Lecture 19

Homographies

(Slides courtesy Mike Langer)

Overview of Today

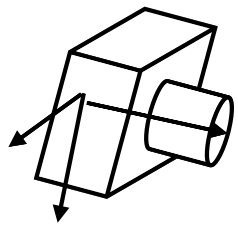
Examples of situations that involve homographies

1. a camera looking at a parameterized scene plane
2. two different cameras looking at a scene plane
3. rotating one camera
4. two cameras looking at a non-planar 3D scene

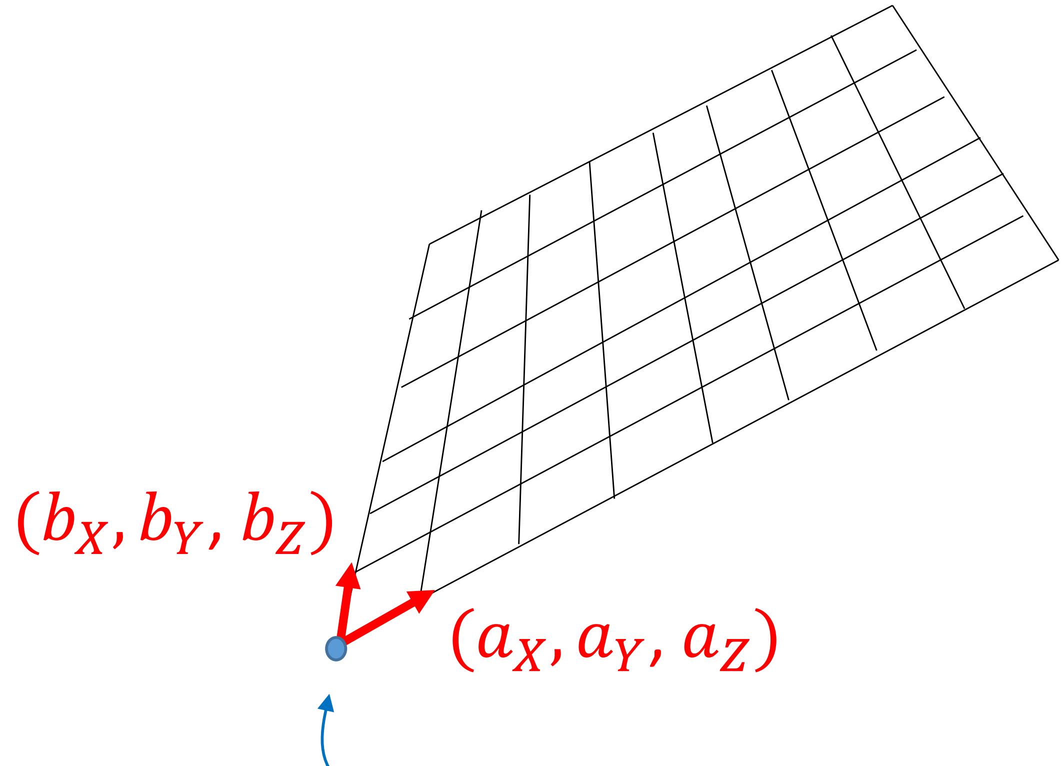
Given two images, how to fit a homography between them?

How to combine the two images?

Recall Case 1 (last lecture)



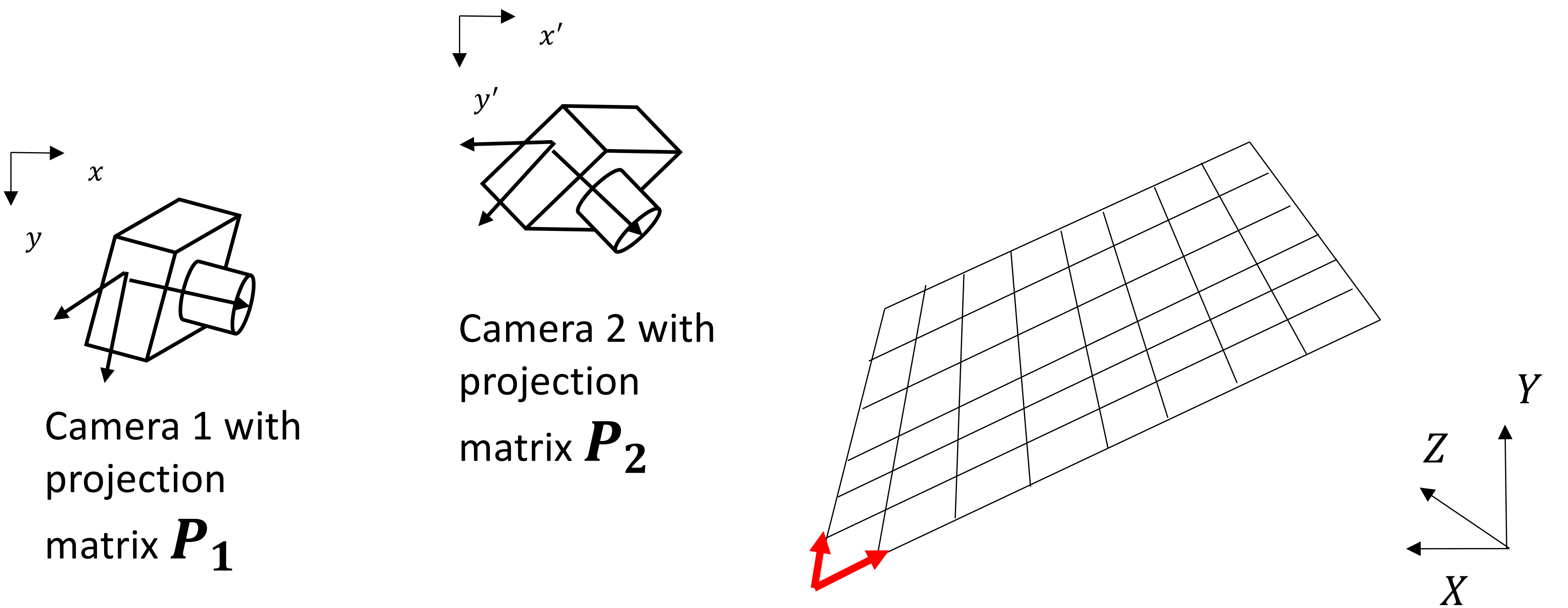
Camera with projection matrix $\boldsymbol{P}$



3D origin (X_0, Y_0, Z_0) corresponds to (s, t)= (0,0).

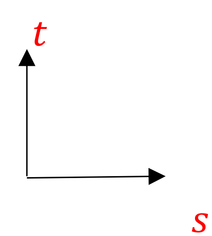
$$\begin{bmatrix} wx \\ wy \\ w \end{bmatrix} = \underbrace{\boldsymbol{P} \begin{bmatrix} a_X & b_X & X_0 \\ a_Y & b_Y & Y_0 \\ a_Z & b_Z & Z_0 \\ 0 & 0 & 1 \end{bmatrix}}_{\boldsymbol{H}_{3\times 3}} \begin{bmatrix} s \\ t \\ 1 \end{bmatrix}$$

