Image Stitching

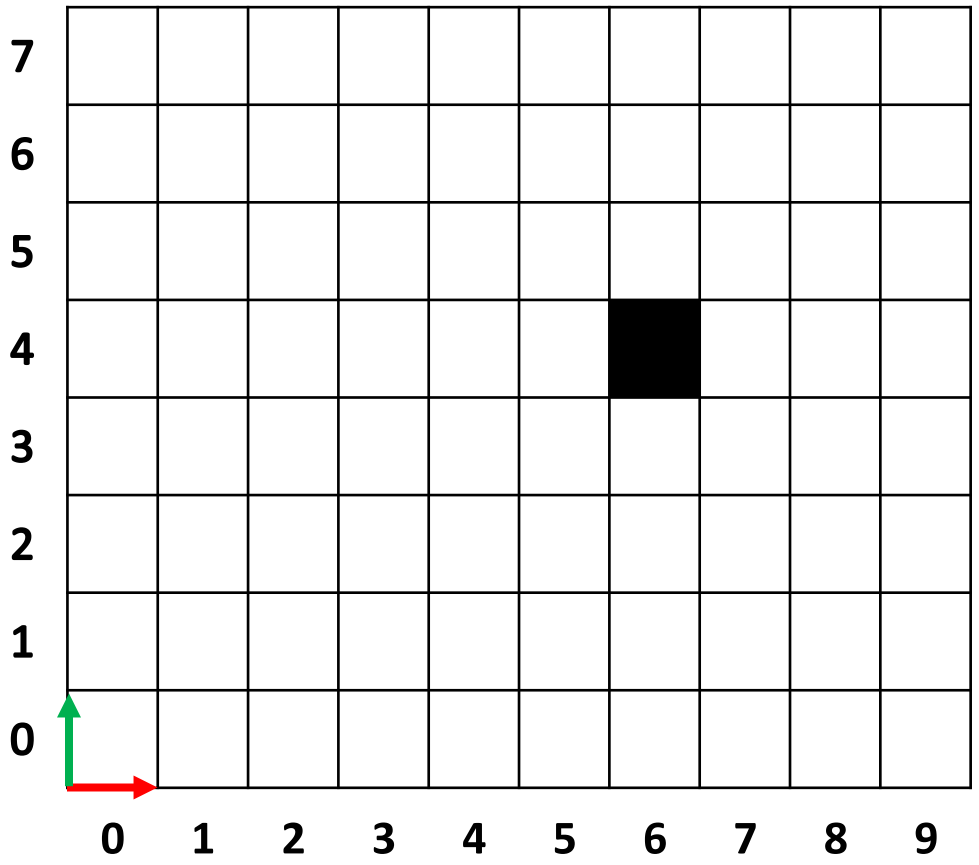
Computational Photography (CSCI 3240U)

Faisal Z. Qureshi
http://vclab.science.ontariotechu.ca
OntarioTech UNIVERSITY

Today

- Image stitching

Cartesian coordinate system

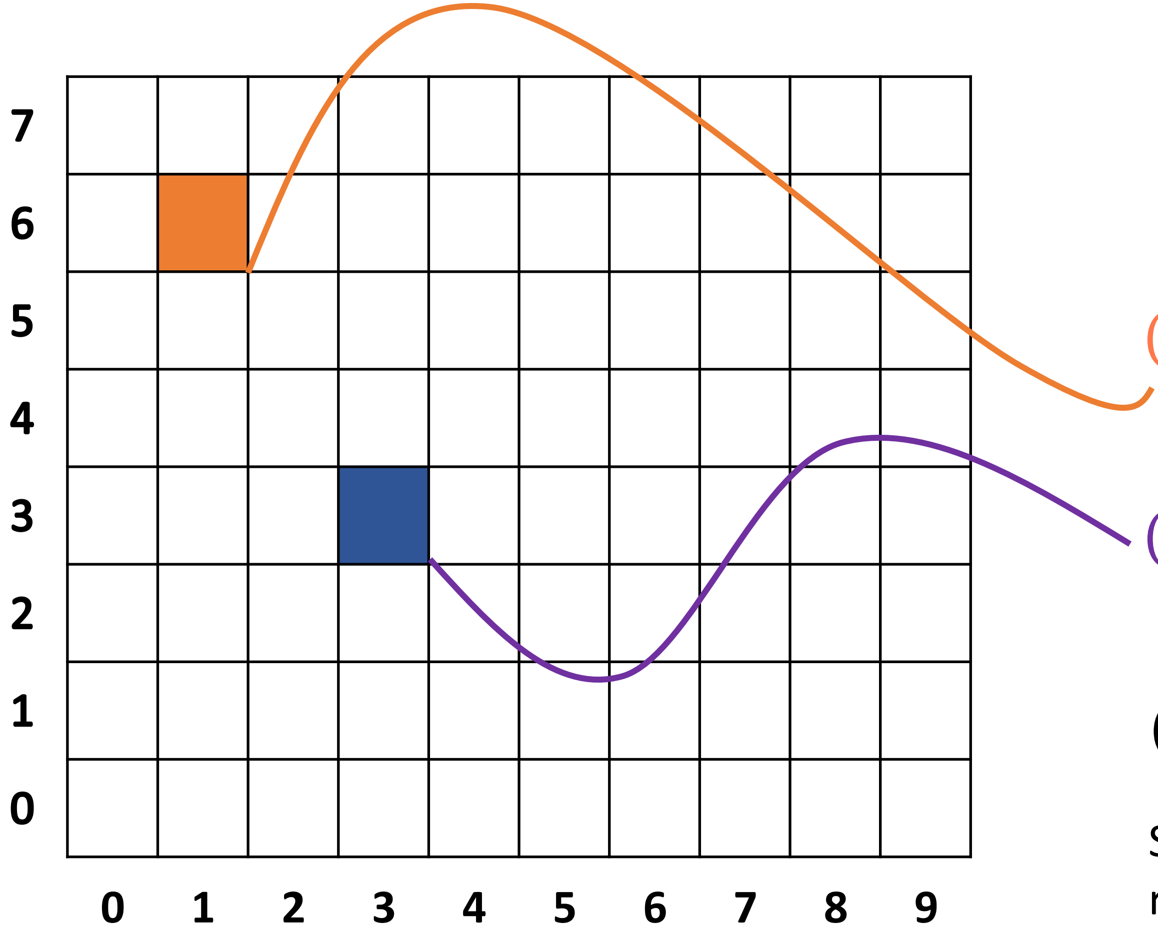


Recall

$$\begin{bmatrix}6\\4\end{bmatrix} = 6\begin{bmatrix}1\\0\end{bmatrix} + 4\begin{bmatrix}0\\1\end{bmatrix}$$

Basis vectors

Euclidean vs. Homogeneous Coordinates



Euclidean (Cartesian) to Homogeneous Coordinates

$(1,6)$ → $(1,6,1)$

$(3,3)$ → $(3,3,1)$

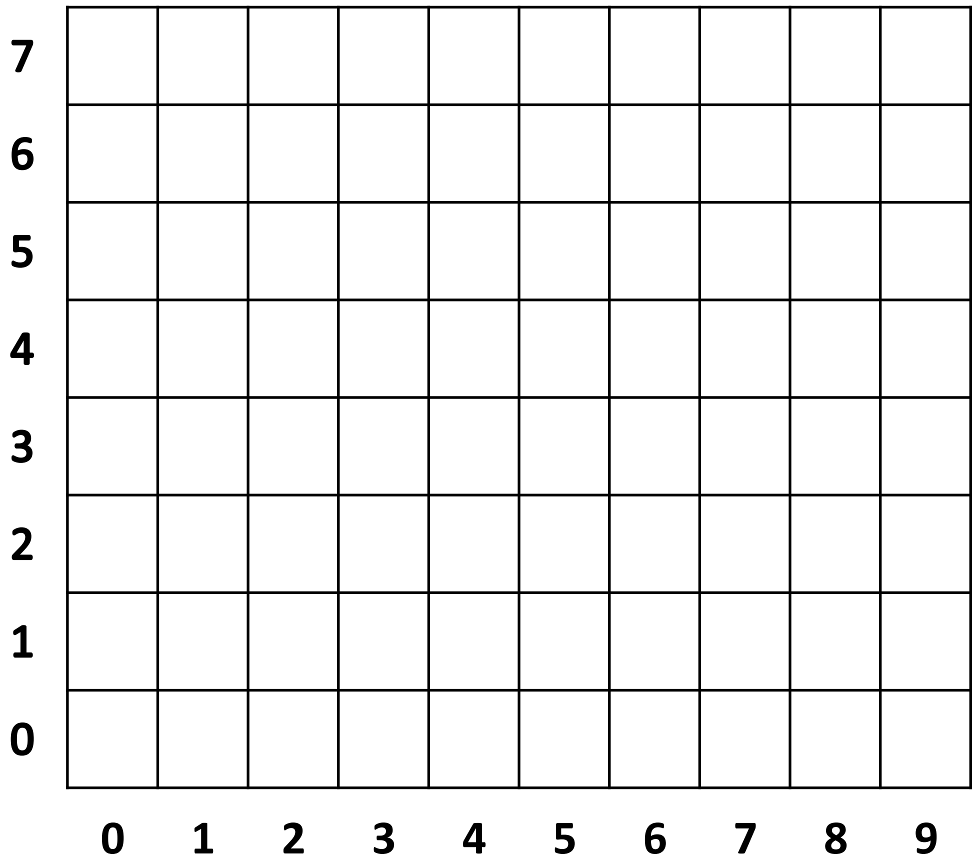
(x, y)

Standard Euclidean representation of points

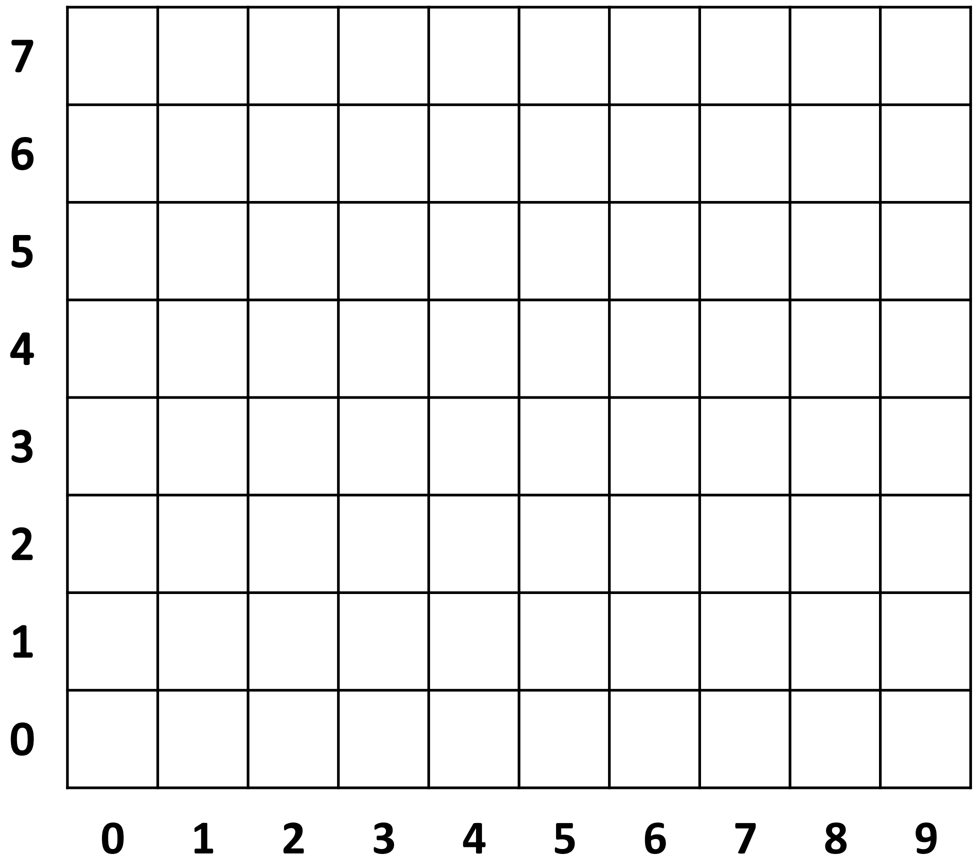
$(\lambda x, \lambda y, \lambda)$
for any $\lambda \neq 0$

