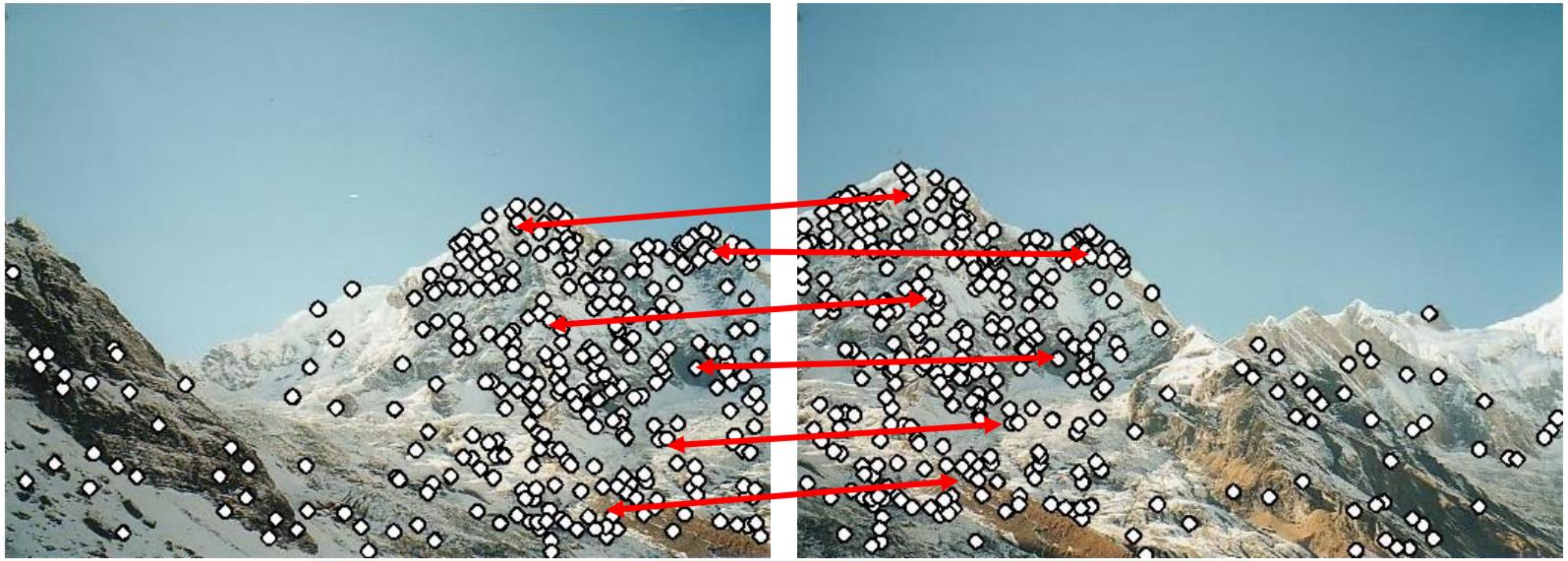
Projective Geometry

Computer Vision

Application: Panorama Stitching

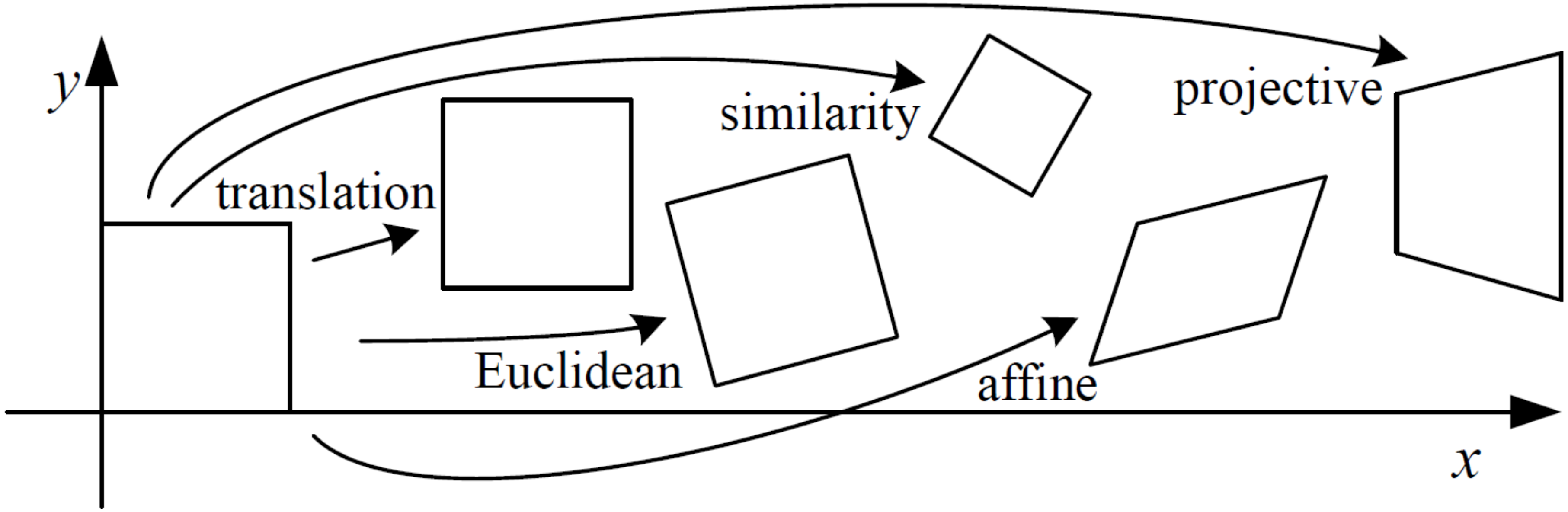
feature extraction
and matching



→ homography

Geometric Transformations

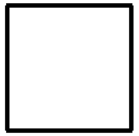
Geometric (Parametric) 2D Transformations



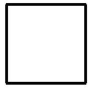
1. relocate individual pixels (without changing its intensity values)
2. calculate new intensity values through interpolation on neighborhood in original image

Linear Transformations

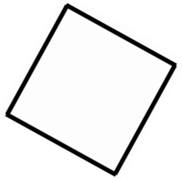
identity: $\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$



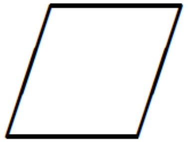
scaling: $\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$



rotation: $\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$



shear: $\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} 1 & s_h \\ s_v & 1 \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$



→ $\begin{pmatrix} x' \\ y' \end{pmatrix} = \underbrace{\begin{bmatrix} a & b \\ c & d \end{bmatrix}}_{\text{parameters}} \begin{pmatrix} x \\ y \end{pmatrix}$

→ can be chained: $\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} e & f \\ g & h \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$

Translation

$$x' = x + t_x$$
$$y' = y + t_y$$

not a linear operation → no matrix representation using 2D coordinates

homogeneous coordinates to the rescue: $\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$

represent 2D point with 3D vector

only defined up to scale: $\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} \stackrel{\text{def}}{=} \begin{pmatrix} wx' \\ wy' \\ w \end{pmatrix}$

Affine Transformations

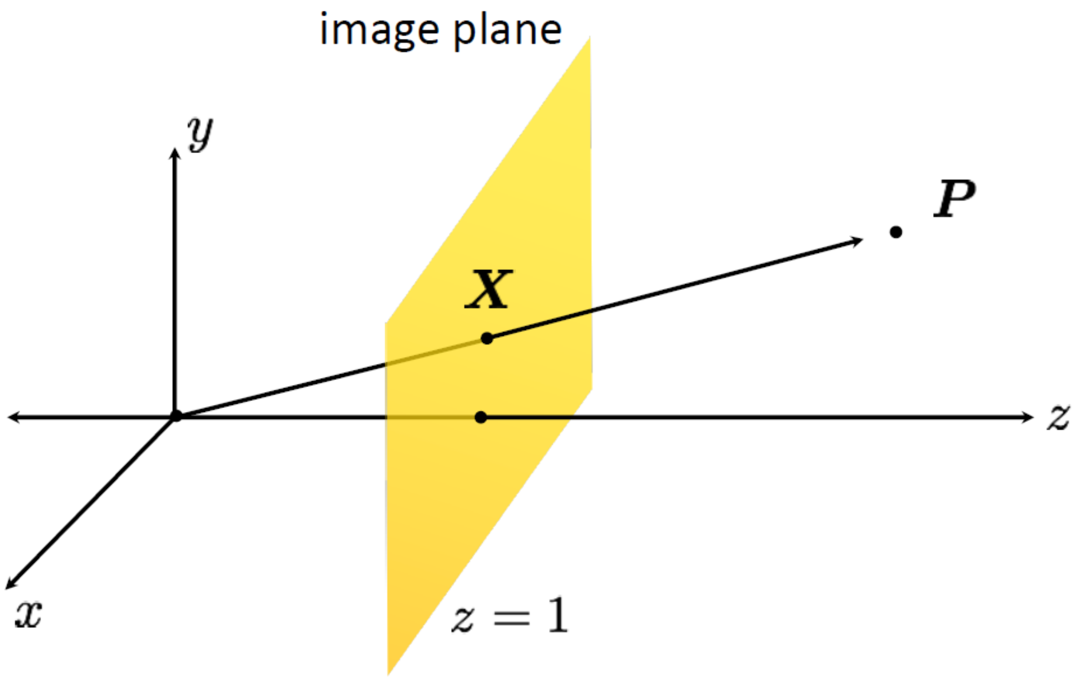
affine transformations:

linear transformations and translation

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Projective Geometry

projective space: 3D scene → 2D image

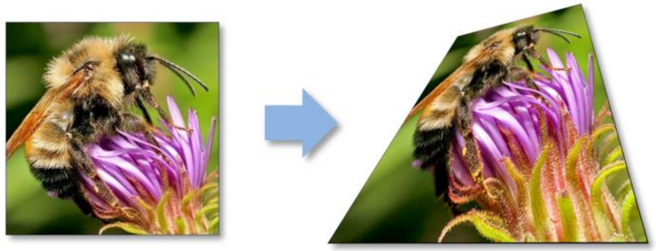


$\boldsymbol{X}$ is projection of point $\boldsymbol{P}$ on image plane

with homogeneous coordinates $\begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$