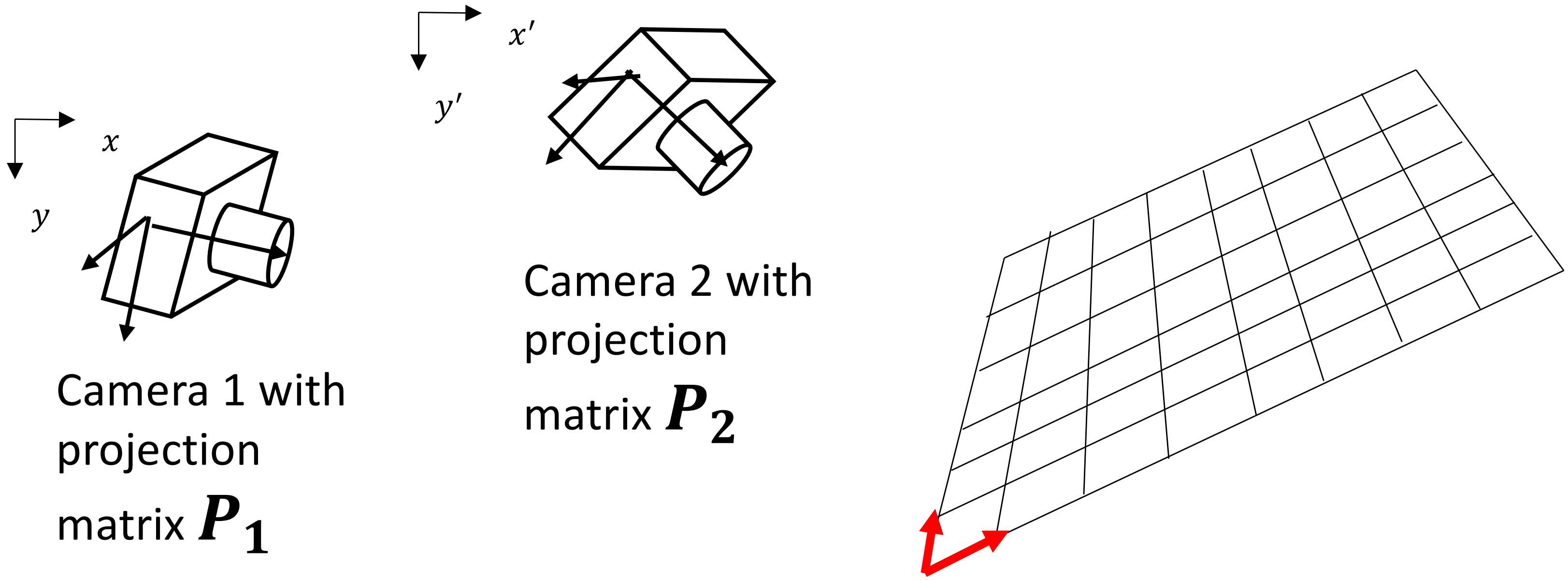
Case 2:
two cameras viewing the same 3D plane



world coordinates

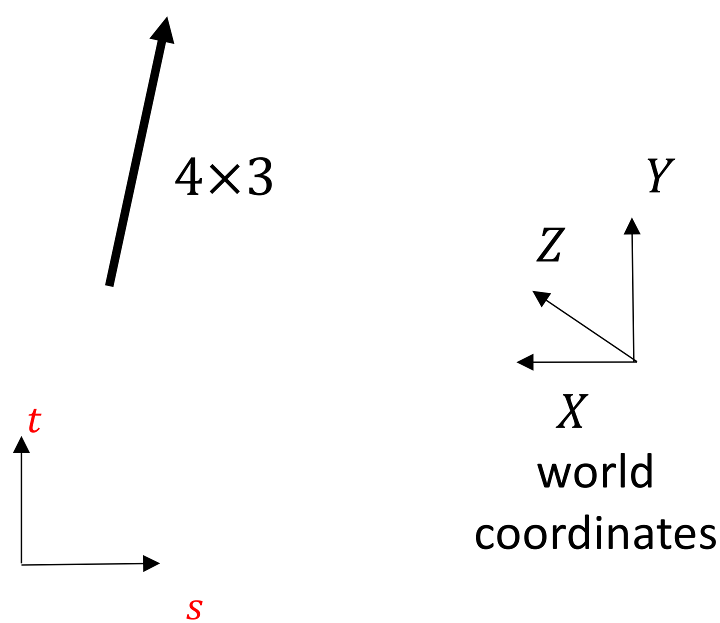
The cameras may have different intrinsic and extrinsic parameters.

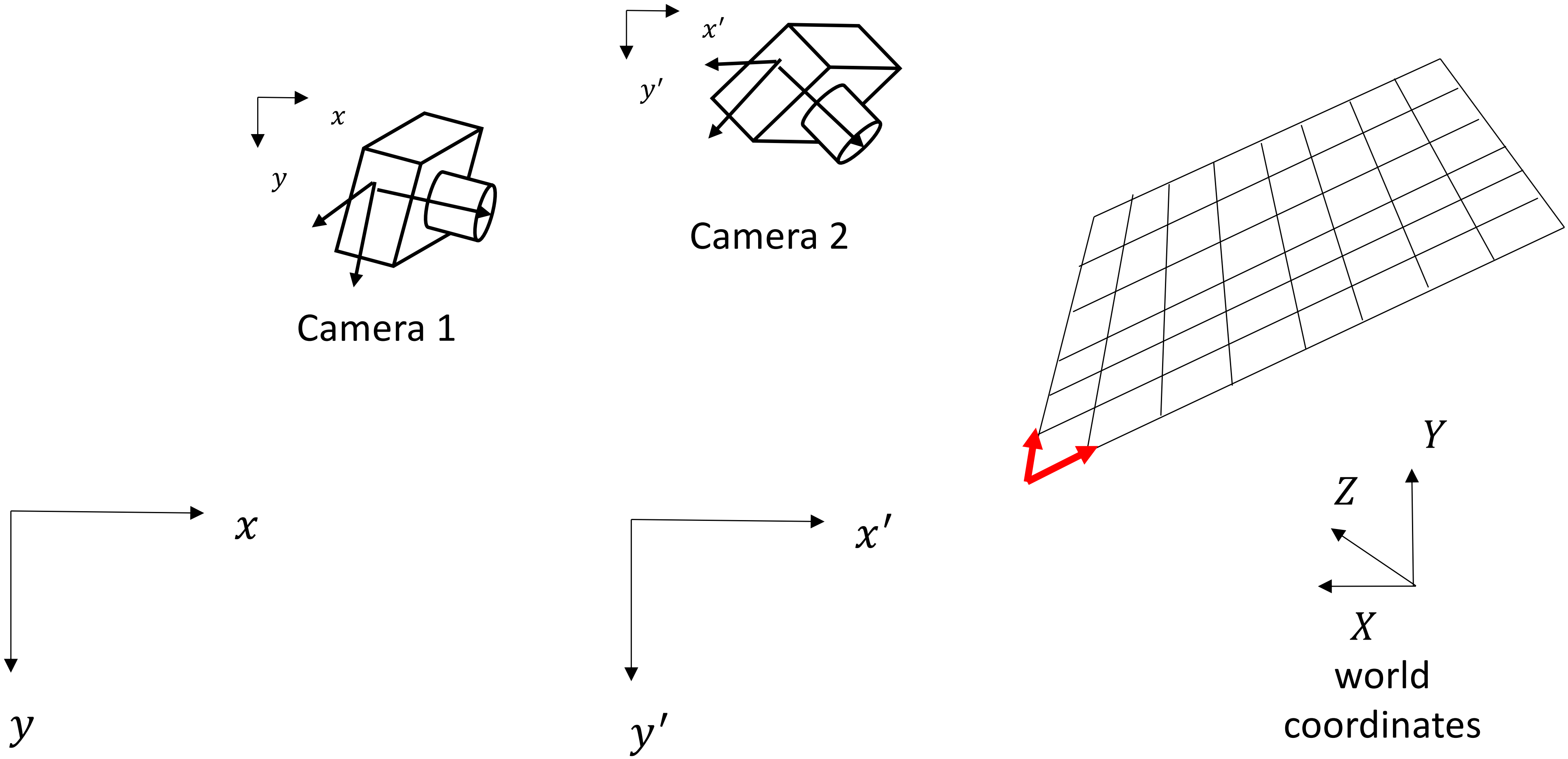


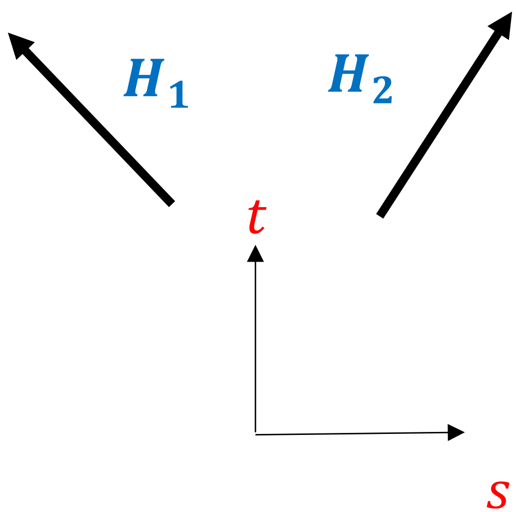


Each camera defines a homography that maps plane coordinates (s, t) to camera pixel coordinates.