Also called *projective representation* of a point

Euclidean vs. Homogeneous Coordinates



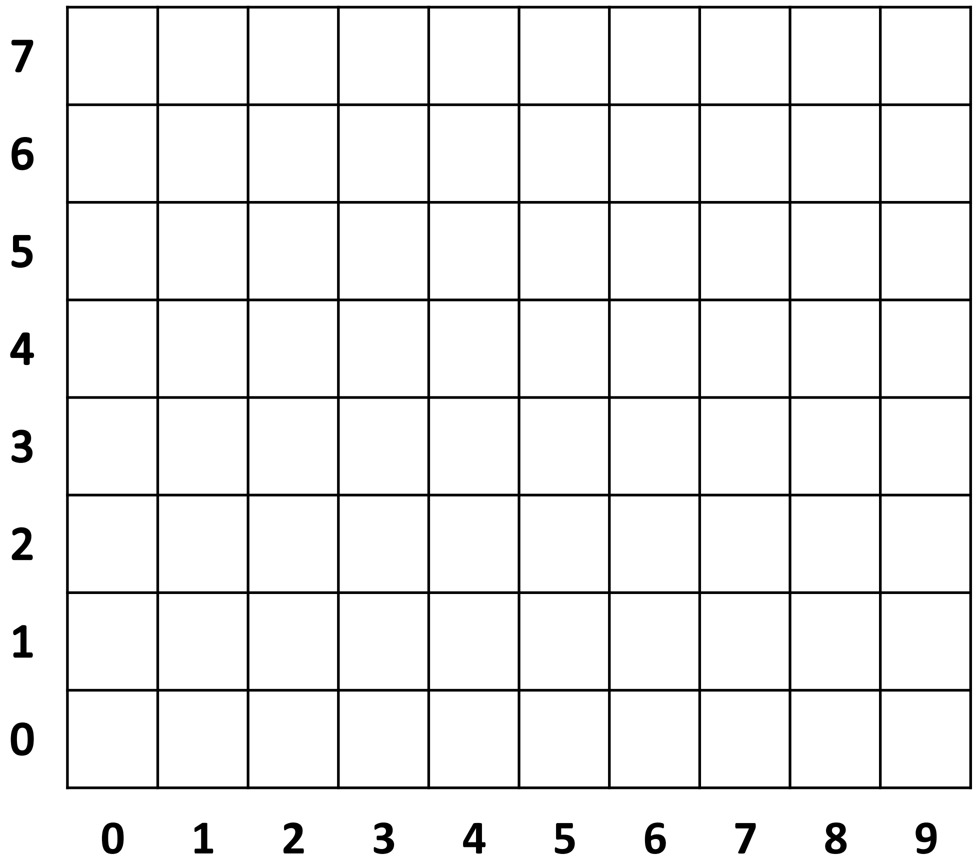
Points at infinity



Say you are given a homogeneous point $(6,4,2)$

How do represent it in Cartesian coordinates?

Points at infinity



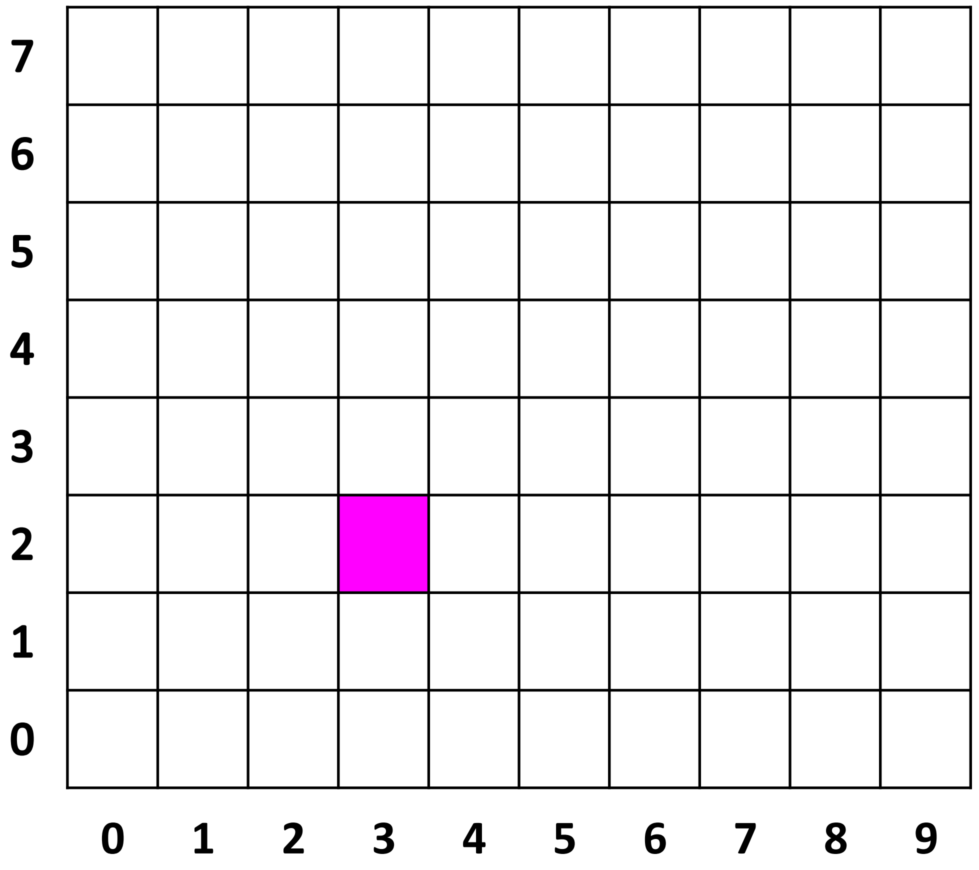
Say you are given a homogeneous point $(6,4,2)$

How do represent it in Cartesian coordinates? **Ans: (3,2)**

Say you are given a homogeneous point $(6,4,0)$

How do represent it in Cartesian coordinates?

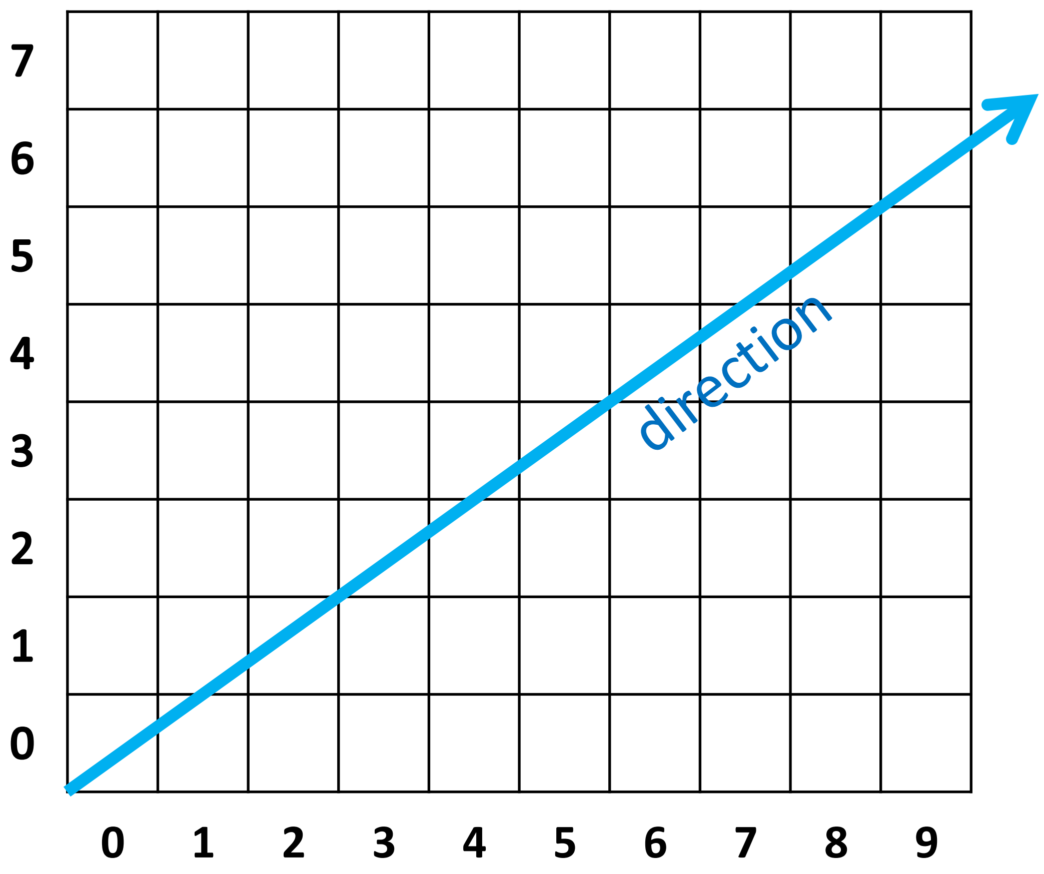
Points at infinity



Say you are given a homogeneous point $(6,4,2)$

How do represent it in Cartesian coordinates? **Ans: (3,2)**

Points at infinity



Say you are given a homogeneous point $(6,4,2)$

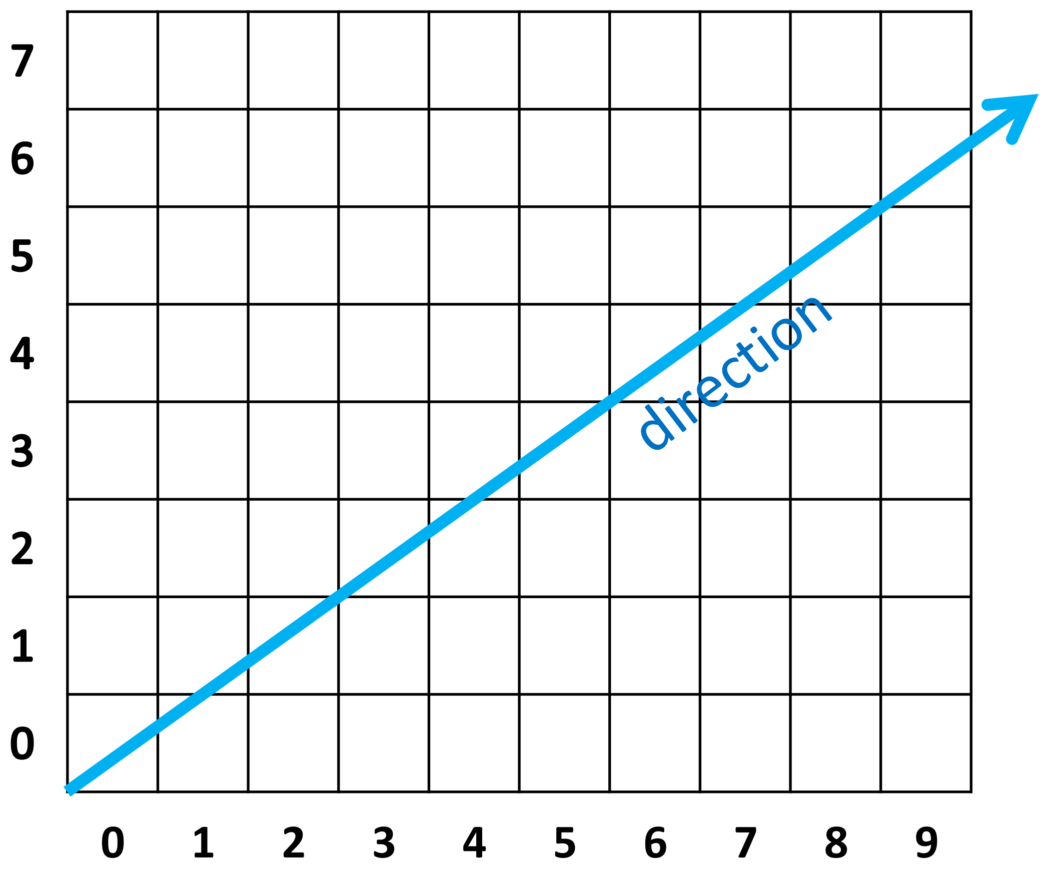
How do represent it in Cartesian coordinates? **Ans: (3,2)**

