Lecture 3

Imaging Geometry

Spring 2021

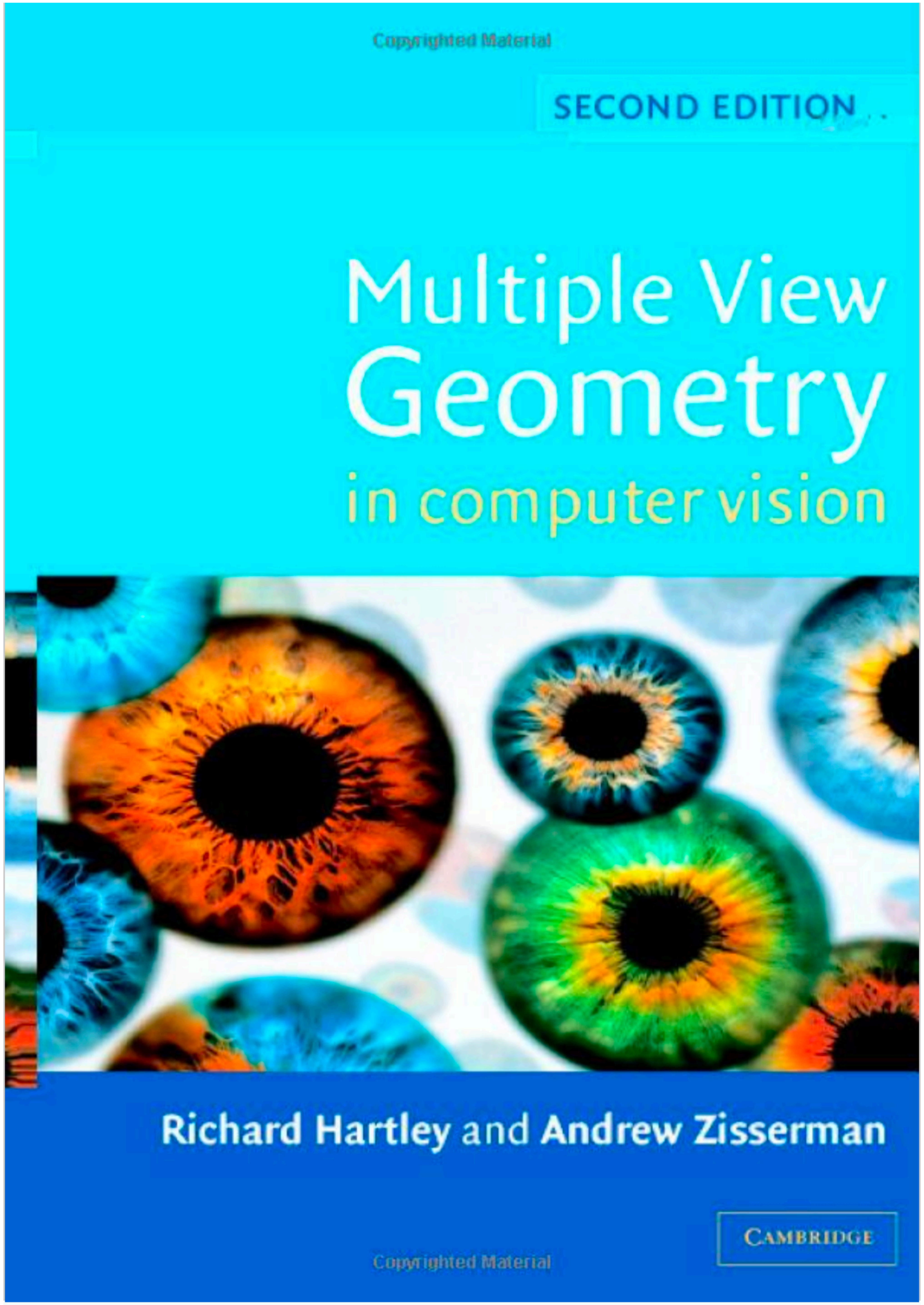
Bill Freeman, Phillip Isola

3. Imaging Geometry

- Applications
- Stereo and how it works
- Homogeneous coordinates for clean description of the geometry
- Intrinsic and extrinsic camera parameters
- Homographies for image stitching, for image rectification, etc.
- Ransac for fitting parameterized models such as homographies.

Relevant readings

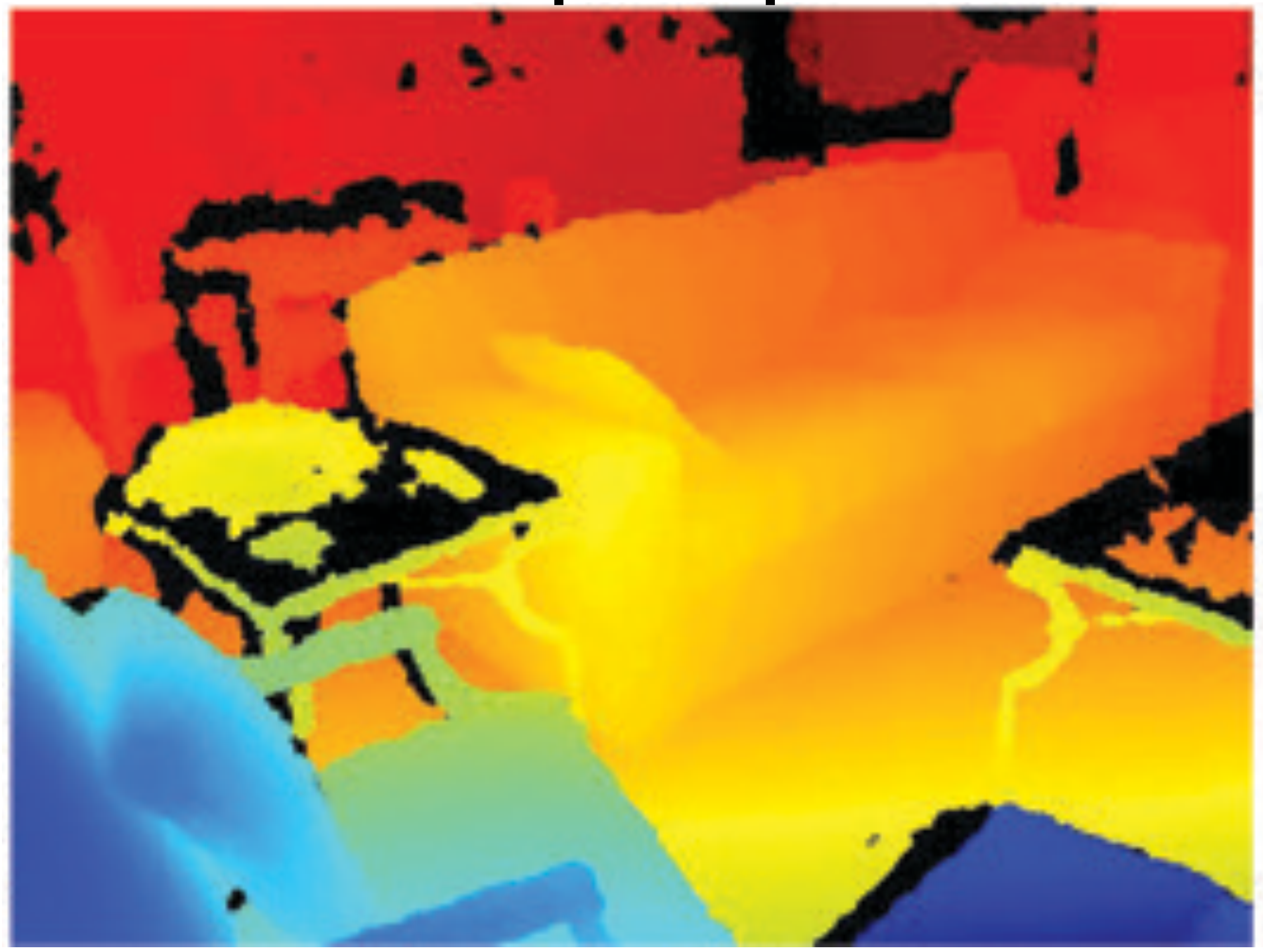
- Relationship between coordinates in the world and coordinates in the image: *geometric camera calibration, see* Szeliski, section 5.2, 5.3.
- (Relationship between intensities in the world and intensities in the image: *photometric image formation*, see Szeliski, sect. 2.2.)
- Class notes.


Copyrighted Material
SECOND EDITION
Multiple View
Geometry
in computer vision
Richard Hartley and Andrew Zisserman
CAMBRIDGE
Copyrighted Material

Devices for depth measurement

Active Stereo

Depth map



Lidar commonly used with self-driving cars.

Applications

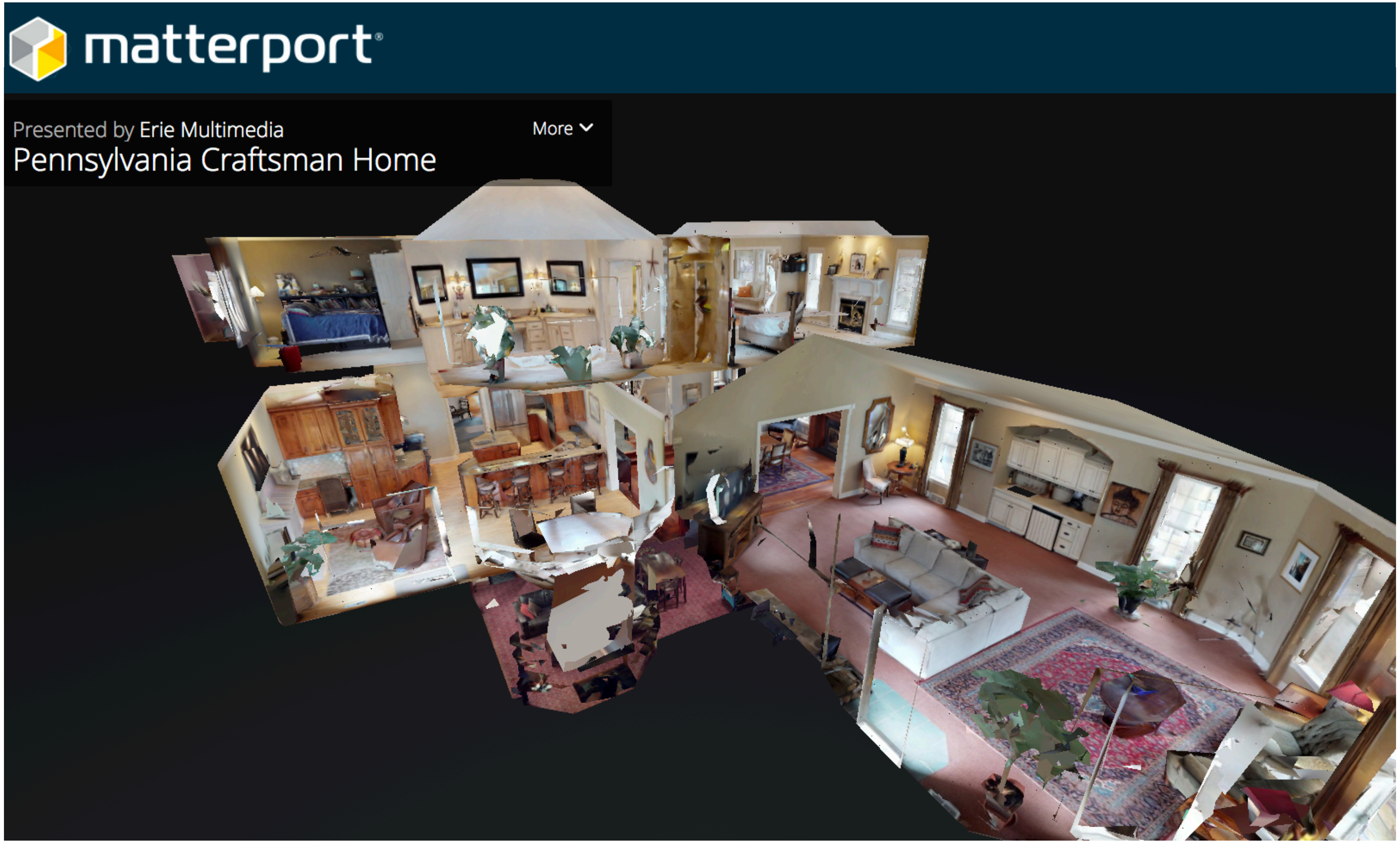
Gaming

1995 demo

Product (different group, different company)