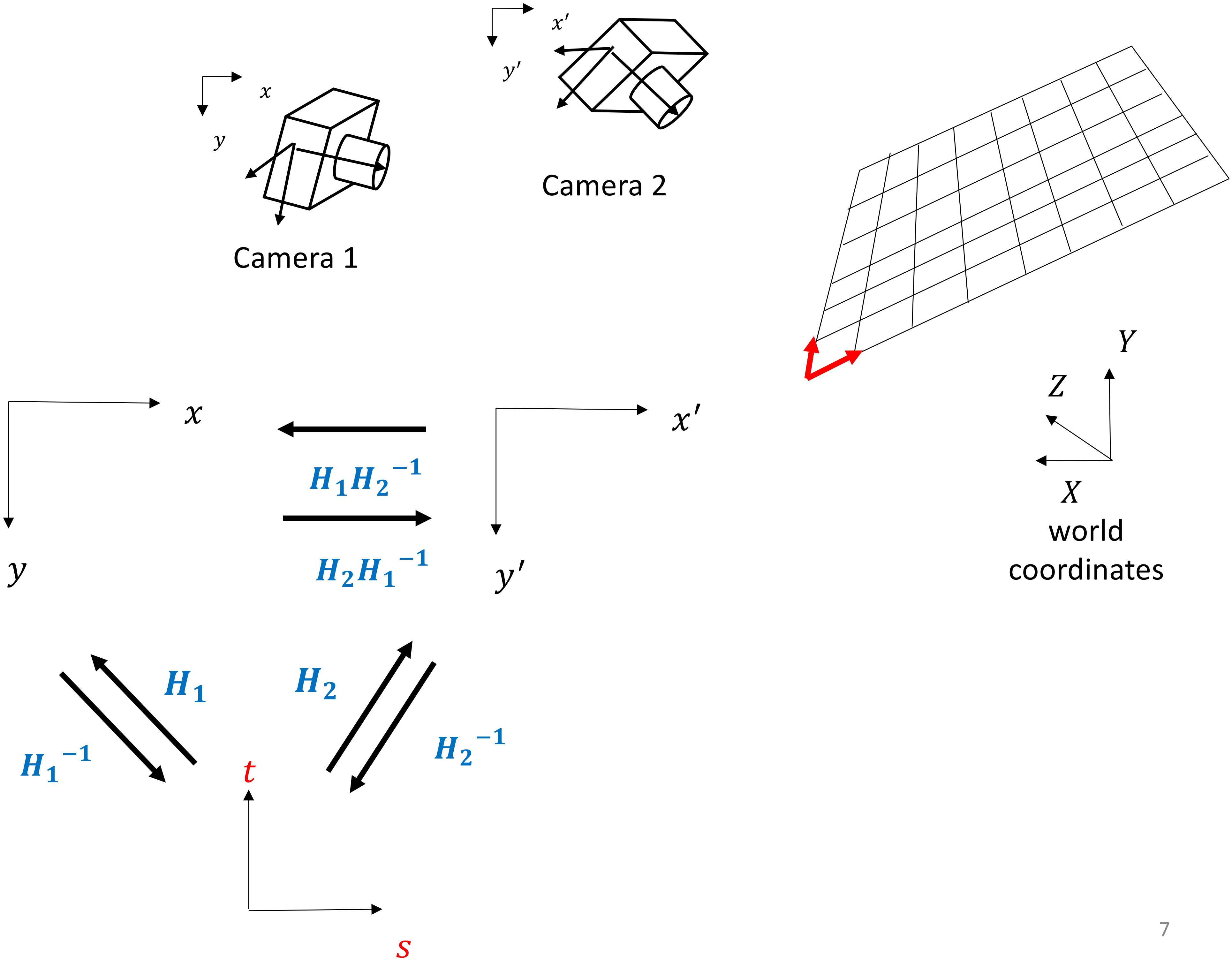
These homographies $\boldsymbol{H_1}$ and $\boldsymbol{H_2}$ map to (x, y) and (x', y'), respectively.





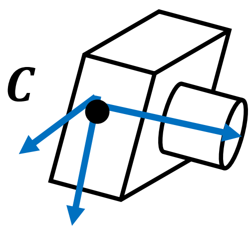


What is the homography between the images seen by the two cameras?


Camera 1
Camera 2
x
y
x'
y'
$H_1 H_2^{-1}$
$H_2 H_1^{-1}$
H_1
H_1^{-1}
H_2
H_2^{-1}
t
s
Y
Z
X
world coordinates

Case 3 : camera rotation
(Application : image stitching / panoramas)

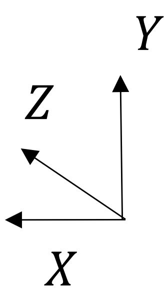
Consider *one* camera at location $\boldsymbol{C}$, and at two orientations $\boldsymbol{R_1}$ and $\boldsymbol{R_2}$.



Camera with projection matrix $\mathbf{P} = \boldsymbol{K}\boldsymbol{R}[\boldsymbol{I} \mid -\boldsymbol{C}]$

(X_0, Y_0, Z_0)

$$\begin{bmatrix} wx \\ wy \\ w \end{bmatrix} = \boldsymbol{K}\ \boldsymbol{R_1}\ [\boldsymbol{I}\ \mid\ -\boldsymbol{C}] \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix}$$

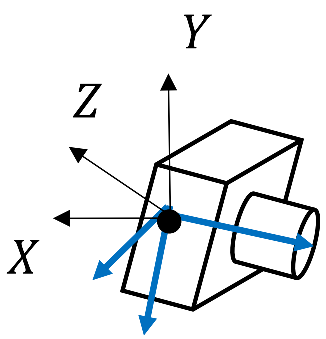


world coordinates

$$\begin{bmatrix} w'x' \\ w'y' \\ w' \end{bmatrix} = \boldsymbol{K}\ \boldsymbol{R_2}\ [\boldsymbol{I}\ \mid\ -\boldsymbol{C}] \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix}$$

Subtracting the camera position effectively places the origin of the world coordinate system to be at the camera position.

$$\boldsymbol{R_1}^{\boldsymbol{T}} \boldsymbol{K}^{-\mathbf{1}} \begin{bmatrix} wx \\ wy \\ w \end{bmatrix} = [\boldsymbol{I} \mid -\boldsymbol{C}] \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix}$$



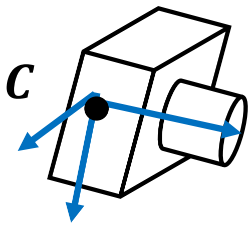
Camera with projection matrix $\mathbf{P} = \boldsymbol{K}\boldsymbol{R}[\boldsymbol{I} \mid -\boldsymbol{C}]$

$(X_0, Y_0, Z_0) - \boldsymbol{C}$

$$\boldsymbol{R_2}^{\boldsymbol{T}} \boldsymbol{K}^{-\mathbf{1}} \begin{bmatrix} w'x' \\ w'y' \\ w' \end{bmatrix} = [\boldsymbol{I} \mid -\boldsymbol{C}] \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix}$$

