RANSAC Recap

RANSAC - Setup

- Given
  - A dataset $D = \{ p_1, p_2, \ldots, p_N\}$
    - Example 1: Line fitting: $\{ (x_1, y_1), \ldots, (x_n, y_n)\}$
    - Example 2: Homography fitting: $\{ \left(\vec{Q}_1, \overrightarrow{q_1}\right), \left(\overrightarrow{Q_2}, \overrightarrow{q_2}\right), \ldots, \left(\overrightarrow{Q_N}, \overrightarrow{q_N}\right)\}$
  - A set of parameters θ that need to be fitted
    - Line fitting: $\theta = (m, b)$
    - Homography estimation $\theta = H, ||h|| = 1$
  - A cost function $C(p, \theta)$
    - Line fitting: $C\big((x, y), (m, b)\big) = ||y - (mx + b)||^2$
    - Homography estimation $C\left(\left(\vec{Q}, \vec{q}\right), H\right) = E(H)$ (Reprojection error)
  - A minimum number needed k
    - Line fitting: 2
    - Homography estimation: 4

RANSAC - Setup

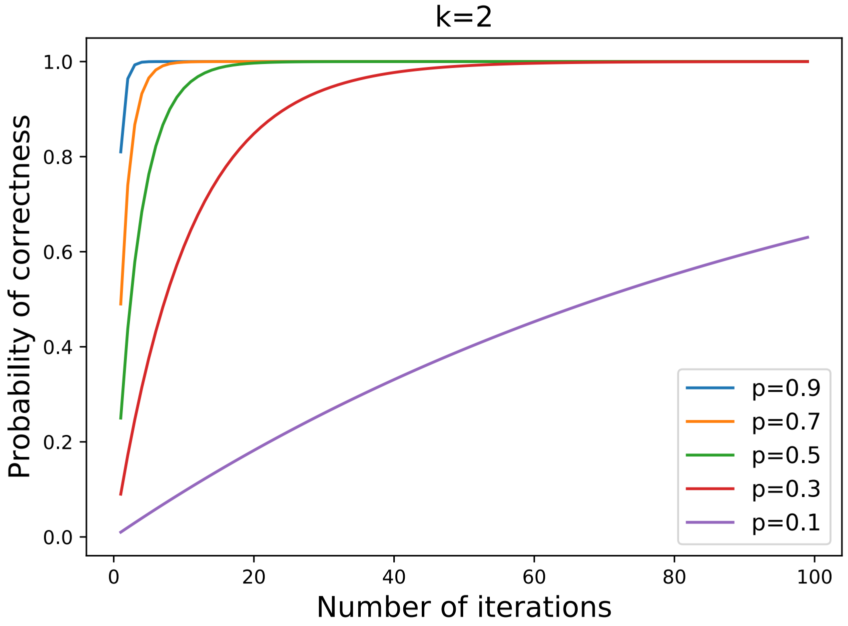
- Given
  - A dataset $D = \{ p_1, p_2, \ldots, p_N\}$
  - A set of parameters θ that need to be fitted
  - A cost function $C(p, \theta)$
  - k
  - $\theta^* = \min_{\theta} \sum_i C(p_i, \theta)$?
  - Problem: outliers

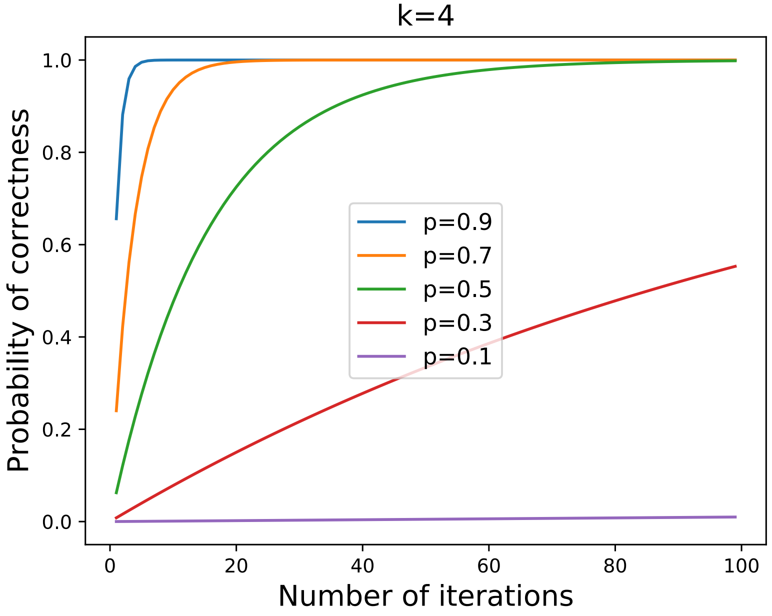
RANSAC - Algorithm

- Given: $D = \{ p_1, p_2, \ldots, p_N\}, C(\theta, p)$,k
- $\theta_{best} \leftarrow \phi, D_{inlier} \leftarrow \phi$
- For i = 1, …, S
  - Sample k points
  - Minimize C for these k points to get θ_{hyp}
  - Compute the set of inliers: $D_{hyp} = \{p \in D: C\left(\theta_{hyp}, p\right) < \delta\}$
  - If size of $D_{hyp}$is more than size of D_{inlier}
    - $\theta_{best} \leftarrow \theta_{hyp}$
    - $D_{inlier} \leftarrow D_{hyp}$
- Minimize θ over D_{inlier}

RANSAC: how many iterations do we need?

