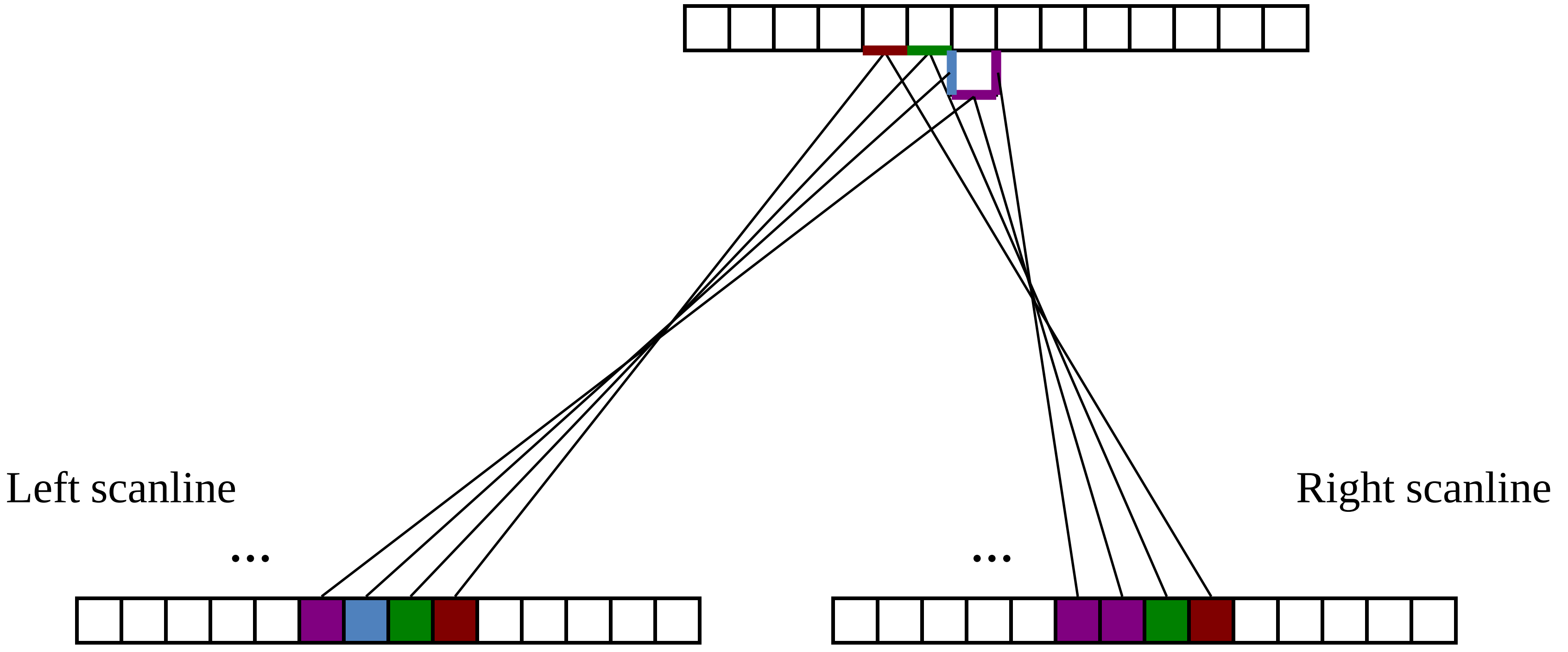
Let grid-point **m** = (a_l,a_r) and let grid-point **m'** = (b_l,b_r).
Then d(**m,m'**) = some measure of similarity between the intensity values in the interval (b_l,a_l) on L_l and the interval (b_r,a_r) on L_r.

Method 4: Dynamic programming

- Occlusions are intervals on the left scanline which have no match in the right scanline – represented by horizontal primitive paths (i = 0, in $\mathbf{i} = (i,j)$).
- Disocclusions are intervals on the right scanline that have no match from the left scanline – represented by vertical primitive paths (j = 0, in $\mathbf{i} = (i,j)$).
- Occlusions and disocclusions are assigned fixed costs.