Projective Transformations: Homographies

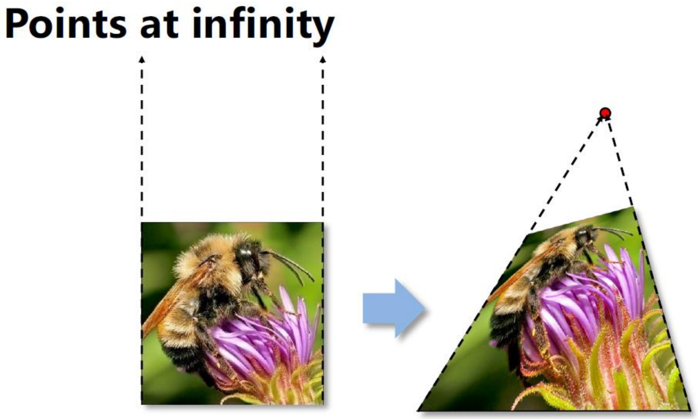
$$\begin{pmatrix} x' \\ y' \\ w' \end{pmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$



homographies: affine transformations and “projective warps”

special points:

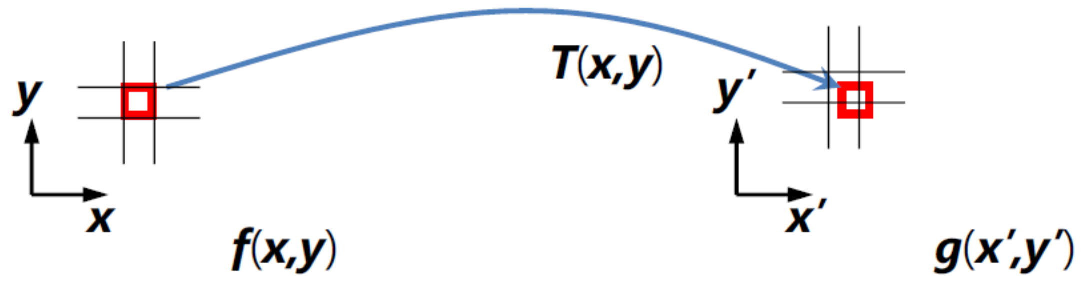
- point at infinity: $\begin{pmatrix} x \\ y \\ 0 \end{pmatrix}$
- undefined: $\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$



After Transformation: Intensity Interpolation

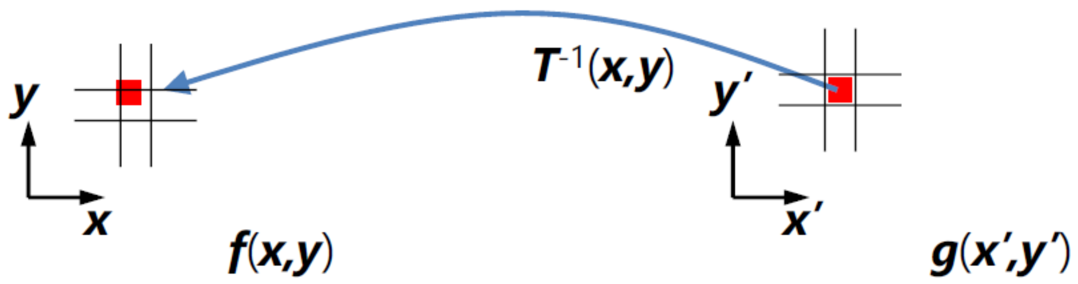
two possibilities:

forward mapping:
send each source pixel to its target location



each pixel in original image can contribute to more than one pixel in transformed image
→ can result in holes

inverse mapping:
get each target pixel from its source location



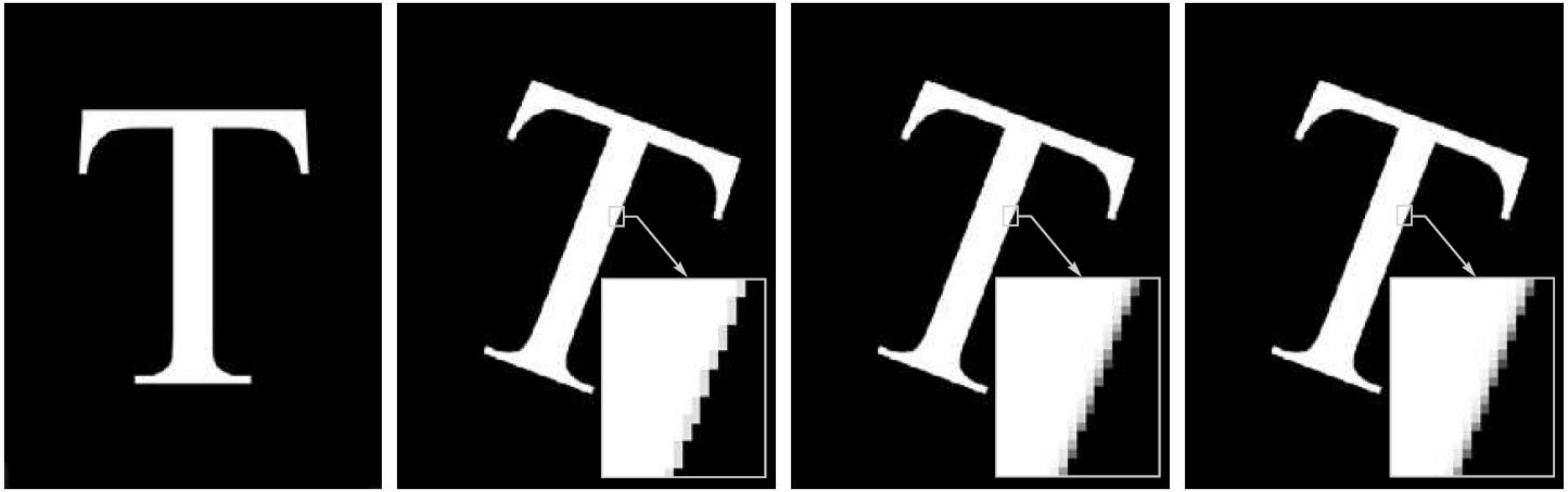
each pixel in transformed image can come from more than one pixel in original image
→ resampling from interpolated (filtered) original image

Example: Rotation

nearest-neighbor interpolation:

bilinear interpolation:

bicubic interpolation:



piecewise-polynomial function, aka spline

$v(x, y) = ax + by + cxy + d$
using 4 neighbors

$v(x, y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^j$
using 16 neighbors

Example: Resizing

scaling means changing image size in terms of number of pixels

(Do not confuse this with pixel size, which depends only on the used display or printer.)

- shrinking: sub- or downsampling (pixel deletion)
- zooming: upsampling (pixel replication)

original to be upsampled: 

Nearest-neighbor interpolation

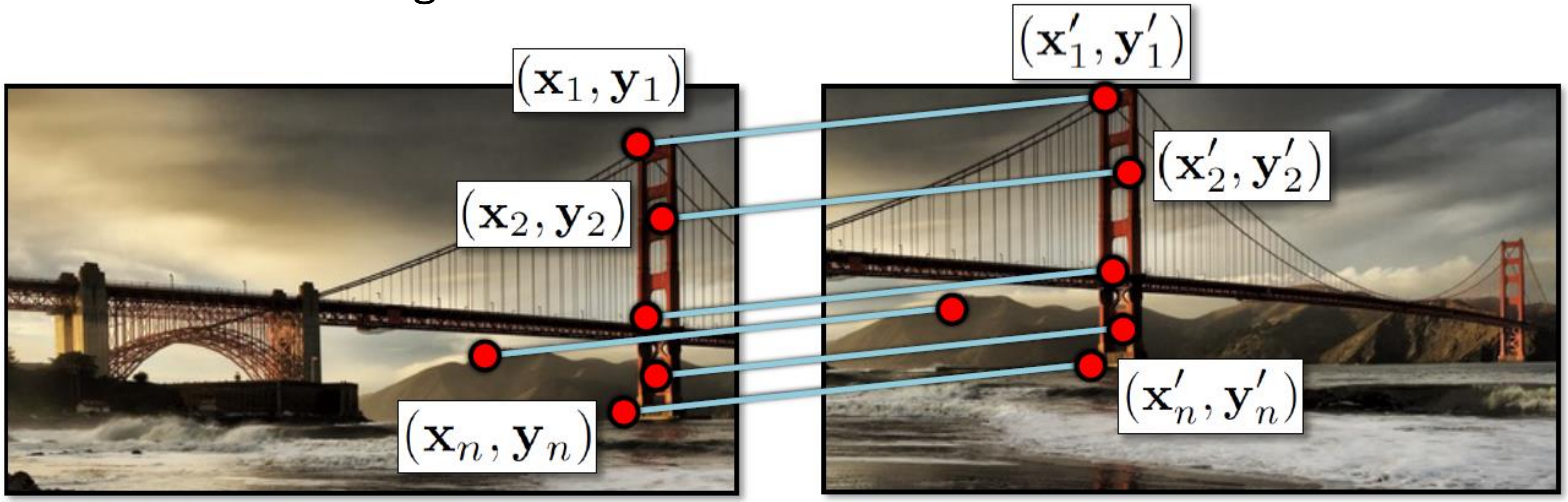
Bilinear interpolation

Bicubic interpolation

Image Alignment

Image Registration

given a set of matches ...