- p = inlier fraction
- k = minimum number of data points
- S = iter
- $P = \left(1 - \left(1 - p^k\right)^S\right)$

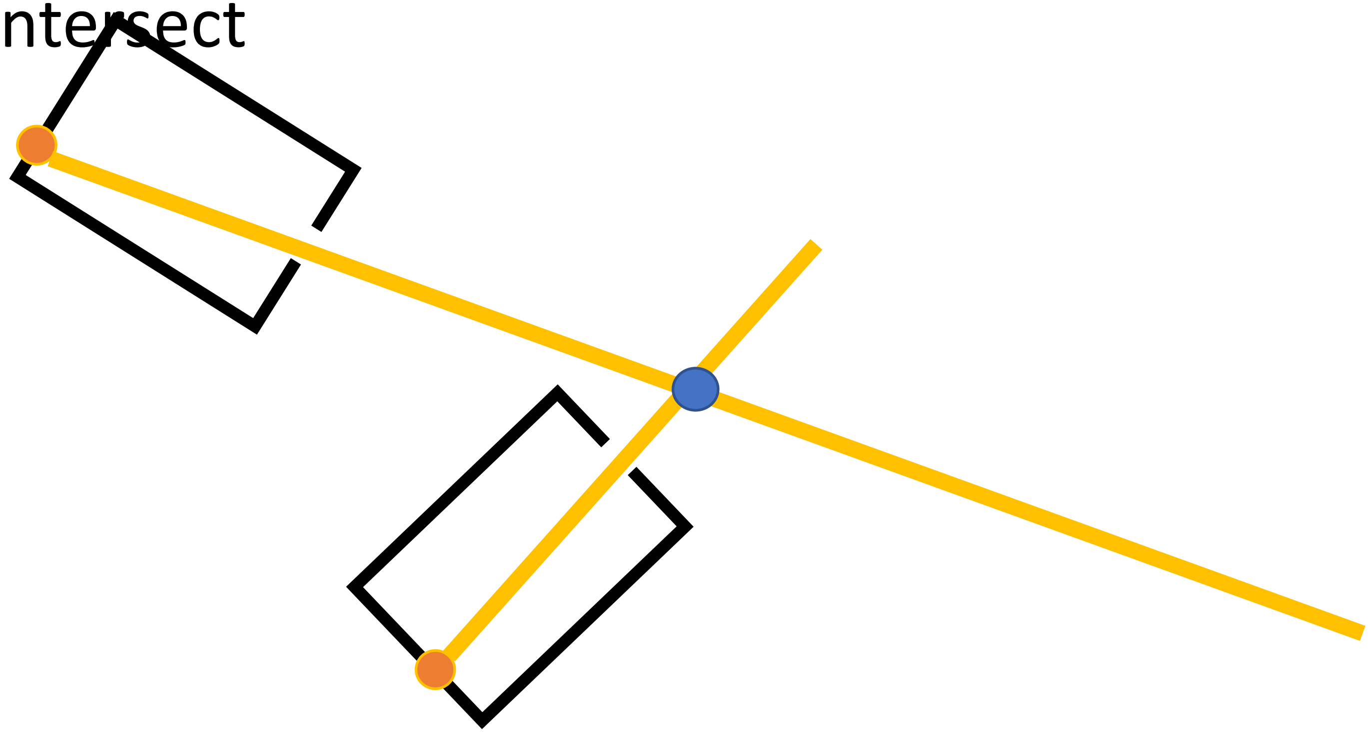




Binocular Stereo

Binocular stereo

- General case: cameras can be arbitrary locations and orientations
- If we know where cameras are, we can shoot rays from corresponding pixels and intersect



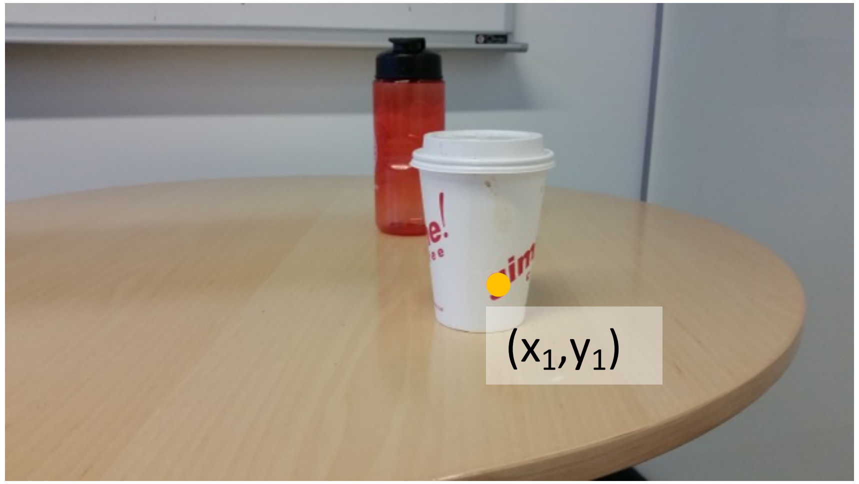
Binocular stereo : Triangulation

- Suppose we have two cameras
  - Calibrated: parameters known
- And a pair of corresponding pixels
- Find 3D location of point!

Triangulation

- Suppose we have two cameras
  - Calibrated: parameters known
- And a pair of corresponding pixels
- Find 3D location of point!

$P^{(1)}$



$P^{(2)}$

Triangulation

$$
\begin{tikzcd}
\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} & \vec{\mathbf{x}}^{(1)}_{img} \equiv P^{(1)} \vec{\mathbf{x}}_w \arrow[l] \arrow[dr] & \\
& & \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \\
\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} & \vec{\mathbf{x}}^{(2)}_{img} \equiv P^{(2)} \vec{\mathbf{x}}_w \arrow[l] \arrow[ur] &
\end{tikzcd}
$$

Triangulation

$$\vec{\mathbf{x}}_{img}^{(1)} \equiv P^{(1)} \vec{\mathbf{x}}_w$$

$$\lambda x_1 = P_{11}^{(1)} X + P_{12}^{(1)} Y + P_{13}^{(1)} Z + P_{14}^{(1)}$$

$$\lambda y_1 = P_{21}^{(1)} X + P_{22}^{(1)} Y + P_{23}^{(1)} Z + P_{24}^{(1)}$$

$$\lambda = P_{31}^{(1)} X + P_{32}^{(1)} Y + P_{33}^{(1)} Z + P_{34}^{(1)}$$

$$(P_{31}^{(1)} X + P_{32}^{(1)} Y + P_{33}^{(1)} Z + P_{34}^{(1)}) x_1 = P_{11}^{(1)} X + P_{12}^{(1)} Y + P_{13}^{(1)} Z + P_{14}^{(1)}$$

$$X(P_{31}^{(1)} x_1 - P_{11}^{(1)}) + Y(P_{32}^{(1)} x_1 - P_{12}^{(1)}) + Z(P_{33}^{(1)} x_1 - P_{13}^{(1)}) + (P_{34}^{(1)} x_1 - P_{14}^{(1)}) = 0$$