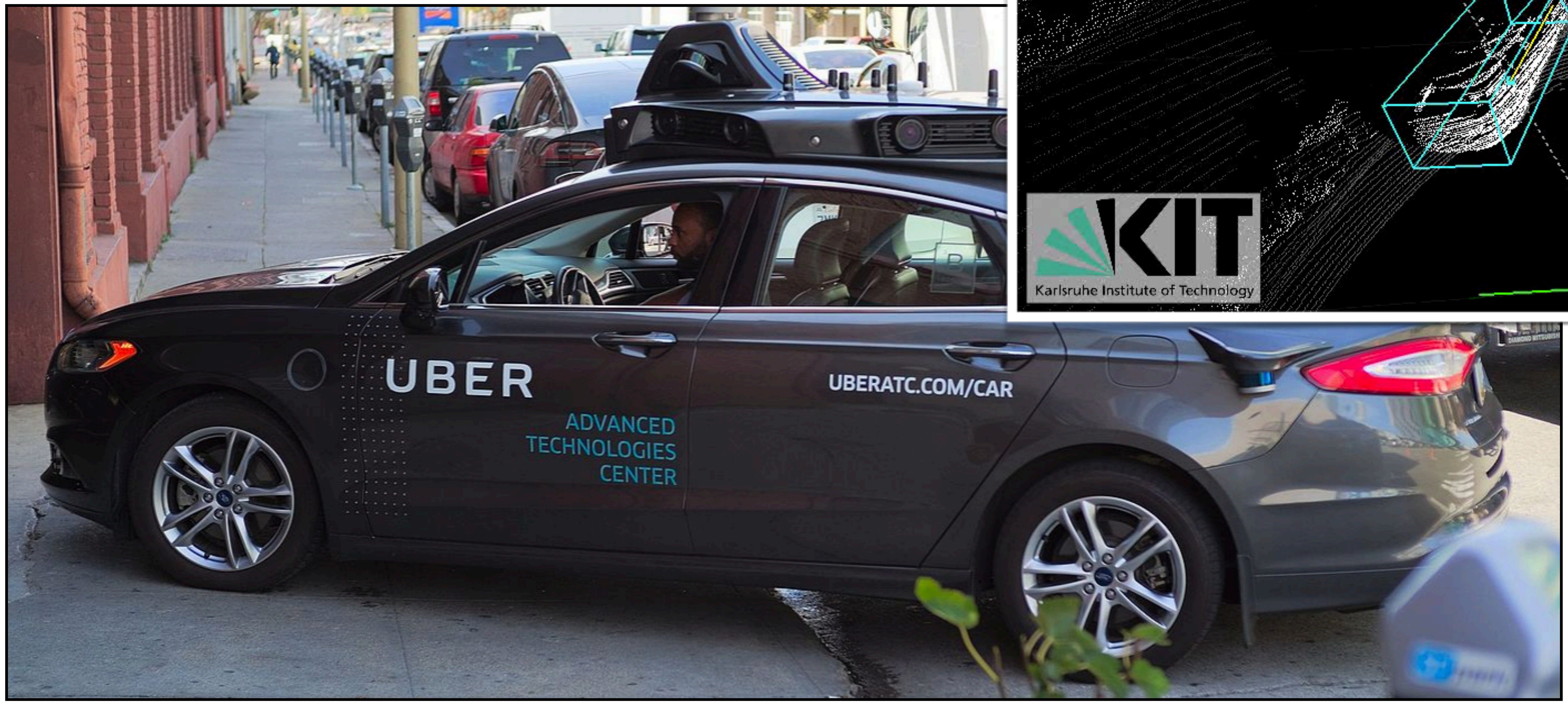
Applications

3D reconstruction for virtual navigation



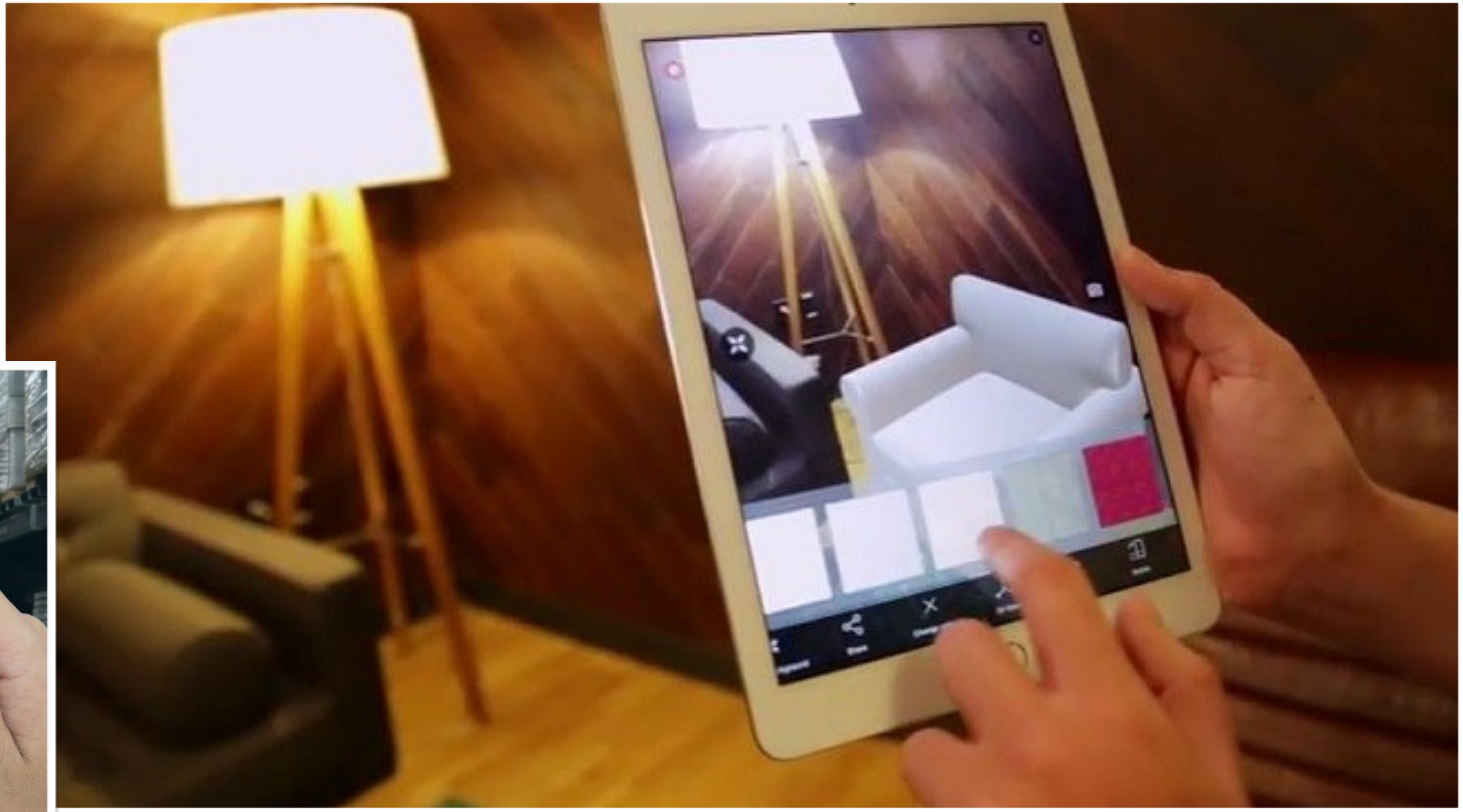
Applications

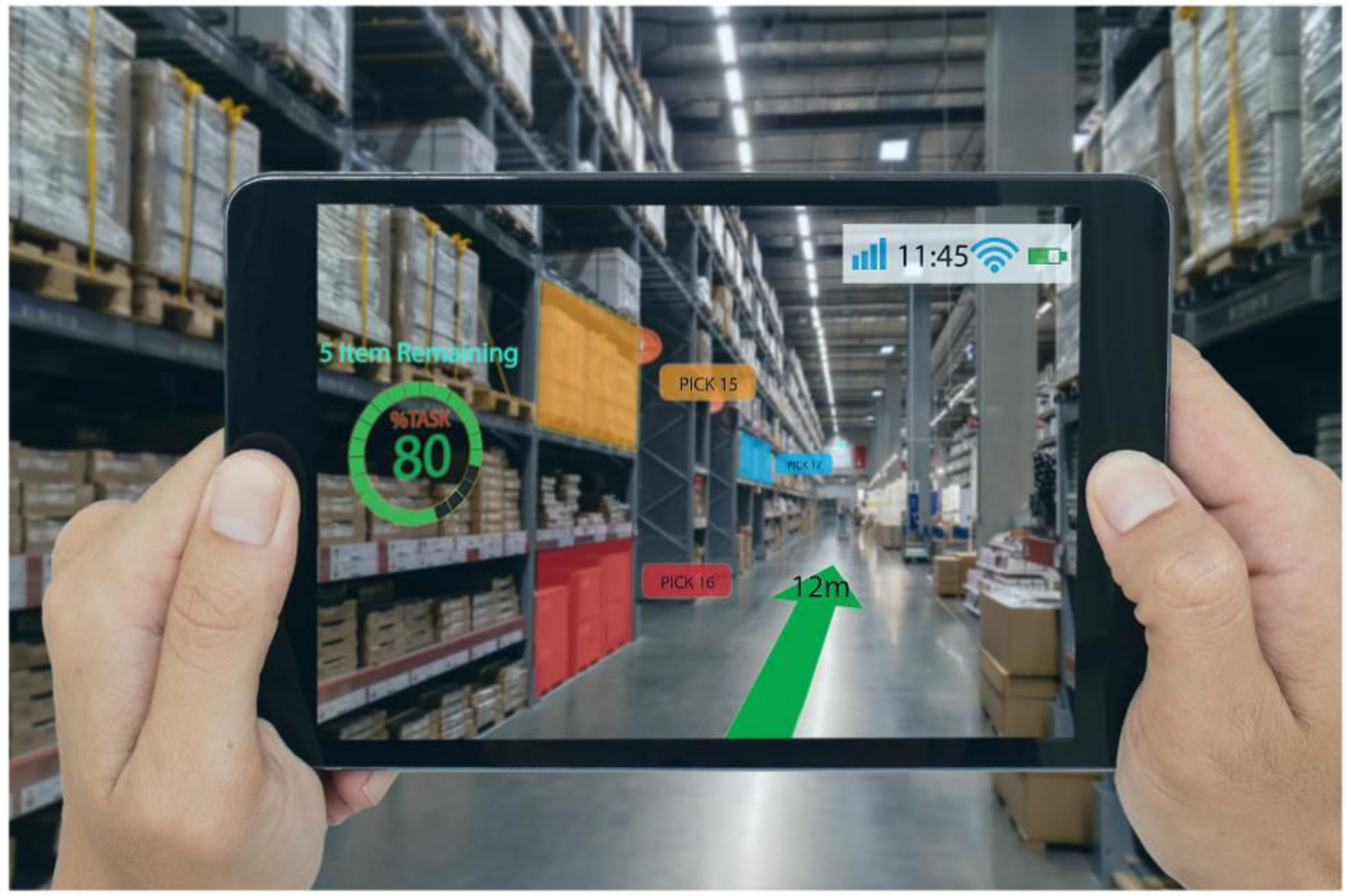
Autonomous driving



Applications

Augmented reality

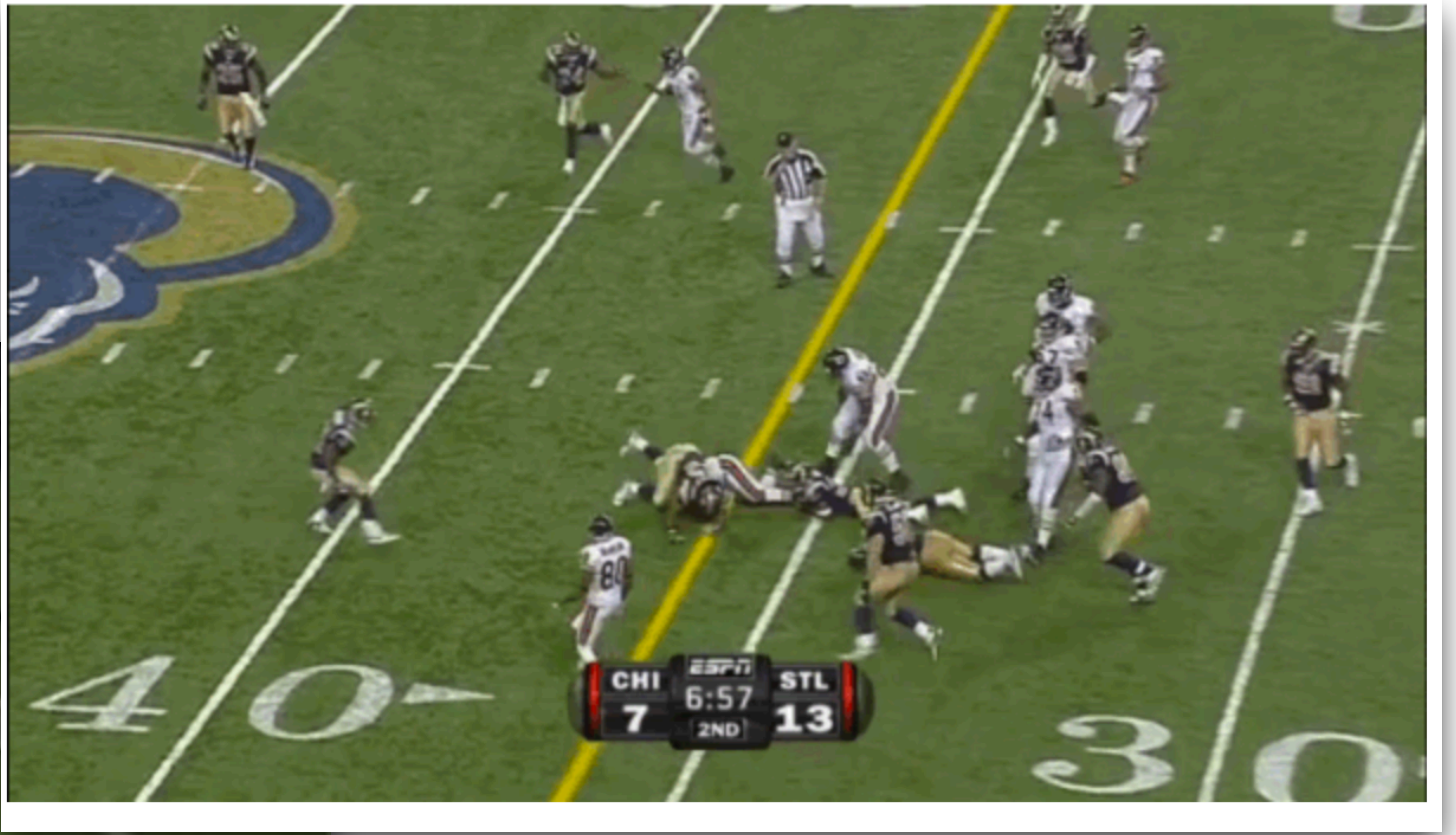




Applications

Augmented reality

https://en.wikipedia.org/wiki/File:First-down-line.jpg



ESPN

Vision systems

One camera

Two cameras

N cameras

Let’s consider two eyes

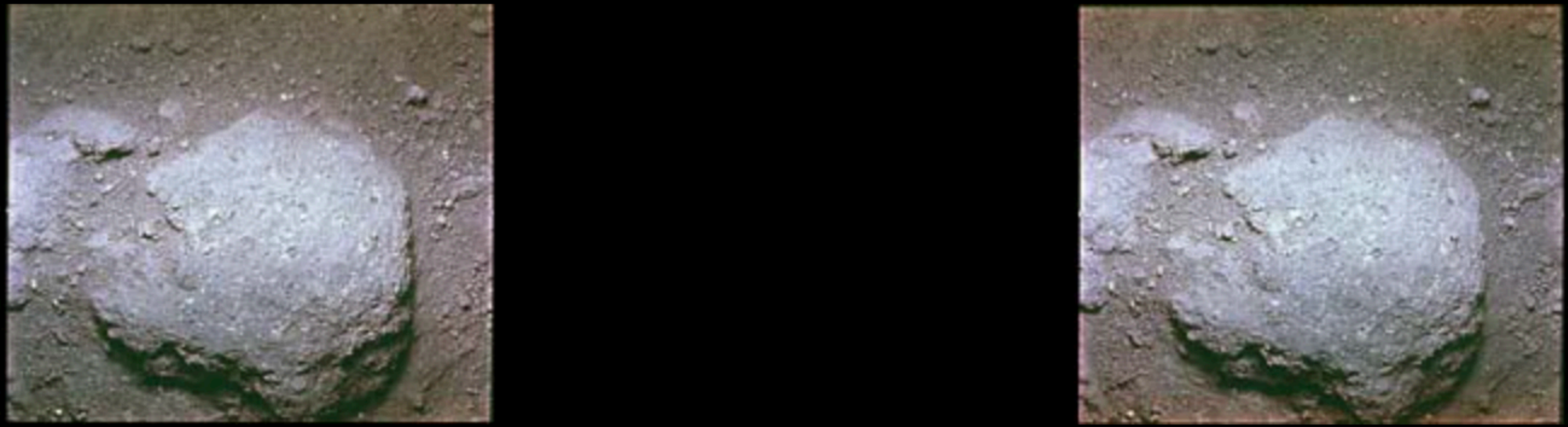
One camera

Two cameras

N cameras

Stereo images of lunar rocks from Apollo 11

Scan Of Transparent Images From Earth To The Moon's Apollo 11 Stereo Slide



NASA Processed Images For Photo ID AS11-45-6712

This artifact is a stereo slide of a pair of images taken during the Apollo 11 EVA using the Apollo Lunar Surface Close-Up Camera (ALSCC). This 35 mm camera provides stereo close up images of the surface of the moon. Each image captured an area of 3 by 3 inches with a resolution of approximately 80 microns. The camera is sometimes referred to as "the Gold camera" in honor of its inventor, British Astronomer Thomas "Tommy" Gold.

These close up images showed detail that could not be seen by the astronauts or by other photographs brought back from the Moon. These special photographs gave geologists a unique insight into the geological processes that shaped the lunar surface.

http://www.earthtothemoon.com/ap_11_Stereo_slide.html

https://www.hq.nasa.gov/alsj/a11/A

Stereoscope