need to find parameters of geometric transformation (in general, homography):

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Fitting

2 equations per correspondence (x, y):

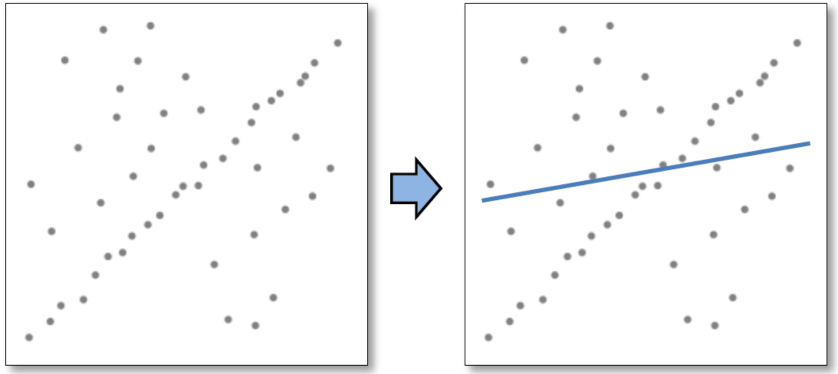
$$x' = \frac{ax+by+c}{gx+hy+i} \qquad y' = \frac{dx+ey+f}{gx+hy+i}$$

in general, non-linear
→ numerical (iterative) optimization

least squares method: minimize sum of squared residuals (cost function)

$$\sum_{j=1}^{n}\left(\left(\frac{ax_j+by_j+c}{gx_j+hy_j+i}-x'_j\right)^2+\left(\frac{dx_j+ey_j+f}{gx_j+hy_j+i}-y'_j\right)^2\right)$$

problem: typically, there are several outliers (wrong matches)



Problem: Fit a line to these datapoints   Least squares fit

Random Sample Consensus (RANSAC)

alignment requirement: robustness against matched outliers

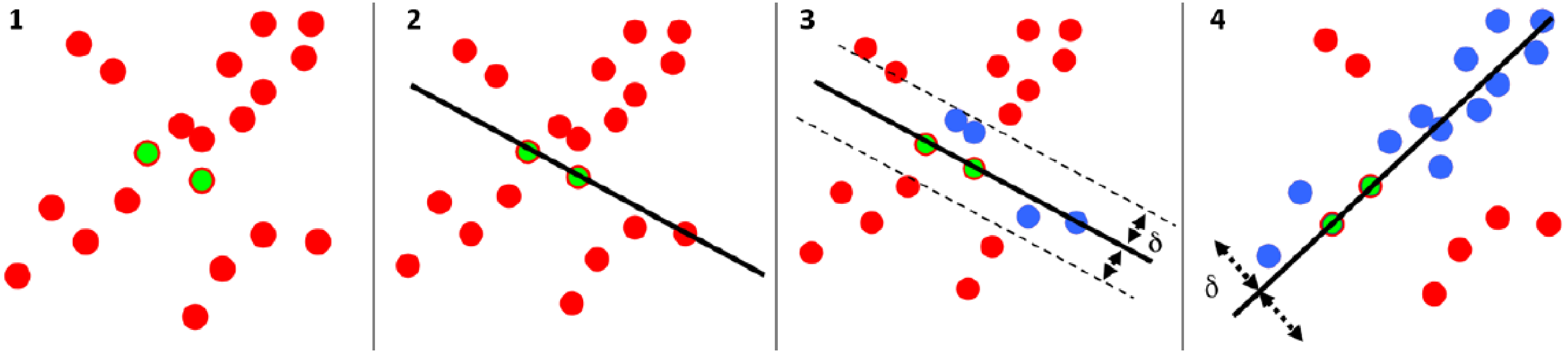
RANSAC approach:

randomly choose as many samples (i.e., matches) as needed for fit

fit a model to these samples (e.g., linear regression)

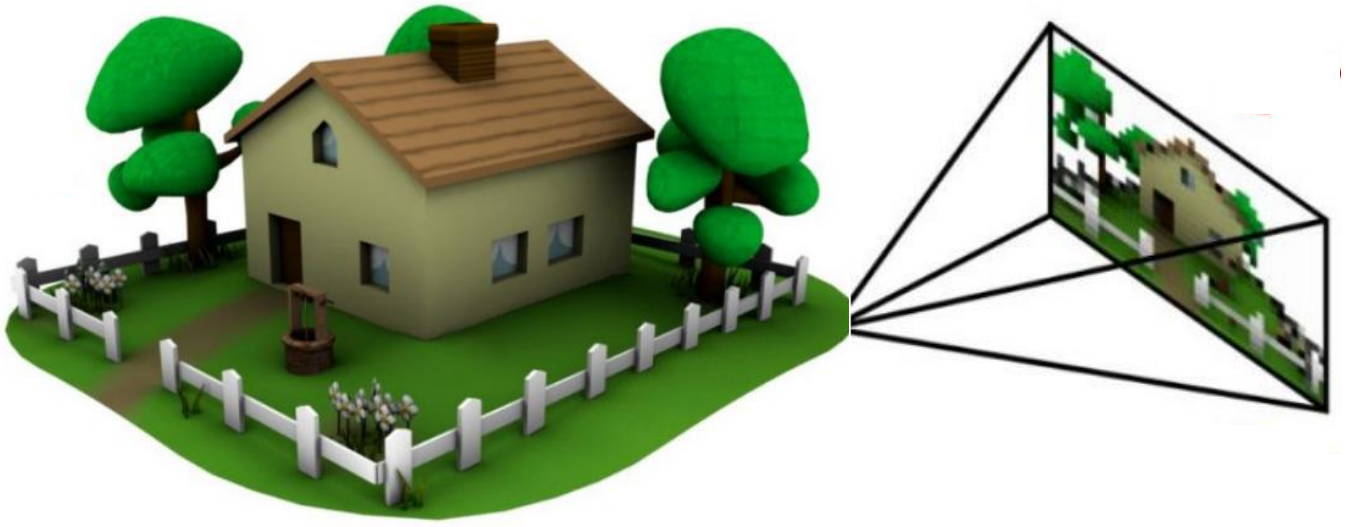
count number of inliers approximately fitting this model

repeat and finally choose the model with most inliers



Stereo Vision

Depth is lost in imaging process.



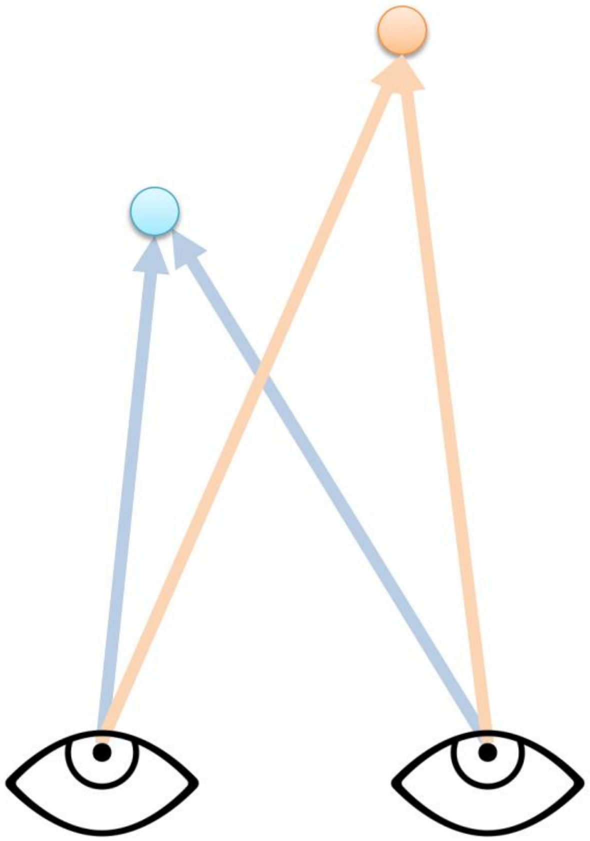
So, how do we get it back?