Say you are given a homogeneous point $(6,4,0)$

How do represent it in Cartesian coordinates? Ans: **direction (6,4)**

This is not a Cartesian point. Rather it denotes a direction

Points at infinity



Case 1

Say you are given a homogeneous point $(6,4,2)$

How do represent it in Cartesian coordinates? **Ans: (3,2)**

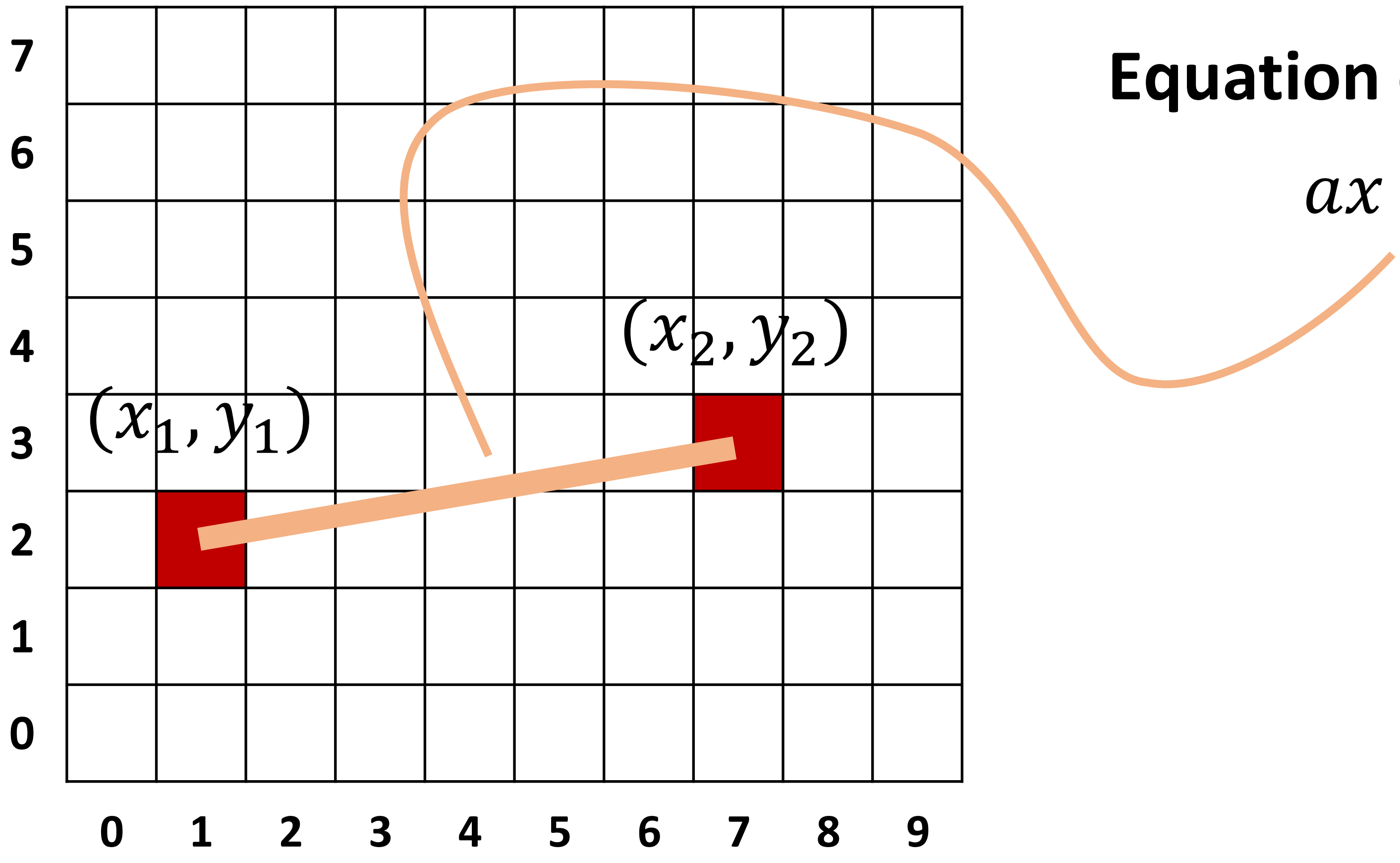
Case 2

Say you are given a homogeneous point $(6,4,0)$

How do represent it in Cartesian coordinates? Ans: **direction (6,4)**

This is not a Cartesian point. Rather it denotes a direction

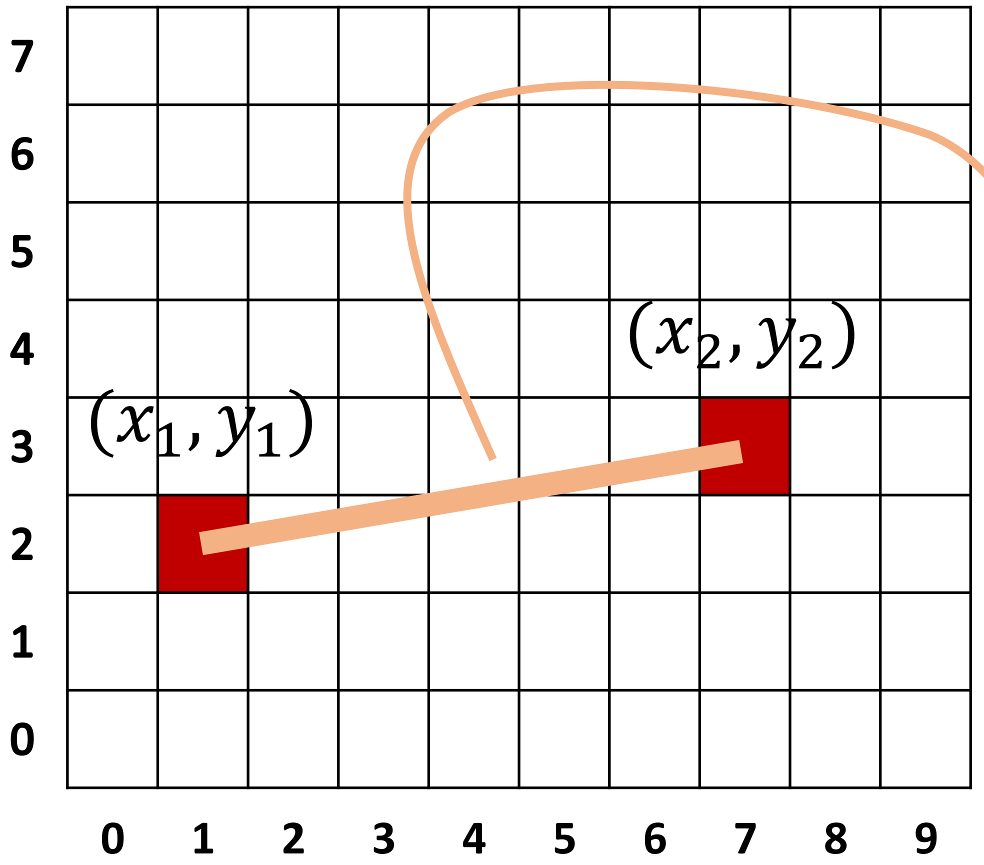
Line equations in homogeneous coordinates



Equation of a line

$$ax + by + c = 0$$

Line equations in homogeneous coordinates



Equation of a line

$$ax + by + c = 0$$

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0$$

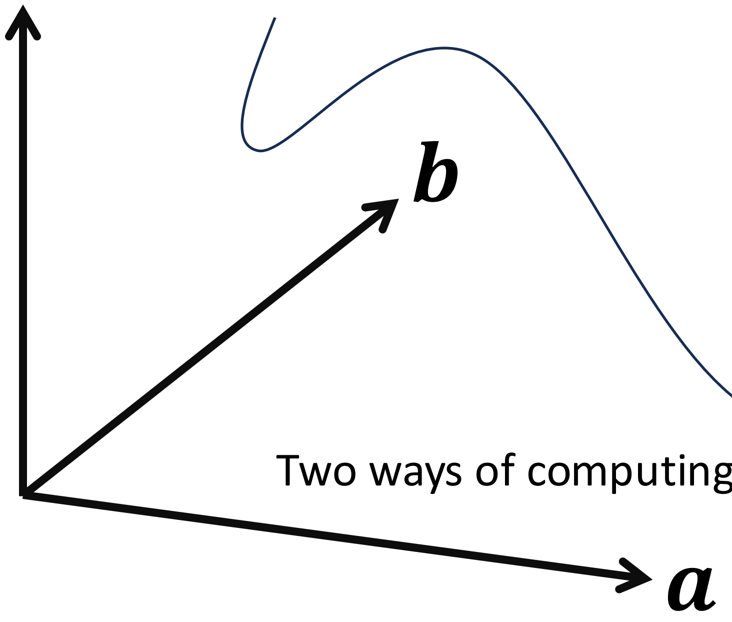
Dot-product

Observations

1. Vectors (a, b, c) and $(x, y, 1)$ are orthogonal to each other
2. This is true for any point (x, y) that lies on the line defined by a, b, and c

Cross-product of two vectors

$$\boldsymbol{c} = \boldsymbol{a} \times \boldsymbol{b}$$



Two ways of computing cross-product

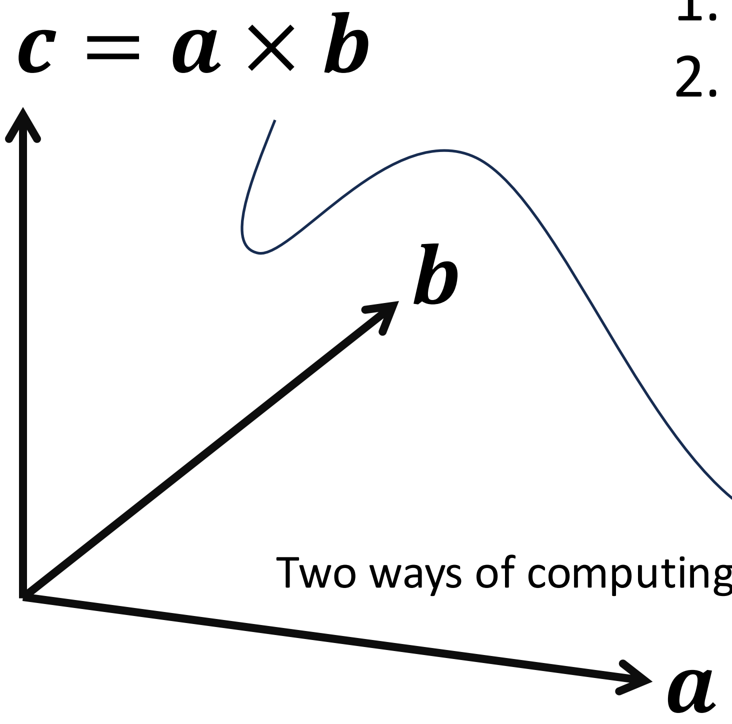
$$\boldsymbol{a} \times \boldsymbol{b} = \begin{vmatrix} \boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k} \\ \boldsymbol{a_x} & \boldsymbol{a_y} & \boldsymbol{a_z} \\ \boldsymbol{b_x} & \boldsymbol{b_y} & \boldsymbol{b_z} \end{vmatrix}$$

determinant

$$\boldsymbol{a} \times \boldsymbol{b} = \begin{bmatrix} 0 & -\boldsymbol{a_z} & \boldsymbol{a_y} \\ \boldsymbol{a_z} & 0 & -\boldsymbol{a_x} \\ -\boldsymbol{a_y} & \boldsymbol{a_x} & 0 \end{bmatrix} \boldsymbol{b}$$

matrix-vector product

Cross-product of two vectors



1. Vector $\boldsymbol{c} = \boldsymbol{a} \times \boldsymbol{b}$ is orthogonal to both vectors $\boldsymbol{a}$ and $\boldsymbol{b}$
2. It follows that $\boldsymbol{c} \cdot \boldsymbol{a} = \mathbf{0}$ and $\boldsymbol{c} \cdot \boldsymbol{b} = \mathbf{0}$