View of Boston, *c. 1860*; an early stereoscopic card for viewing a scene from nature

More details

Soule, John P., 1827-1904 -- Photographer - This image is available from the New York Public Library's Digital Library under the digital ID G90F336_113F: digitalgallery.nypl.org → digitalcollections.nypl.org

File: Charles Street Mall, Boston Common, by Soule, John P., 1827-1904 3.jpg

Created: Coverage: 1860?-1890?. Source Imprint: 1860?-1890?. Digital item published 7-28-2005; updated 4-23-2009.

Brewster-type stereoscope, 1870

More details

Alessandro Nassiri - Museo della Scienza e della Tecnologia "Leonardo da Vinci"

Visore stereoscopico portatile di tipo Brewster, J. Fleury - Hermagis, 1870, con messa a fuoco manuale. Per la visione di lastre e stampe stereoscopiche 8,5x17cm. Museo nazionale della scienza e della tecnologia Leonardo da Vinci, Milano.

File: IGB 006055 Visore stereoscopico portatile Museo scienza e tecnologia Milano.jpg

Created: 1 July 2014

Depth without objects

Random dot stereograms (Bela Julesz)

1	0	1	0	1	0	0	1	0	1
1	0	0	1	0	1	0	1	0	0
0	0	1	1	0	1	1	0	1	0
0	1	0	Y	A	A	B	B	0	1
1	1	1	X	B	A	B	A	0	1
0	0	1	X	A	A	B	A	1	0
1	1	1	Y	B	B	A	B	0	1
1	0	0	1	1	0	1	1	0	1
1	1	0	0	1	1	0	1	1	1
0	1	0	0	0	1	1	1	1	0

1	0	1	0	1	0	0	1	0	1
1	0	0	1	0	1	0	1	0	0
0	0	1	1	0	1	1	0	1	0
0	1	0	A	A	B	B	X	0	1
1	1	1	B	A	B	A	Y	0	1
0	0	1	A	A	B	A	Y	1	0
1	1	1	B	B	A	B	X	0	1
1	0	0	1	1	0	1	1	0	1
1	1	0	0	1	1	0	1	1	1
0	1	0	0	0	1	1	1	1	0

Julesz, 1971

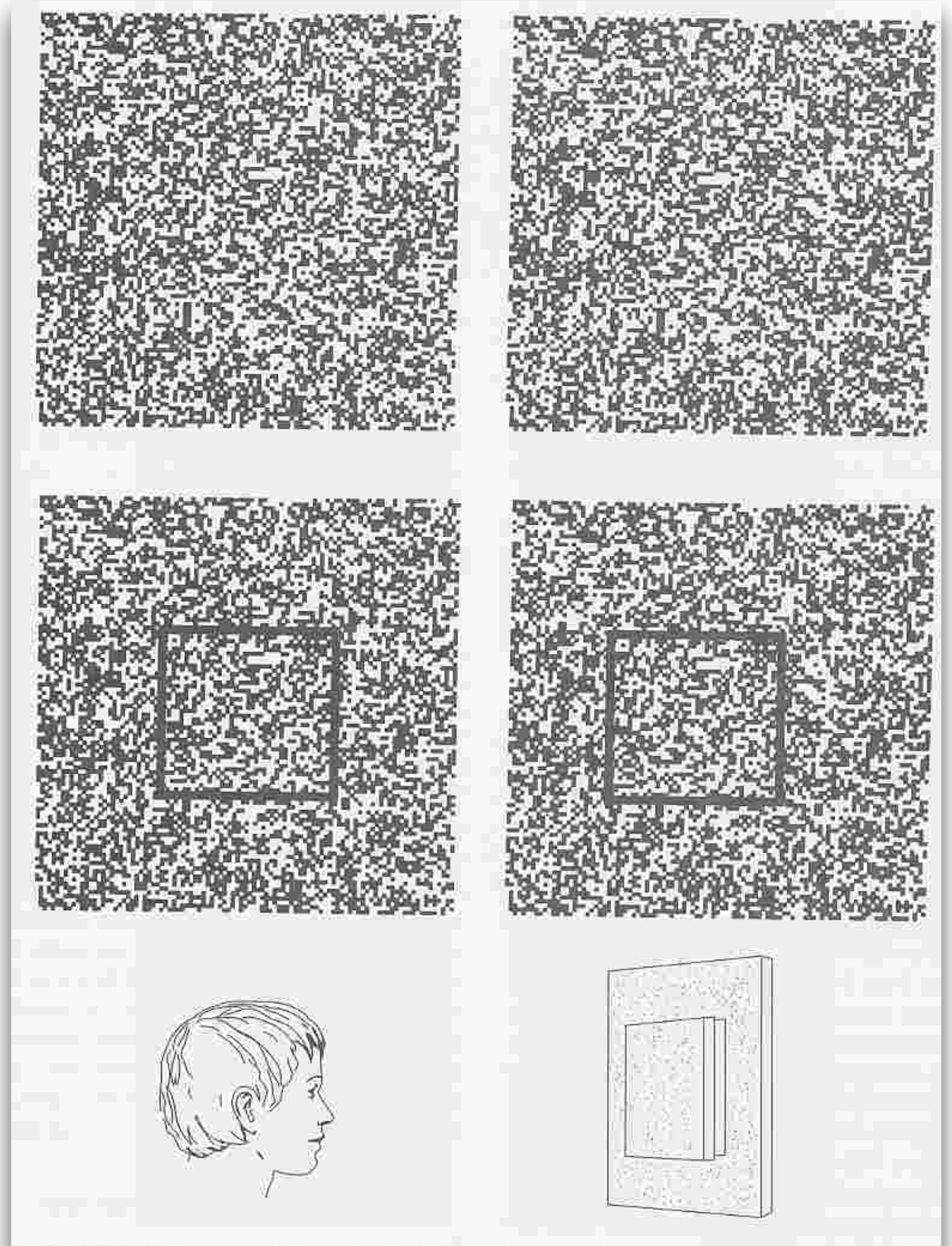
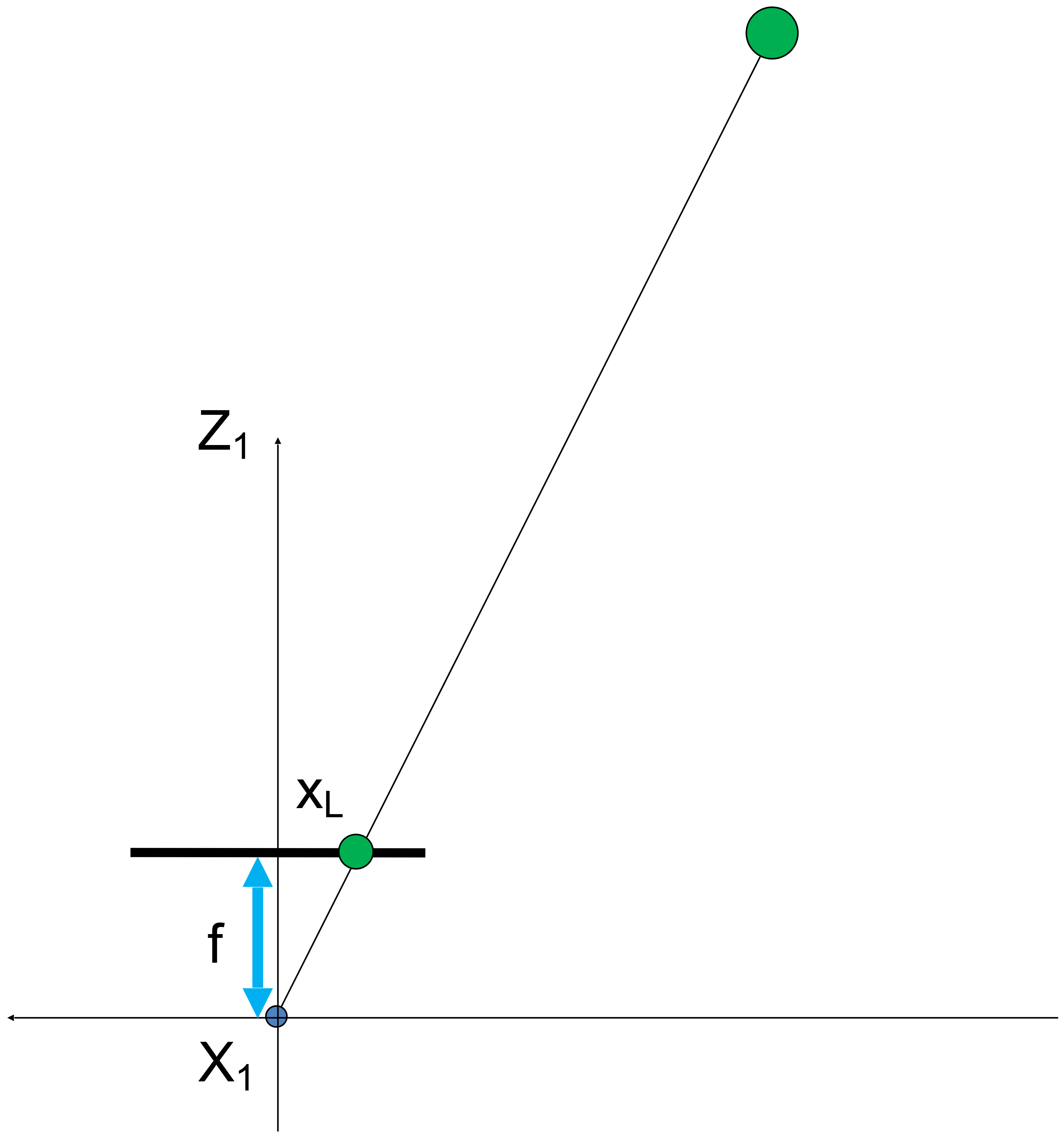
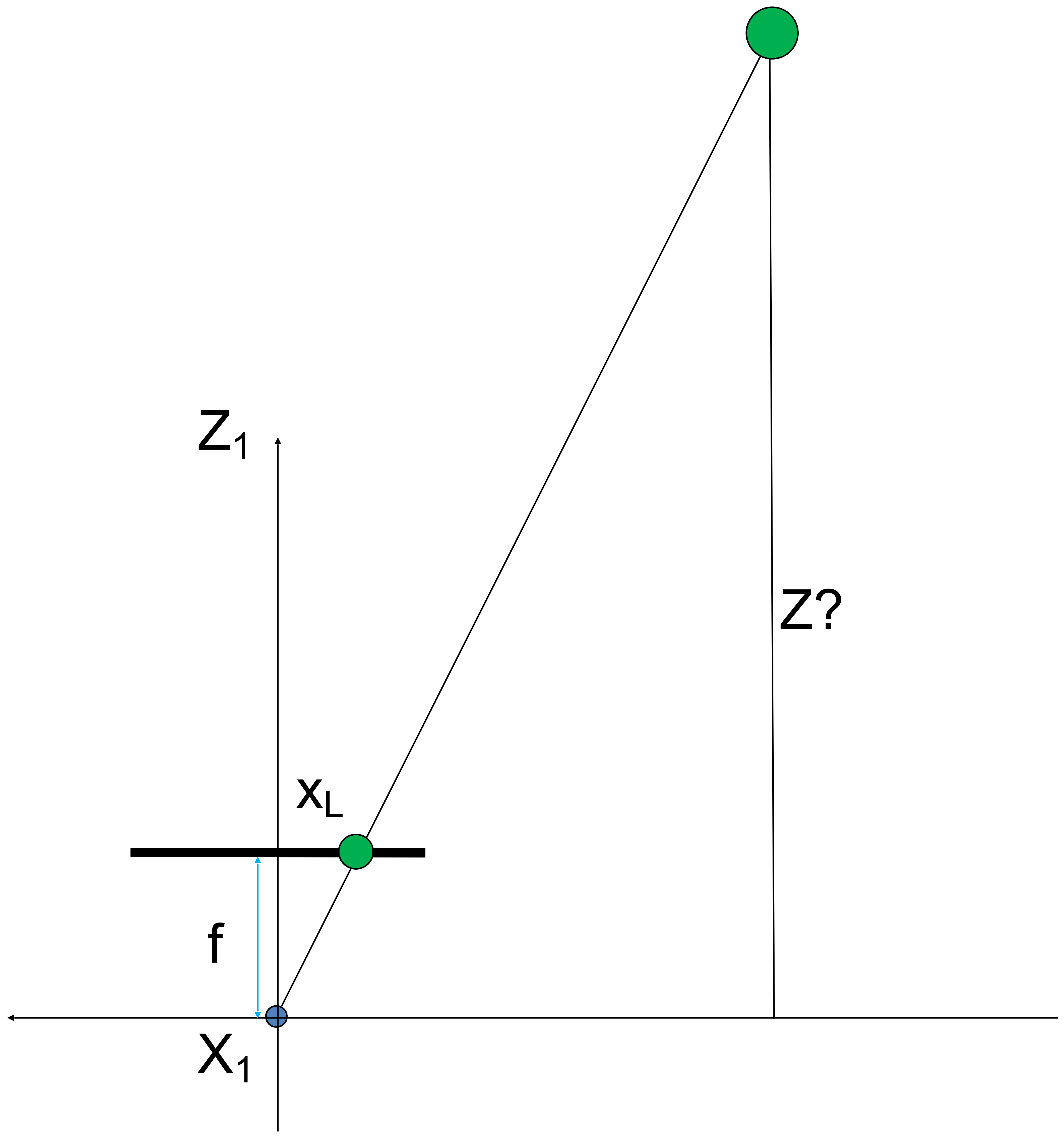