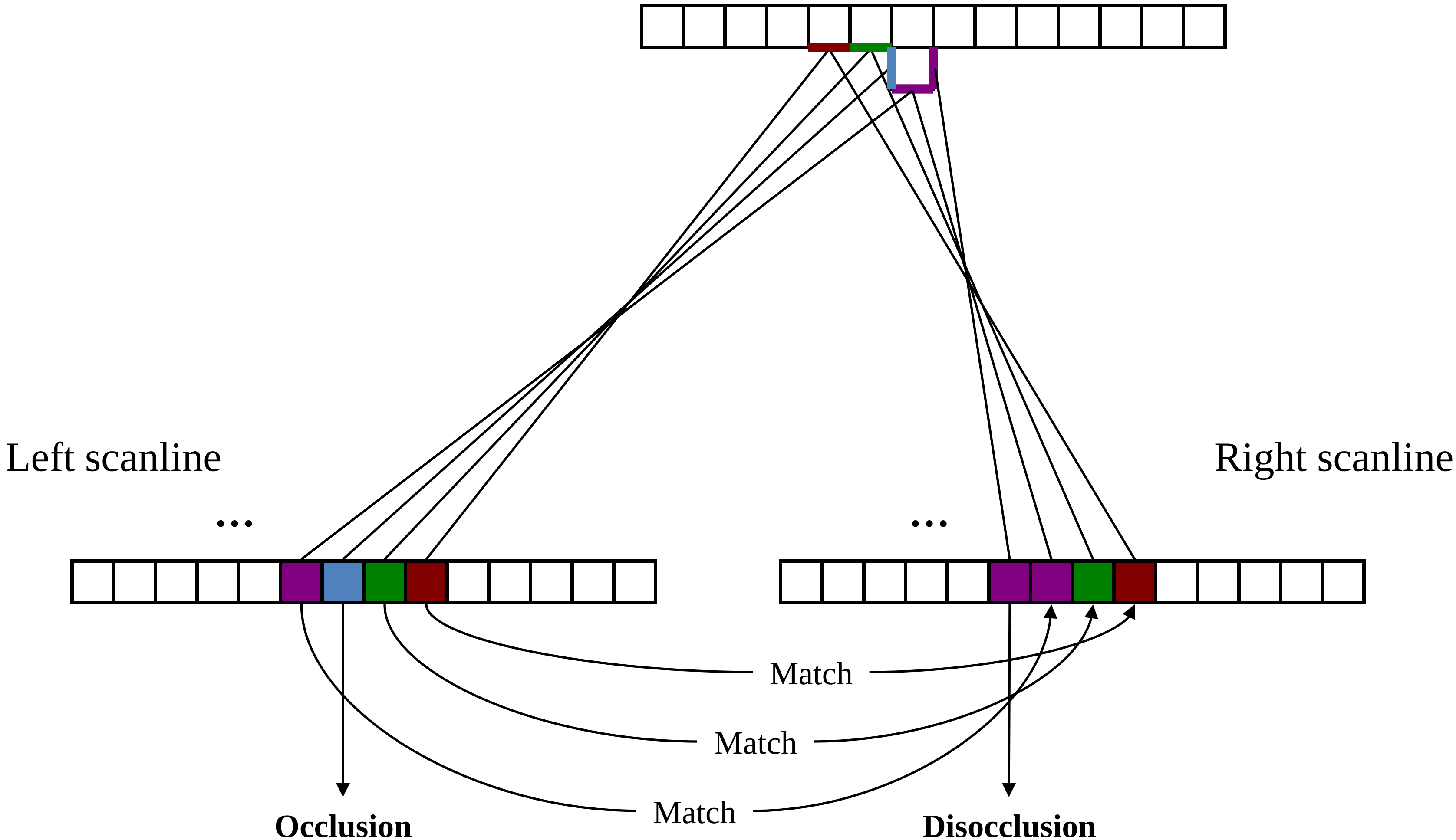
Slide taken from a University of Washington course on computer vision – Steve Seitz

Stereo Correspondences



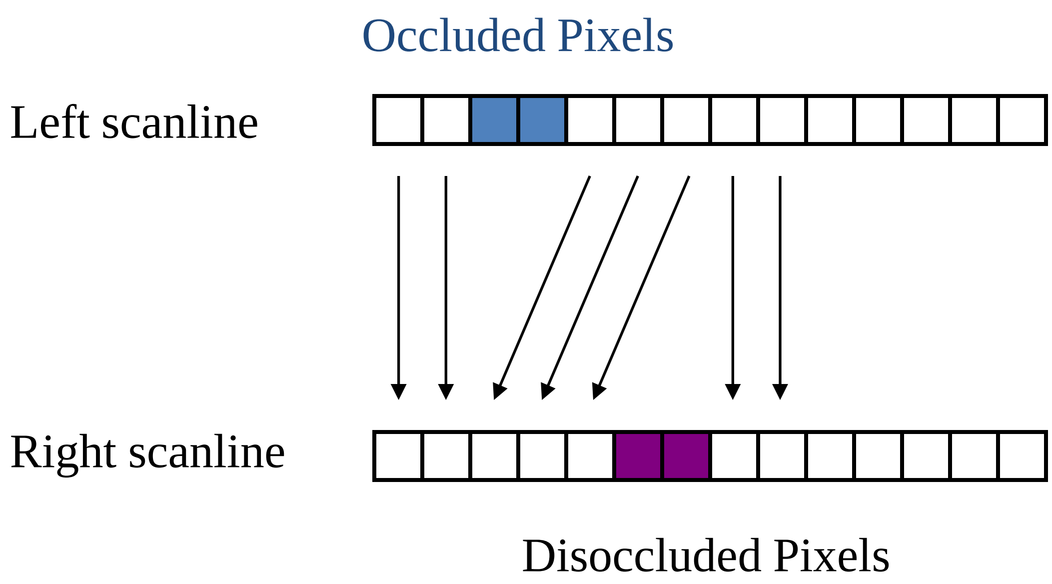
Slide taken from a University of Washington course on computer vision – Steve Seitz