Two ways of computing cross-product

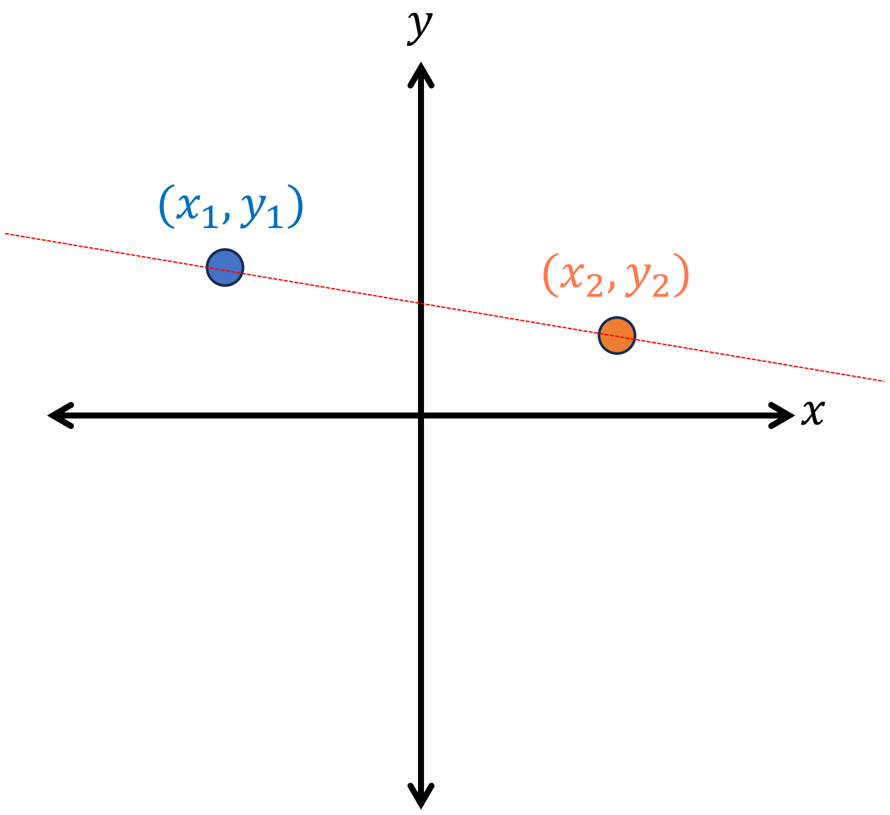
$$\boldsymbol{a} \times \boldsymbol{b} = \begin{vmatrix} \boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k} \\ \boldsymbol{a_x} & \boldsymbol{a_y} & \boldsymbol{a_z} \\ \boldsymbol{b_x} & \boldsymbol{b_y} & \boldsymbol{b_z} \end{vmatrix}$$

determinant

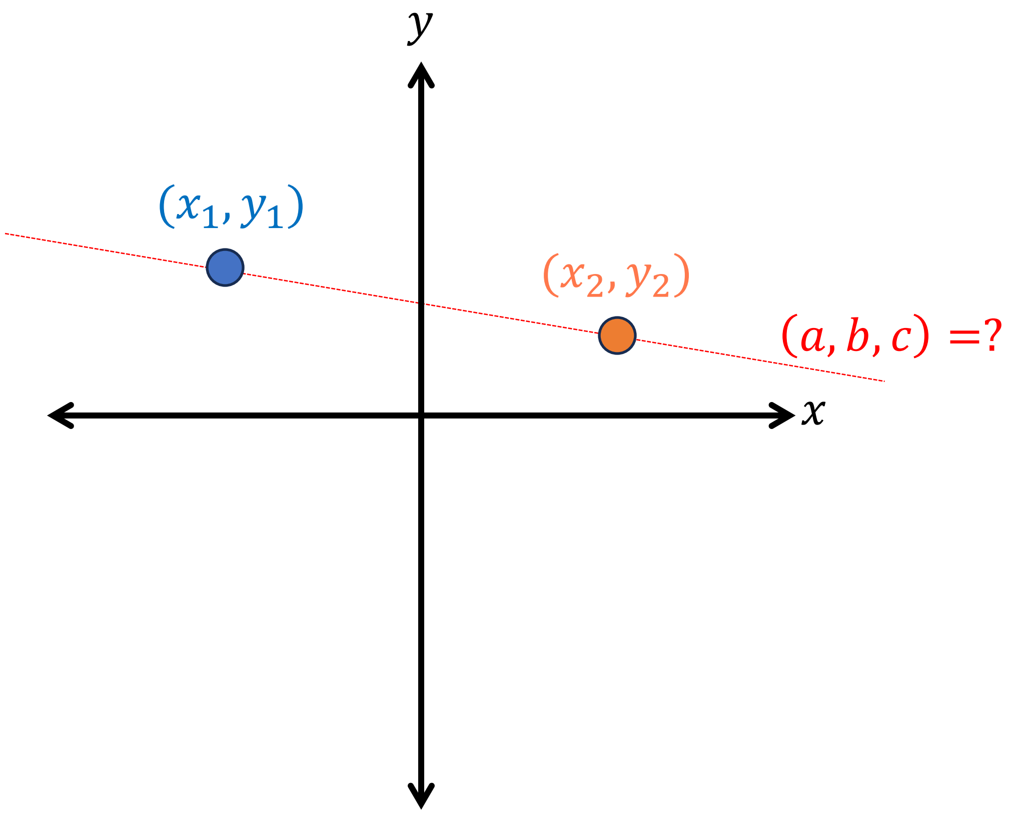
$$\boldsymbol{a} \times \boldsymbol{b} = \begin{bmatrix} \mathbf{0} & -\boldsymbol{a_z} & \boldsymbol{a_y} \\ \boldsymbol{a_z} & \mathbf{0} & -\boldsymbol{a_x} \\ -\boldsymbol{a_y} & \boldsymbol{a_x} & \mathbf{0} \end{bmatrix} \boldsymbol{b}$$

matrix-vector product

The line passing through two points



The line passing through two points



From previous slides, we know

$$(x_1, y_1, 1) \cdot (a, b, c) = 0$$

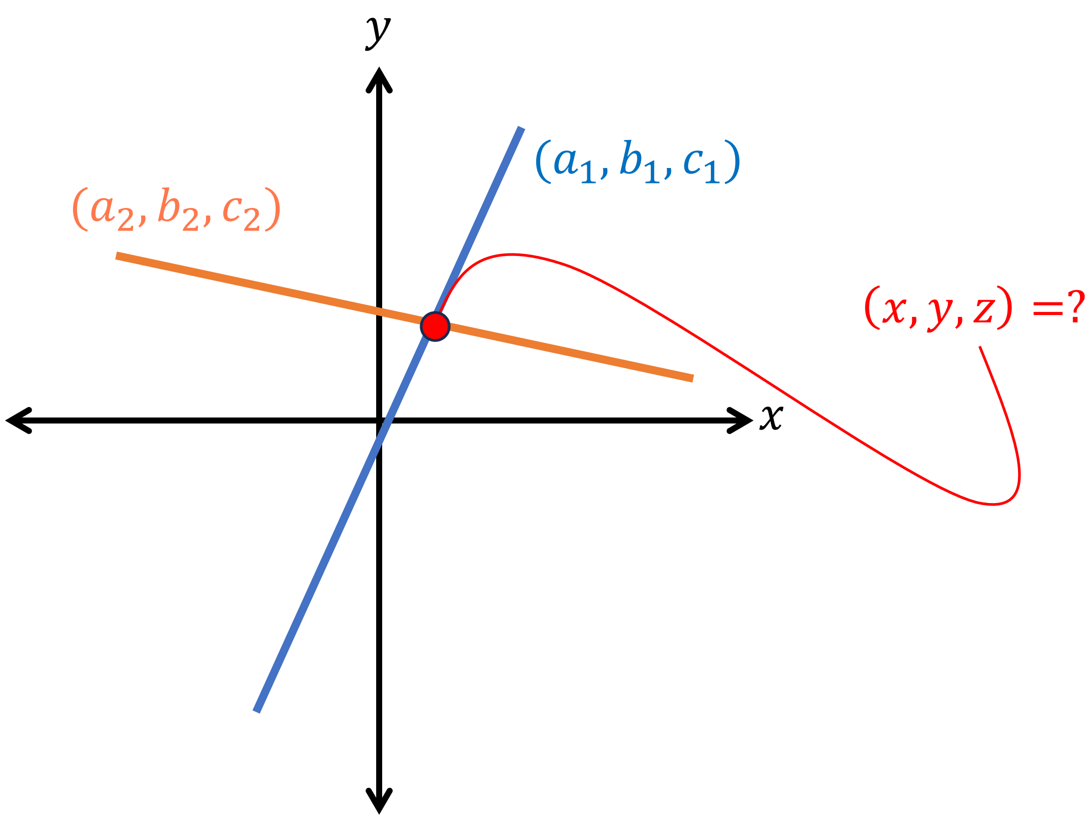
$$(x_2, y_2, 1) \cdot (a, b, c) = 0$$

Therefore, we can estimate the line parameters as follows

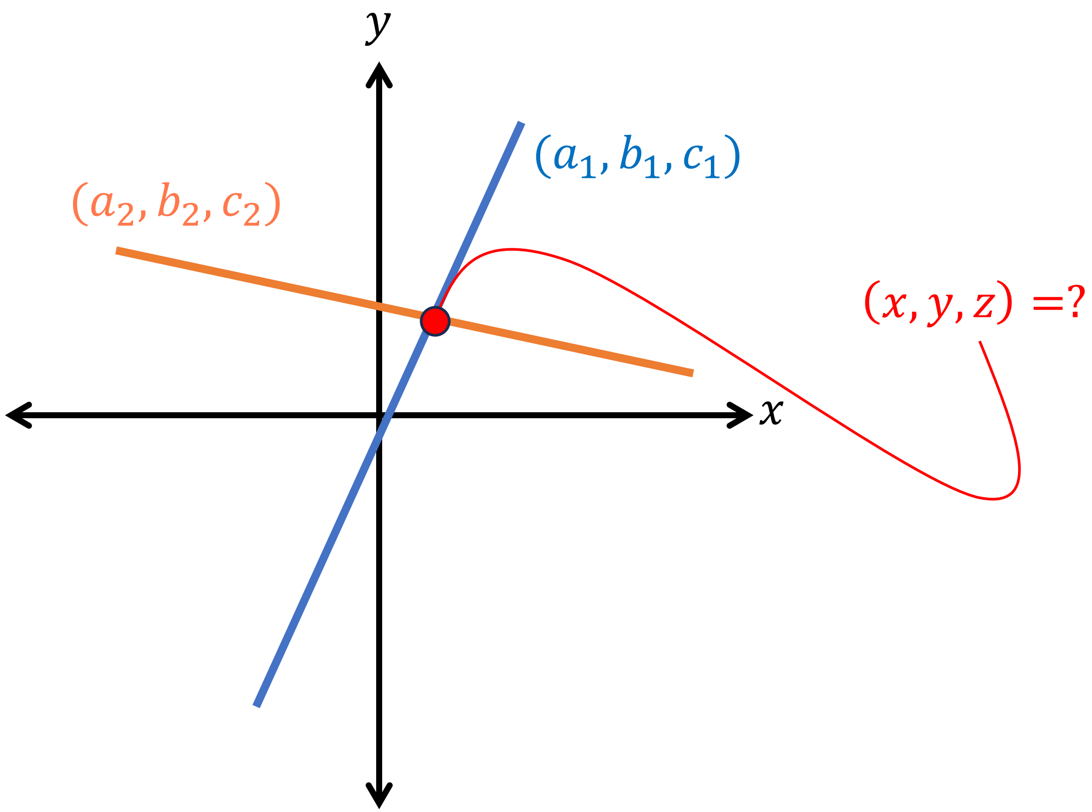
$$(a, b, c) = (x_1, y_1, 1) \times (x_2, y_2, 1)$$

Homogeneous coordinates

The point of intersection of two lines



The point of intersection of two lines



From previous slides, we know

$$(x, y, 1) \cdot (a_1, b_1, c_1) = 0$$

$$(x, y, 1) \cdot (a_2, b_2, c_2) = 0$$

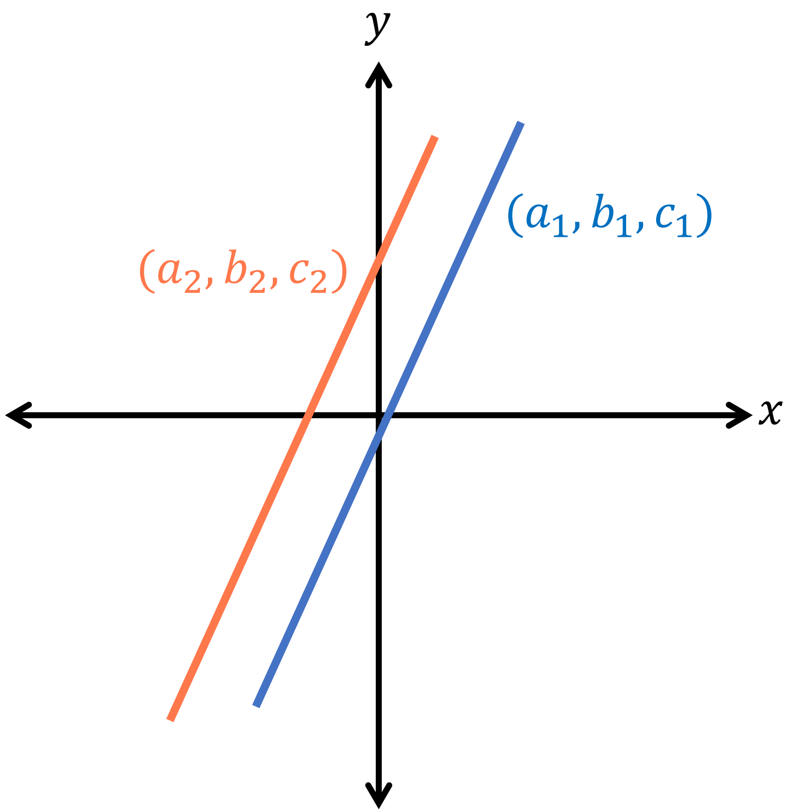
Therefore, we can estimate the intersection point in homogeneous coordinates as follows