FIGURE 8.13

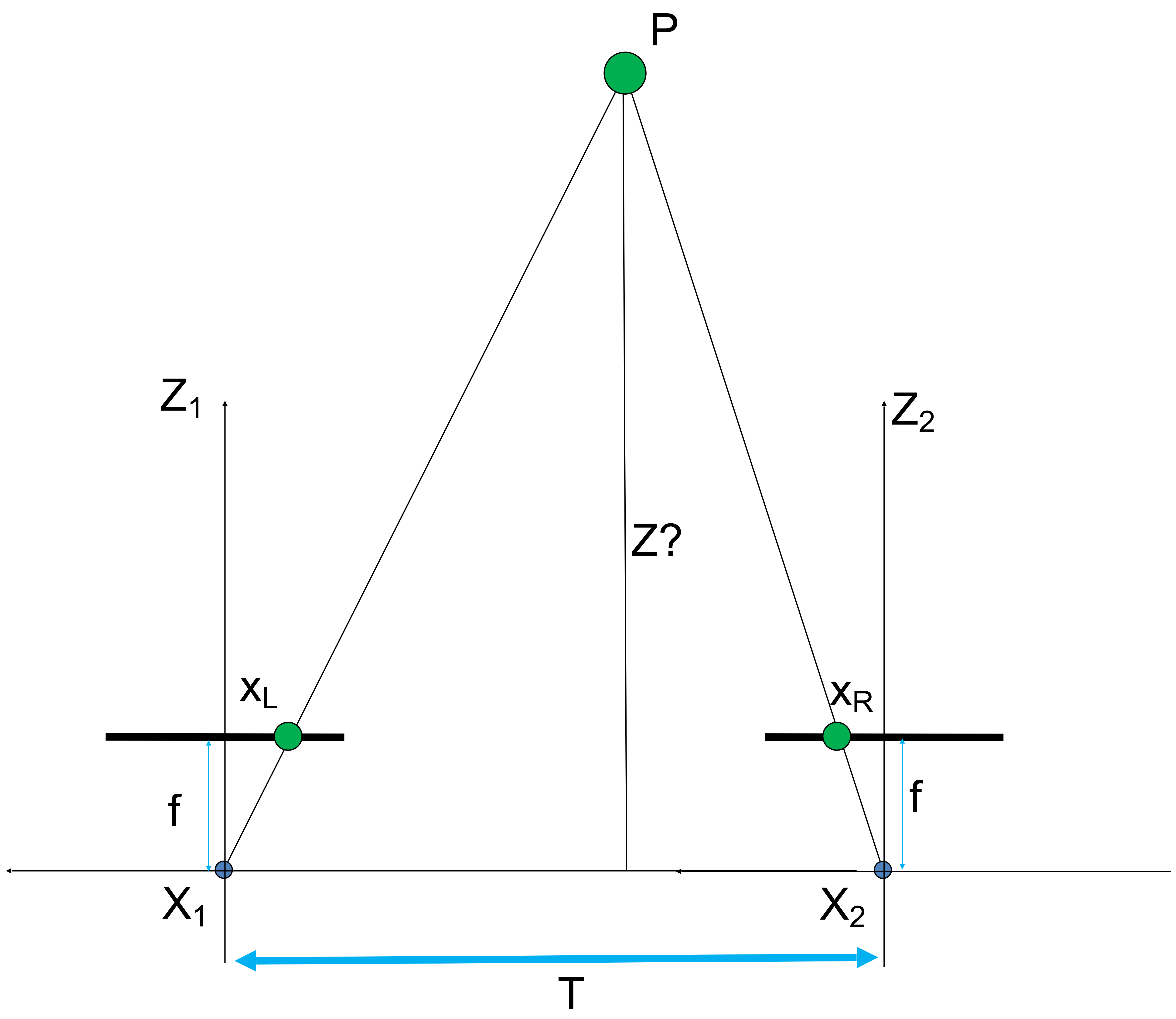
Geometry for a simple stereo system



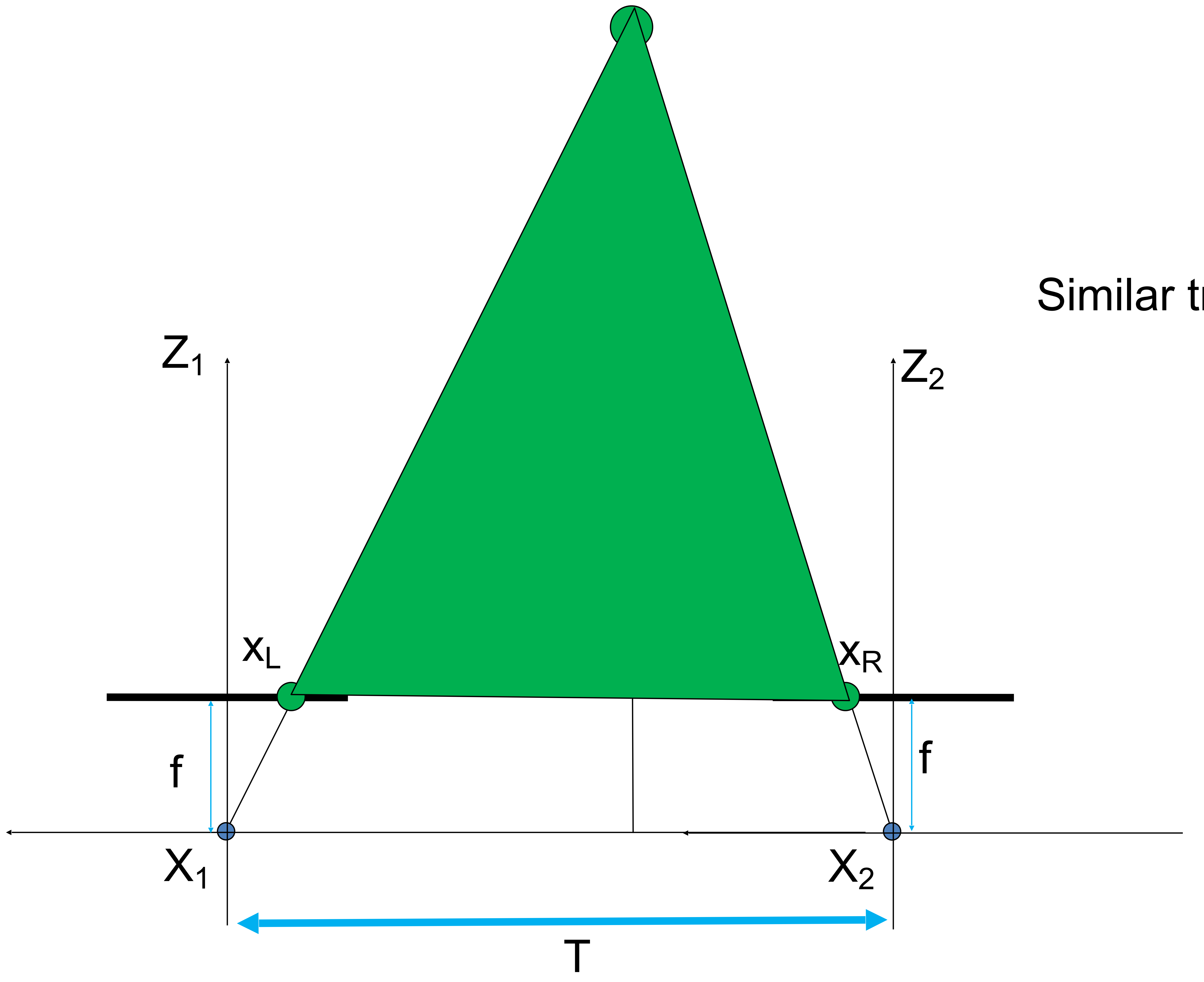
Geometry for a simple stereo system



Geometry for a simple stereo system

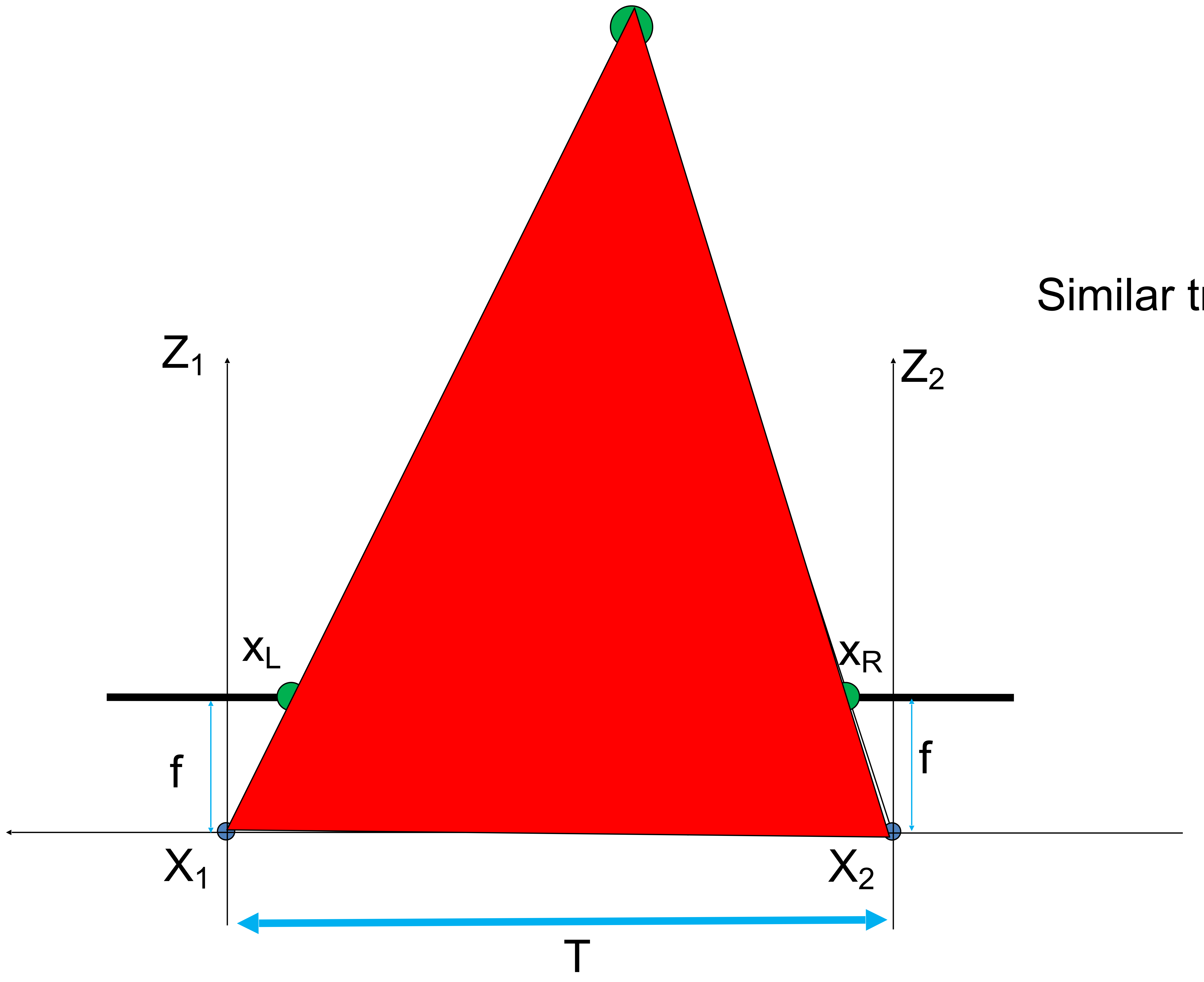


Geometry for a simple stereo system



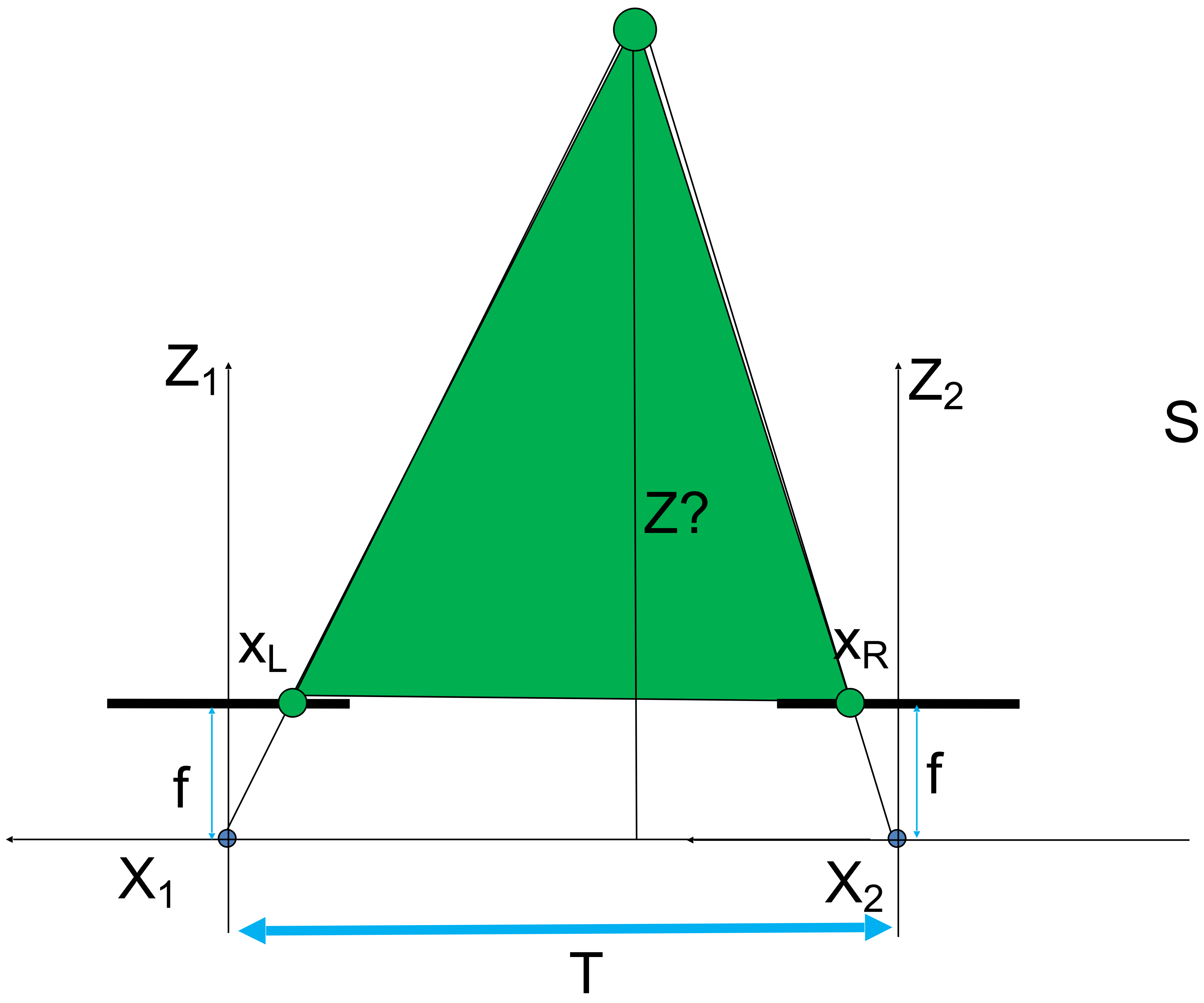
Similar triangles

Geometry for a simple stereo system



Similar triangles

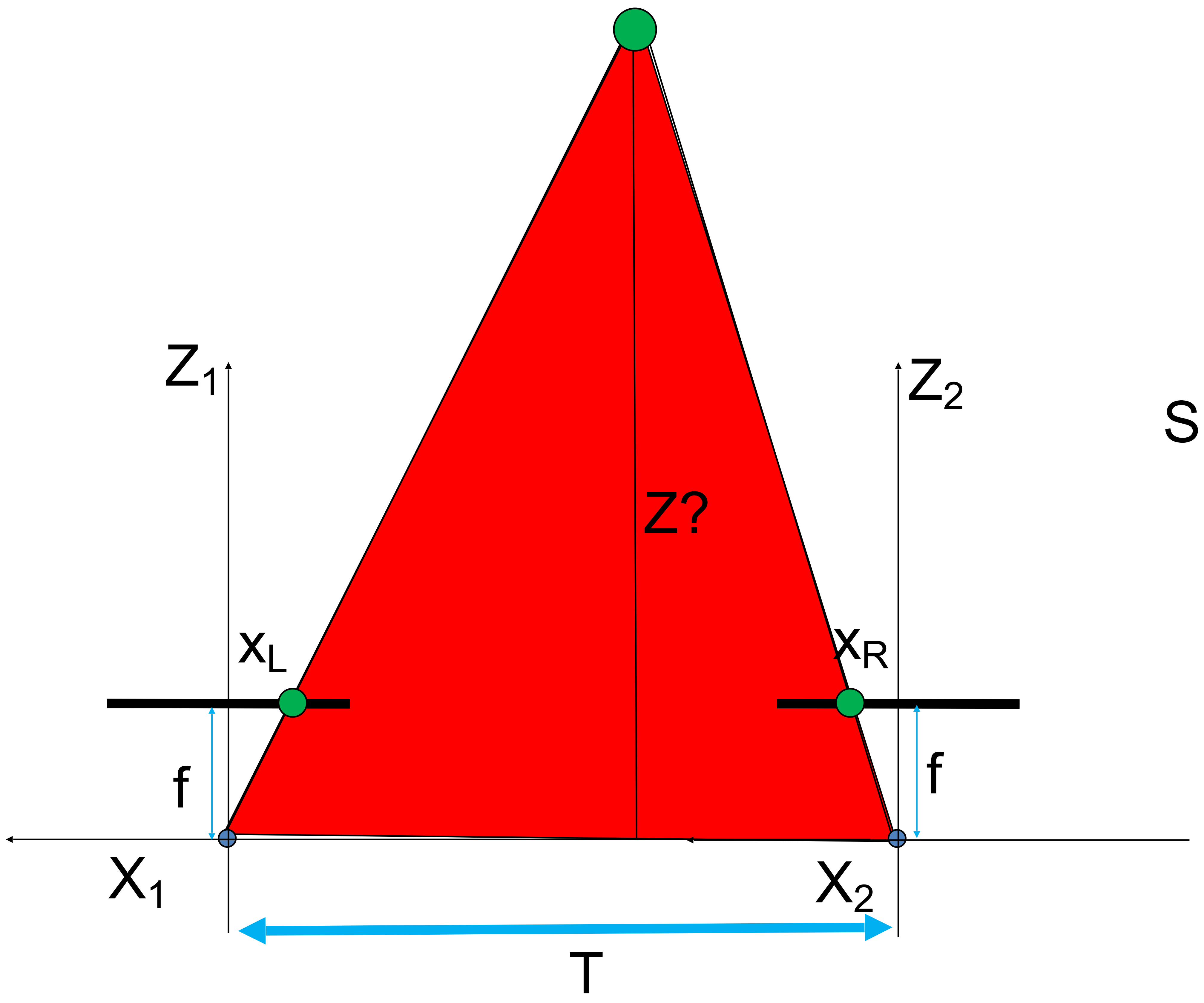
Geometry for a simple stereo system



Similar triangles:

$$\frac{T + X_R - X_L}{Z - f} =$$

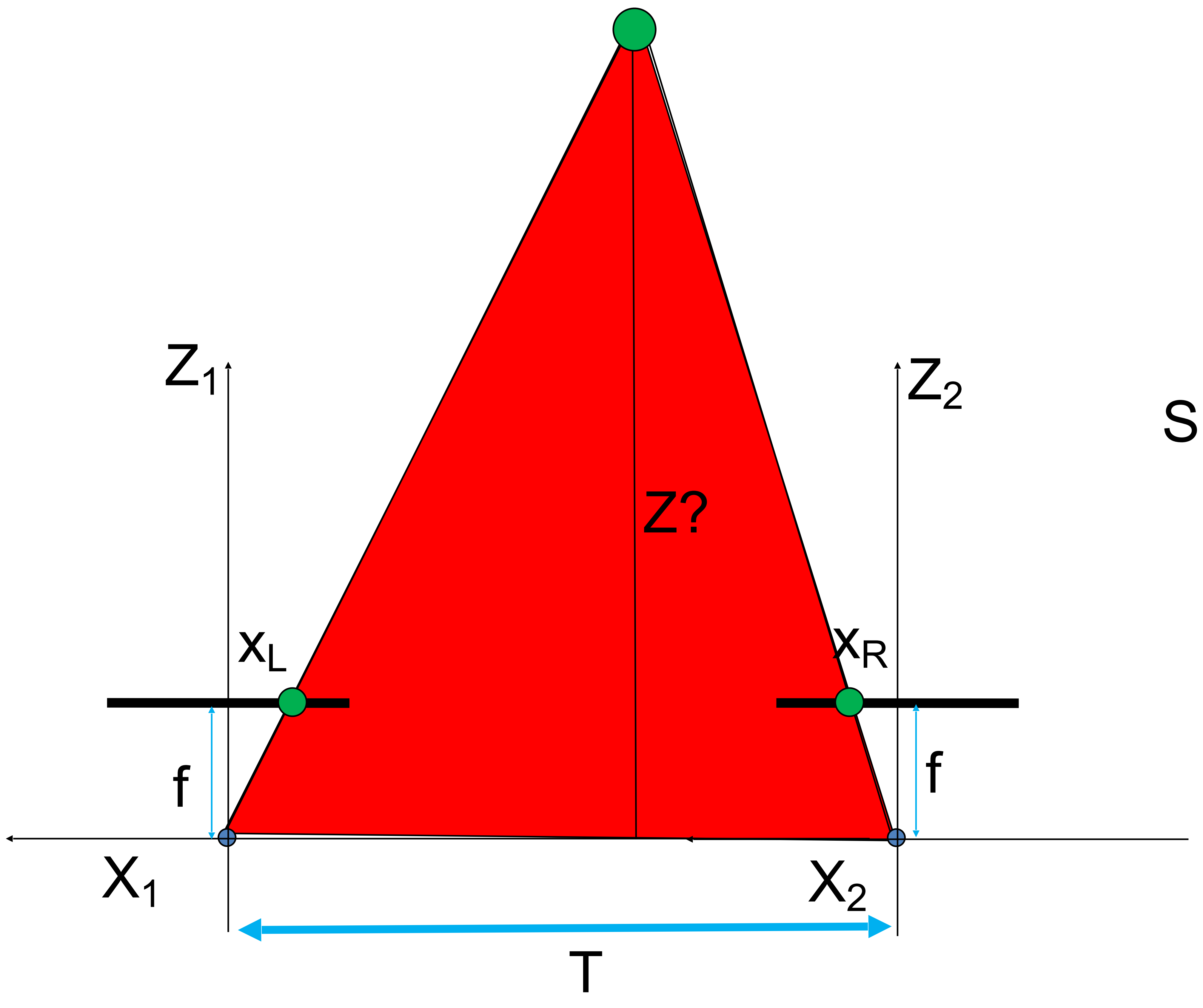
Geometry for a simple stereo system



Similar triangles:

$$\frac{T + X_R - X_L}{Z - f} = \frac{T}{Z}$$

Geometry for a simple stereo system



Similar triangles:

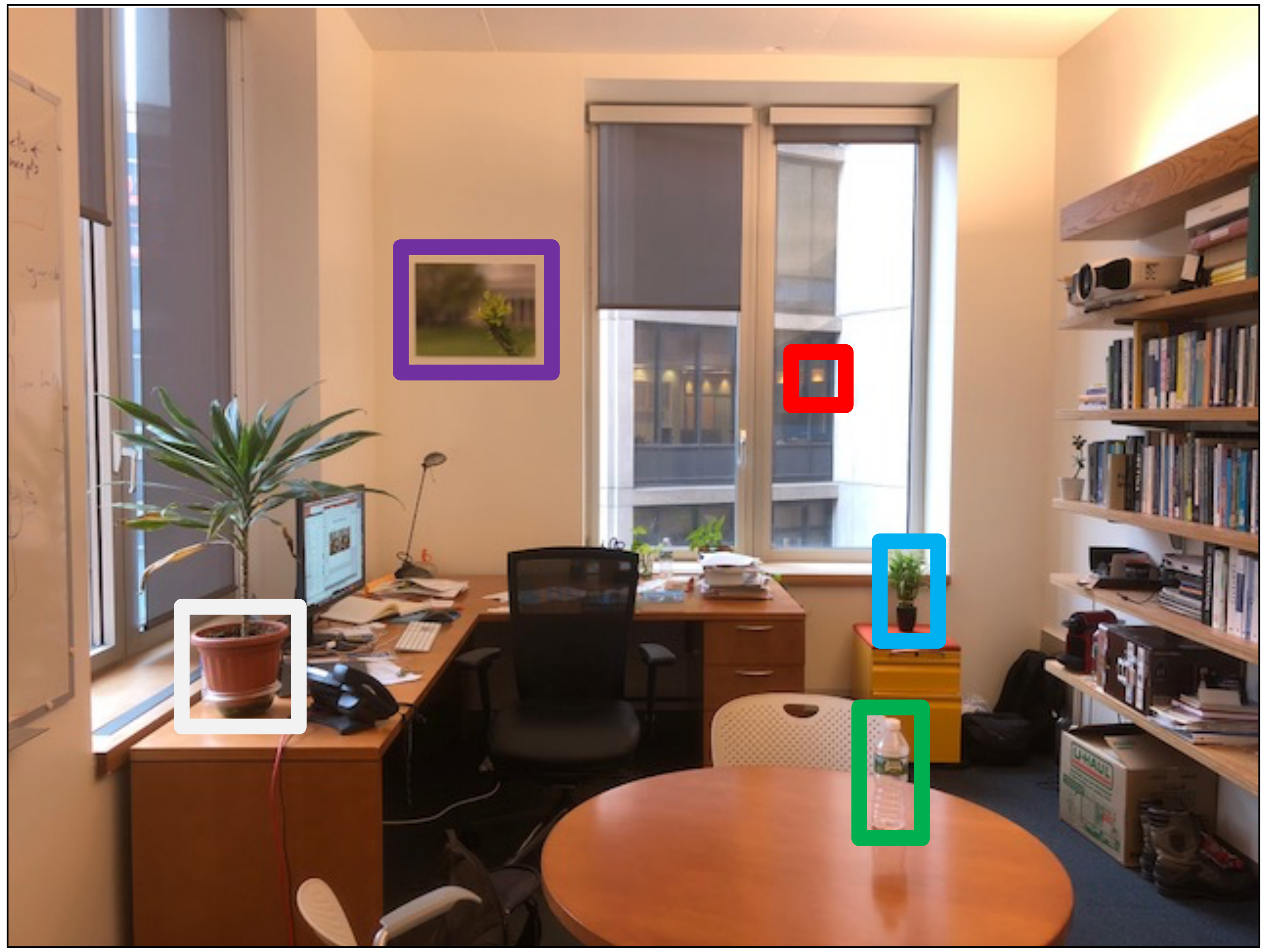
$$\frac{T+X_R-X_L}{Z-f} = \frac{T}{Z}$$

Solving for Z:

$$Z = f \frac{T}{X_L - X_R}$$

Disparity

Measuring disparity

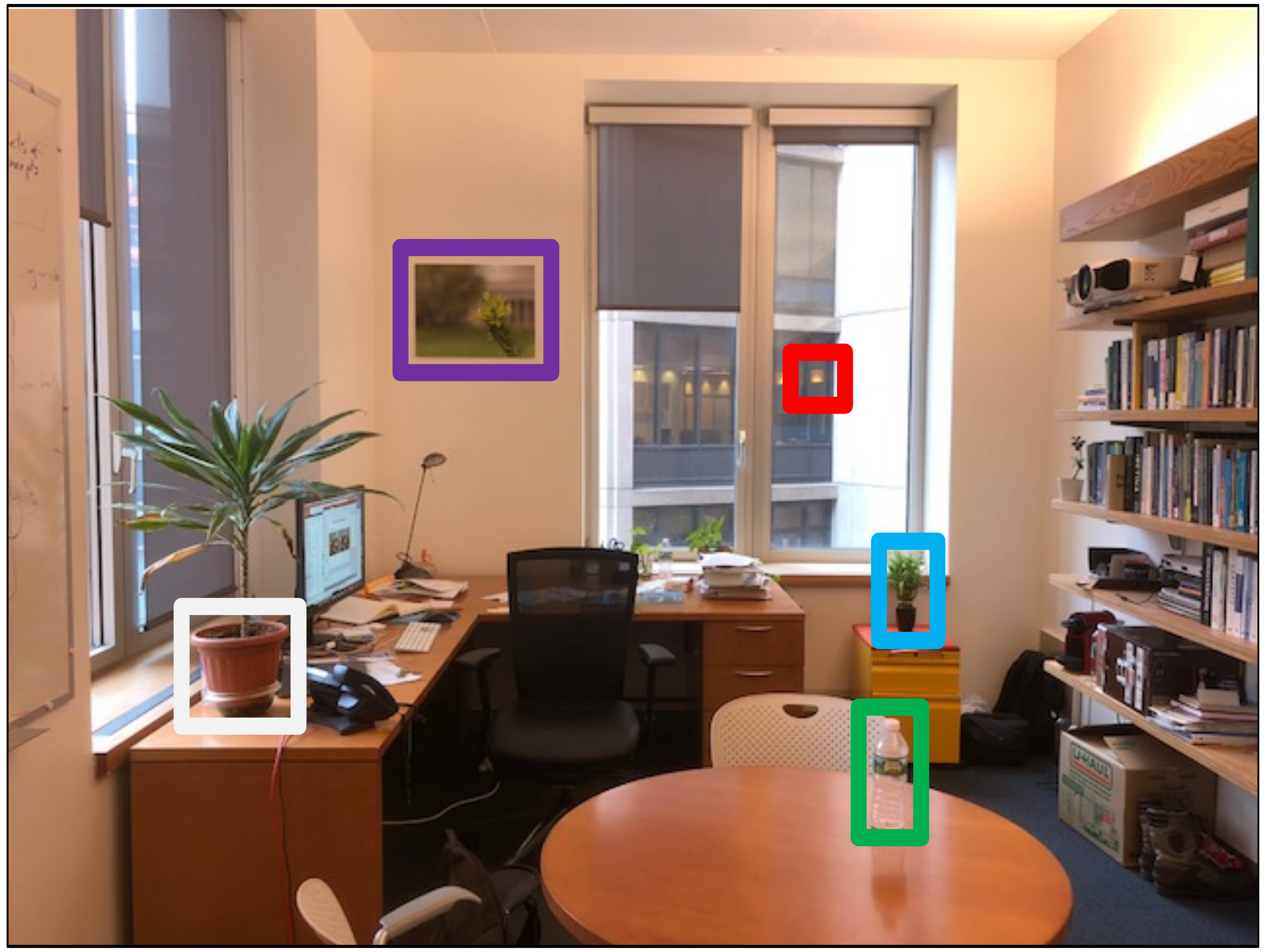


Left image

Right image

I took one picture, then I moved ~1m to the right and took a second picture.