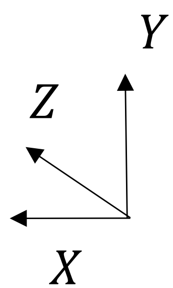
The two left sides must be equal and independent of $\boldsymbol{C}$ and (X_0, Y_0, Z_0). Thus,

$$\boldsymbol{R_2}^{\boldsymbol{T}} \boldsymbol{K}^{-\mathbf{1}} \begin{bmatrix} w'x' \\ w'y' \\ w' \end{bmatrix} = \boldsymbol{R_1}^{\boldsymbol{T}} \boldsymbol{K}^{-\mathbf{1}} \begin{bmatrix} wx \\ wy \\ w \end{bmatrix}$$



Camera with
projection matrix
$\mathbf{P} = \boldsymbol{K}\boldsymbol{R}[\boldsymbol{I} \mid -\boldsymbol{C}]$

(X_0, Y_0, Z_0)



world
coordinates

$$\boldsymbol{R_2}^{\boldsymbol{T}}\boldsymbol{K}^{-1}\begin{bmatrix} w'x' \\ w'y' \\ w' \end{bmatrix} = \boldsymbol{R_1}^{\boldsymbol{T}}\boldsymbol{K}^{-1}\begin{bmatrix} wx \\ wy \\ w \end{bmatrix}$$

implies

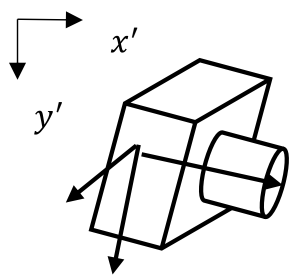
$$\begin{bmatrix} w'x' \\ w'y' \\ w' \end{bmatrix} = \underbrace{\boldsymbol{K}\,\boldsymbol{R_2}\,\boldsymbol{R_1}^{\boldsymbol{T}}\boldsymbol{K}^{-1}}_{\boldsymbol{H}_{3\times 3}}\begin{bmatrix} wx \\ wy \\ w \end{bmatrix}$$

Homography that maps each pixel position in image 1 to pixel positions in image 2.
(In image stitching applications, one attempts to estimate this homography directly by matching image points)

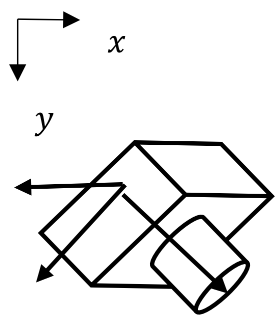
Later this lecture, we discuss the following problem.

Given two images taken by two orientations of the camera, how can we estimate a homography that maps positions in one image to positions in the other?

Case 4: Image “Rectification”



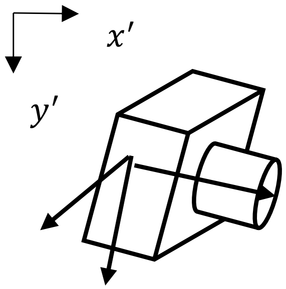
Camera 2 with projection matrix $\boldsymbol{P_2}$



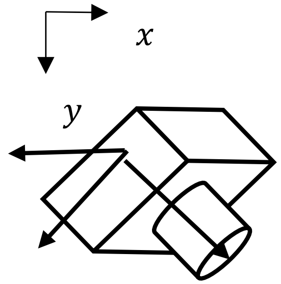
Camera 1 with projection matrix $\boldsymbol{P_1}$

Suppose we have non-planar 3D scene viewed by two cameras.

The two cameras may have different intrinsic and extrinsic parameters.



Camera 2 with projection matrix $\boldsymbol{P_2}$

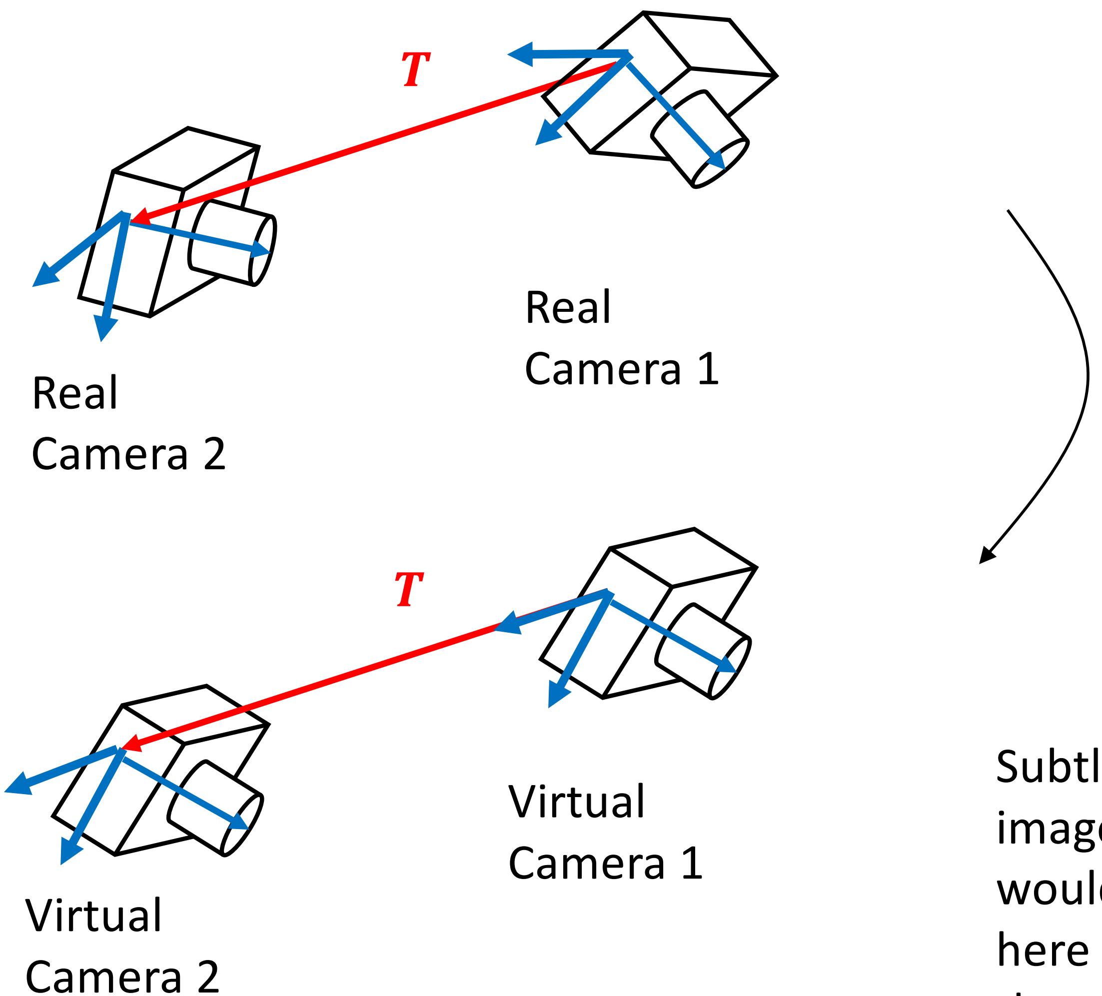


Camera 1 with projection matrix $\boldsymbol{P_1}$

Assume that we have performed camera calibration for both cameras so that we know $\boldsymbol{P_1}$ and $\boldsymbol{P_2}$ and we have decomposed them into $\boldsymbol{K_1}, \boldsymbol{R_1}, \boldsymbol{C_1}, \text{and } \boldsymbol{K_2}, \boldsymbol{R_2}, \boldsymbol{C_2}$ respectively.