$$(\lambda x, \lambda y, \lambda) = (a_1, b_1, c_1) \times (a_2, b_2, c_2)$$

Homogeneous coordinates

Convert to Cartesian (x, y) by dividing by λ

Intersecting two parallel lines



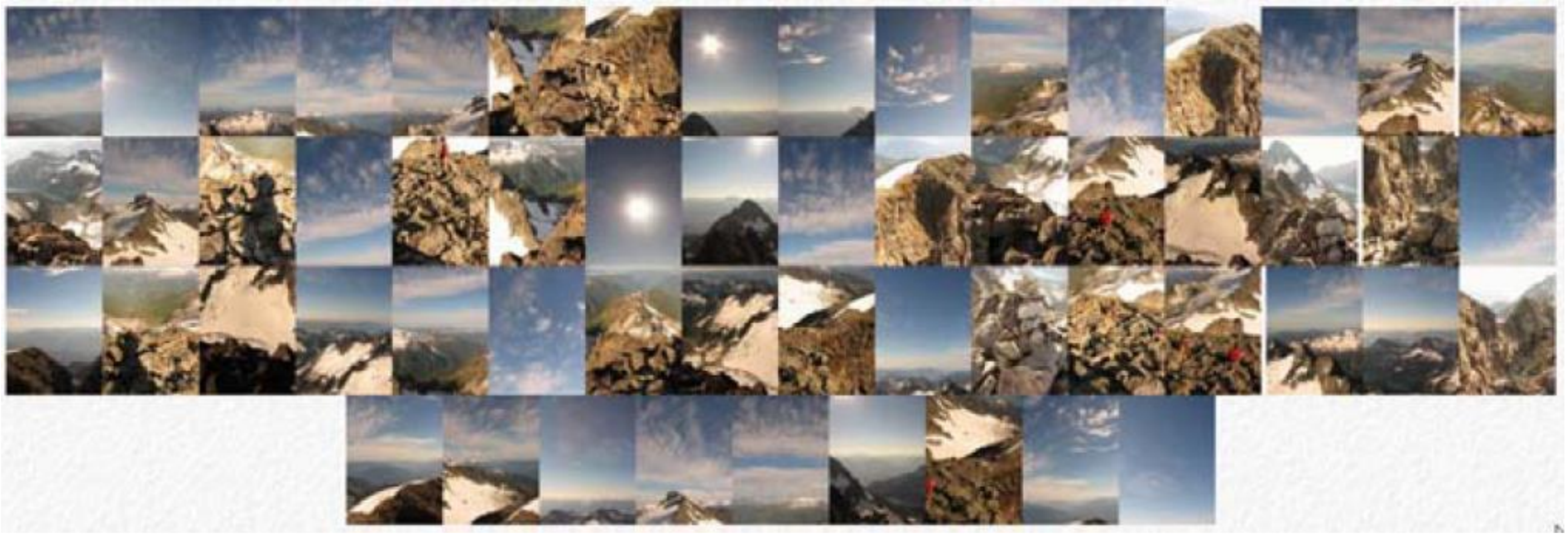
We can estimate the intersection point of two parallel lines in homogeneous coordinates as follows

$$(d, e, f) = (a_1, b_1, c_1) \times (a_2, b_2, c_2)$$

Since parallel lines **do not** intersect, (d, e, f) will not be a valid Cartesian coordinate and the homogeneous parameter f will be equal to 0.

The above equation therefore provides the direction that points to the **point at infinity** where the two lines intersect.

Image stitching



57 images

Camera should change orientation only, not position.
Keep camera settings (gain, focus, speed, aperture) fixed, if possible.

Image stitching

Using 28 out of 57 images

Image stitching

Using all 57 images

Image stitching (Autostitch)

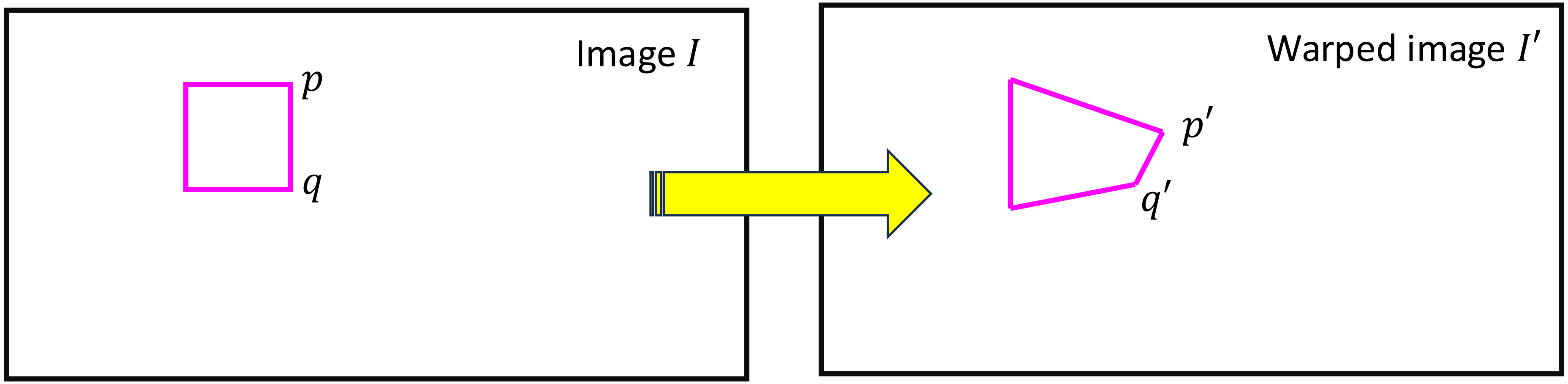
Seams are not visible

Using all 57 images. Laplacian blending.

Brown & Lowe; ICCV 2003

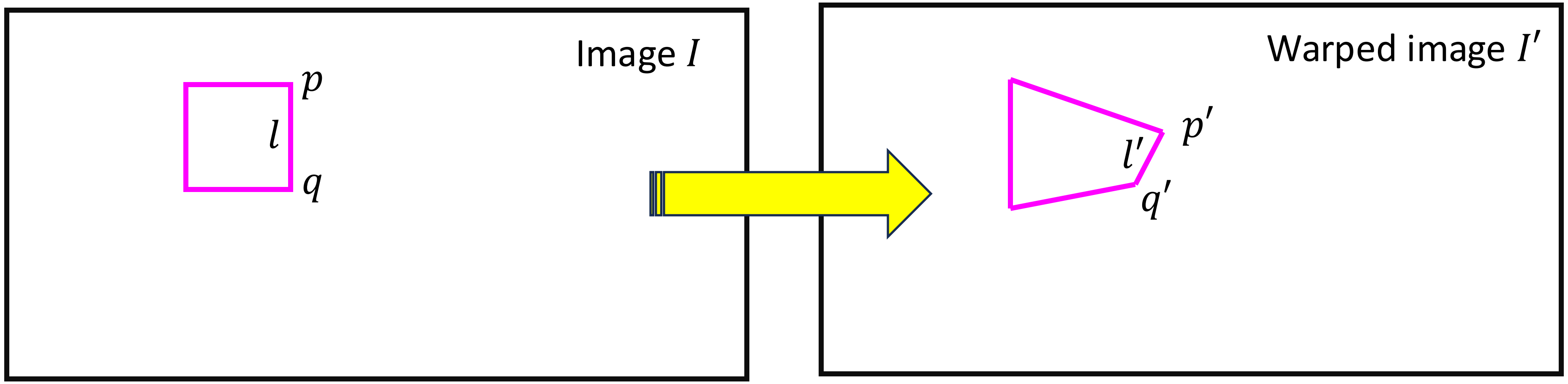
Linear image wraps

- To align multiple photos for image stitching, we must warp these images in such a way that all lines are preserved.
  - Lines before warping remain lines after warping
- Linear image wraps and *homographies*



Linear image wraps

- Definition: an image warp is linear if every 2D line l in the original image is transformed into a line l' in the warped image
- Property: Every linear warp can be expressed as a 3×3 matrix H that transforms homogeneous image coordinates (we won't discuss the proof here)



Warping images using homography

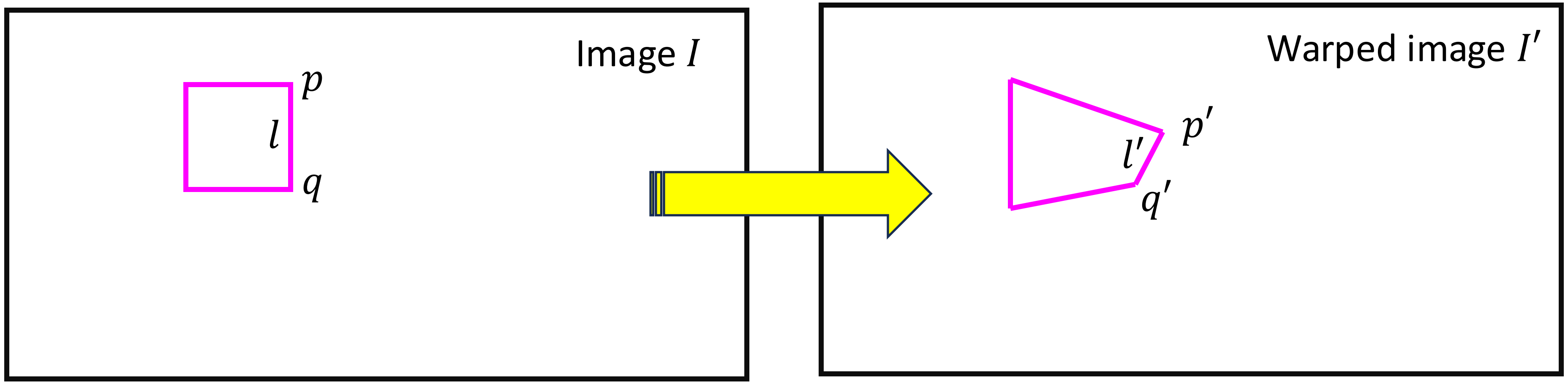
Linear warping equation: $I(\boldsymbol{p}) = I'(H\boldsymbol{p})$

Intensity at pixel in the source image I with homogeneous coordinates $\boldsymbol{p}$

Intensity at pixel in the warped image I' with homogeneous coordinates $H\boldsymbol{p}$

Matrix H is called homography

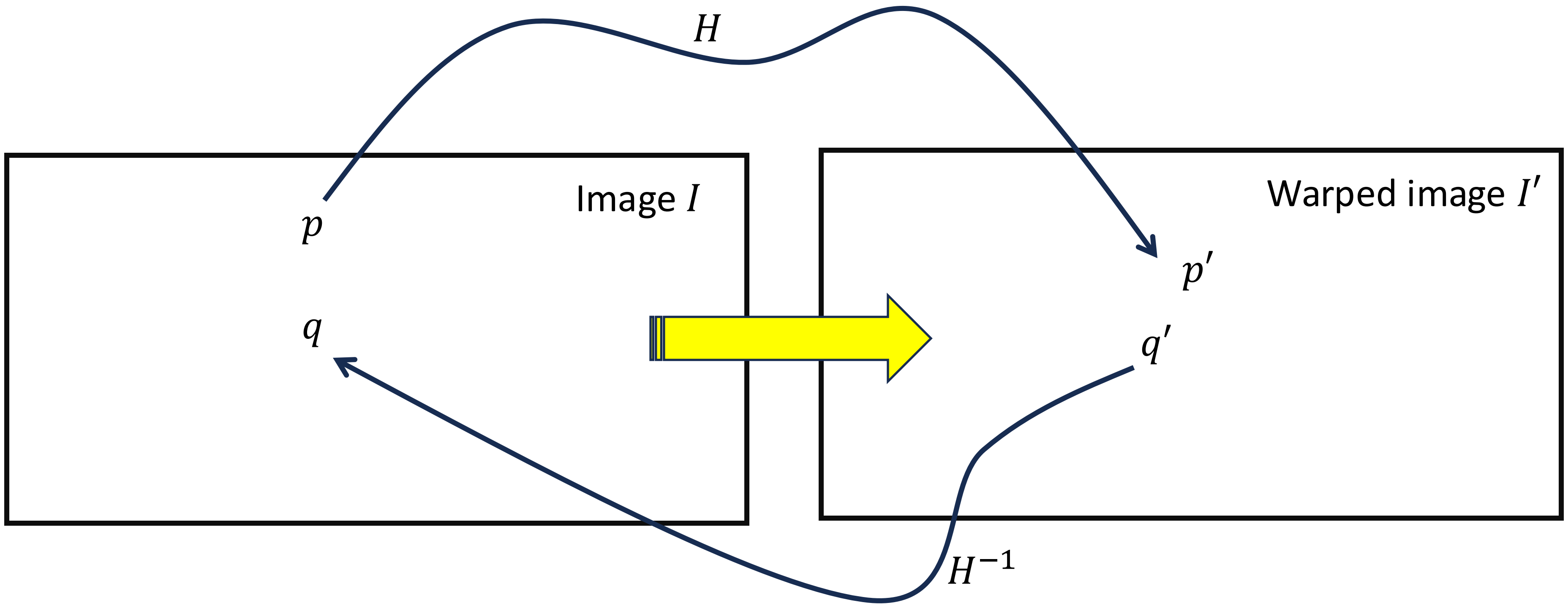
Scaling H by a factor $\lambda \neq 0$ does not change homography