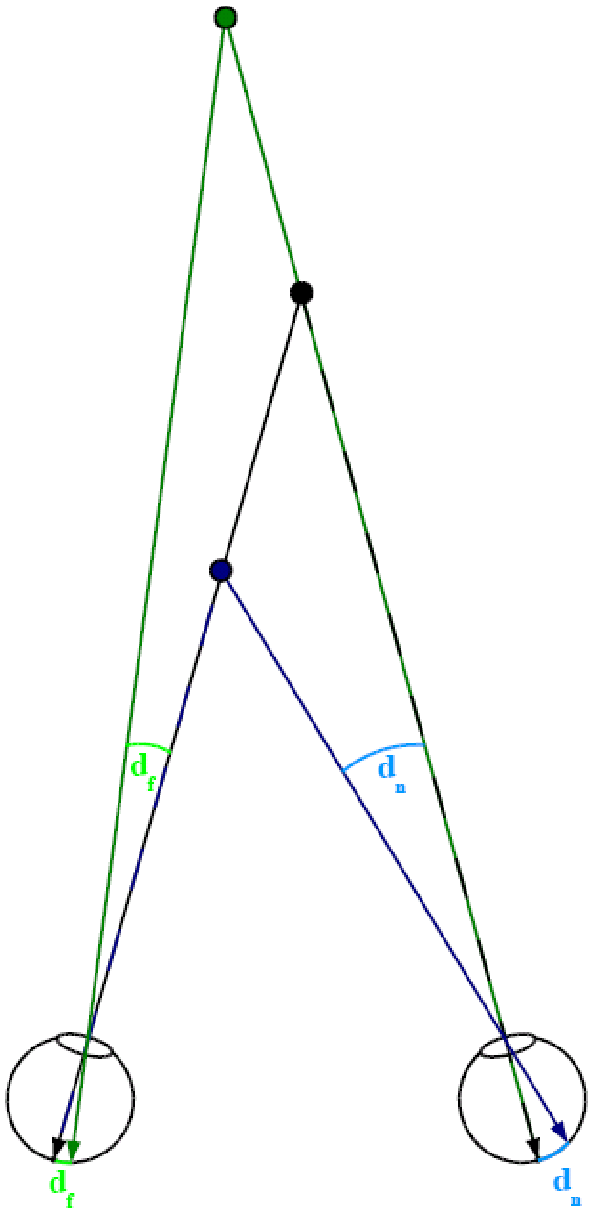
Binocular Disparity

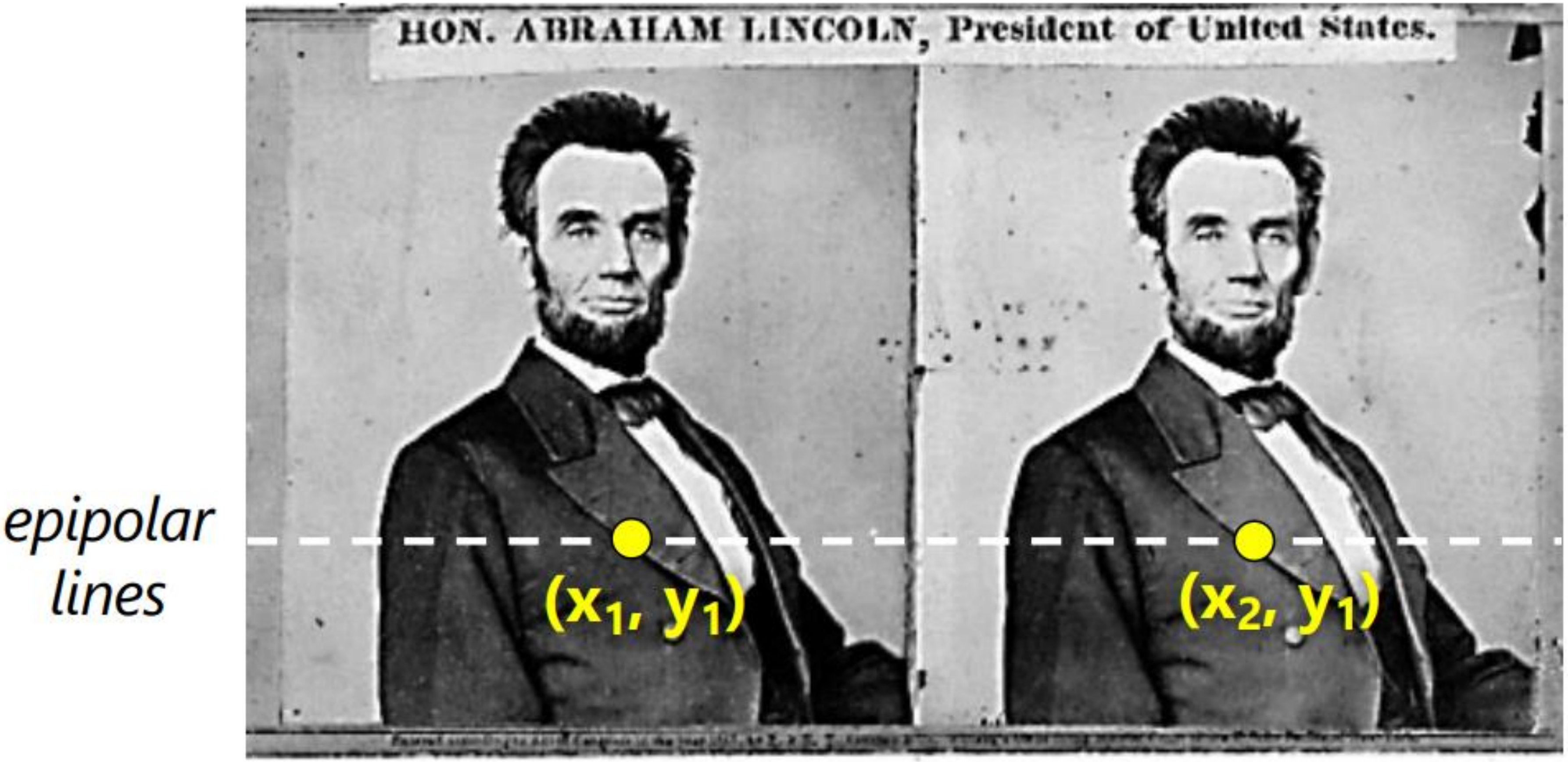
almost all animals have (at least) two eyes

stereo vision:

- (at least) two cameras in different views
- with matched correspondences for the points for which the depth should be estimated (all pixels for a full depth map)





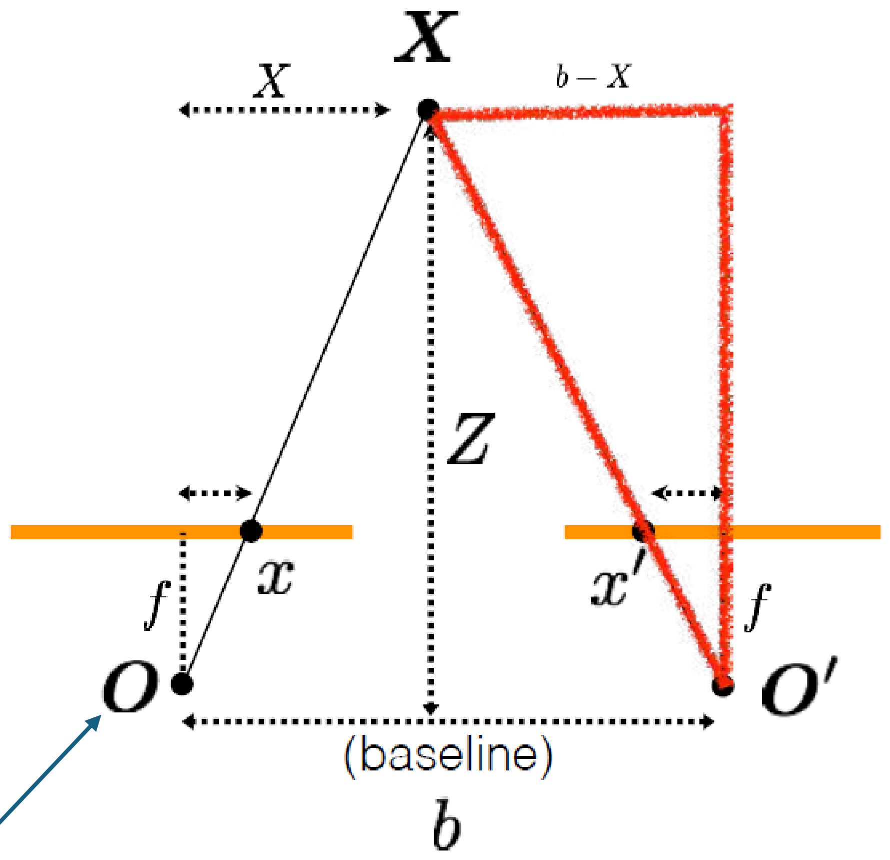


Two images captured by a purely horizontal translating camera
(*rectified* stereo pair)

$x_2 - x_1$ = the *disparity* of pixel (x_1, y_1)

Triangulation

$$\frac{X}{Z} = \frac{x}{f}$$



$$\frac{b - X}{Z} = \frac{x'}{f}$$

disparity:

(with respect to one camera origin in image plane
→ flip sign of x')