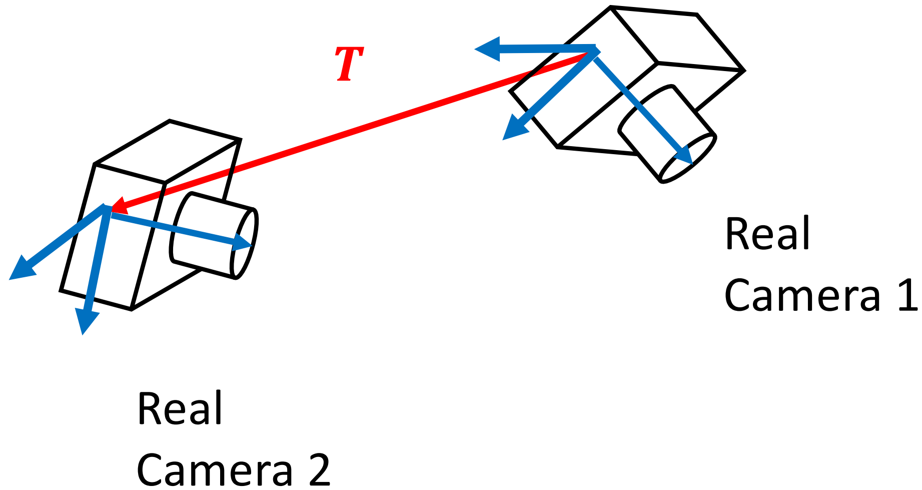
Let's now transform the two images by applying homographies to each, so that the images are "rectified" (to be described below.)

First, apply homographies that virtually rotate each real camera such that the XYZ axes of the two virtual cameras are parallel. In particular, the X axes are parallel to the vector T from camera 1 to camera 2.



How ?

Subtle point: The resulting deformed images are different from what we would get with a real rotation, since here we aren't changing the part of the scene that is visible.

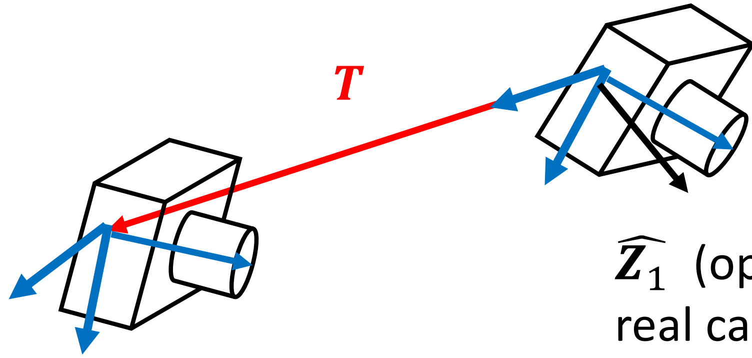


How? Define rotation $\boldsymbol{R}_{rect}$ which virtually rotates camera 1 so that its X axis becomes parallel to the $\boldsymbol{T}$ vector:

$$\begin{bmatrix} \|\boldsymbol{T}\| \\ 0 \\ 0 \end{bmatrix} = \boldsymbol{R}_{rect}\,\boldsymbol{T}.$$

For this to be a rotation matrix, the first row must be of unit length.

$$\boldsymbol{R}_{rect} \equiv \begin{bmatrix} \longleftarrow & \boldsymbol{T}/\|\boldsymbol{T}\| & \longrightarrow \\ \longleftarrow & & \longrightarrow \\ \longleftarrow & & \longrightarrow \end{bmatrix}_{3\times 3}$$



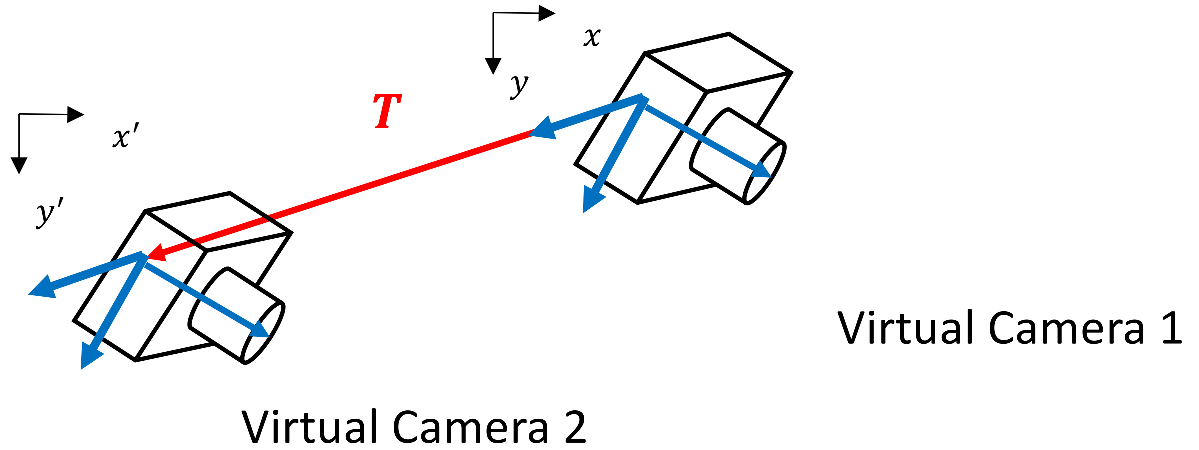
Define the other two rows of $\boldsymbol{R}_{rect}$ such that its three rows are orthonormal. Here $\widehat{\boldsymbol{Z}_1}$ is the optical axis of *real* camera 1

(So $\widehat{\boldsymbol{Z}_1}$ is not parallel to $\boldsymbol{T}$ and so $\widehat{\boldsymbol{Z}_1} \times \boldsymbol{T} \neq \boldsymbol{0}$.

$$\boldsymbol{R}_{rect} \equiv \begin{bmatrix} unit(\boldsymbol{T}) \\ unit(\widehat{\boldsymbol{Z}_1} \times \boldsymbol{T}) \\ unit(\boldsymbol{T} \times (\widehat{\boldsymbol{Z}_1} \times \boldsymbol{T})) \end{bmatrix}$$

These are virtual camera 1's axes, expressed in world coordinates.

where $unit(\,)$ is an operator that normalizes a vector.



To rotate real camera 2's coordinate systems to be parallel to virtual camera 1's coordinate system, apply $\boldsymbol{R}_{rect}\, \boldsymbol{R}_1 \boldsymbol{R}_2^T$.

(It rotates to world coordinates, then to real camera 1 coordinates, then to virtual camera 1 coordinates.)

What else?

- The images are in pixel coordinates. In order to apply a rotation, we must *first* (inverse) map the pixels to the image projection planes.
- The cameras have different internal parameters $\boldsymbol{K}_1$ and $\boldsymbol{K}_2$. We will instead give virtual camera 2 the same parameters as real camera 1 ($\boldsymbol{K}_1$).