Warping images using homography

Linear warping equation:

$$I(\boldsymbol{p}) = I'(H\boldsymbol{p}) \text{ and also } I'(\boldsymbol{q}') = I(H^{-1}\boldsymbol{q}')$$

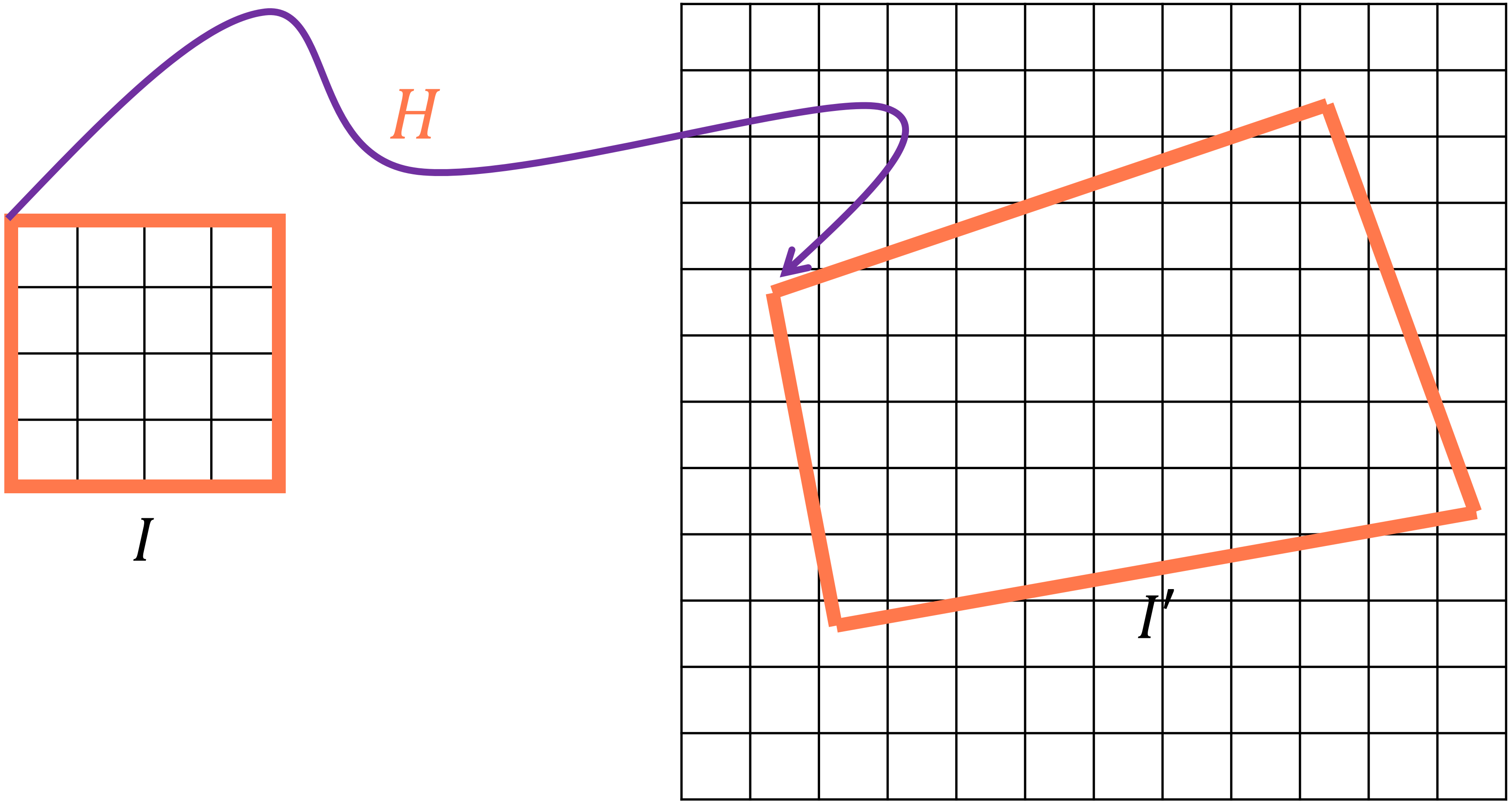


Computing warp I' from I and H

- Compute H^{-1}
- To compute the color of pixel (u, v) in the warped image
  - Compute $\begin{bmatrix} a \\ b \\ c \end{bmatrix} = H^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$
  - Copy color from $I\left(\frac{a}{c}, \frac{b}{c}\right)$

What if location $\left(\frac{a}{c}, \frac{b}{c}\right)$ is not valid pixel locations?

Computing warp I' from I and H

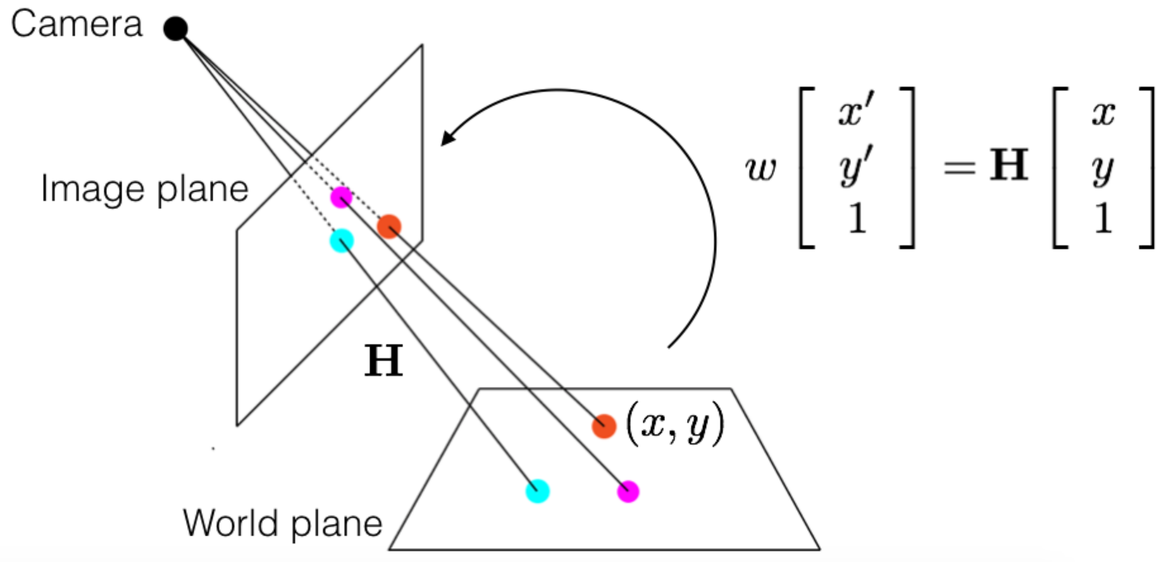


Homography & image mosaicing

- Every photo taken from a tripod-mounted camera is related by a homography
- Assumptions
  - No lens distortion
  - Camera's center of projection does not move while camera is mounted on the tripod
- Problem
  - These homographies that relate photos taken from a tripod-mounted camera are *unknown*
    - We need to estimate them

Homography

- Generally speaking, points that lie on two planes are related via homography.

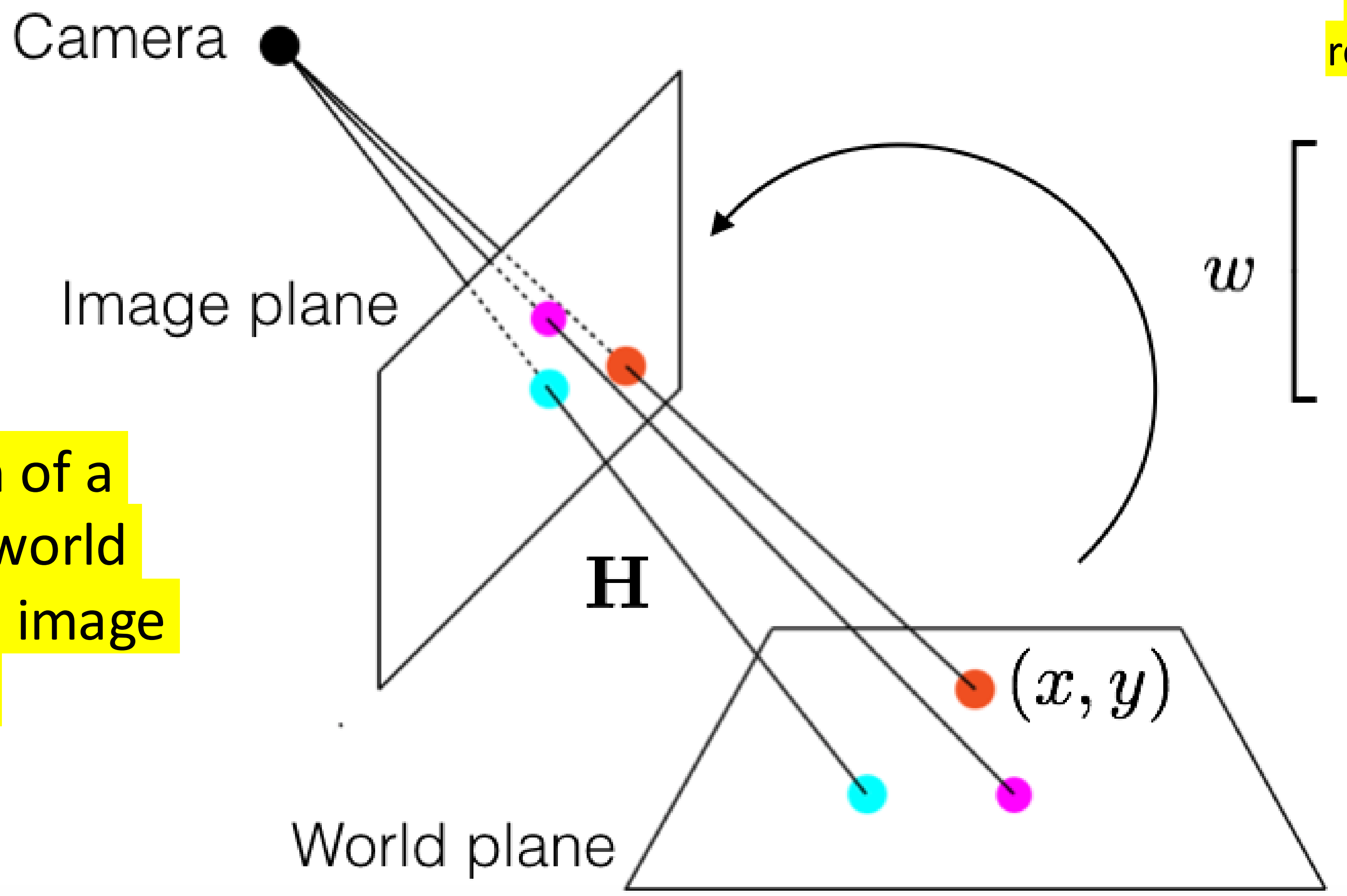


Homography

The location of the point and that of its projection are related via a Homography.