Triangulation

$$\vec{\mathbf{x}}_{img}^{(1)} \equiv P^{(1)}\vec{\mathbf{x}}_w$$

$$X(P_{31}^{(1)}x_1 - P_{11}^{(1)}) + Y(P_{32}^{(1)}x_1 - P_{12}^{(1)}) + Z(P_{33}^{(1)}x_1 - P_{13}^{(1)}) + (P_{34}^{(1)}x_1 - P_{14}^{(1)}) = 0$$

$$X(P_{31}^{(1)}y_1 - P_{21}^{(1)}) + Y(P_{32}^{(1)}y_1 - P_{22}^{(1)}) + Z(P_{33}^{(1)}y_1 - P_{23}^{(1)}) + (P_{34}^{(1)}y_1 - P_{24}^{(1)}) = 0$$

- 1 image gives 2 equations
- Need 2 images!
- Solve linear equations to get 3D point location

Linear vs non-linear optimization

$$\lambda x_1 = P_{11}^{(1)}X + P_{12}^{(1)}Y + P_{13}^{(1)}Z + P_{14}^{(1)}$$

$$\lambda y_1 = P_{21}^{(1)}X + P_{22}^{(1)}Y + P_{23}^{(1)}Z + P_{24}^{(1)}$$

$$\lambda = P_{31}^{(1)}X + P_{32}^{(1)}Y + P_{33}^{(1)}Z + P_{34}^{(1)}$$

$$x_1 = \frac{P_{11}^{(1)}X + P_{12}^{(1)}Y + P_{13}^{(1)}Z + P_{14}^{(1)}}{P_{31}^{(1)}X + P_{32}^{(1)}Y + P_{33}^{(1)}Z + P_{34}^{(1)}}$$

$$y_1 = \frac{P_{21}^{(1)}X + P_{22}^{(1)}Y + P_{23}^{(1)}Z + P_{24}^{(1)}}{P_{31}^{(1)}X + P_{32}^{(1)}Y + P_{33}^{(1)}Z + P_{34}^{(1)}}$$

Linear vs non-linear optimization

$$x_1 = \frac{P^{(1)}_{11}X + P^{(1)}_{12}Y + P^{(1)}_{13}Z + P^{(1)}_{14}}{P^{(1)}_{31}X + P^{(1)}_{32}Y + P^{(1)}_{33}Z + P^{(1)}_{34}}$$

$$y_1 = \frac{P^{(1)}_{21}X + P^{(1)}_{22}Y + P^{(1)}_{23}Z + P^{(1)}_{24}}{P^{(1)}_{31}X + P^{(1)}_{32}Y + P^{(1)}_{33}Z + P^{(1)}_{34}}$$

$$\boxed{\begin{aligned}&(x_1 - \frac{P^{(1)}_{11}X + P^{(1)}_{12}Y + P^{(1)}_{13}Z + P^{(1)}_{14}}{P^{(1)}_{31}X + P^{(1)}_{32}Y + P^{(1)}_{33}Z + P^{(1)}_{34}})^2 \\ +&(y_1 - \frac{P^{(1)}_{21}X + P^{(1)}_{22}Y + P^{(1)}_{23}Z + P^{(1)}_{24}}{P^{(1)}_{31}X + P^{(1)}_{32}Y + P^{(1)}_{33}Z + P^{(1)}_{34}})^2\end{aligned}}$$

Reprojection error

Linear vs non-linear optimization

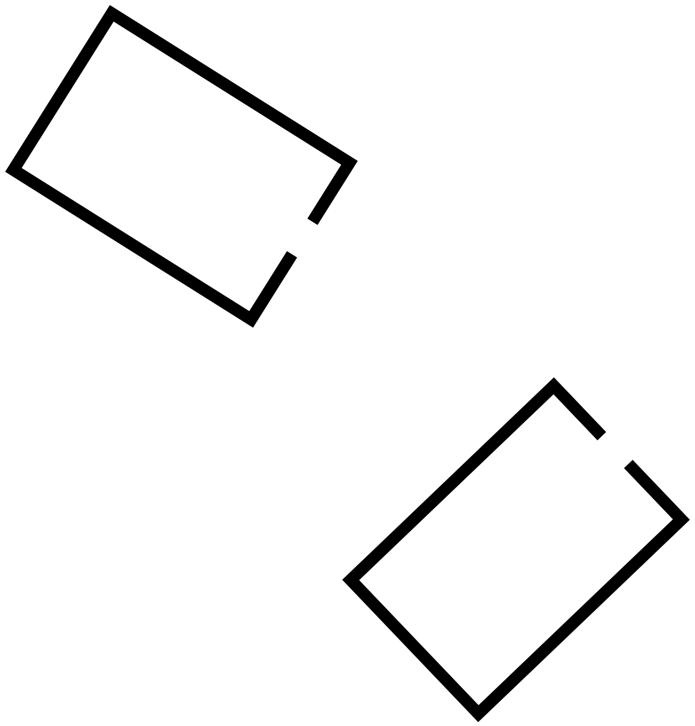
$$\boxed{\begin{aligned}&(x_1 - \frac{P^{(1)}_{11}X + P^{(1)}_{12}Y + P^{(1)}_{13}Z + P^{(1)}_{14}}{P^{(1)}_{31}X + P^{(1)}_{32}Y + P^{(1)}_{33}Z + P^{(1)}_{34}})^2 \\ +&(y_1 - \frac{P^{(1)}_{21}X + P^{(1)}_{22}Y + P^{(1)}_{23}Z + P^{(1)}_{24}}{P^{(1)}_{31}X + P^{(1)}_{32}Y + P^{(1)}_{33}Z + P^{(1)}_{34}})^2\end{aligned}}$$

Reprojection error

- Reprojection error is the squared error between the true image coordinates of a point and the projected coordinates of hypothesized 3D point
- Actual error we care about
- Minimize total sum of reprojection error across all images
- *Non-linear optimization*