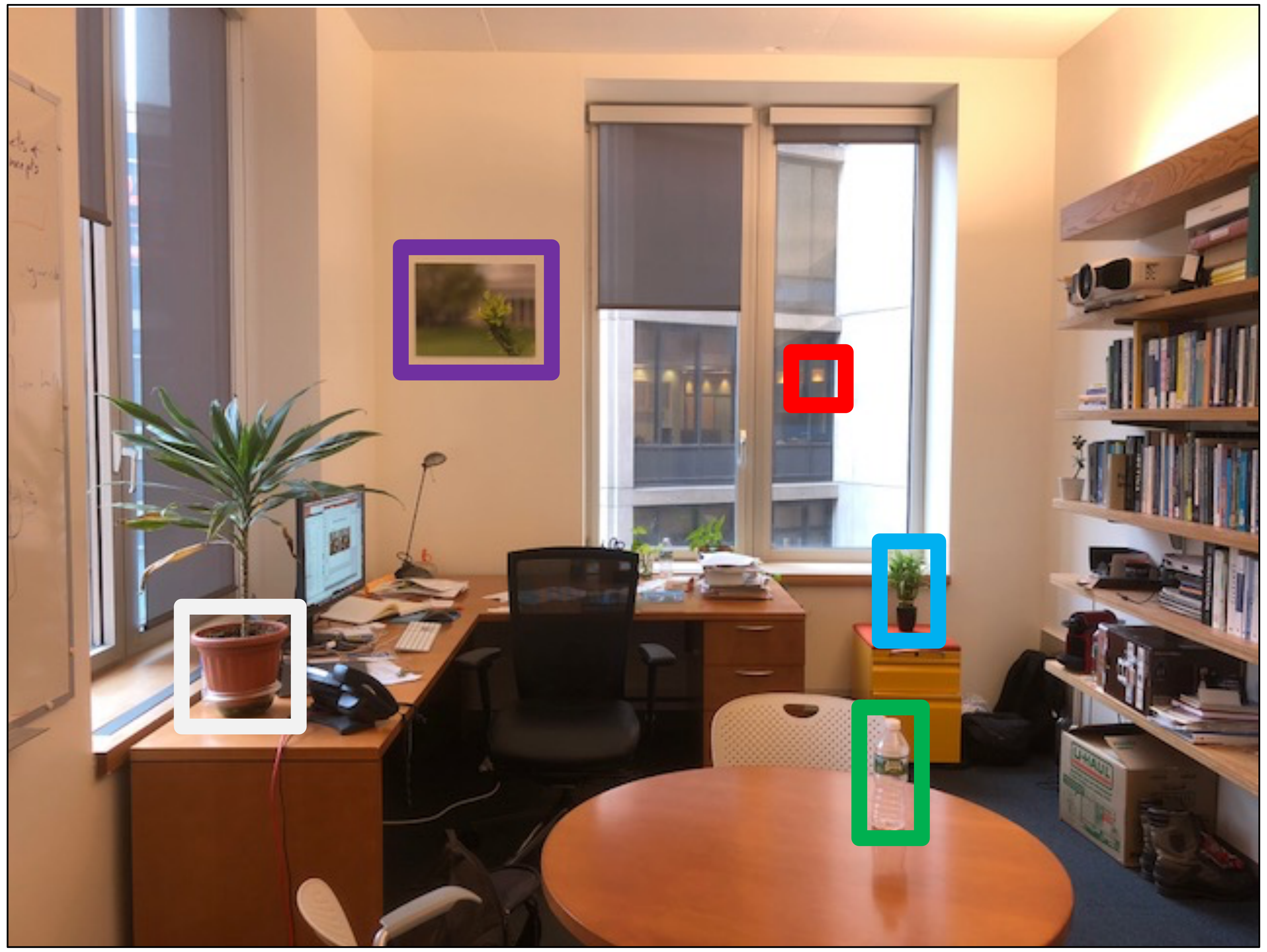
Measuring disparity



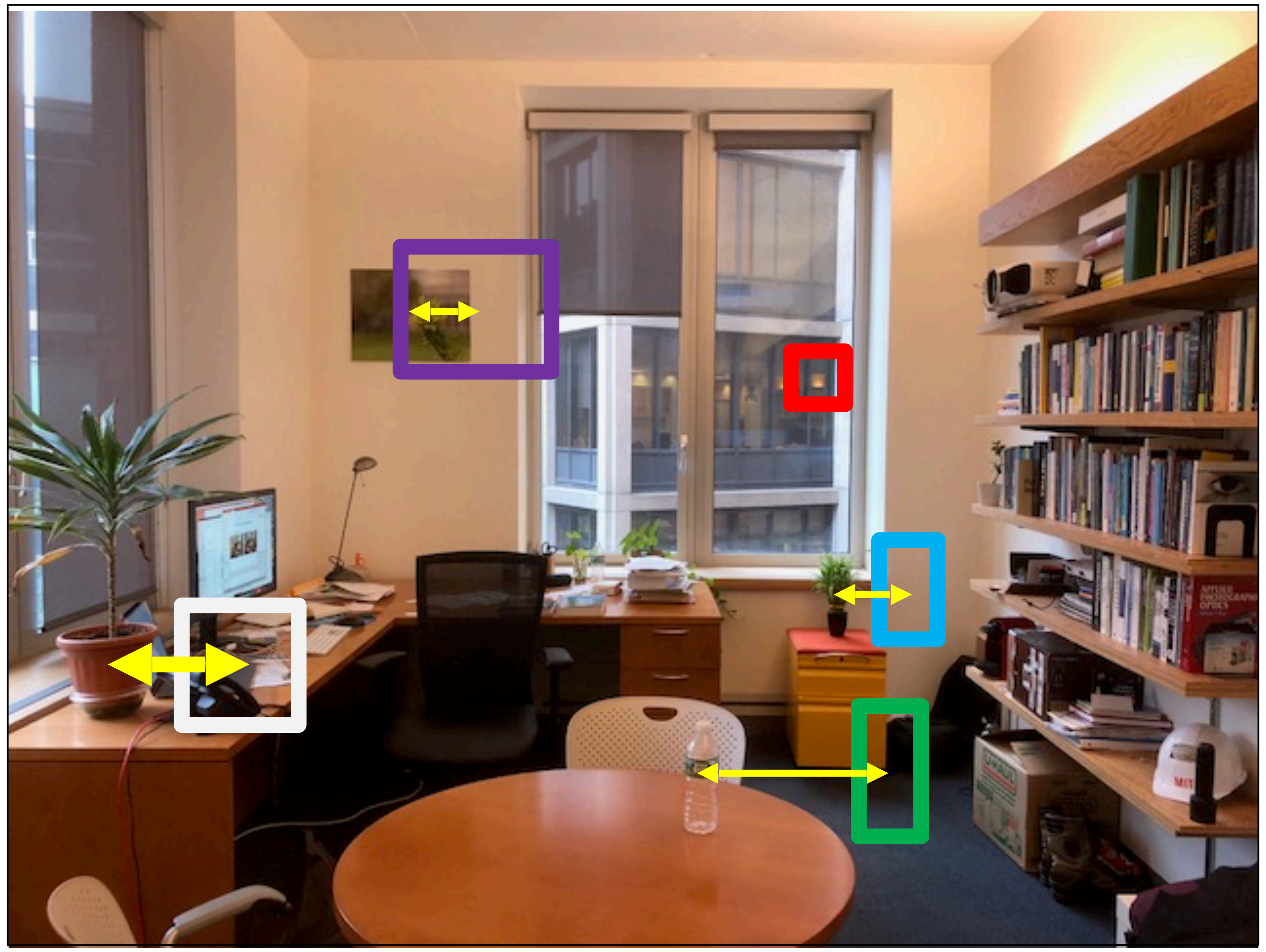
Left image

Right image

Measuring disparity



Left image



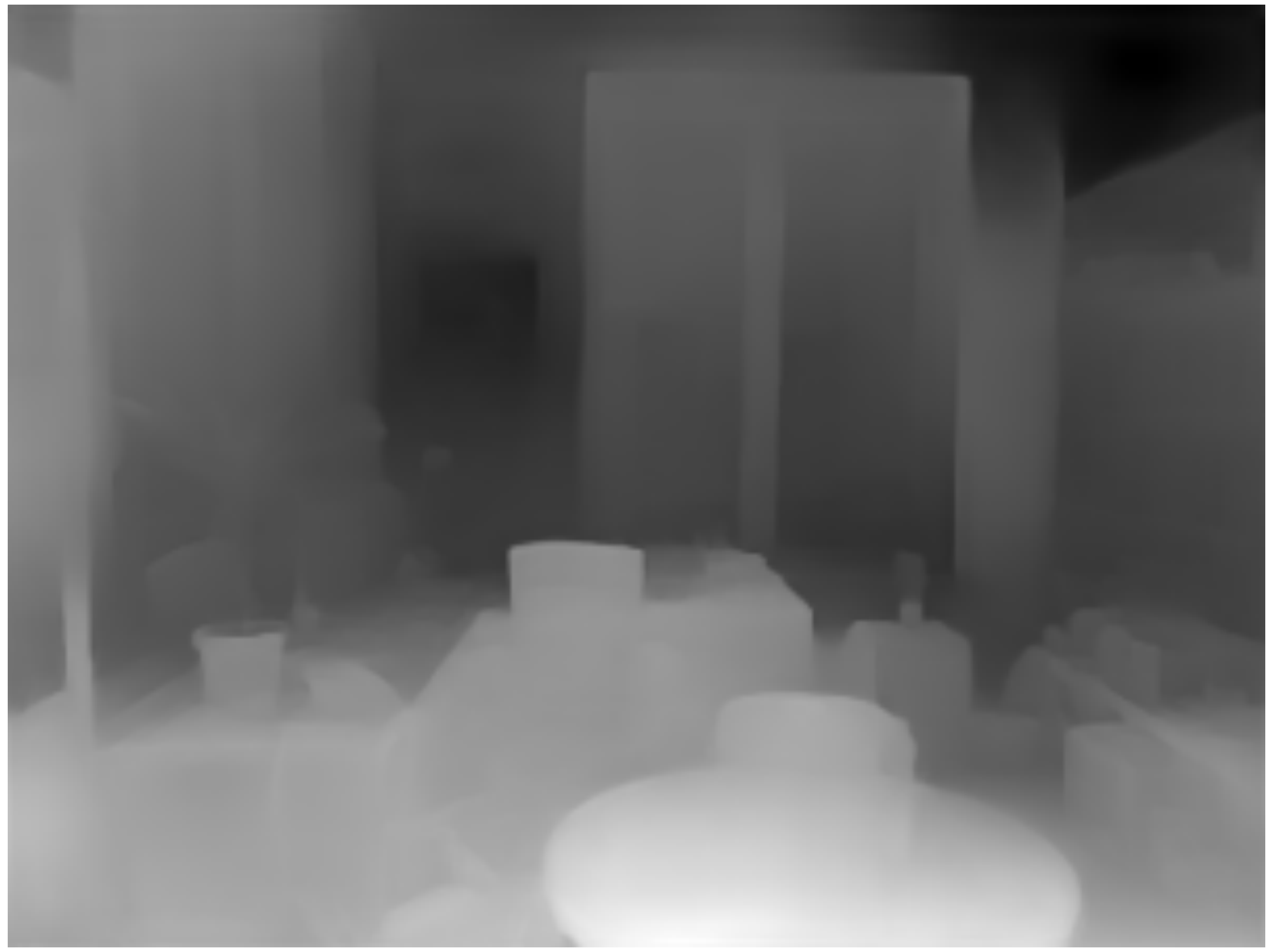
Right image

Disparity map

I(x,y)

I'(x,y) = I(x+D(x,y), y)

D(x,y)

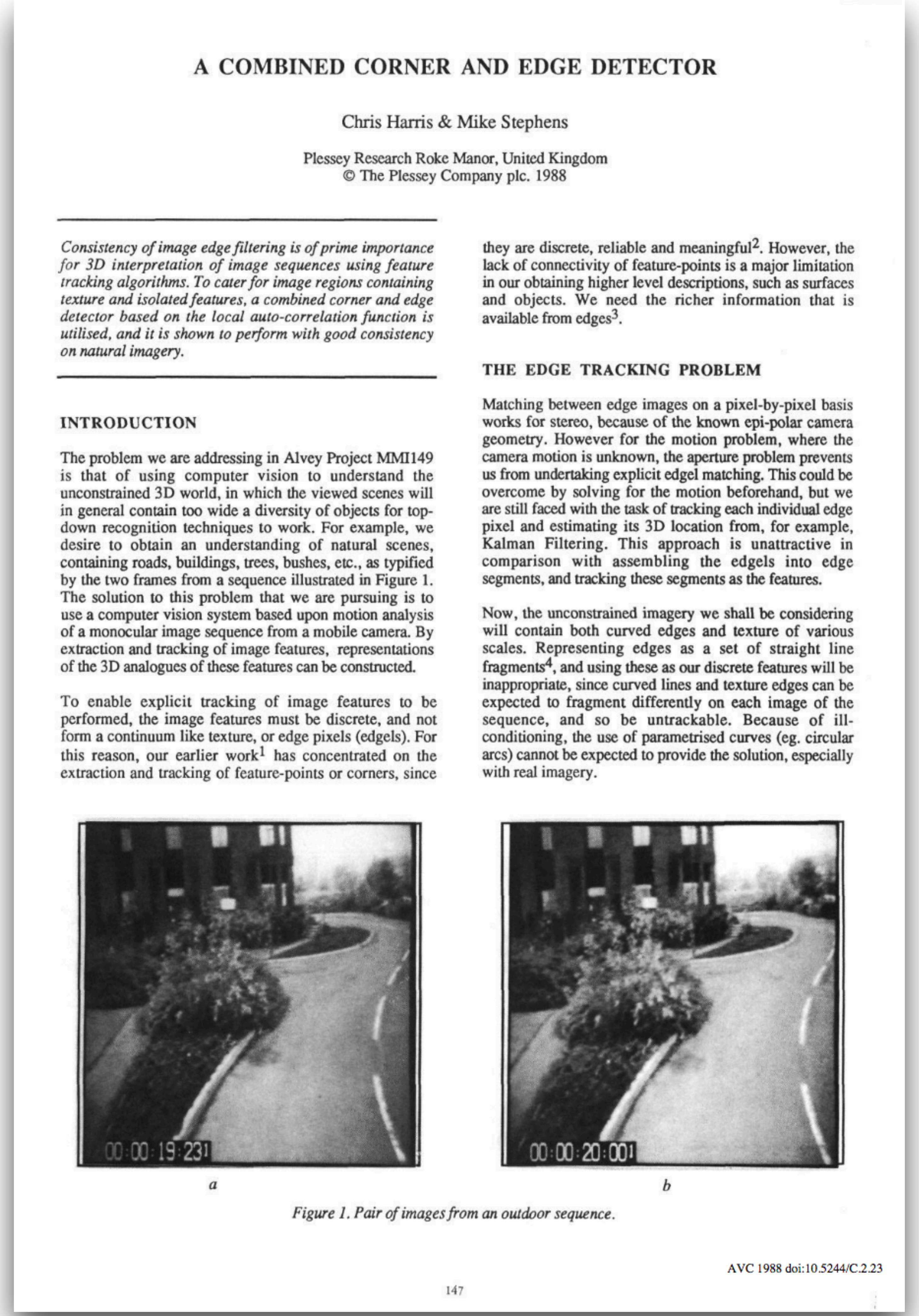


$$Z(x,y) = \frac{a}{D(x,y)}$$

Finding correspondences

We only need to search for matches along horizontal lines.

Computing disparity


A COMBINED CORNER AND EDGE DETECTOR

Chris Harris & Mike Stephens

Plessey Research Roke Manor, United Kingdom
© The Plessey Company plc. 1988

Consistency of image edge filtering is of prime importance for 3D interpretation of image sequences using feature tracking algorithms. To cater for image regions containing texture and isolated features, a combined corner and edge detector based on the local auto-correlation function is utilised, and it is shown to perform with good consistency on natural imagery.

INTRODUCTION

The problem we are addressing in Alvey Project MMI149 is that of using computer vision to understand the unconstrained 3D world, in which the viewed scenes will in general contain too wide a diversity of objects for top-down recognition techniques to work. For example, we desire to obtain an understanding of natural scenes, containing roads, buildings, trees, bushes, etc., as typified by the two frames from a sequence illustrated in Figure 1. The solution to this problem that we are pursuing is to use a computer vision system based upon motion analysis of a monocular image sequence from a mobile camera. By extraction and tracking of image features, representations of the 3D analogues of these features can be constructed.

To enable explicit tracking of image features to be performed, the image features must be discrete, and not form a continuum like texture, or edge pixels (edgels). For this reason, our earlier work[1] has concentrated on the extraction and tracking of feature-points or corners, since they are discrete, reliable and meaningful[2]. However, the lack of connectivity of feature-points is a major limitation in our obtaining higher level descriptions, such as surfaces and objects. We need the richer information that is available from edges[3].

THE EDGE TRACKING PROBLEM

Matching between edge images on a pixel-by-pixel basis works for stereo, because of the known epi-polar camera geometry. However for the motion problem, where the camera motion is unknown, the aperture problem prevents us from undertaking explicit edgel matching. This could be overcome by solving for the motion beforehand, but we are still faced with the task of tracking each individual edge pixel and estimating its 3D location from, for example, Kalman Filtering. This approach is unattractive in comparison with assembling the edgels into edge segments, and tracking these segments as the features.

Now, the unconstrained imagery we shall be considering will contain both curved edges and texture of various scales. Representing edges as a set of straight line fragments[4], and using these as our discrete features will be inappropriate, since curved lines and texture edges can be expected to fragment differently on each image of the sequence, and so be untrackable. Because of ill-conditioning, the use of parametrised curves (eg. circular arcs) cannot be expected to provide the solution, especially with real imagery.

00:00:19:23   00:00:20:00

a   b

Figure 1. Pair of images from an outdoor sequence.

AVC 1988 doi:10.5244/C.2.23

147

Harris corner detector

Harris & Stephens, 1988

SIFT descriptor

David Lowe, 1999


Object Recognition from Local Scale-Invariant Features

David G. Lowe
Computer Science Department
University of British Columbia
Vancouver, B.C., V6T 1Z4, Canada
lowe@cs.ubc.ca

Abstract

An object recognition system has been developed that uses a new class of local image features. The features are invariant to image scaling, translation, and rotation, and partially invariant to illumination changes and affine or 3D projection. These features share similar properties with neurons in inferior temporal cortex that are used for object recognition in primate vision. Features are efficiently detected through a staged filtering approach that identifies stable points in scale space. Image keys are created that allow for local geometric deformations by representing blurred image gradients in multiple orientation planes and at multiple scales. The keys are used as input to a nearest-neighbor indexing method that identifies candidate object matches. Final verification of each match is achieved by finding a low-residual least-squares solution for the unknown model parameters. Experimental results show that robust object recognition can be achieved in cluttered partially-occluded images with a computation time of under 2 seconds.

1. Introduction

Object recognition in cluttered real-world scenes requires local image features that are unaffected by nearby clutter or partial occlusion. The features must be at least partially invariant to illumination, 3D projective transforms, and common object variations. On the other hand, the features must also be sufficiently distinctive to identify specific objects among many alternatives. The difficulty of the object recognition problem is due in large part to the lack of success in finding such image features. However, recent research on the use of dense local features (e.g., Schmid & Mohr [19]) has shown that efficient recognition can often be achieved by using local image descriptors sampled at a large number of repeatable locations.

This paper presents a new method for image feature generation called the Scale Invariant Feature Transform (SIFT). This approach transforms an image into a large collection of local feature vectors, each of which is invariant to image translation, scaling, and rotation, and partially invariant to illumination changes and affine or 3D projection. Previous approaches to local feature generation lacked invariance to scale and were more sensitive to projective distortion and illumination change. The SIFT features share a number of properties in common with the responses of neurons in inferior temporal (IT) cortex in primate vision. This paper also describes improved approaches to indexing and model verification.

The scale-invariant features are efficiently identified by using a staged filtering approach. The first stage identifies key locations in scale space by looking for locations that are maxima or minima of a difference-of-Gaussian function. Each point is used to generate a feature vector that describes the local image region sampled relative to its scale-space coordinate frame. The features achieve partial invariance to local variations, such as affine or 3D projections, by blurring image gradient locations. This approach is based on a model of the behavior of complex cells in the cerebral cortex of mammalian vision. The resulting feature vectors are called SIFT keys. In the current implementation, each image generates on the order of 1000 SIFT keys, a process that requires less than 1 second of computation time.

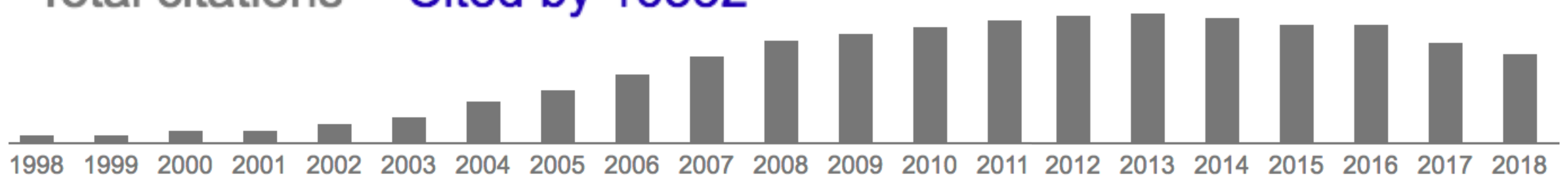
The SIFT keys derived from an image are used in a nearest-neighbour approach to indexing to identify candidate object models. Collections of keys that agree on a potential model pose are first identified through a Hough transform hash table, and then through a least-squares fit to a final estimate of model parameters. When at least 3 keys agree on the model parameters with low residual, there is strong evidence for the presence of the object. Since there may be dozens of SIFT keys in the image of a typical object, it is possible to have substantial levels of occlusion in the image and yet retain high levels of reliability.

The current object models are represented as 2D locations of SIFT keys that can undergo affine projection. Sufficient variation in feature location is allowed to recognize perspective projection of planar shapes at up to a 60 degree rotation away from the camera or to allow up to a 20 degree rotation of a 3D object.