Case 4: Image "Rectification" : Solution

Inverse map the camera 1 pixels to the image projection plane. Then rotate as just described. Then remap to (virtual) camera 1's pixel coordinates.

$$\begin{bmatrix} wx_{rect} \\ wy_{rect} \\ w \end{bmatrix} = \boldsymbol{K}_1 \boldsymbol{R}_{rect} \boldsymbol{K}_1^{-1} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Inverse map the camera 2 pixels to the image projection plane. Then rotate from real camera 2's orientation to real camera 1's orientation. Then rotate using the same rectifying map. Then remap to (virtual) camera 1's pixel coordinates (not camera 2's – why not?).

$$\begin{bmatrix} wx'_{rect} \\ wy'_{rect} \\ w \end{bmatrix} = \boldsymbol{K}_1 (\boldsymbol{R}_{rect} \boldsymbol{R}_1 \boldsymbol{R}_2^T) \boldsymbol{K}_2^{-1} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$$

Overview of Today

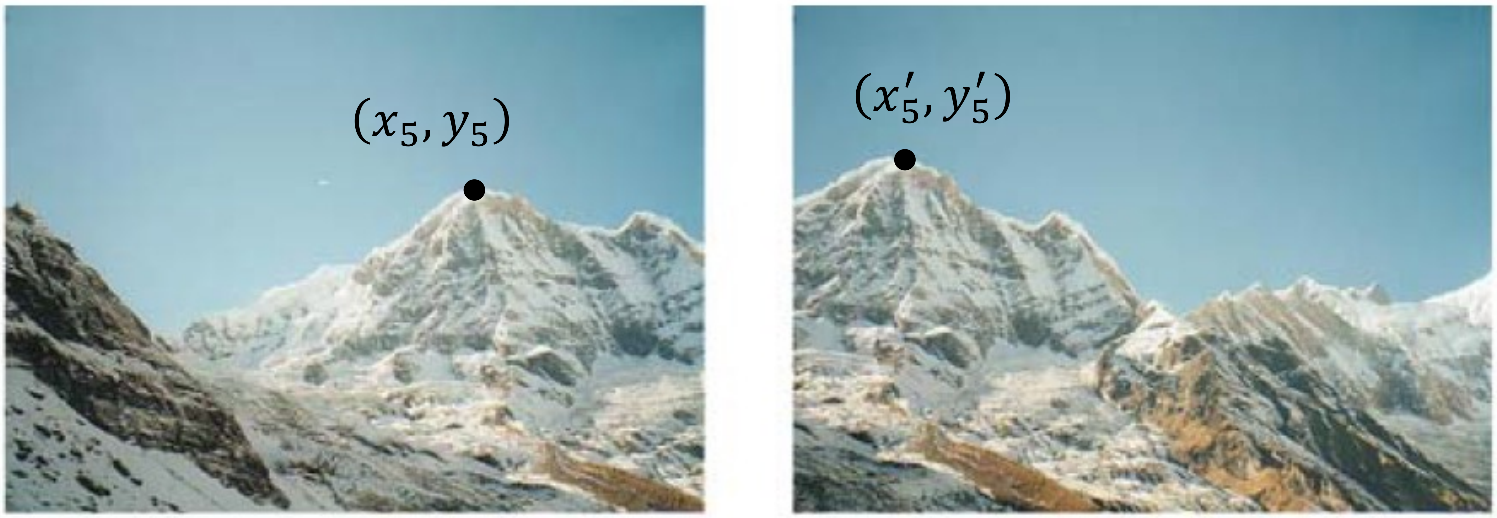
Examples of situations that involve homographies

1. a camera looking at a parameterized scene plane
2. two different cameras looking at a scene plane
3. rotating one camera
4. two cameras looking at a non-planar 3D scene

Given two images, how to fit a homography between them?

How to combine the two images?

Given two images, how do you fit a homography that maps one to the other? (e.g. Case 3: image stitching)



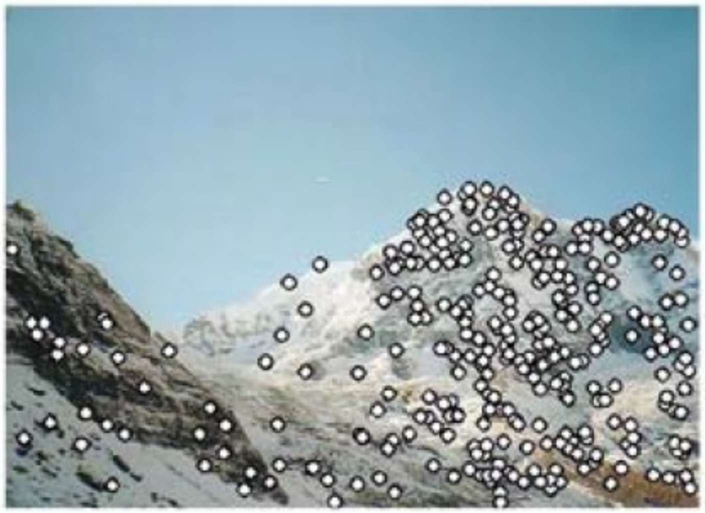
$$\{(x_i, y_i, x'_i, y'_i)\}$$

Assume that we can find matching points (e.g. SIFT features).

Then we can apply similar least squares methods that we used for camera calibration.

Given two images, how do we choose matching points ?

[Case 3: image stitching (Brown and Lowe, 2007)]



$\{(x_i, y_i)\}$ $\{(x'_k, y'_k)\}$

Use SIFT features. For each SIFT keypoint in image 1, find the most similar feature (or k features) in image 2, and vice versa, where similarity is defined by the 128-D SIFT descriptor.

This gives us a set of candidate matching pairs $\{ (x_i, y_i, x'_k, y'_k) \}$.

In the case of a homography $\boldsymbol{H}$ between two images:

$$\begin{bmatrix} w'x_i' \\ w'y_i' \\ w' \end{bmatrix} \approx \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

$$x_i' = \frac{w_i x_i'}{w_i} \approx \frac{H_{11}x_i + H_{12}y_i + H_{13}}{H_{31}x_i + H_{32}y_i + H_{33}}$$

$$y_i' = \frac{w_i y_i'}{w_i} \approx \frac{H_{21}x_i + H_{22}y_i + H_{23}}{H_{31}x_i + H_{32}y_i + H_{33}}$$

Each matching pair of points gives us two linear “equations”.

$$x_i'(H_{31}x_i + H_{32}y_i + H_{33}) \approx H_{11}x_i + H_{12}y_i + H_{13}$$

$$y_i'(H_{31}x_i + H_{32}y_i + H_{33}) \approx H_{21}x_i + H_{22}y_i + H_{23}.$$

We have N matching pairs of points $\{(x_i, y_i, x'_i, y'_i)\}$.
Stack the N pairs of equations:

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1x_1 & -x'_1y_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1x_1 & -y'_1y_1 & -y'_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_N & y_N & 1 & 0 & 0 & 0 & -x'_Nx_N & -x'_Ny_N & -x'_N \\ 0 & 0 & 0 & x_N & y_N & 1 & -y'_Nx_N & -y'_Ny_N & -y'_N \end{bmatrix} \begin{bmatrix} H_{11} \\ H_{12} \\ H_{13} \\ H_{21} \\ H_{22} \\ H_{23} \\ H_{31} \\ H_{32} \\ H_{33} \end{bmatrix} \approx \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

This is the familiar form $\boldsymbol{Ax} \approx 0$, where $\boldsymbol{A}$ is an $2N \times 9$ data matrix.

The solution for $\boldsymbol{H}$ is the eigenvector of $\boldsymbol{A^T A}$ with the smallest eigenvalue.

Data Normalization

Normalize so that the mean and standard deviation are 0 and 1, respectively.

$$(x_i, y_i) \rightarrow (\frac{x_i - \bar{x}}{\sigma_1}, \frac{y_i - \bar{y}}{\sigma_1})$$

$$(x'_i, y'_i) \rightarrow (\frac{x'_i - \bar{x}'}{\sigma_2}, \frac{y'_i - \bar{x}'}{\sigma_2})$$

$$\underbrace{\begin{bmatrix} 1/\sigma_1 & 0 & 0 \\ 0 & 1/\sigma_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -\bar{x} \\ 0 & 1 & -\bar{y} \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{M_1}}$$

$$\underbrace{\begin{bmatrix} 1/\sigma_2 & 0 & 0 \\ 0 & 1/\sigma_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -\bar{\bar{x}} \\ 0 & 1 & -\bar{\bar{y}} \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{M_2}}$$

Solve the least squares problem for $\boldsymbol{H_{normalized}}$ using normalized values.

$$w_i \mathbf{M}_2 \begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} \approx \mathbf{H}_{normalized} \mathbf{M}_1 \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

Then the solution for the homography in the original image space is:

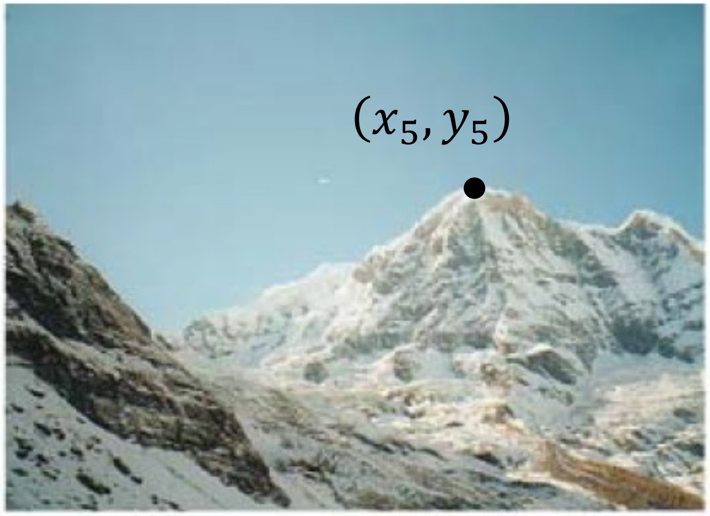
$$\mathbf{H} = (\mathbf{M}_2)^{-1} \mathbf{H}_{normalized} \mathbf{M}_1$$

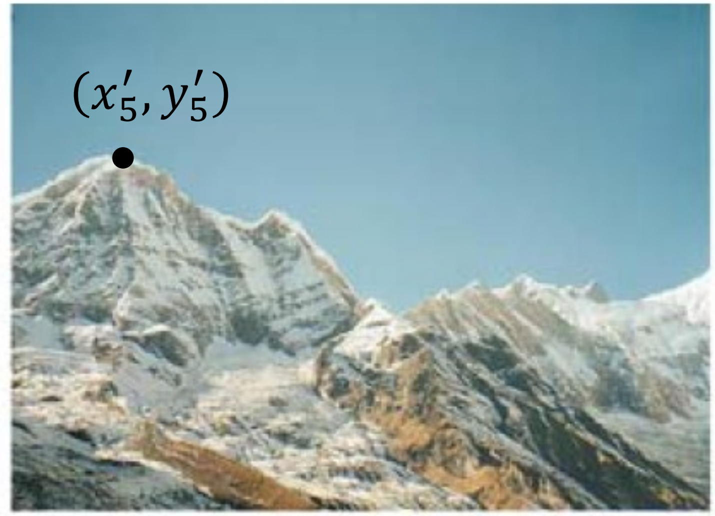
$\{(x_i, y_i)\}$ $\{(x'_k, y'_k)\}$

We have assumed that we can find a set of candidate matching pairs $\{\ (x_i, y_i, x'_k, y'_k)\ \}$.

However, *some of the matches might be incorrect*. What to do?

Use RANSAC. RANSAC requires fitting an *exact model,* using the *minimum number of samples*.





$$\{\, (x_i, y_i, x'_i, y'_i) \,\}.$$

Here we re-index so that we can talk about the ith matching pair.

We have N matching pairs of points $\{(x_i, y_i, x'_i, y'_i)\}$.

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1x_1 & -x'_1y_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1x_1 & -y'_1y_1 & -y'_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_N & y_N & 1 & 0 & 0 & 0 & -x'_Nx_N & -x'_Ny_N & -x'_N \\ 0 & 0 & 0 & x_N & y_N & 1 & -y'_Nx_N & -y'_Ny_N & -y'_N \end{bmatrix} \begin{bmatrix} H_{11} \\ H_{12} \\ H_{13} \\ H_{21} \\ H_{22} \\ H_{23} \\ H_{31} \\ H_{32} \\ H_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

$\boldsymbol{A}$ is an $2N \times 9$ data matrix.

Here we are not looking for a least squares solution. Rather we are looking for an exact solution (the null space of $\boldsymbol{A}$).

How many matching pairs of points N do we need for an exactly fitting homography $\boldsymbol{H}$?

RANSAC
(Random Sample Consensus)

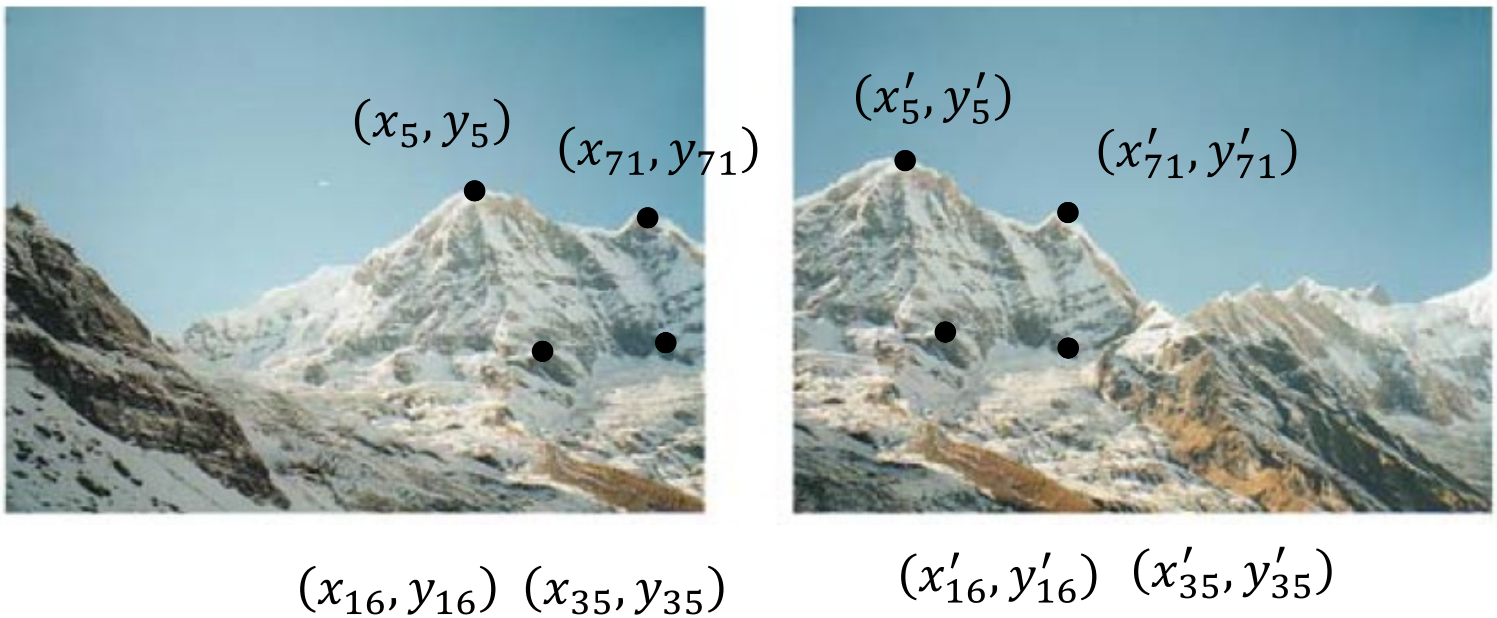
repeat {

- randomly sample $N = 4$ matching pairs of points i.e. (x_i, y_i, x'_i, y'_i) from the two images;
- fit a homography (*exact* model fit) that maps these (x_i, y_i) to (x'_i, y'_i)
- examine remaining matching pairs of points (quadruples) and count how many are a *good* match for the model $\boldsymbol{H}$. (Let C be the "consensus set").
- if C is sufficiently large, then refit $\mathbf{H}$ using *all* points in the consensus set, using least squares. If the model fit is the best so far, then save it.