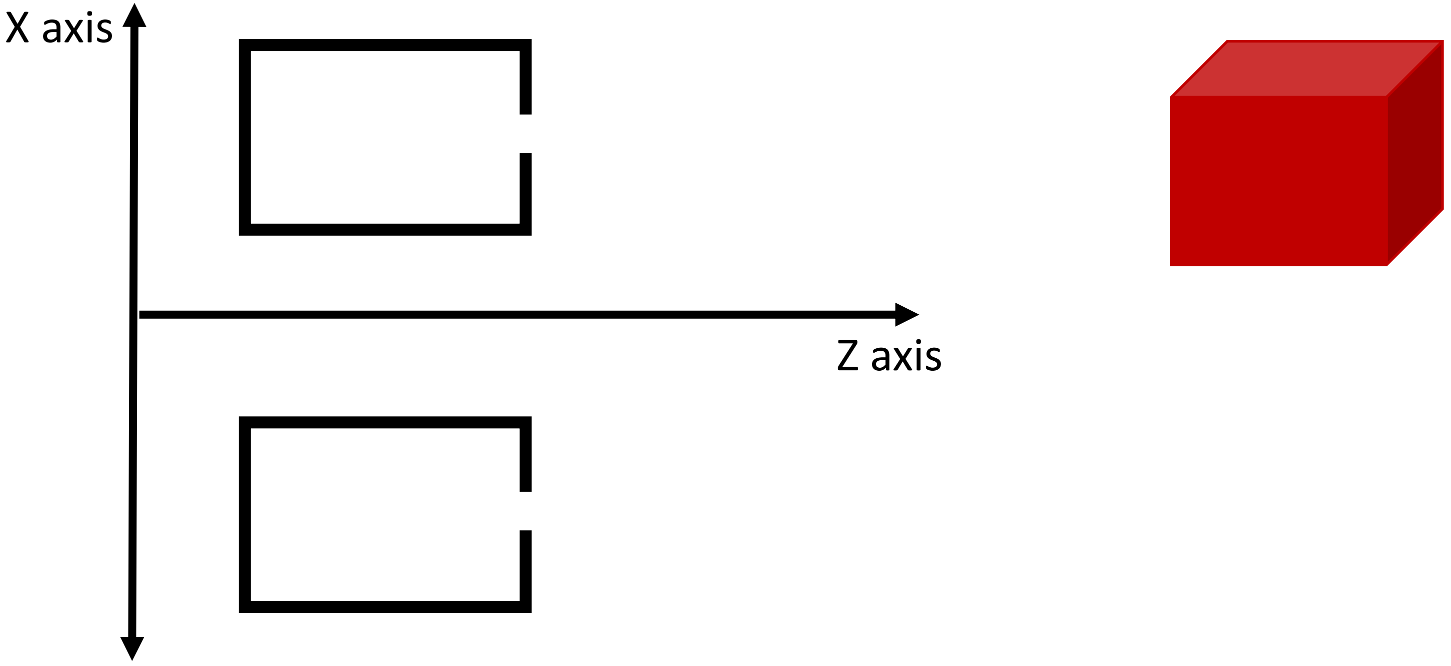
Binocular stereo

- General case: cameras can be arbitrary locations and orientations



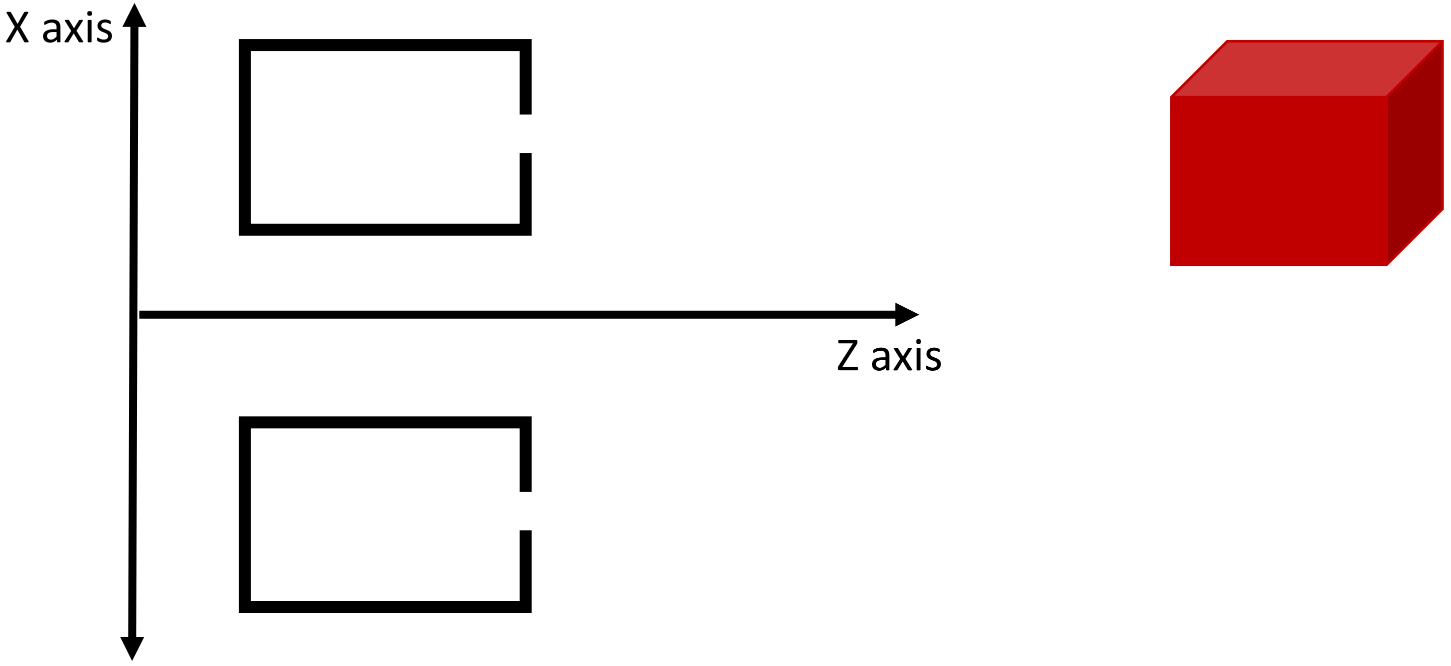
Binocular stereo

- Special case: cameras are parallel to each other and translated along X axis

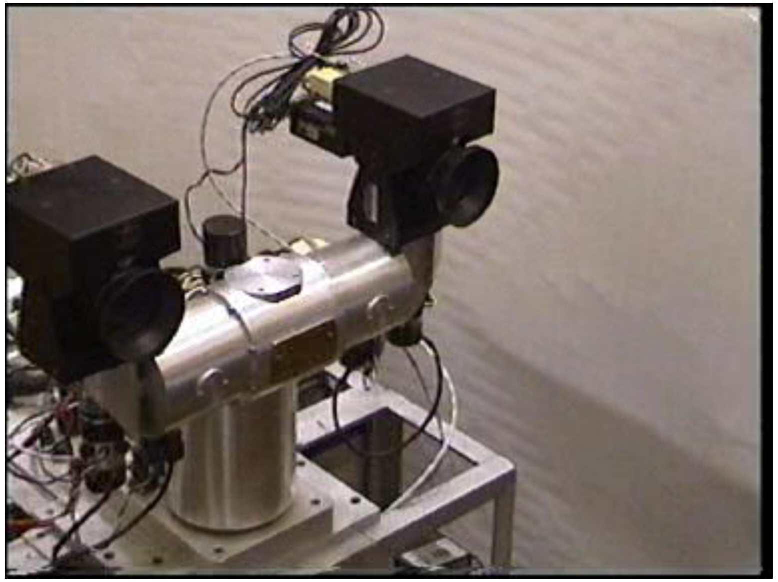


Stereo with *rectified cameras*

- Special case: cameras are parallel to each other and translated along X axis

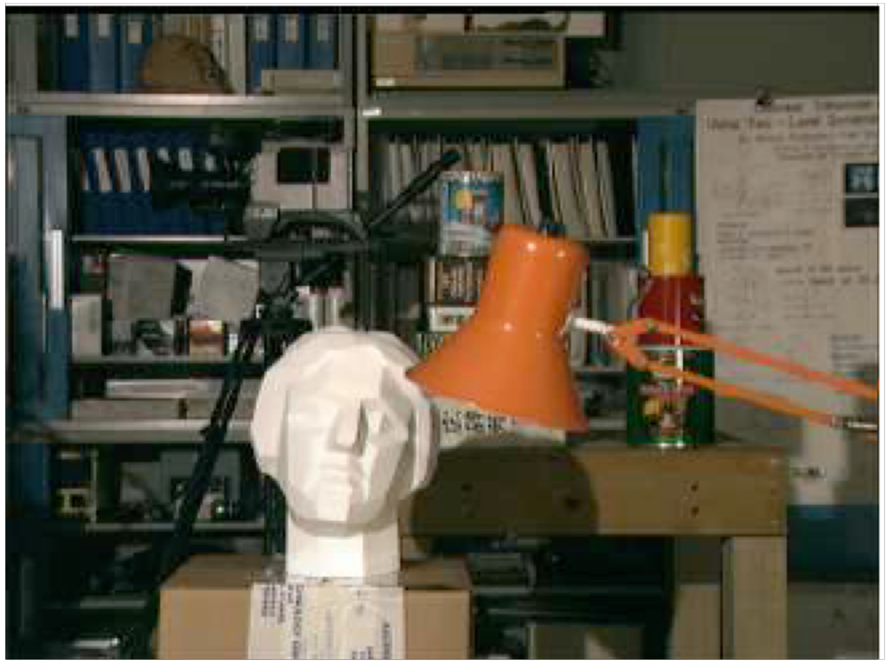


Stereo head



Kinect / depth cameras

Stereo with “rectified cameras”



Perspective projection in rectified cameras

- Without loss of generality, assume origin is at pinhole of 1st camera

$$\vec{\mathbf{x}}_{img}^{(1)} \equiv \begin{bmatrix} I & \mathbf{0} \end{bmatrix} \vec{\mathbf{x}}_w$$

$$\vec{\mathbf{x}}_{img}^{(2)} \equiv \begin{bmatrix} I & \mathbf{t} \end{bmatrix} \vec{\mathbf{x}}_w$$

$$\mathbf{t} = \begin{bmatrix} t_x \\ 0 \\ 0 \end{bmatrix}$$

Perspective projection in rectified cameras

- Without loss of generality, assume origin is at pinhole of 1st camera

$$\vec{\mathbf{x}}_{img}^{(1)} \equiv \begin{bmatrix} I & \mathbf{0} \end{bmatrix} \vec{\mathbf{x}}_w$$

$$\vec{\mathbf{x}}_{img}^{(2)} \equiv \begin{bmatrix} I & \mathbf{t} \end{bmatrix} \vec{\mathbf{x}}_w$$

$$\mathbf{t} = \begin{bmatrix} t_x \\ 0 \\ 0 \end{bmatrix} \vec{\mathbf{x}}_w = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{x}_w \\ 1 \end{bmatrix}$$

Perspective projection in rectified cameras

- Without loss of generality, assume origin is at pinhole of 1st camera

$$\vec{\mathbf{x}}_{img}^{(1)} \equiv \begin{bmatrix} I & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x}_w \\ 1 \end{bmatrix} = \mathbf{x}_w = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