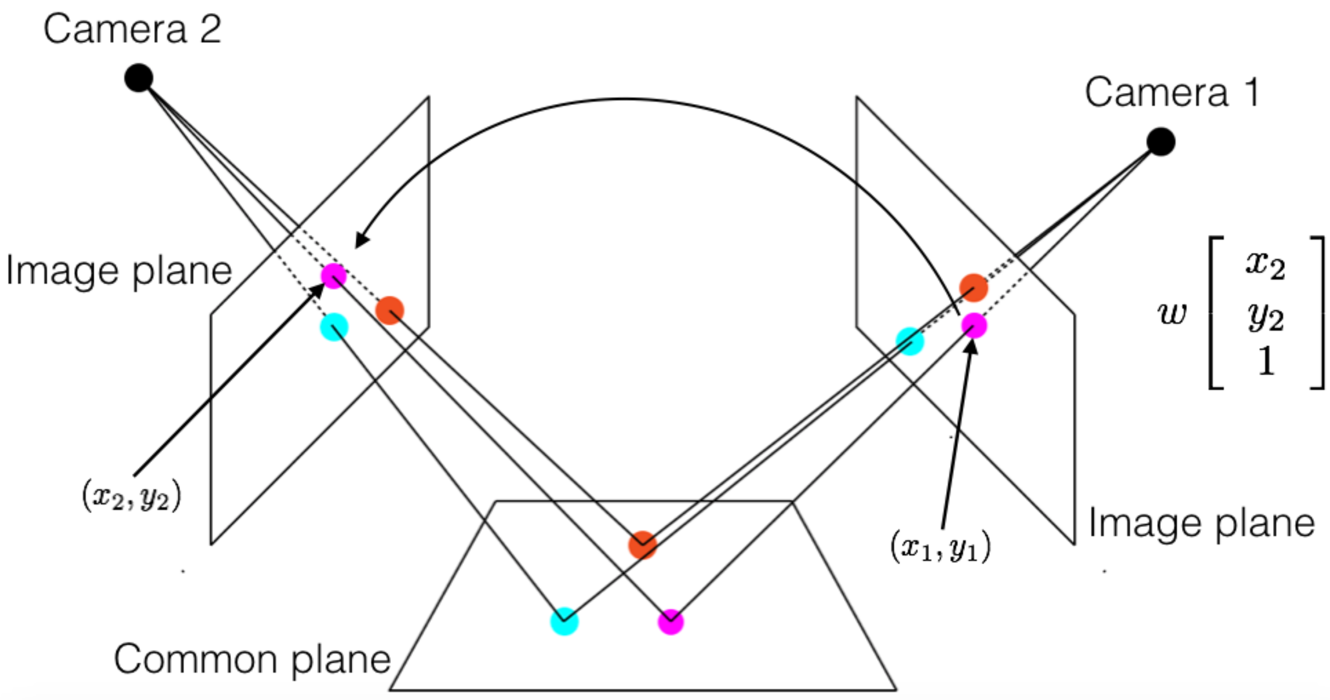
$$w \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \mathbf{H} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Projection of a point in a world plane to the image plane

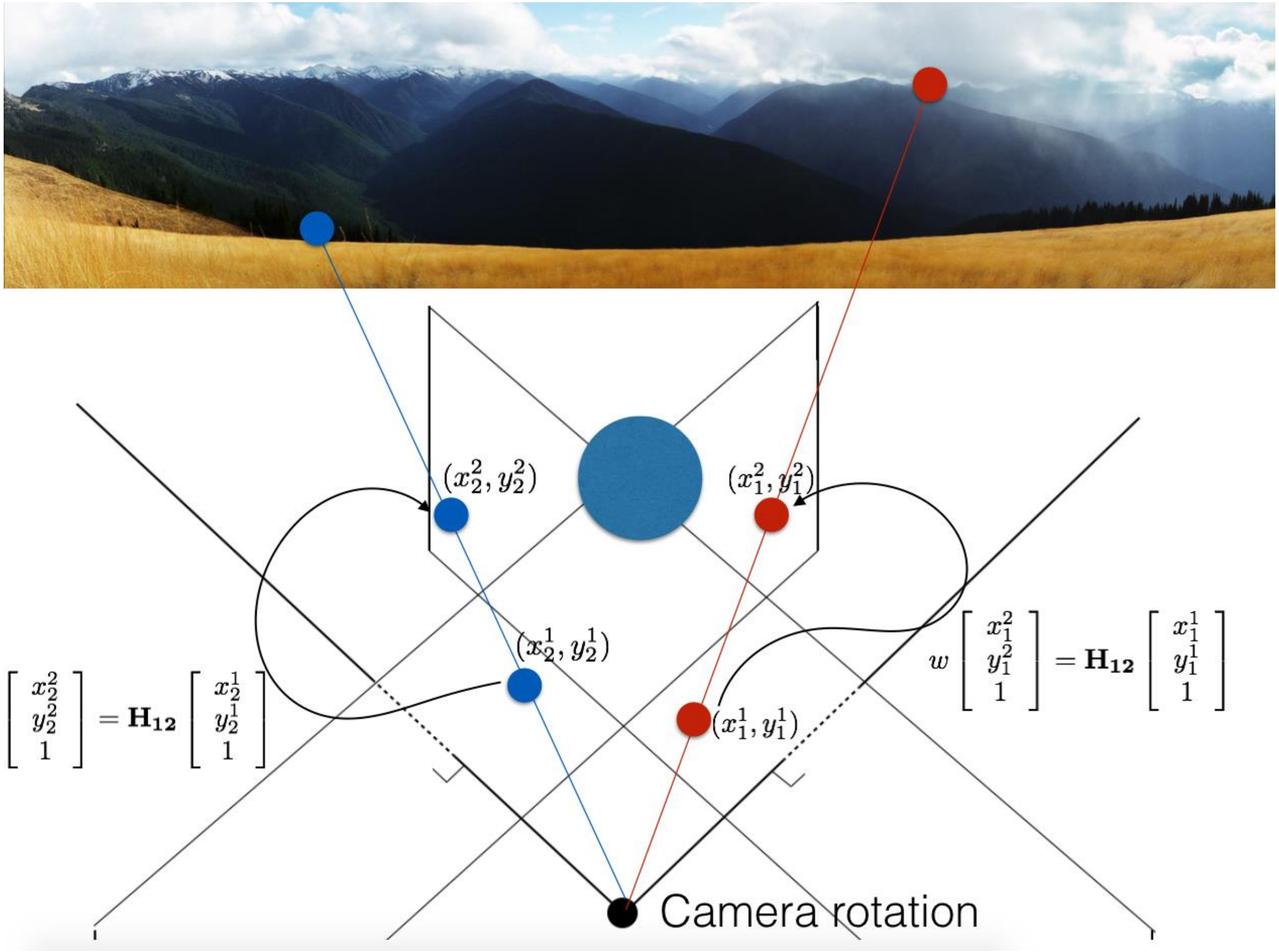


Homography

- Generally speaking, points that lie on two planes are related via homography.
  - This also means that the projections of points (that lie on a common plane) in two cameras are related via homography.

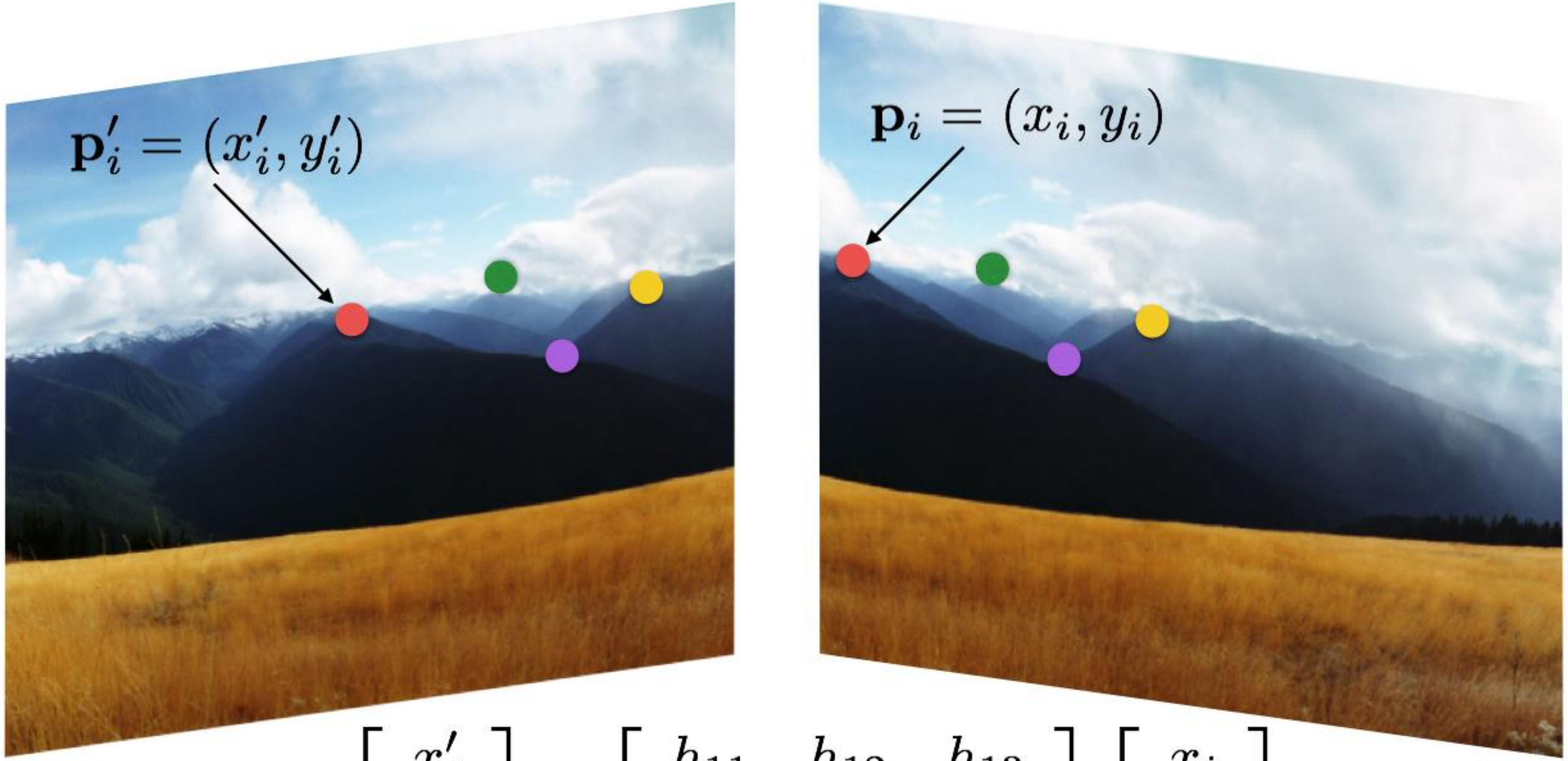


$$w \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \mathbf{H_{12}} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$



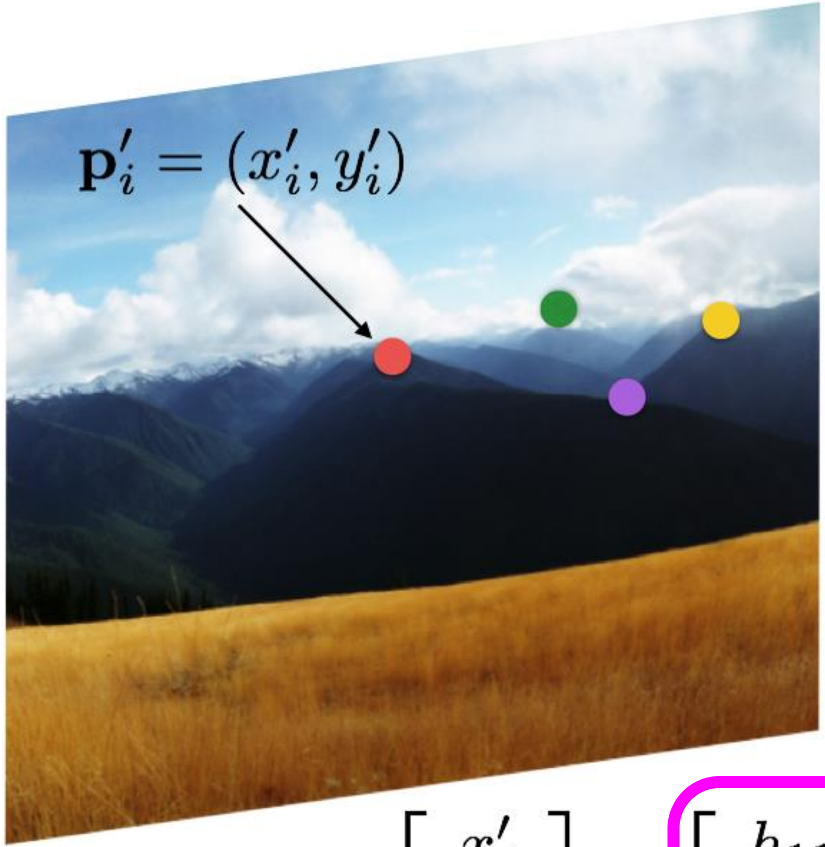
(x_2^2, y_2^2)
(x_1^2, y_1^2)
(x_2^1, y_2^1)
(x_1^1, y_1^1)
$\begin{bmatrix} x_2^2 \\ y_2^2 \\ 1 \end{bmatrix} = \mathbf{H_{12}} \begin{bmatrix} x_2^1 \\ y_2^1 \\ 1 \end{bmatrix}$
$w \begin{bmatrix} x_1^2 \\ y_1^2 \\ 1 \end{bmatrix} = \mathbf{H_{12}} \begin{bmatrix} x_1^1 \\ y_1^1 \\ 1 \end{bmatrix}$
Camera rotation