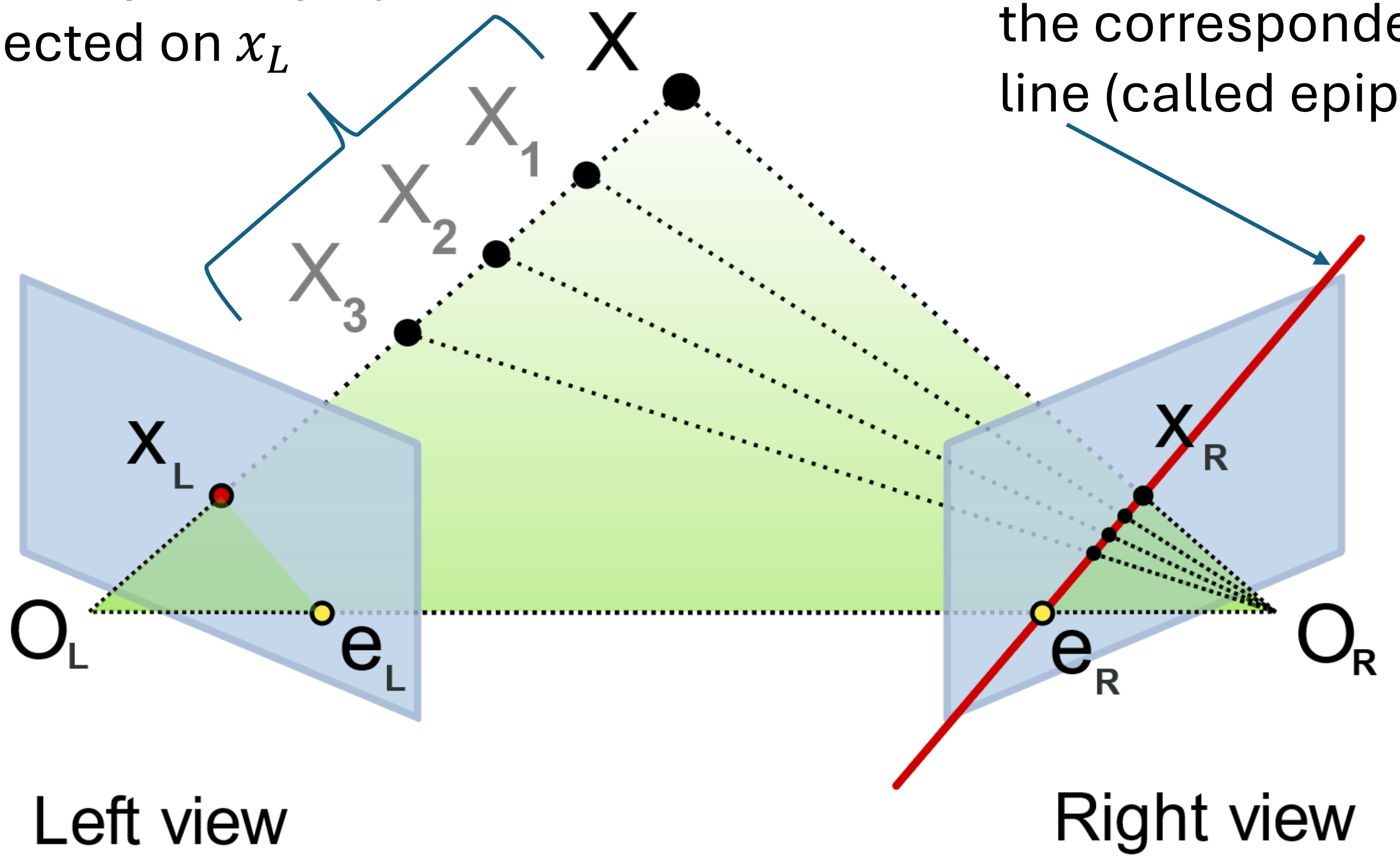
$$\boxed{d = x - x' = \frac{bf}{Z}}$$

→ inverse depth

Epipolar Geometry

all these 3D points projected
are projected on x_L

the correspondence x_R to x_L must lie on this
line (called epipolar line)



Find Best Match on Epipolar Line

for each pixel ...

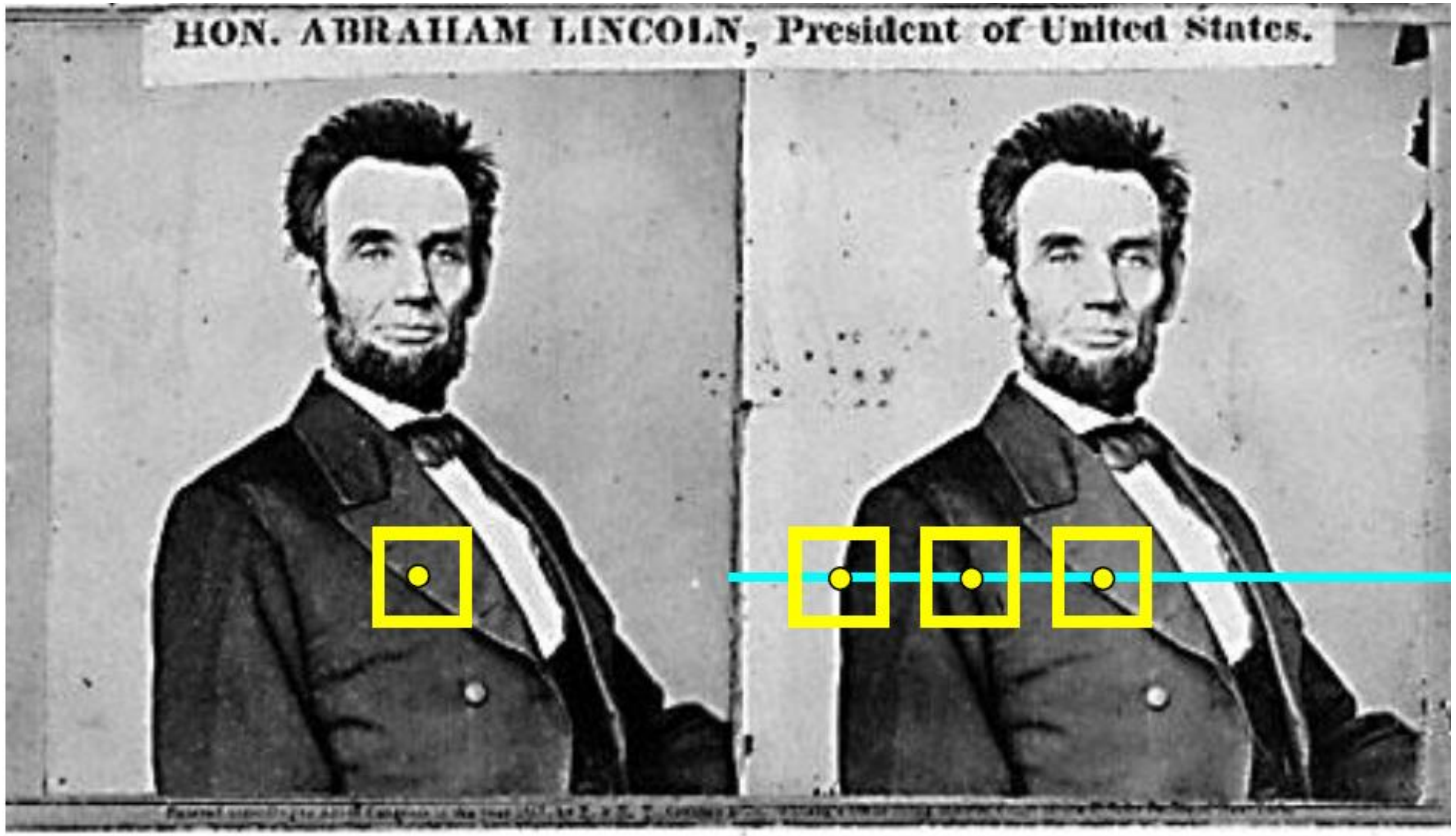
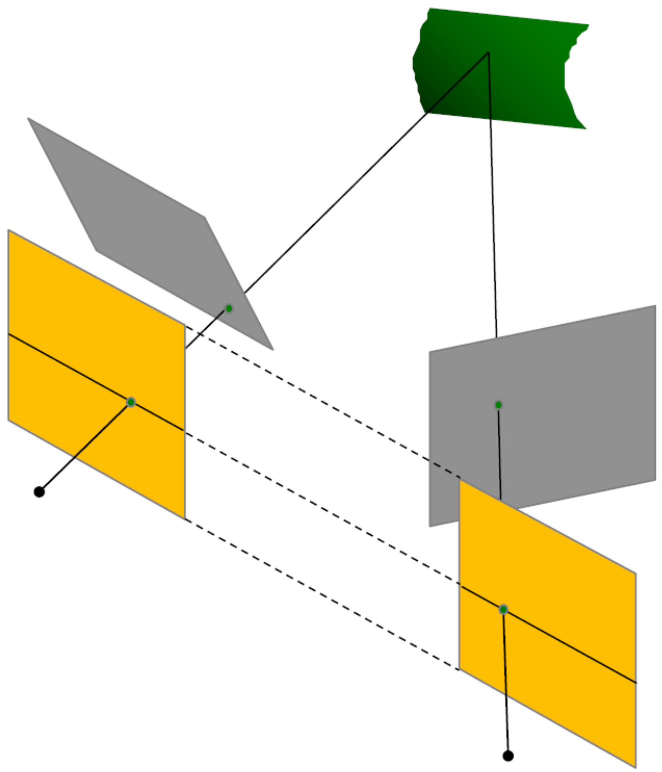


Image Rectification

epipolar lines become horizontal

homographies:



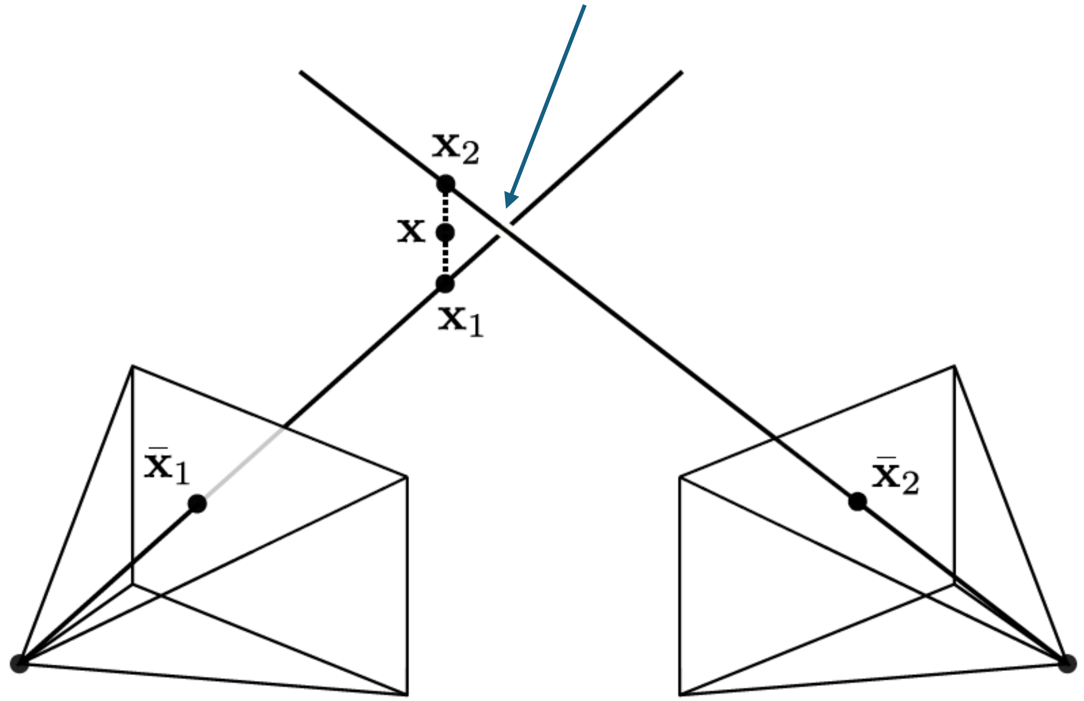
→ simplifies triangulation process

Basic Stereo Matching Pipeline

1. image rectification
2. for each pixel in image 1 find correspondence in image 2
3. calculate depth from disparity
4. create depth map