$$\vec{\mathbf{x}}_{img}^{(2)} \equiv \begin{bmatrix} I & \mathbf{t} \end{bmatrix} \begin{bmatrix} \mathbf{x}_w \\ 1 \end{bmatrix} = \mathbf{x}_w + \mathbf{t} = \begin{bmatrix} X + t_x \\ Y \\ Z \end{bmatrix}$$

Perspective projection in rectified cameras

- Without loss of generality, assume origin is at pinhole of 1st camera

$$\vec{\mathbf{x}}_{img}^{(1)} \equiv \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

$$\vec{\mathbf{x}}_{img}^{(2)} \equiv \begin{bmatrix} X + t_x \\ Y \\ Z \end{bmatrix}$$

Perspective projection in rectified cameras

- Without loss of generality, assume origin is at pinhole of 1st camera

$$\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \equiv \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

$$\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} \equiv \begin{bmatrix} X + t_x \\ Y \\ Z \end{bmatrix}$$

Perspective projection in rectified cameras

- Without loss of generality, assume origin is at pinhole of 1st camera

$$\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \equiv \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

$$\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} \equiv \begin{bmatrix} X + t_x \\ Y \\ Z \end{bmatrix}$$

Perspective projection in rectified cameras

- Without loss of generality, assume origin is at pinhole of 1st camera

$$\begin{bmatrix}\lambda x_1 \\ \lambda y_1 \\ \lambda\end{bmatrix} = \begin{bmatrix}X \\ Y \\ Z\end{bmatrix}$$

$$\begin{bmatrix}\lambda x_2 \\ \lambda y_2 \\ \lambda\end{bmatrix} = \begin{bmatrix}X + t_x \\ Y \\ Z\end{bmatrix}$$

Perspective projection in rectified cameras

- Without loss of generality, assume origin is at pinhole of 1st camera

X coordinate differs by t_x/Z

$$x_1 = \frac{X}{Z} \qquad x_2 = \frac{X + t_x}{Z}$$

$$y_1 = \frac{Y}{Z} \qquad y_2 = \frac{Y}{Z}$$

Y coordinate is the same!