} until some condition is satisfied

- randomly sample $N = 4$ matching pairs of points i.e. (x_i, y_i, x'_i, y'_i) from the two images;

- fit a homography (*exact* model fit) that maps these (x_i, y_i) to (x'_i, y'_i)



RANSAC (Random Sample Consensus)

repeat {

- randomly sample $N = 4$ matching pairs of points i.e. (x_i, y_i, x'_i, y'_i) from the two images;
- fit a homography (*exact* model fit) that maps these (x_i, y_i) to (x'_i, y'_i)
- examine remaining matching pairs of points (quadruples) and count how many are a *good* match for the model $\boldsymbol{H}$. (Let C be the "consensus set"). If we are using SIFT keypoints, then we can require that the scales and dominant orientations are a good fit as well. (Details omitted here.)
- if C is sufficiently large, then refit $\mathbf{H}$ using *all* matching pairs in the consensus set, using least squares. If the model fit is the best so far, then save it.

} until some condition is satisfied

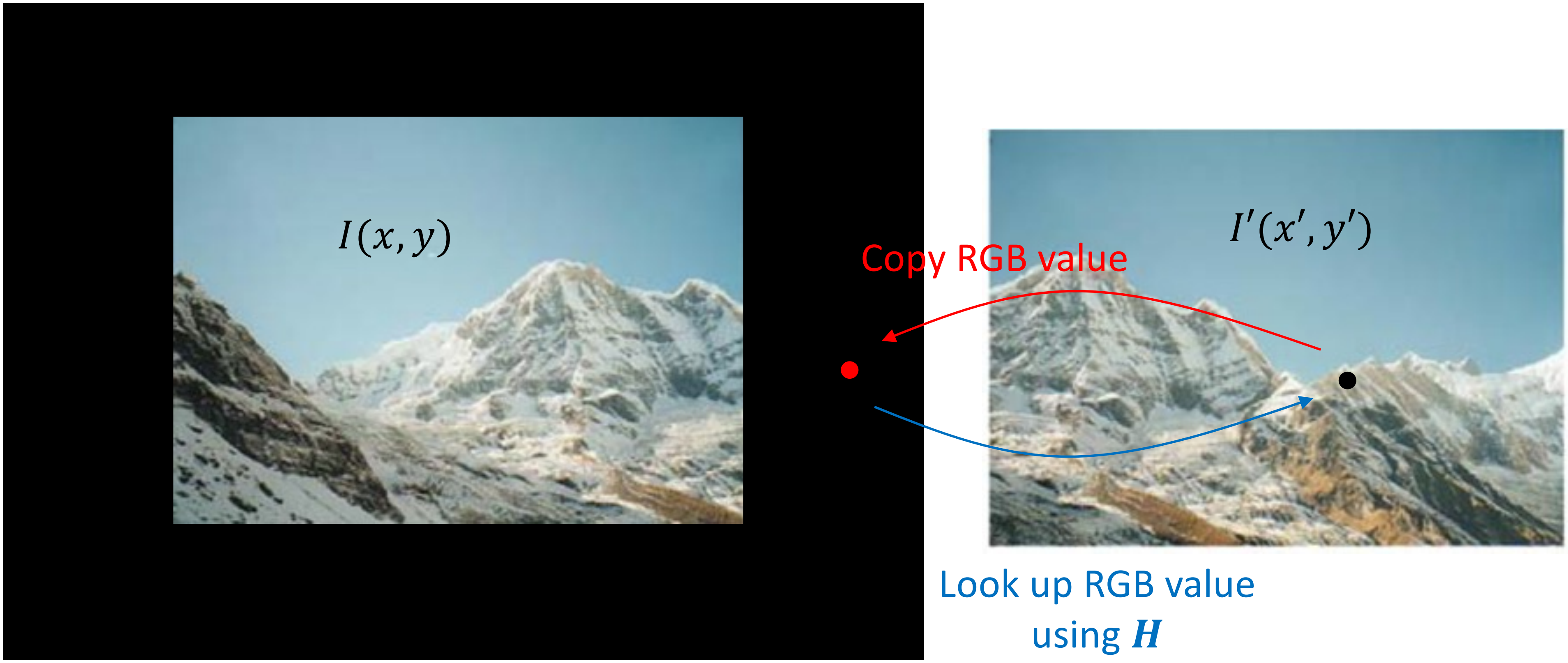
Given the homography $\mathbf{H}$ that maps (x, y) to (x', y') …

$I(x, y)$

$I'(x', y')$

$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \boldsymbol{H} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

… how do we combine the two images to obtain a bigger image?

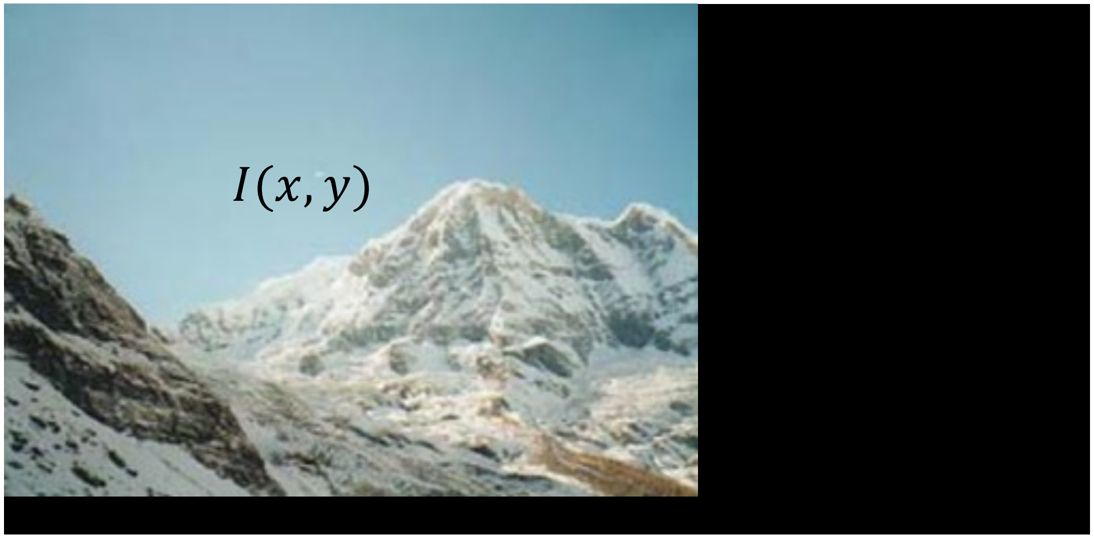


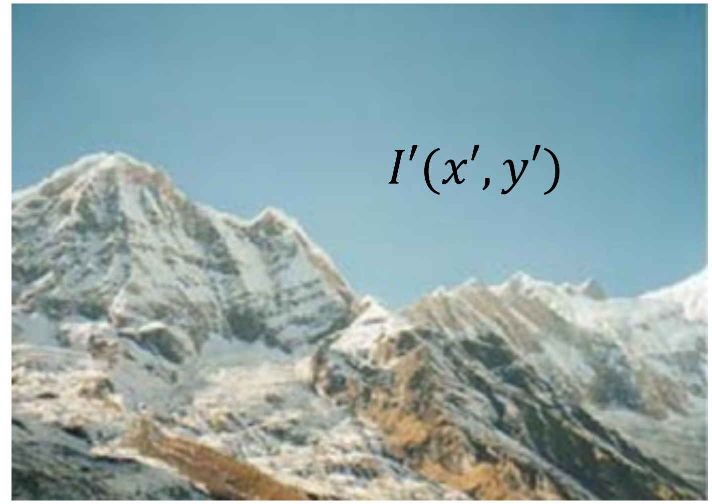
For each (x, y) in the unassigned black region of $I(x, y)$, check if the corresponding position (x', y')

$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \boldsymbol{H} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

is within the other image. If so, then get the value $I'(\,round(x'), round(y'))$ and assign it to $I(x, y)$.

One should choose the black region based on where 'new' pixels are available in the other image.





Next week

Two cameras looking a 3D scene

- what if the scene is not planar ?
- what if the camera is not calibrated ?

How can we estimate 3D positions of scene points in the above situation?