Stereo Correspondences



Slide taken from a University of Washington course on computer vision – Steve Seitz

Search Over Correspondences

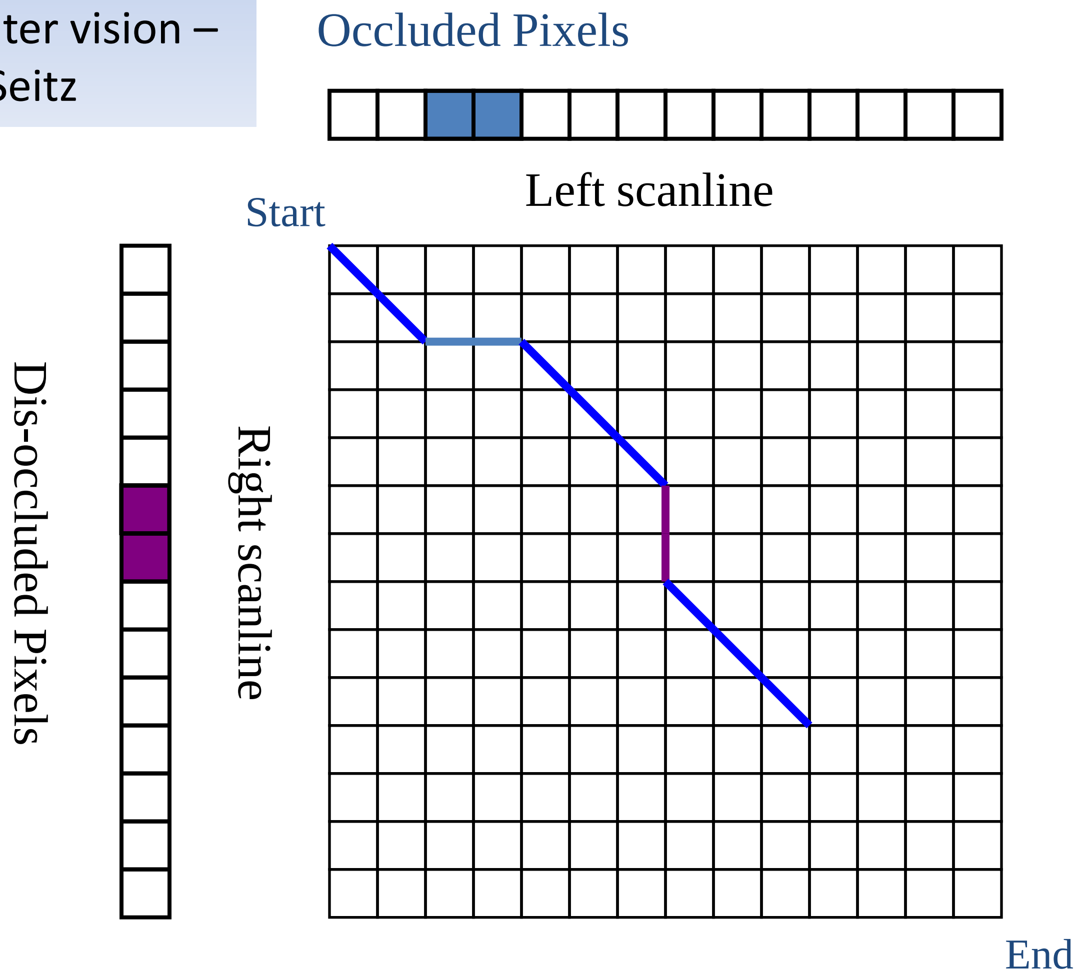


Three cases:

- Sequential – add cost of match (small if intensities agree)
- Occluded – add cost of no match (large cost)
- Disoccluded – add cost of no match (large cost)

Stereo Matching with Dynamic Programming

Slide taken from a University of Washington course on computer vision – Steve Seitz



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