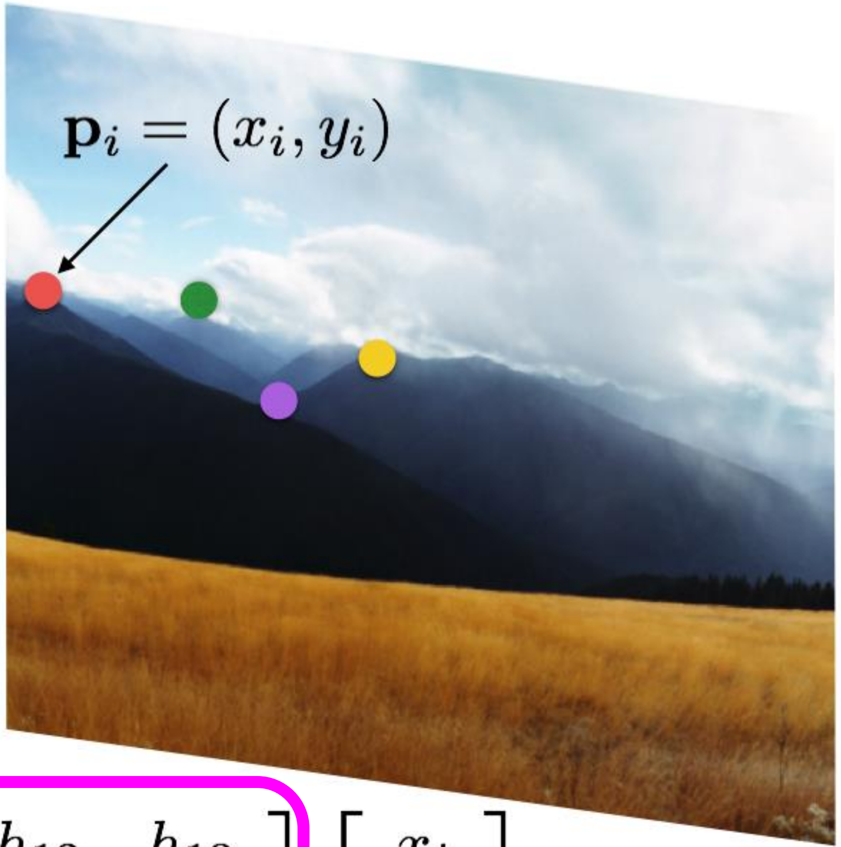
Image stitching



$$w \begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

Image stitching





$$w \begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

How do we solve H?

Solving homography

$$w\begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

How many degrees-of-freedom?

Solving for homography (Step 1)

- Re-write homography relationship as homogeneous equations

Solving for homography (Step 2)

- We can then write these as matrix-vector product

Solving for homography (Step 3)

- Given n correspondences between two images, setup $A\boldsymbol{x} = 0$ and solve for $\boldsymbol{x}$.

Solving $\boldsymbol{Ax} = 0$

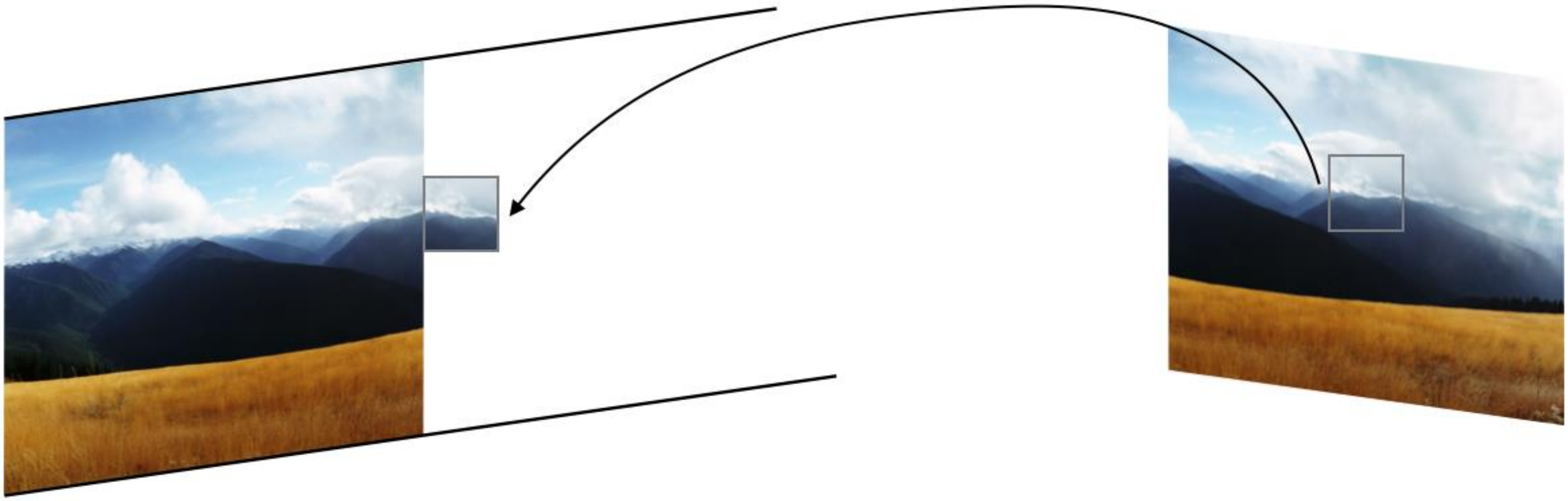
- Estimate using least-square fitting

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{\operatorname{argmax}} \; \|A\boldsymbol{x}\|^2 \quad \text{s.t.} \quad \|x\| = 1$$

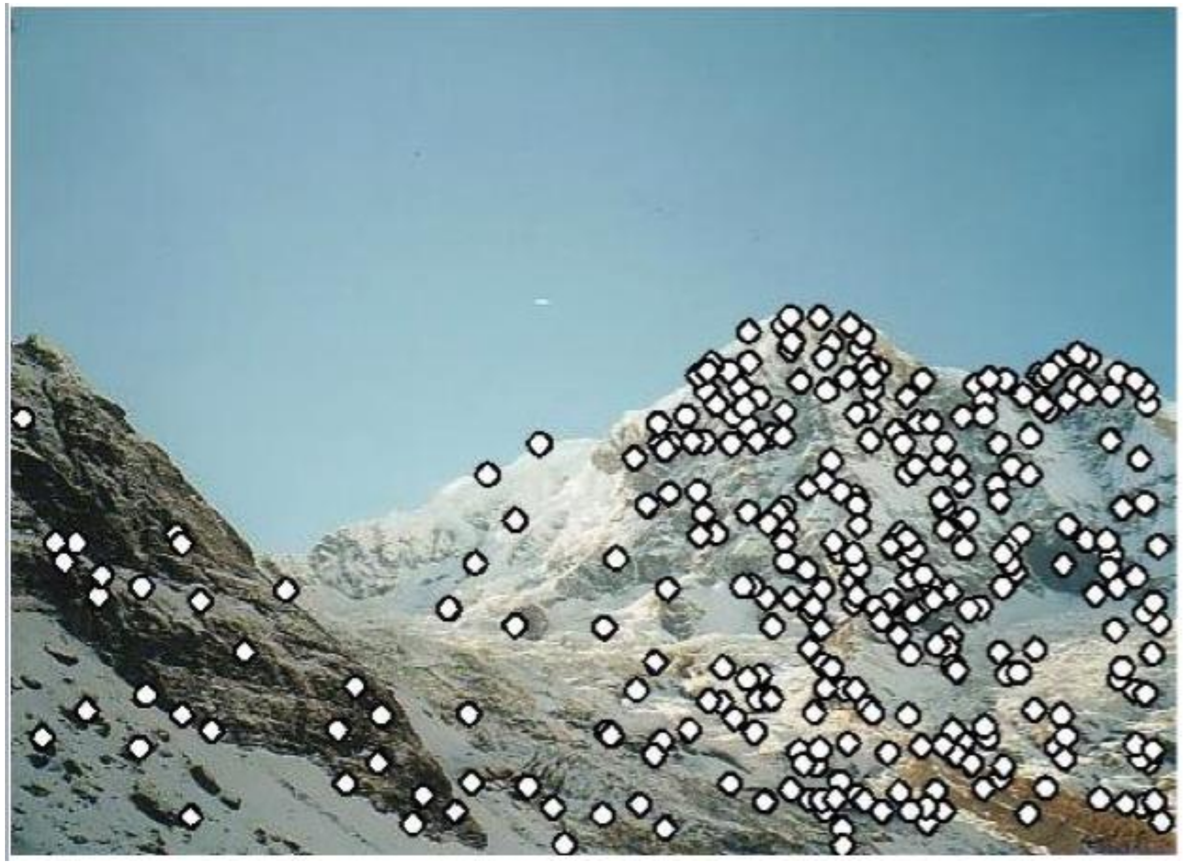
- The solution is the right *null-space* of A; therefore, the solution is the eigenvector corresponding to the smallest eigenvalue of $A^T A$

Image stitching

- Estimate homography
- Use it to fill the colors from the “other” image

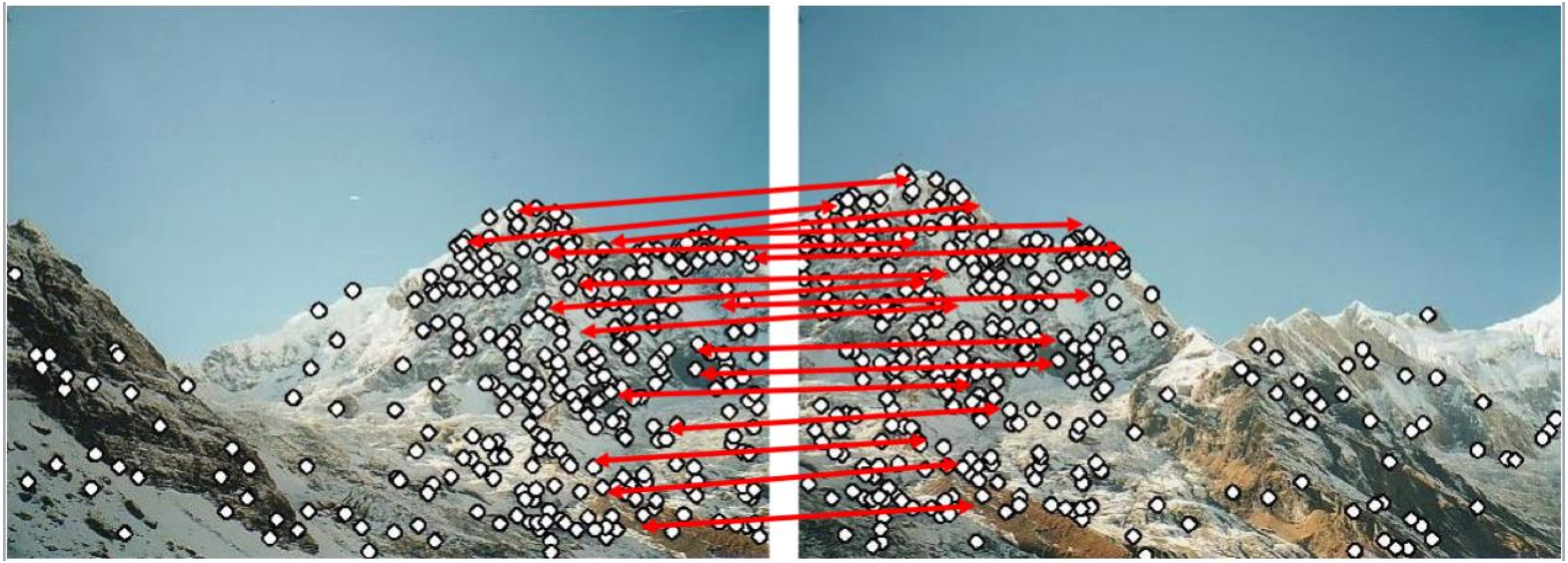


Extract features



Find matches

Use RANSAC to estimate homography



Perform image stitching