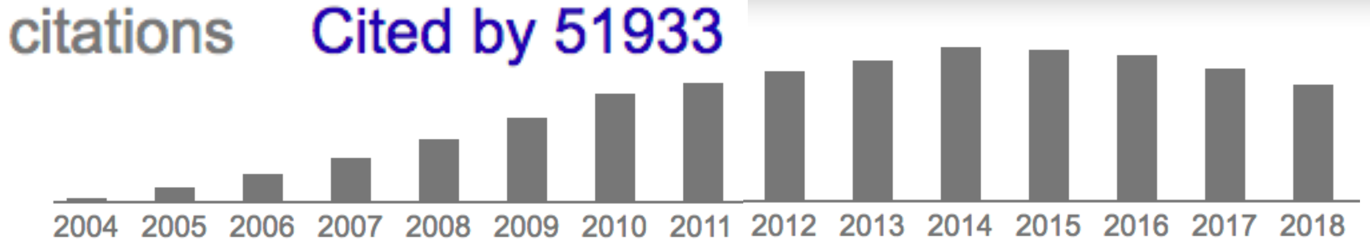
Proc. of the International Conference on Computer Vision, Corfu (Sept. 1999)

1

Total citations   Cited by 16332

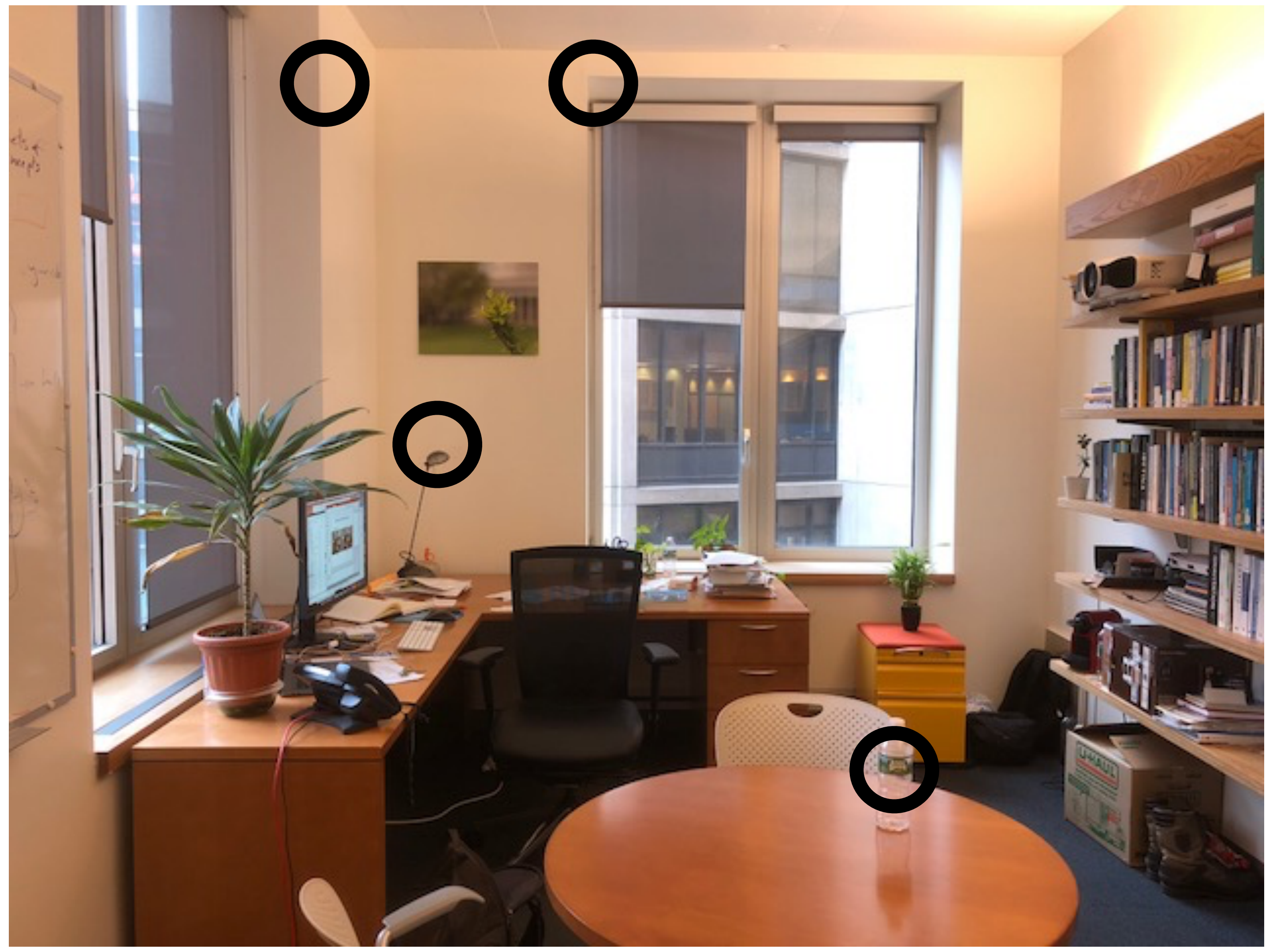


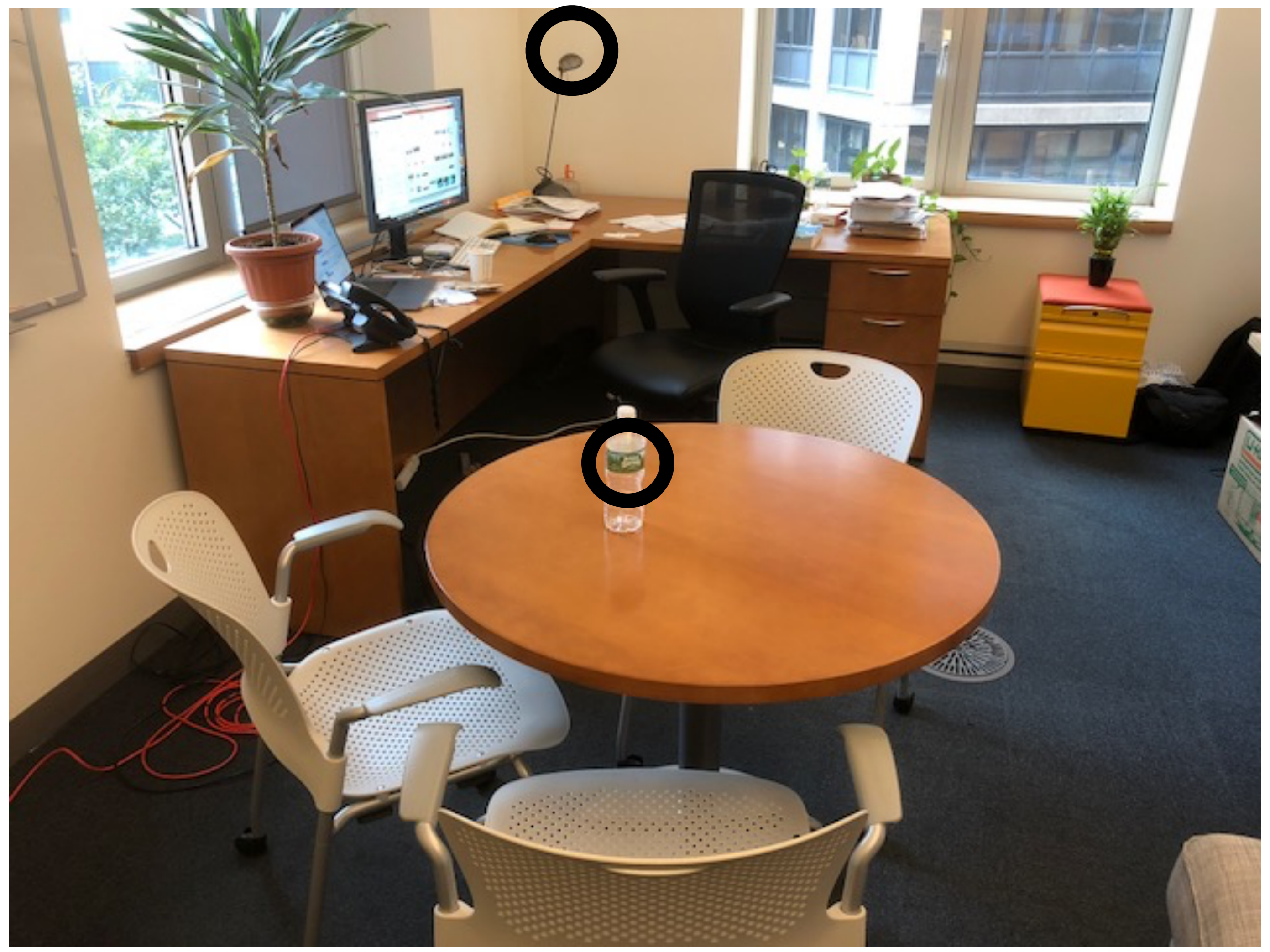
Total citations   Cited by 51933



Finding correspondences

1. Detect features using SIFT [Lowe, IJCV 2004]



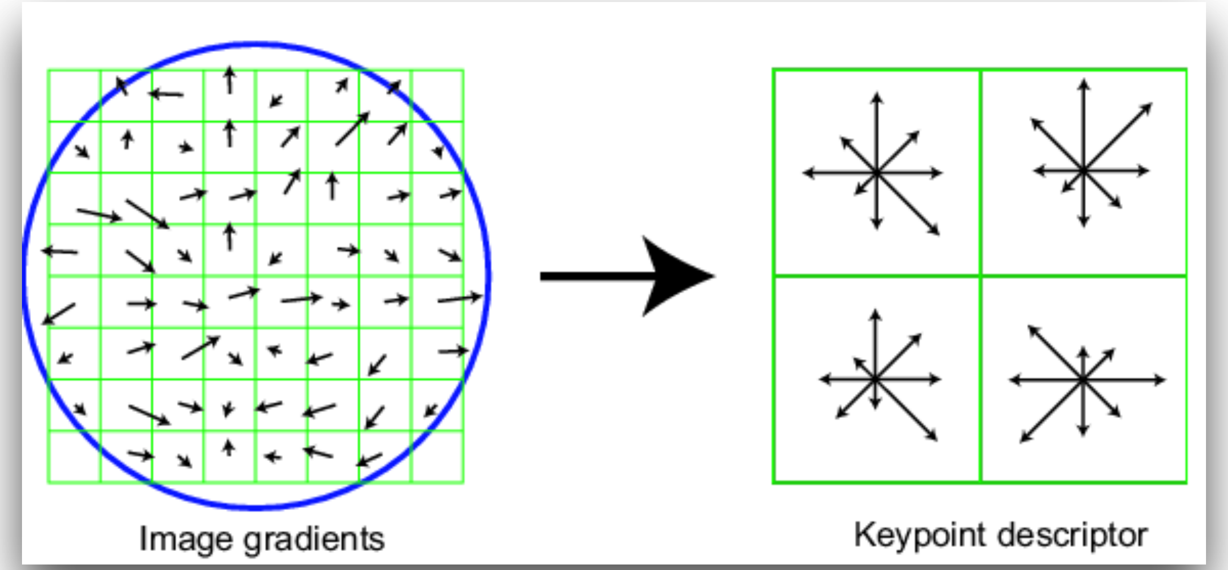


Finding correspondences (SIFT)

1) detect keypoints

points = detectHarrisFeatures(img);

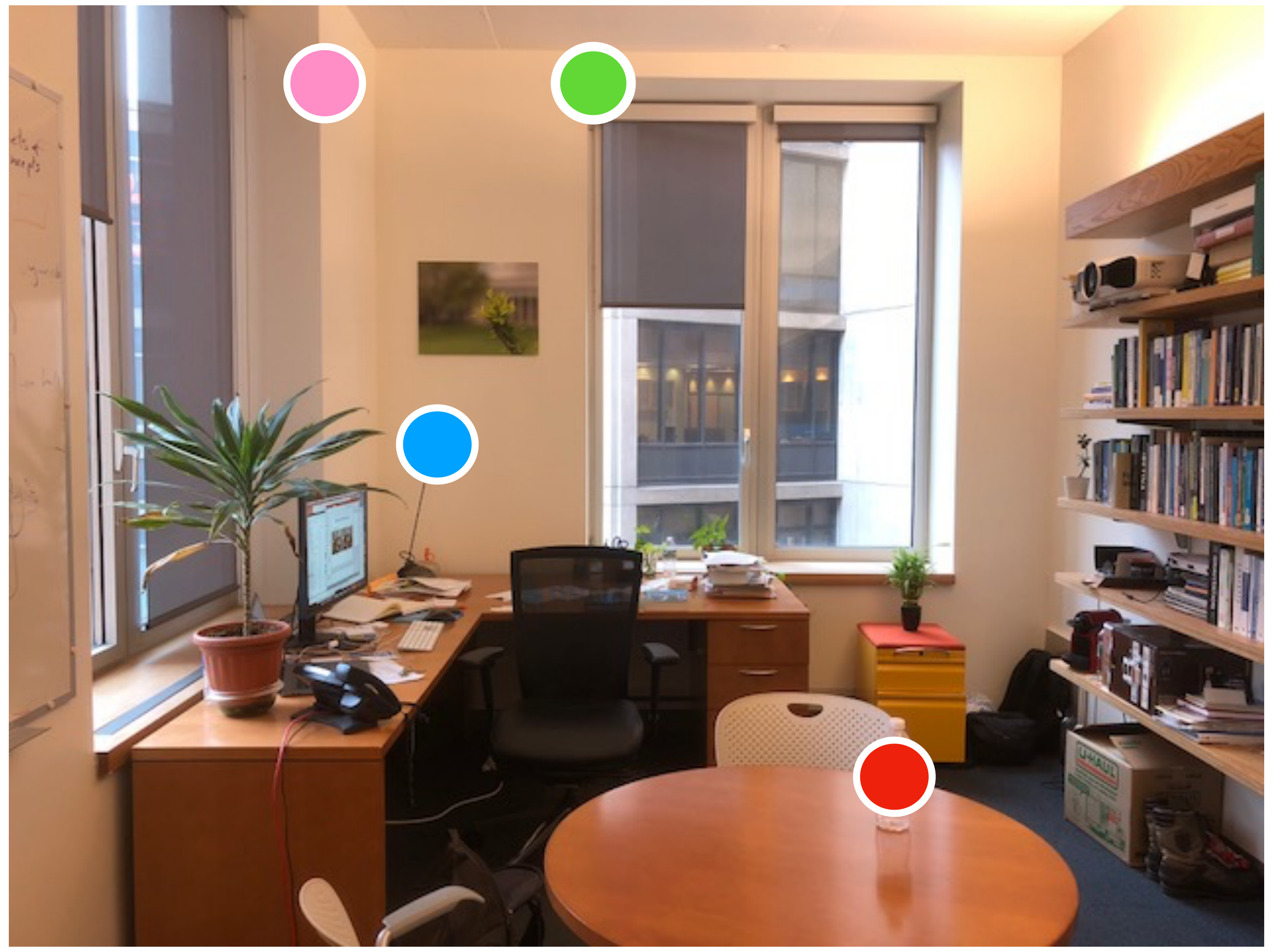
2) extract SIFT
at each keypoint

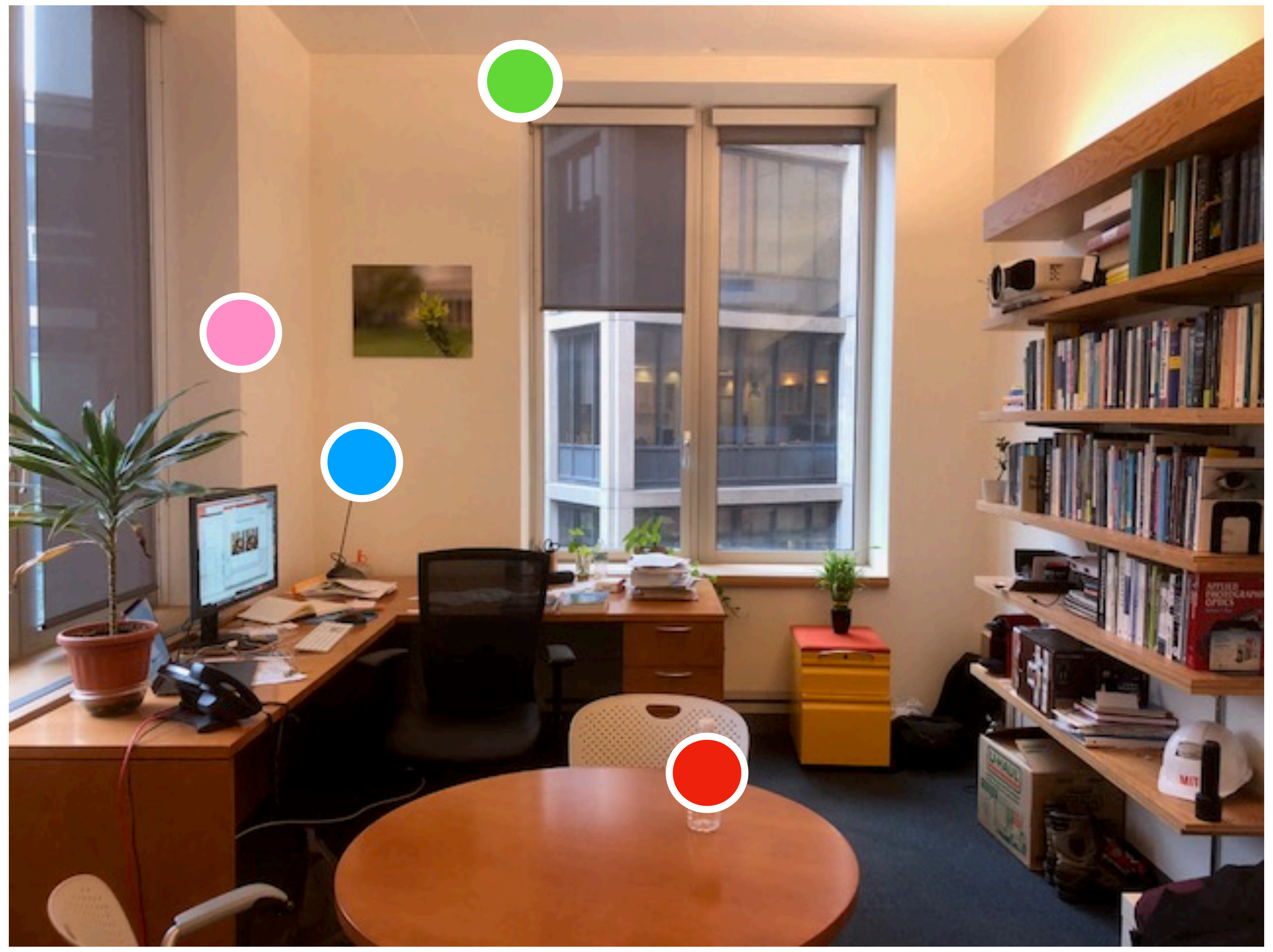


Finding correspondences (SIFT)