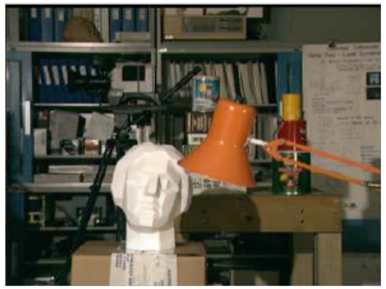
Perspective projection in rectified cameras

- X coordinate differs by t_x/Z
- That is, difference in X coordinate is *inversely proportional to depth*
- Difference in X coordinate is called *disparity*
- Translation between cameras (tx) is called *baseline*

- *disparity = baseline / depth*

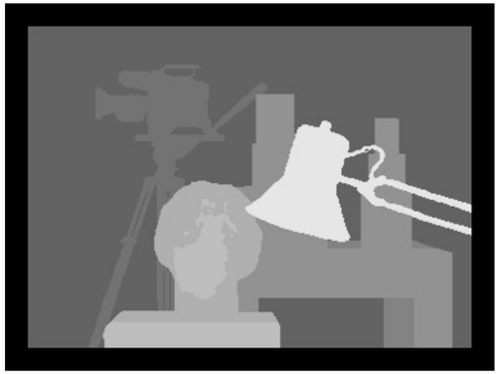
The disparity image

- For pixel (x,y) in one image, only need to know disparity to get correspondence
- Create an image with color at (x,y) = disparity



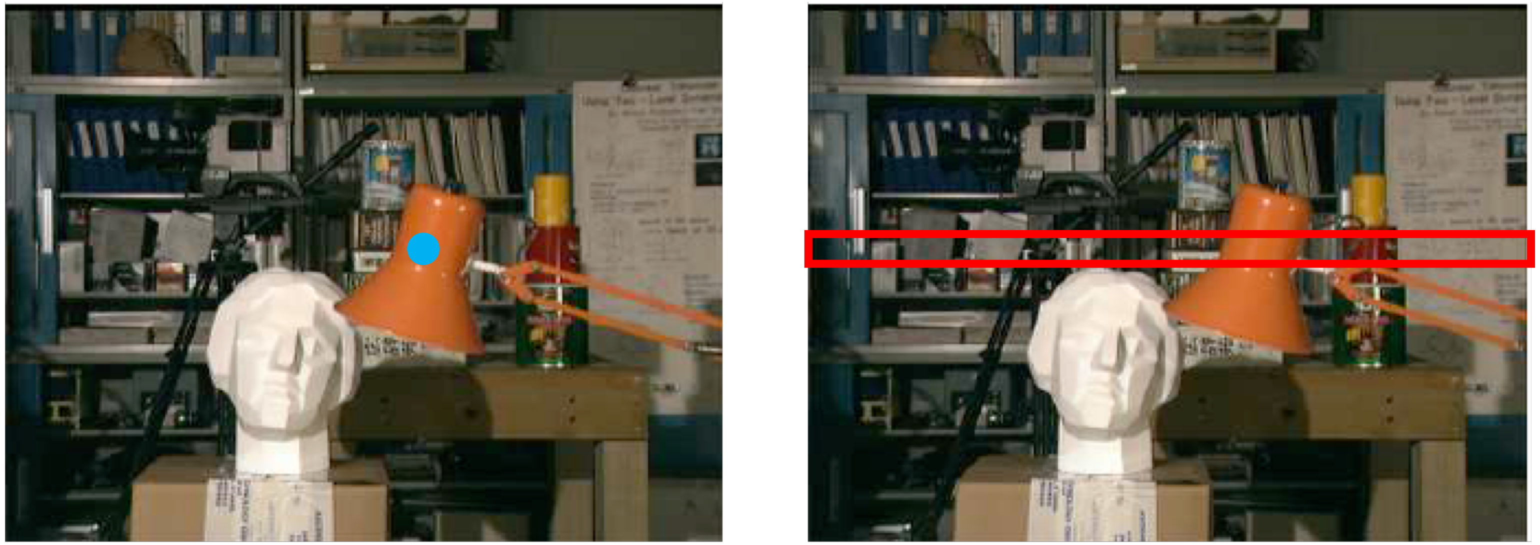
right image

left image



disparity

Perspective projection in rectified cameras



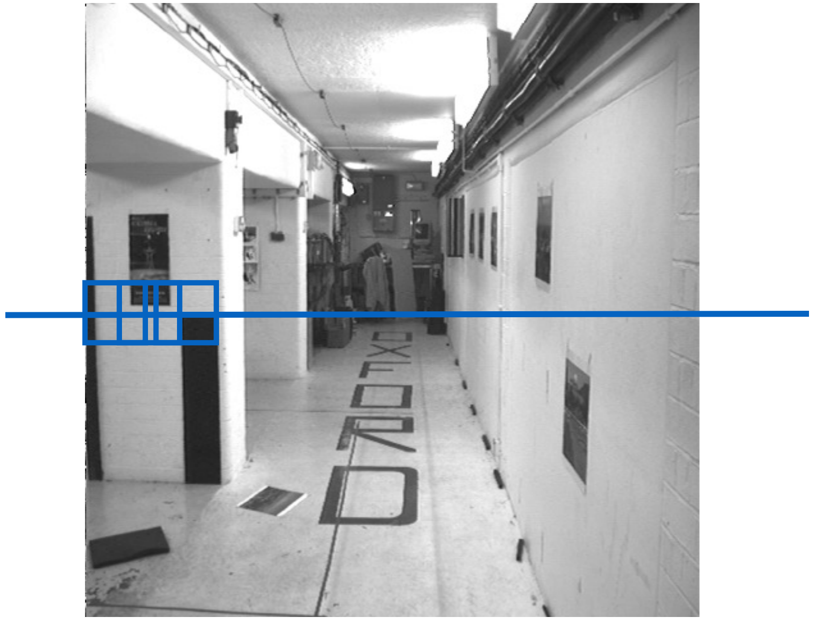
- For rectified cameras, correspondence problem is easier
- Only requires searching along a particular *row.*

NCC - Normalized Cross Correlation

- Lighting and color change pixel intensities
- Example: increase brightness / contrast
- $I' = \alpha I + \beta$
- Subtract patch mean: invariance to β
- Divide by norm of vector: invariance to α
- $x' = x - < x >$
- $x'' = \frac{x'}{||x'||}$
- *similarity* = $x'' \cdot y''$

Why not SIFT?