Issue: Noisy Images

might not intersect in one 3D point



→ optimization problem

Prerequisite for Triangulation

$$Z = \frac{bf}{d}$$

focal length (f), camera positions (for b), and camera coordinate systems (for d) need to be known

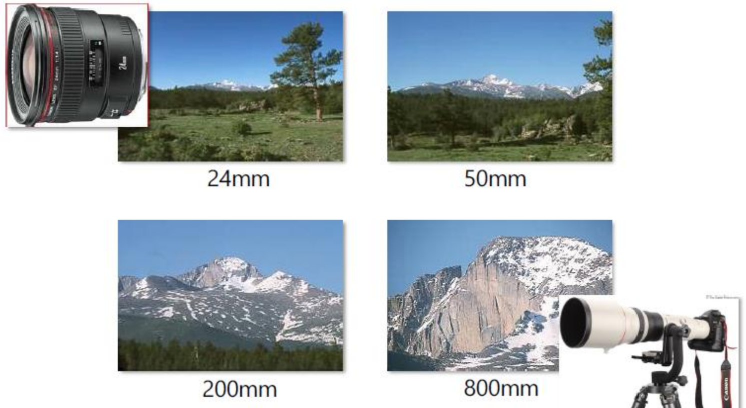
→ camera calibration

Geometric Camera Calibration

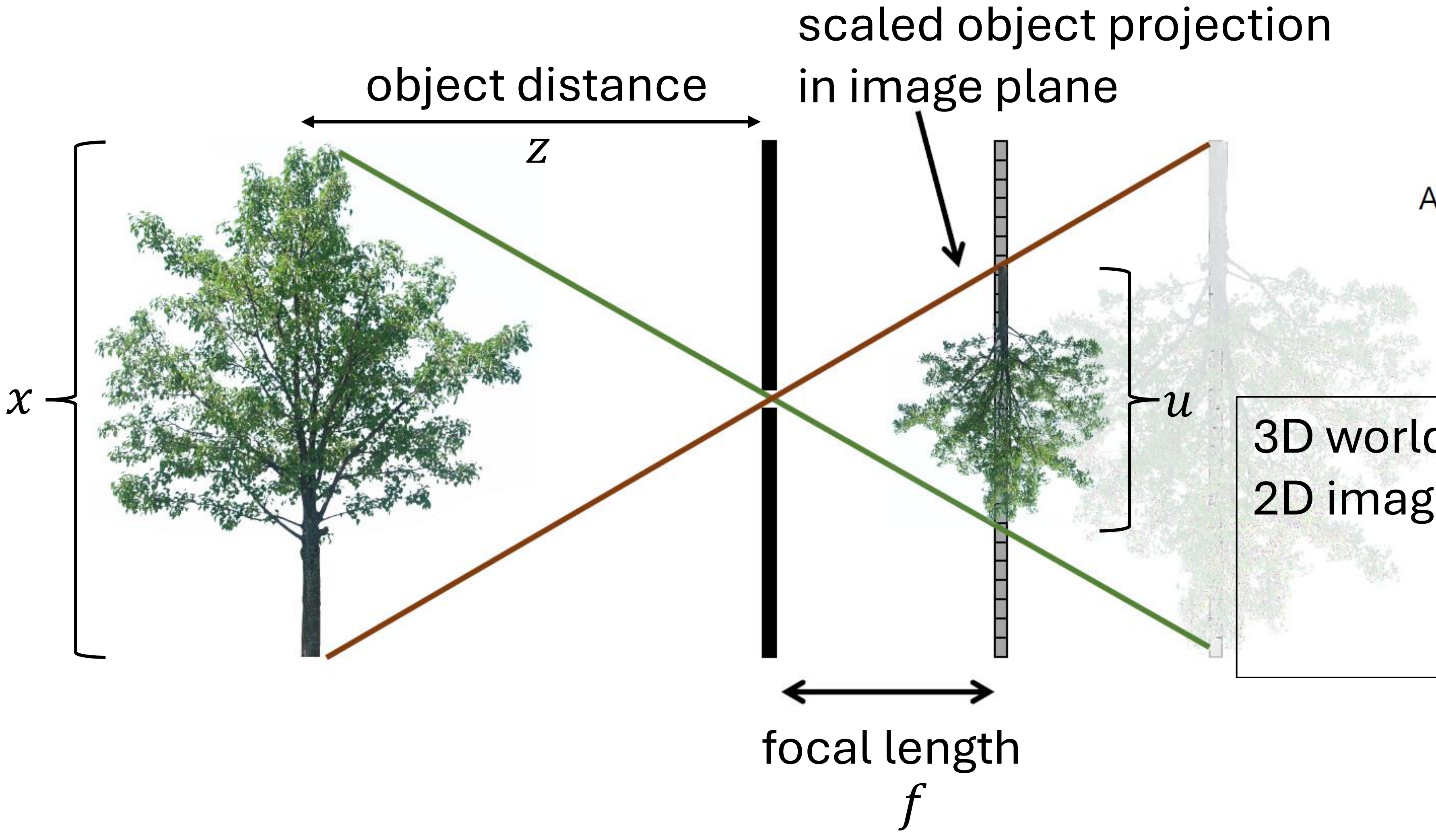
Pinhole Camera Model

focal length:

Can think of as "zoom"



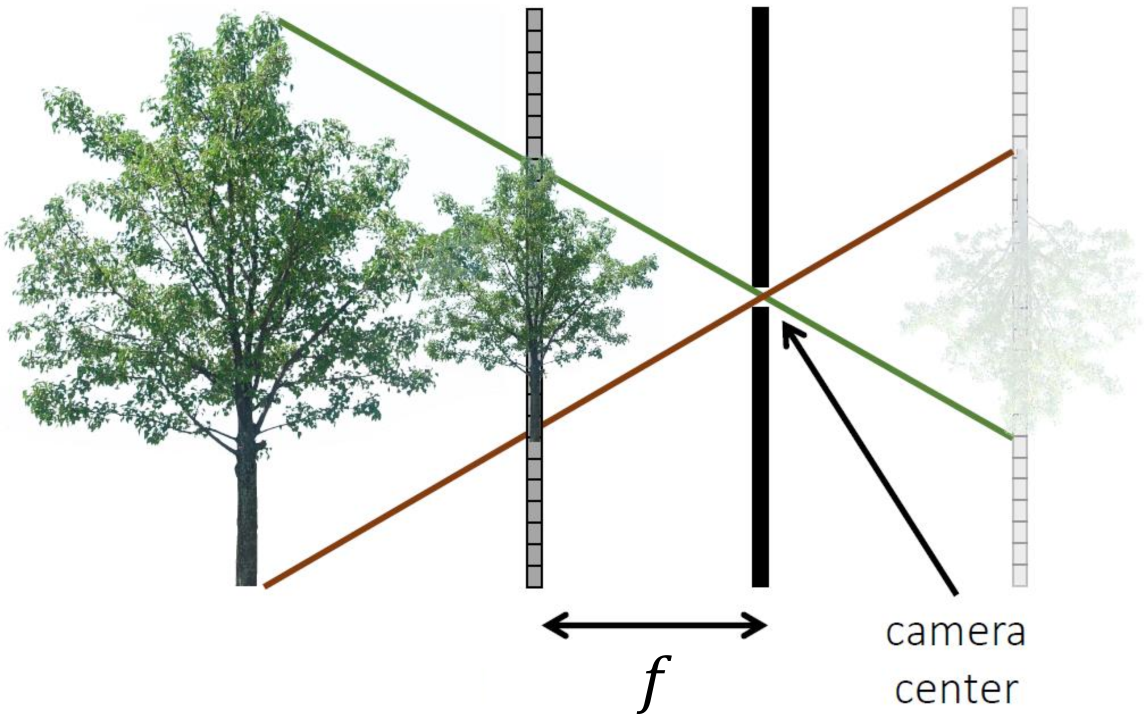
Also related to *field of view*



3D world coordinates (x, y, z) to
2D image coordinates (u, v):

$$\begin{pmatrix} u \\ v \end{pmatrix} = -\frac{f}{z}\begin{pmatrix} x \\ y \end{pmatrix}$$

Rearranged Pinhole Camera Setup



in homogeneous coordinates:

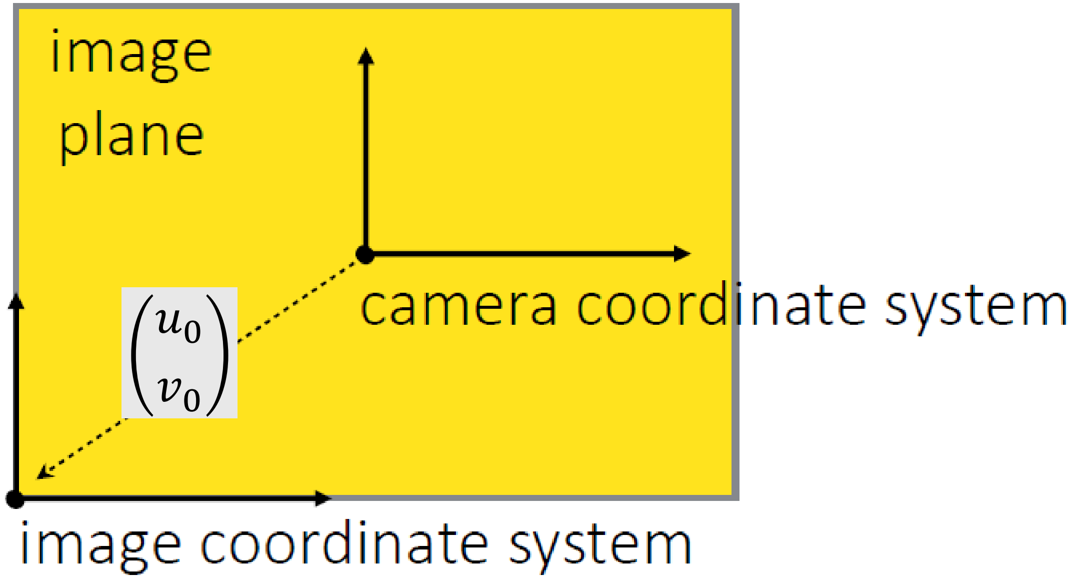
$$\begin{pmatrix} wu \\ wv \\ w \end{pmatrix} = \boldsymbol{C} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

with 3×4 projection matrix $\boldsymbol{C}$ (camera matrix)

$$\boldsymbol{C} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

in general 11 degrees of freedom

Generalized Camera Intrinsics



$$\rightarrow \boldsymbol{C} = \begin{bmatrix} f & 0 & u_0 & 0 \\ 0 & f & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

more potential generalizations:

- pixels may not be square (aspect ratio α)
- sensor may be skewed (γ)

$$\boldsymbol{C} = \begin{bmatrix} f & \gamma & u_0 & 0 \\ 0 & \alpha f & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Decomposition of Camera Matrix

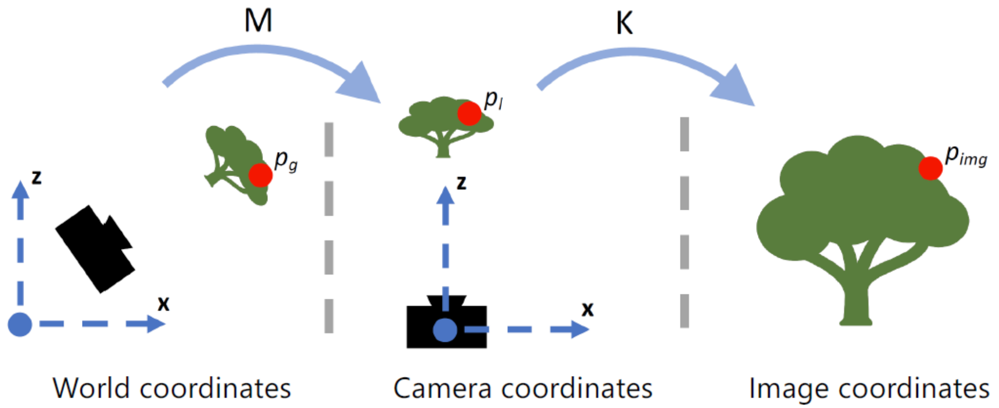
$$\boldsymbol{C} = \boldsymbol{K}[\boldsymbol{R}|\boldsymbol{t}]$$

$$\boldsymbol{C} = \underbrace{\begin{bmatrix} f & \gamma & u_0 \\ 0 & \alpha f & v_0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{intrinsic parameters}} \underbrace{\left[\begin{array}{ccc:c} r_1 & r_2 & r_3 & t_1 \\ r_4 & r_5 & r_6 & t_2 \\ r_7 & r_8 & r_9 & t_3 \end{array}\right]}_{\text{extrinsic parameters}}$$

intrinsic parameters
(camera-to-image transformation)

5 parameters

extrinsic parameters
(world-to-camera transformation)



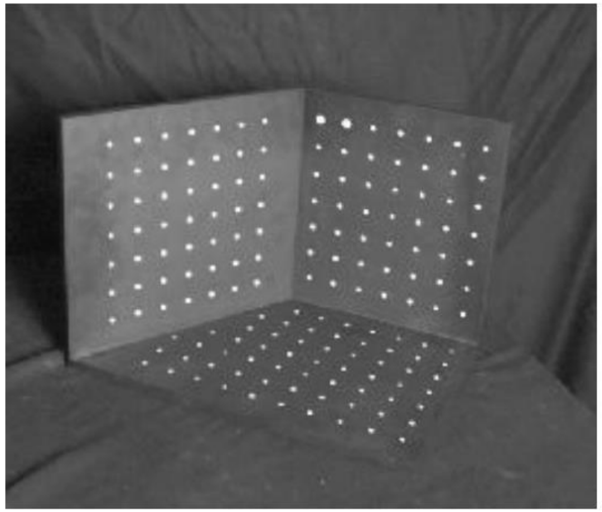
camera pose:

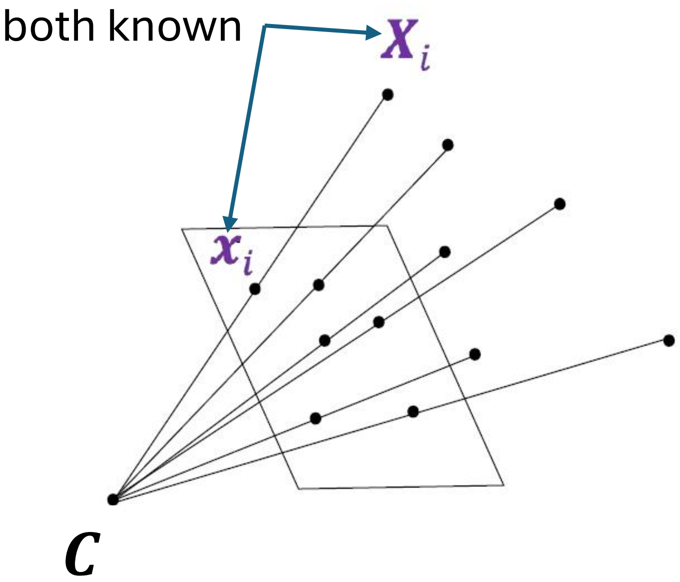
3D translation (origin of world in camera coordinates)
3 parameters

3D rotation (camera orientation)
3 parameters (rotation angles)

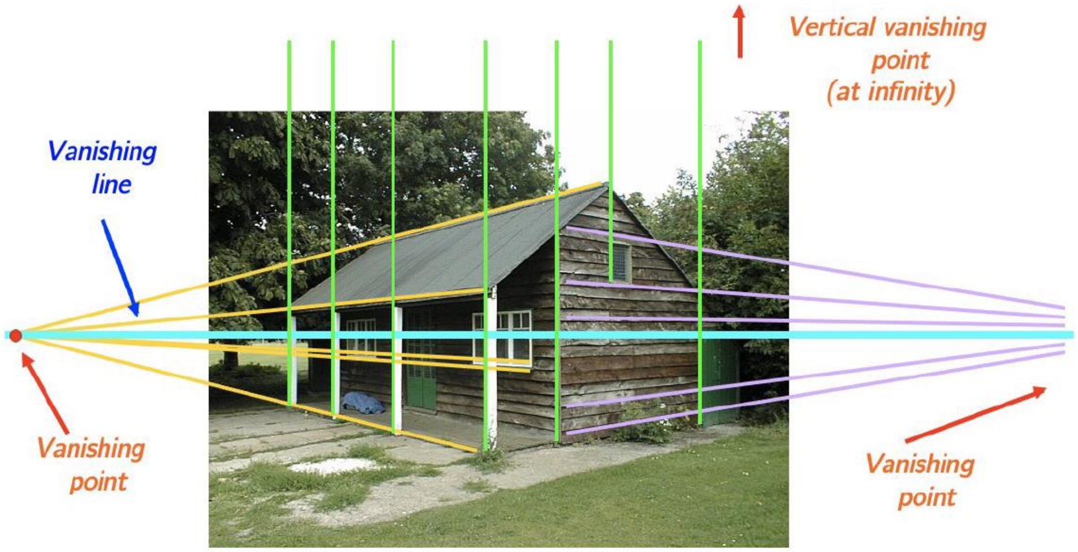
Estimation of Camera Parameters

typically via images of known calibration target (e.g., checkerboard)