Finding correspondences

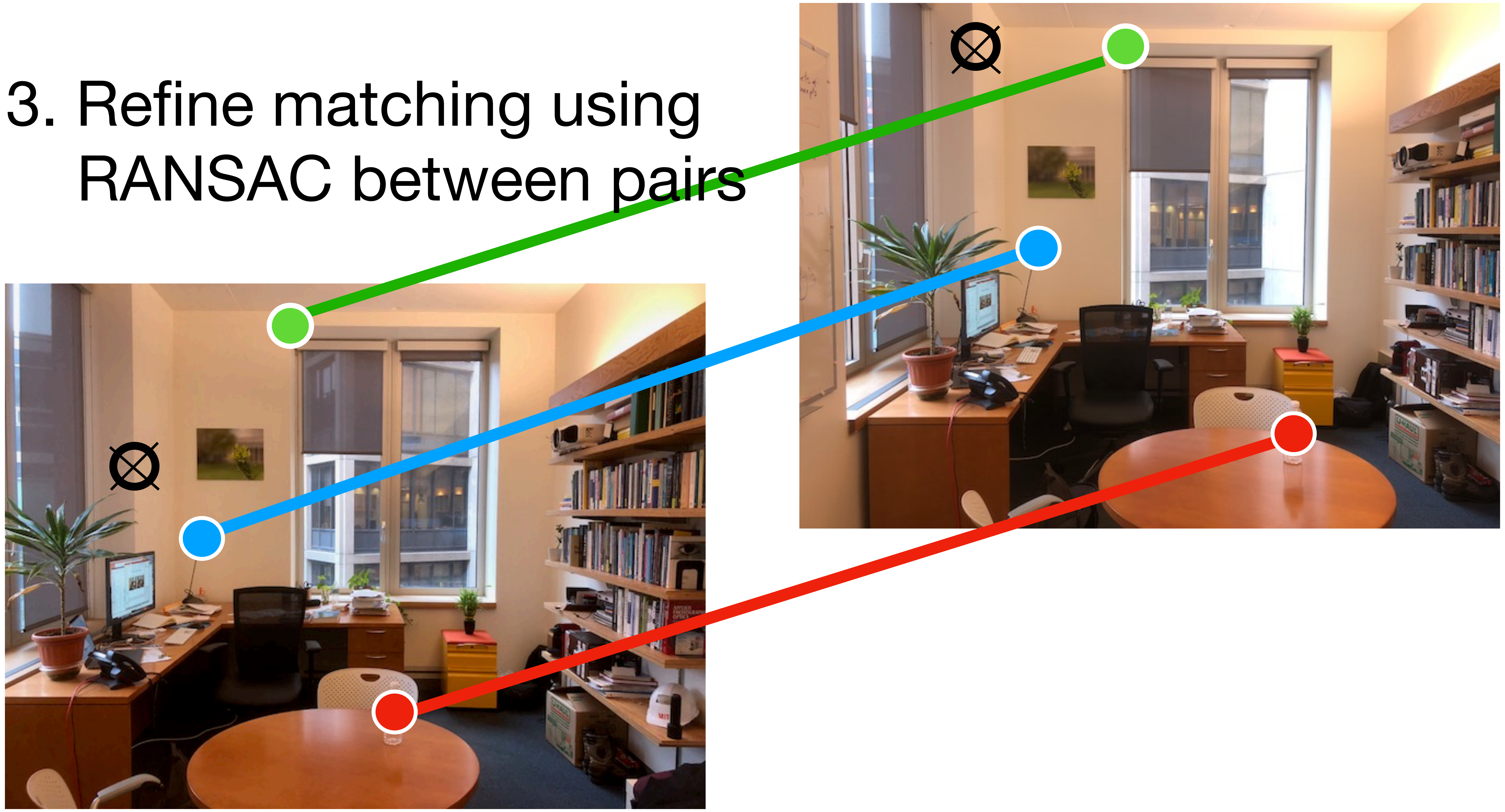
2. Match features between each pair of images



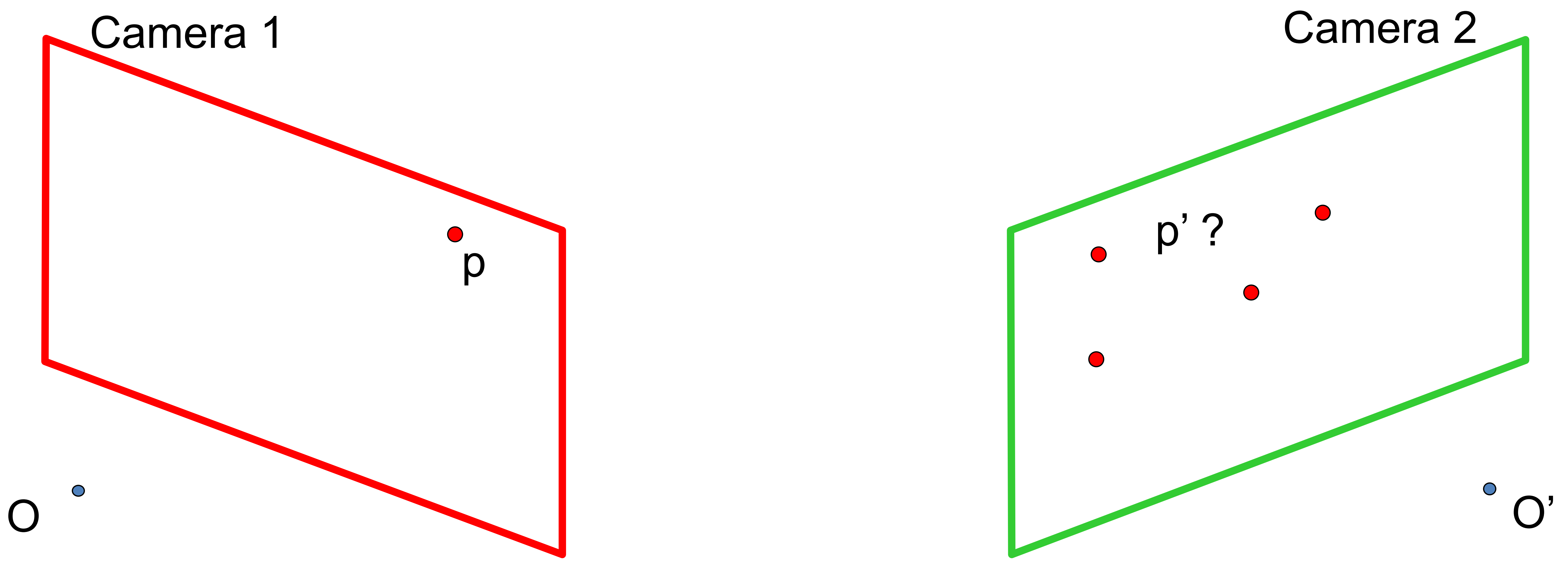


Finding correspondences

3. Refine matching using RANSAC between pairs

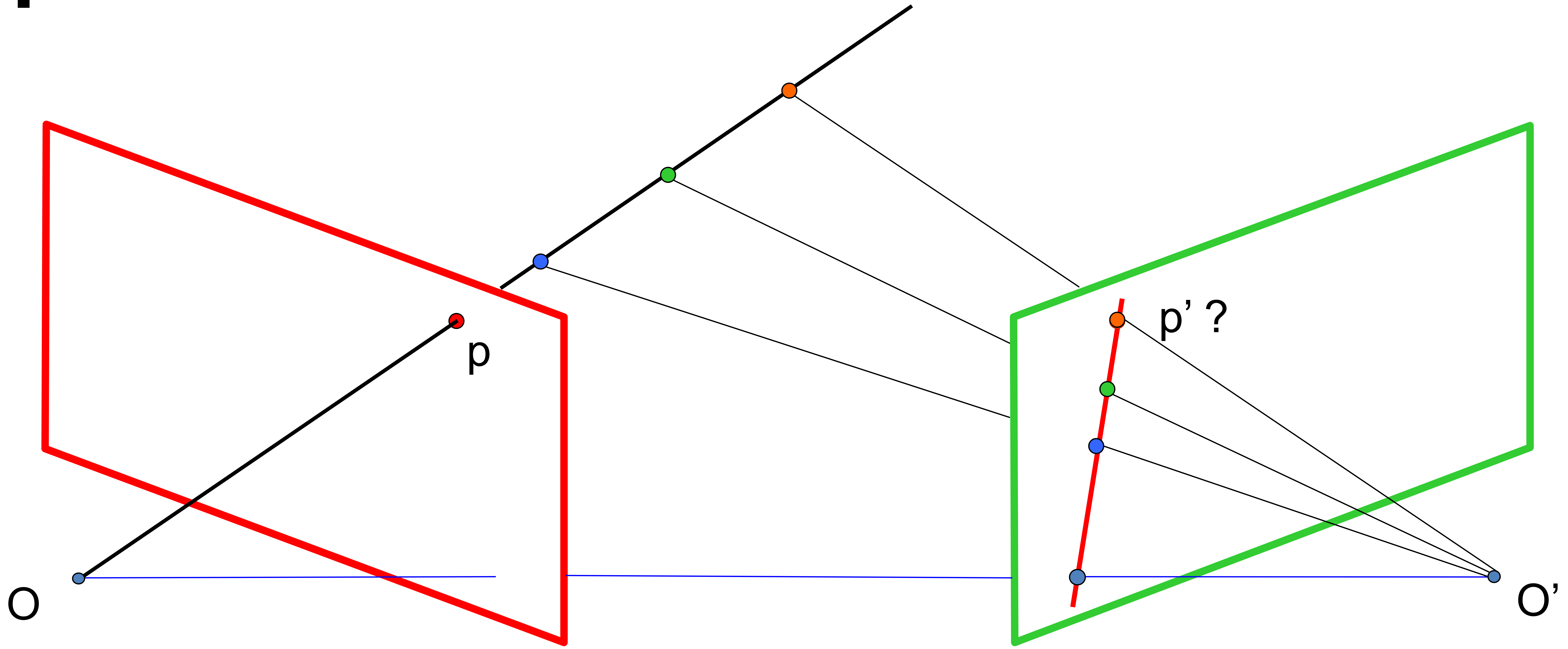


Stereo correspondence constraints



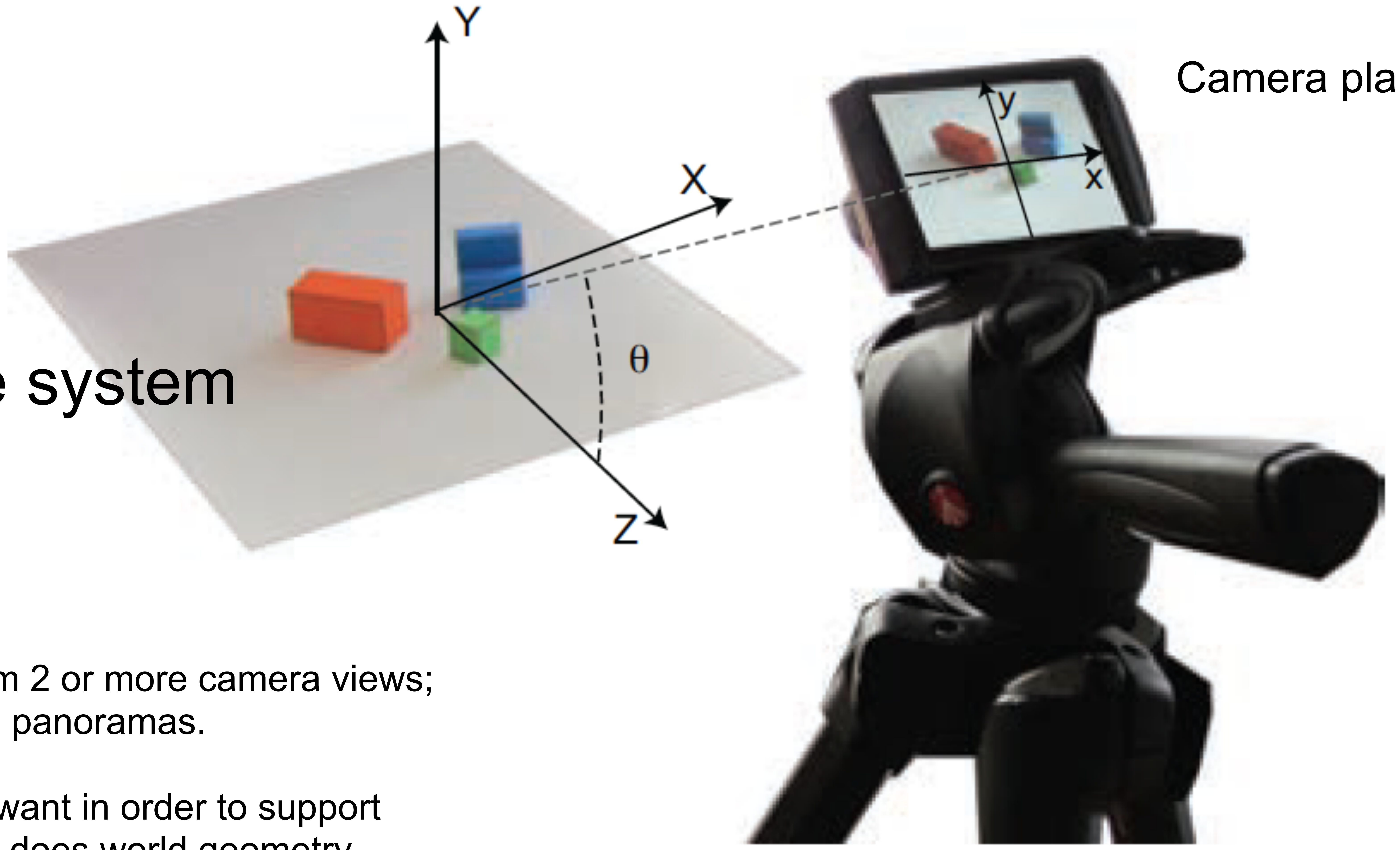
If we see a point in camera 1, are there any constraints on where we will find it on camera 2?

Epipolar constraint



World and camera coordinate systems

Camera plane

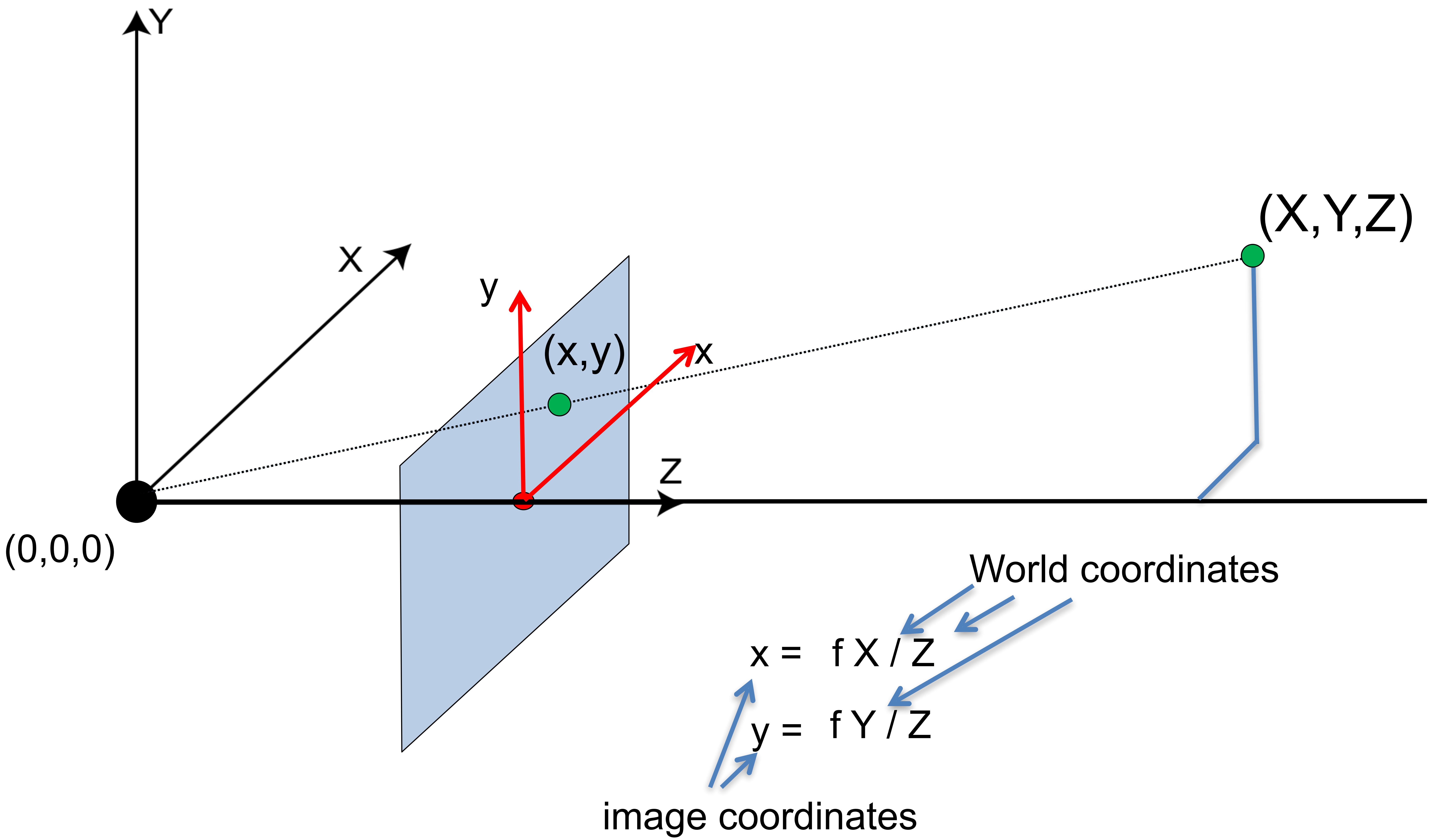


World reference system

Desired applications:
—reconstruct shape from 2 or more camera views;
—align multi-shot image panoramas.

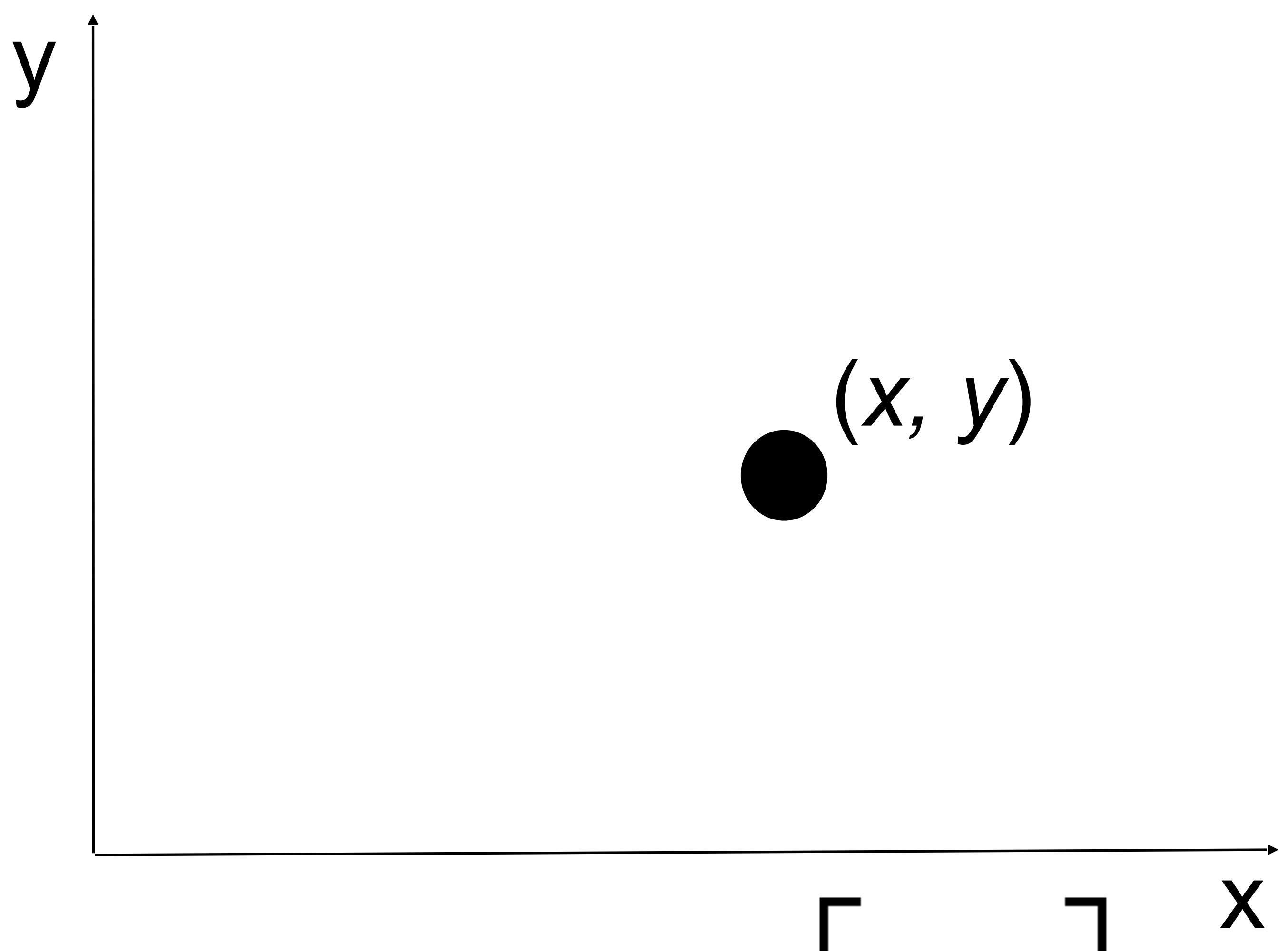
The understanding you want in order to support those applications: how does world geometry relate to pixel positions in a particular camera at a particular position and orientation?