Cross-correlation of neighborhood



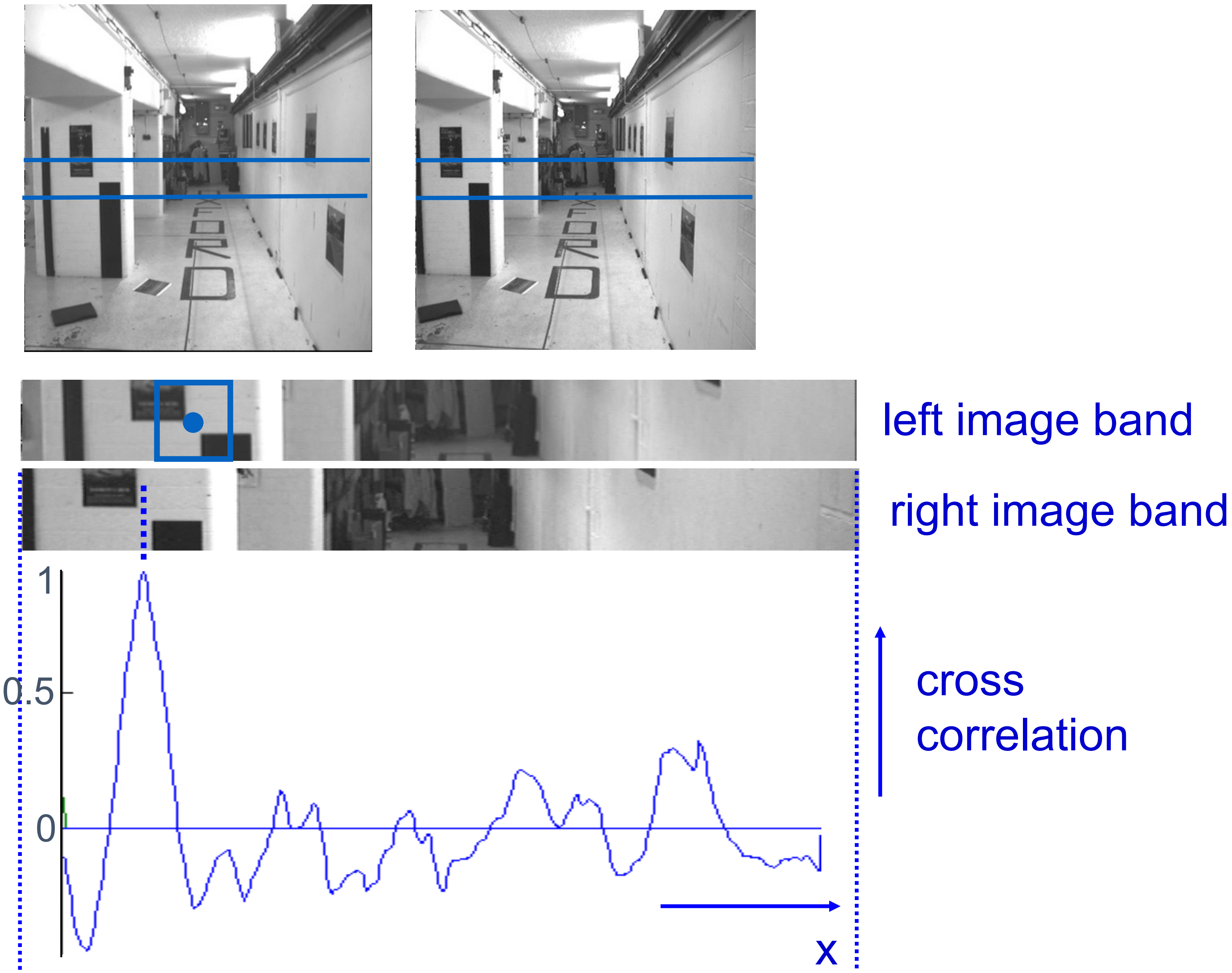
regions $\mathtt{A}, \mathtt{B}$, write as vectors $\mathbf{a}, \mathbf{b}$