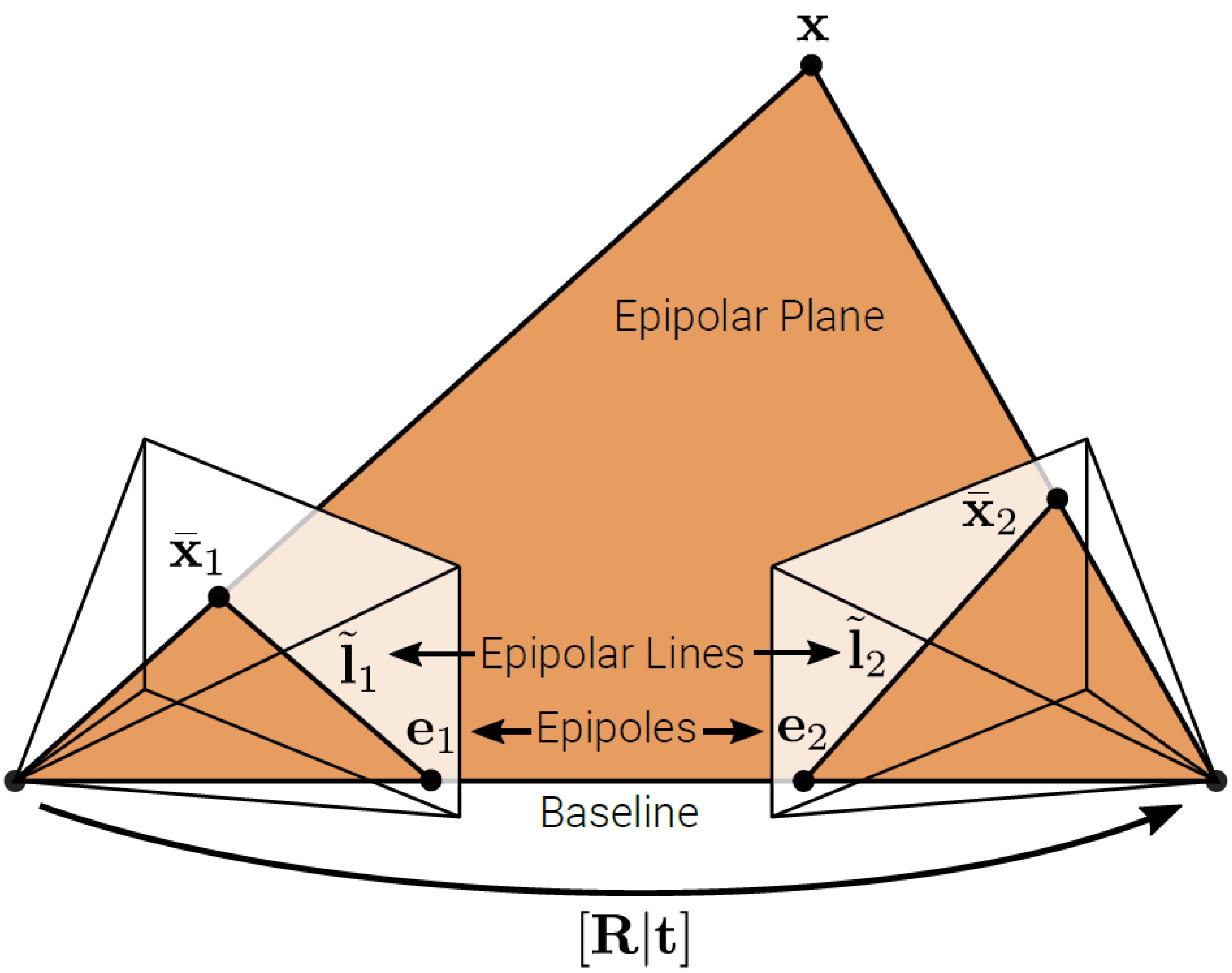




one alternative: use of vanishing points

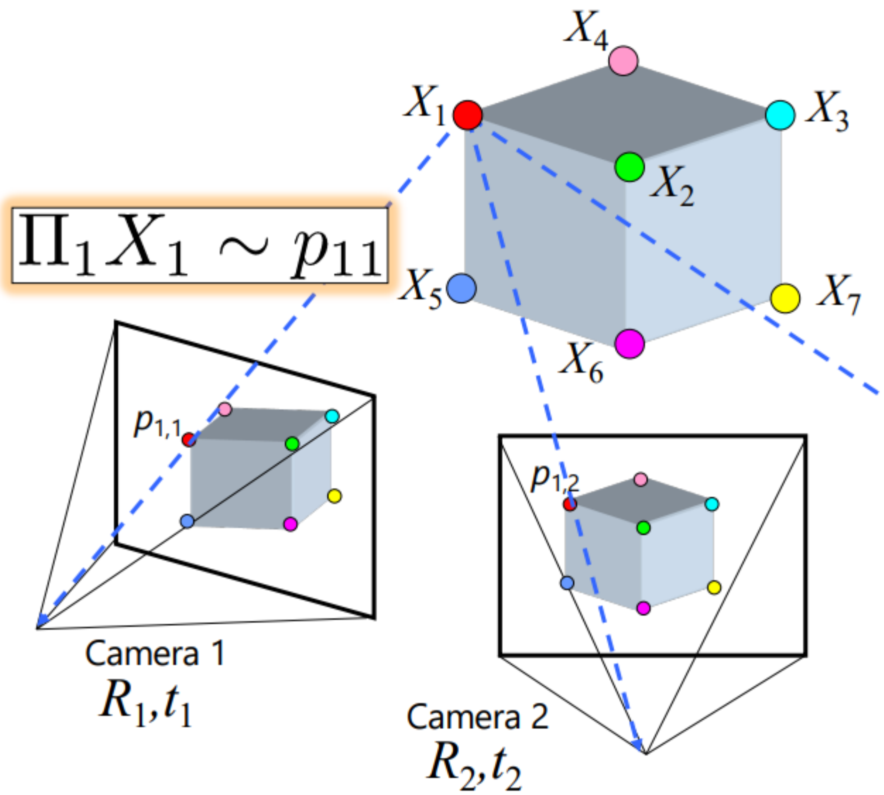


Pose Estimation = Inverse Triangulation



Structure from Motion

uncalibrated stereo: simultaneous calculation of both 3D locations and camera poses (typically using many images)



minimize reprojection error

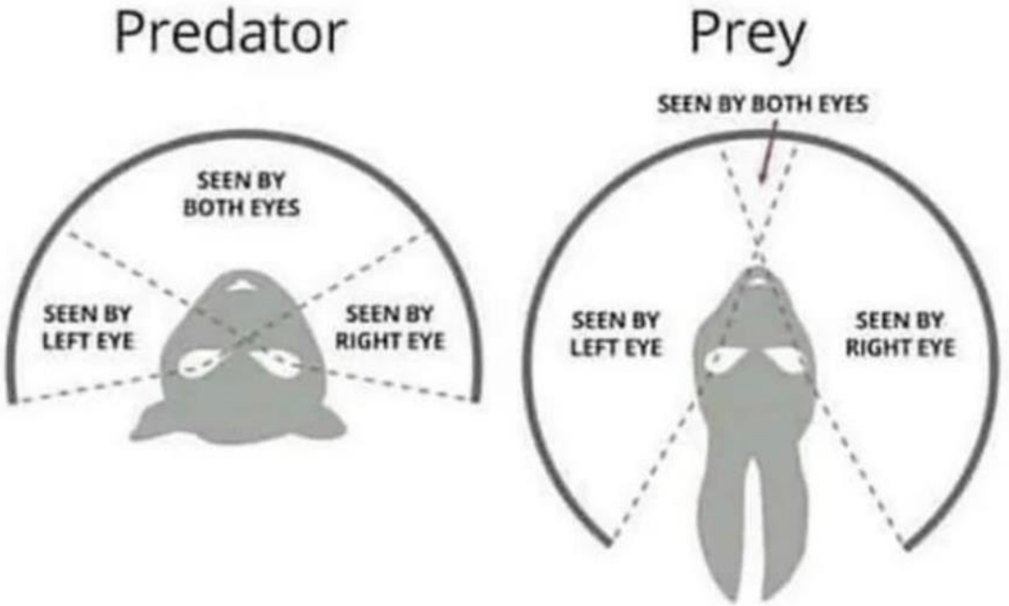
$$\mathbf{g}(\mathbf{R}, \mathbf{T}, \mathbf{X})$$

non-linear least squares

application: Visual SLAM
(camera poses from video)

Alternatives for Depth Estimation

Not All Animals See Stereo



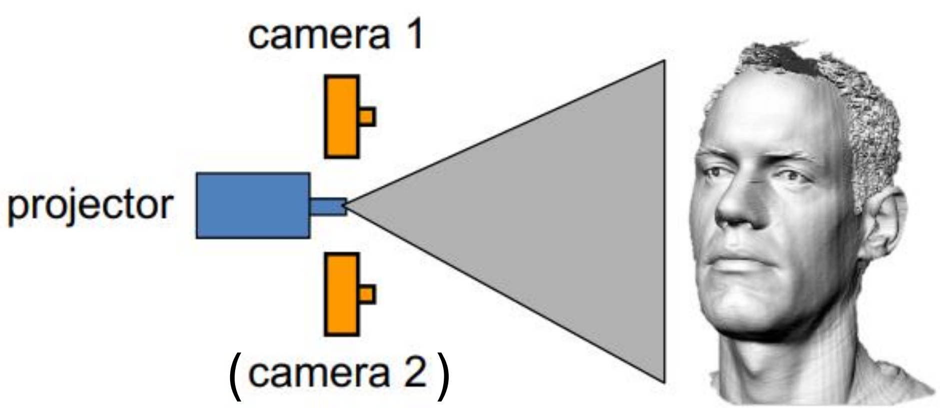
stereopsis: depth perception through binocular disparity

stereoblind: need to use motion parallax for depth perception (wide field of view to spot predators)

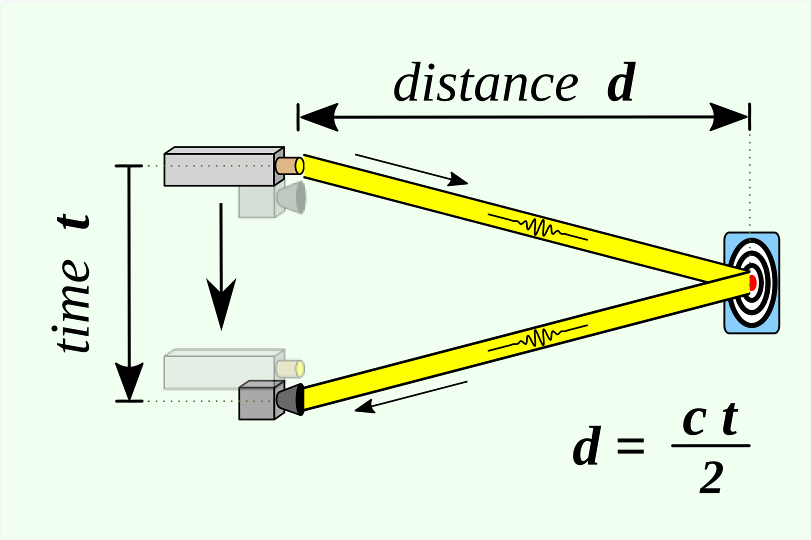
→ move the camera, find matching correspondences

Active Depth Sensors

structured light



time-of-light



e.g., Kinect RGBD cameras

Shape from X

- shape from shading and photometric stereo
- shape from texture
- shape from focus

humans:

using also perspective effects and familiar objects in scene