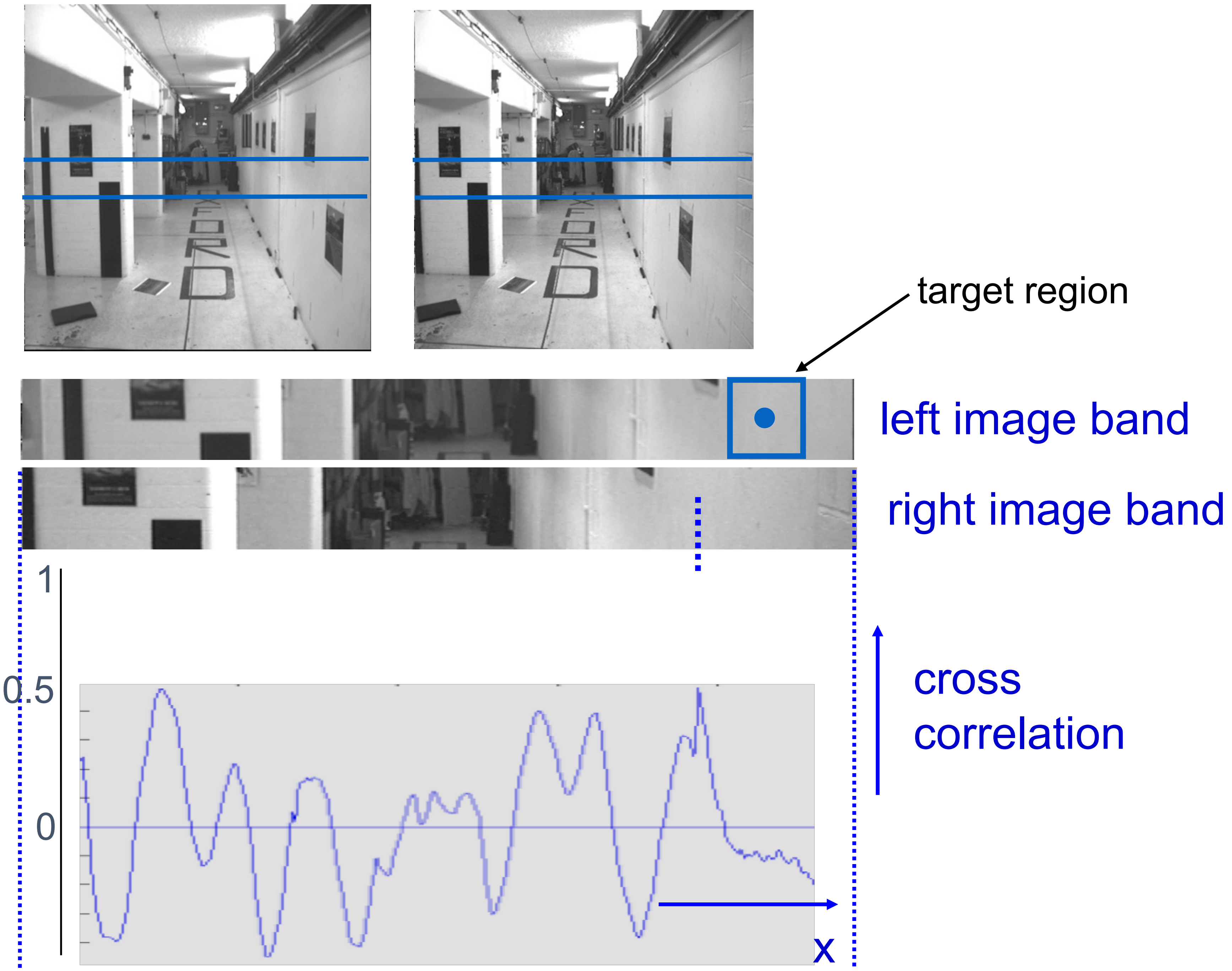
translate so that mean is zero

$\mathtt{a} \rightarrow \mathtt{a} - \langle \mathbf{a} \rangle, \quad \mathtt{b} \rightarrow \mathtt{b} - \langle \mathbf{b} \rangle$

$$\text{cross correlation} = \frac{\mathbf{a}.\mathbf{b}}{|\mathbf{a}||\mathbf{b}|}$$

Invariant to $I \rightarrow \alpha I + \beta$


left image band
right image band
1
0.5
0
cross
correlation
x


target region
left image band
right image band
1
0.5
0
cross
correlation
x

The NCC cost volume

- Consider M x N image
- Suppose there are D possible disparities.
- For every pixel, D possible scores
- Can be written as an M x N x D array
- To get disparity, take max along 3rd axis

Computing the NCC volume

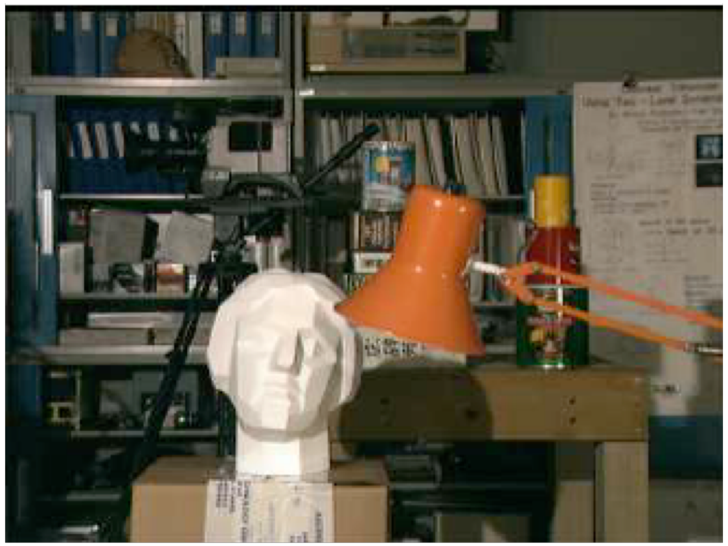
1. For every pixel (x, y)
    1. For every disparity d
        1. Get normalized patch from image 1 at (x, y)
        2. Get normalized patch from image 2 at (x + d, y)
        3. Compute NCC

Computing the NCC volume

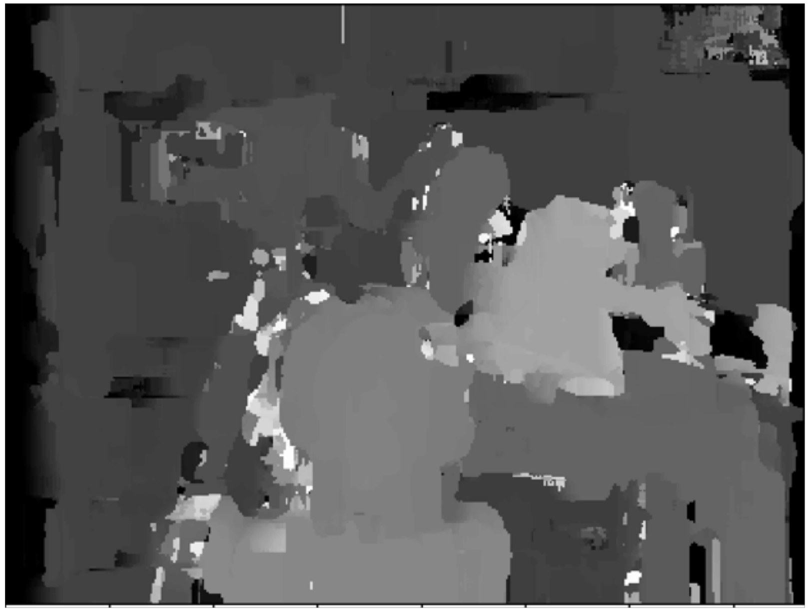
1. For every disparity d
    1. For every pixel (x, y)
        1. Get normalized patch from image 1 at (x, y)
        2. Get normalized patch from image 2 at (x + d, y)
        3. Compute NCC

Assume all pixels lie at same disparity d (i.e., lie on same plane) and compute cost for each

Plane sweep stereo



NCC volume



Disparity