Perspective projection



Homogeneous coordinates

2D



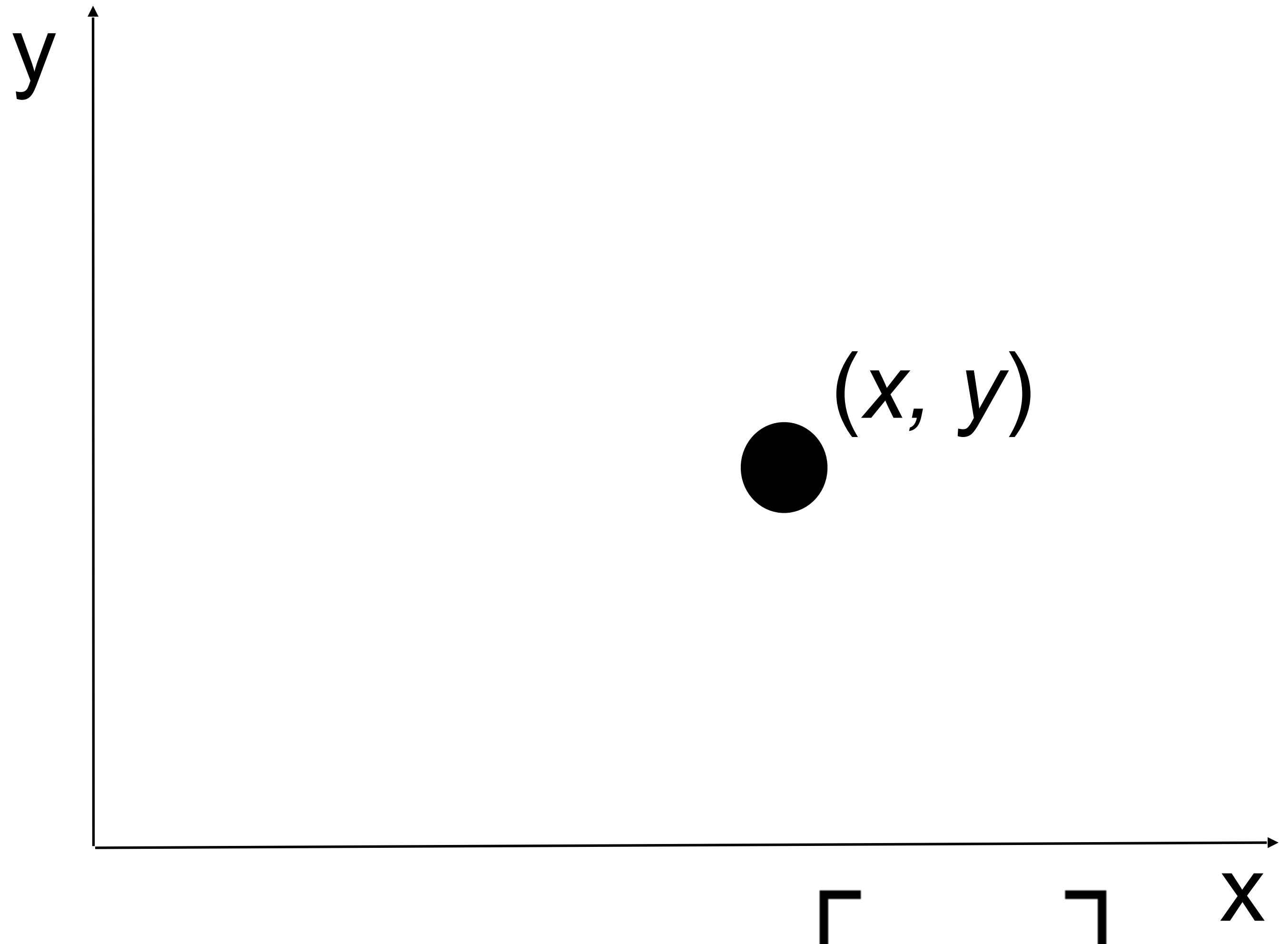
$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

heterogeneous image coordinates

homogeneous image coordinates

Homogeneous coordinates

2D

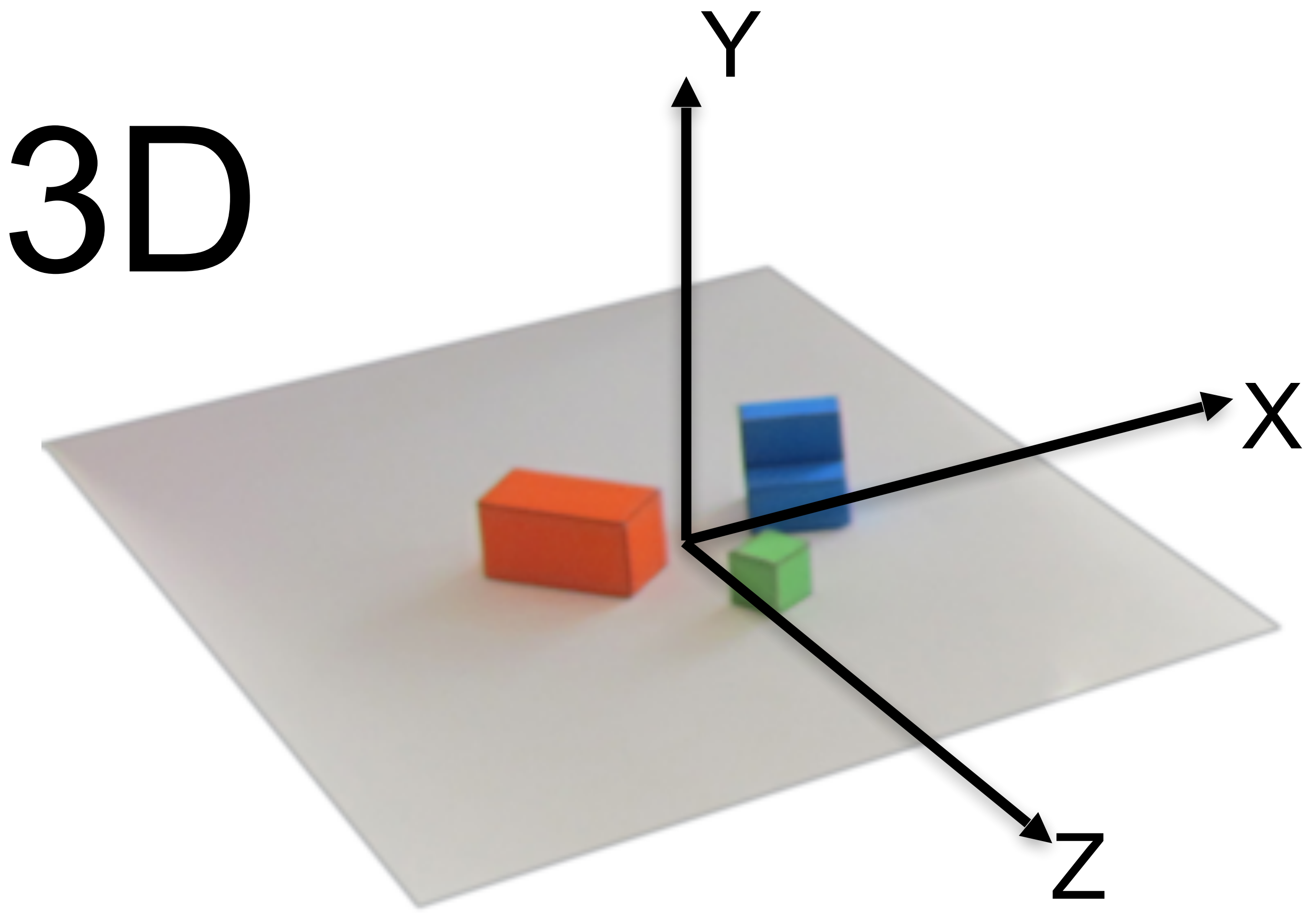


$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

heterogeneous image coordinates

homogeneous image coordinates

3D



$$(x, y, z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Homogeneous coordinates

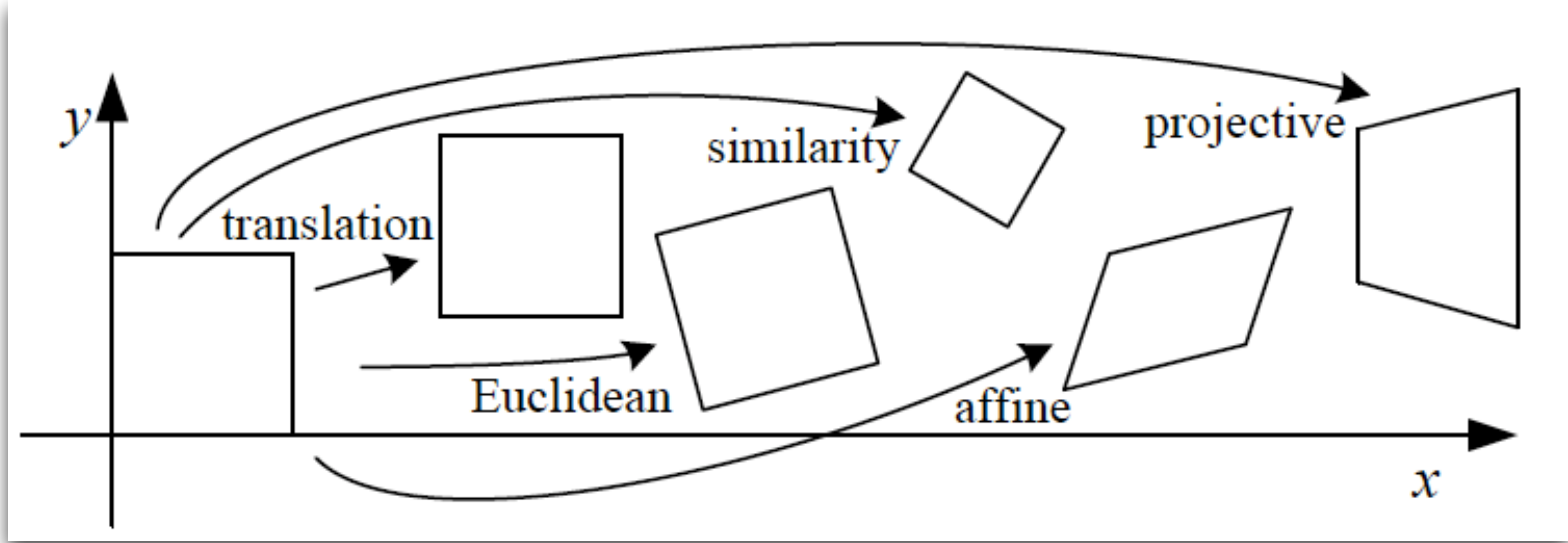
From heterogeneous to homogeneous:

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \text{a } x \\ \text{a } y \\ \text{a} \end{bmatrix}$$

From homogeneous to heterogeneous:

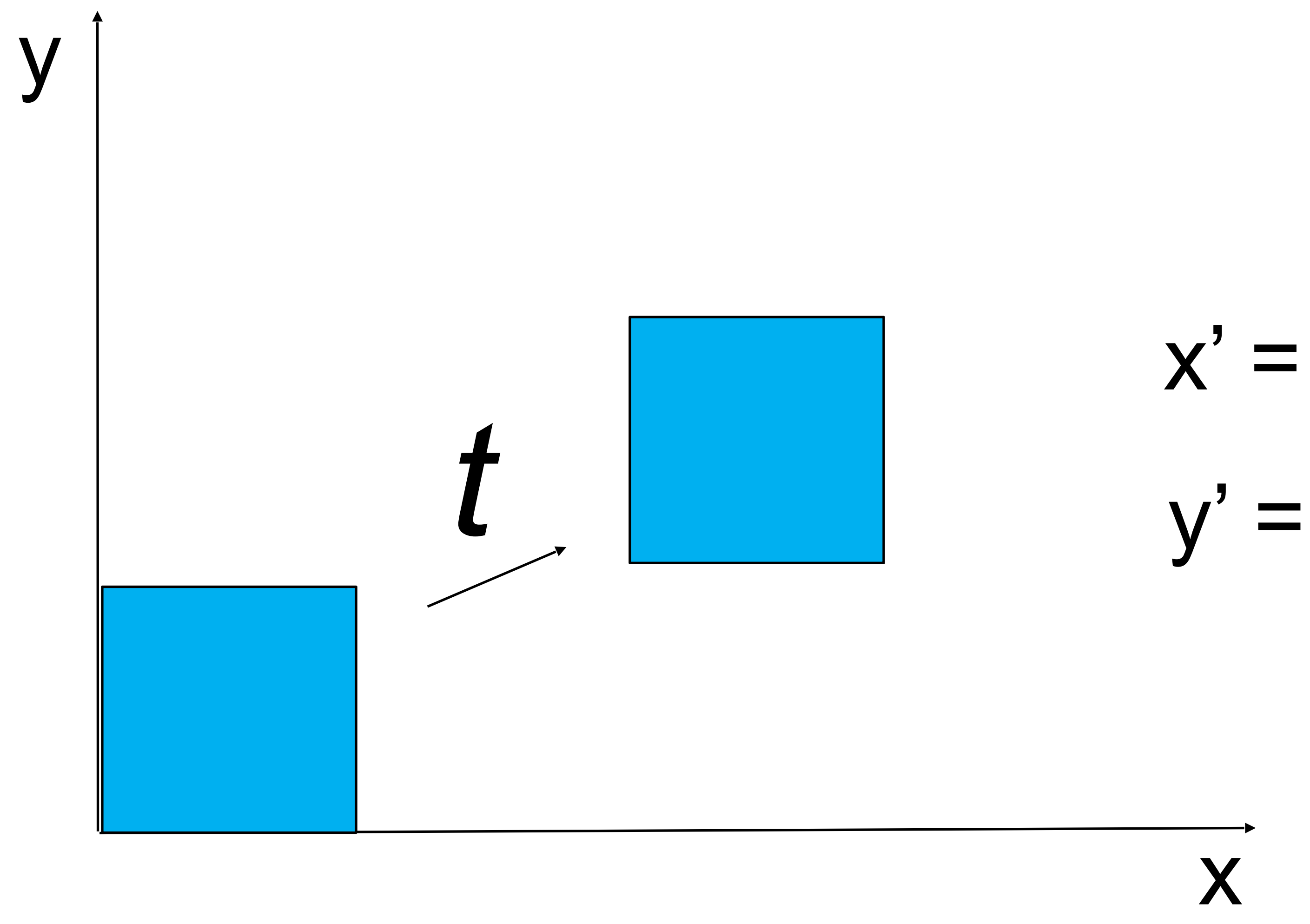
$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)$$

2D Transformations



2D Transformations

Translation

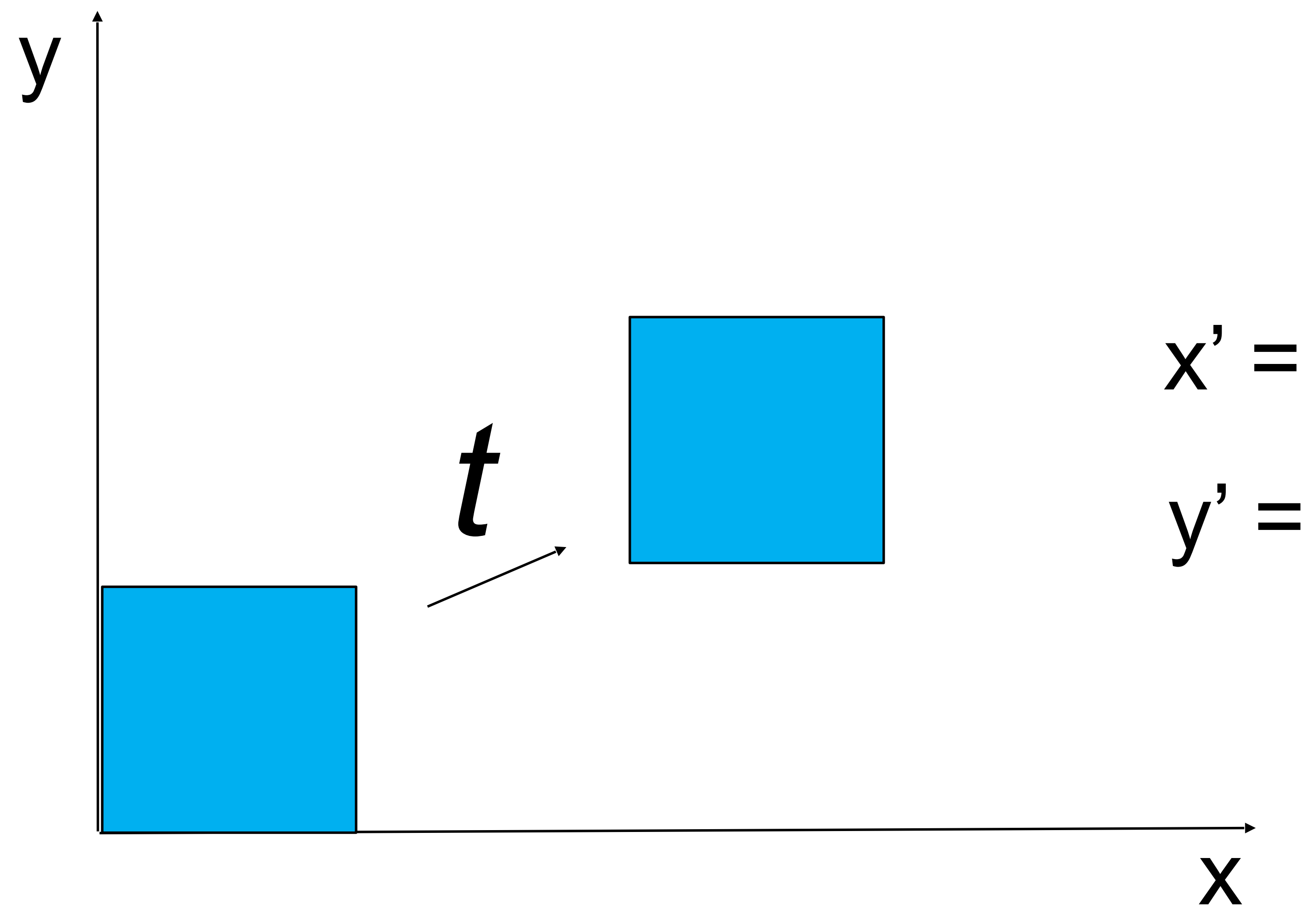


$x' = x + t_x$

$y' = y + t_y$

2D Transformations

Translation



$x' = x + t_x$

$y' = y + t_y$

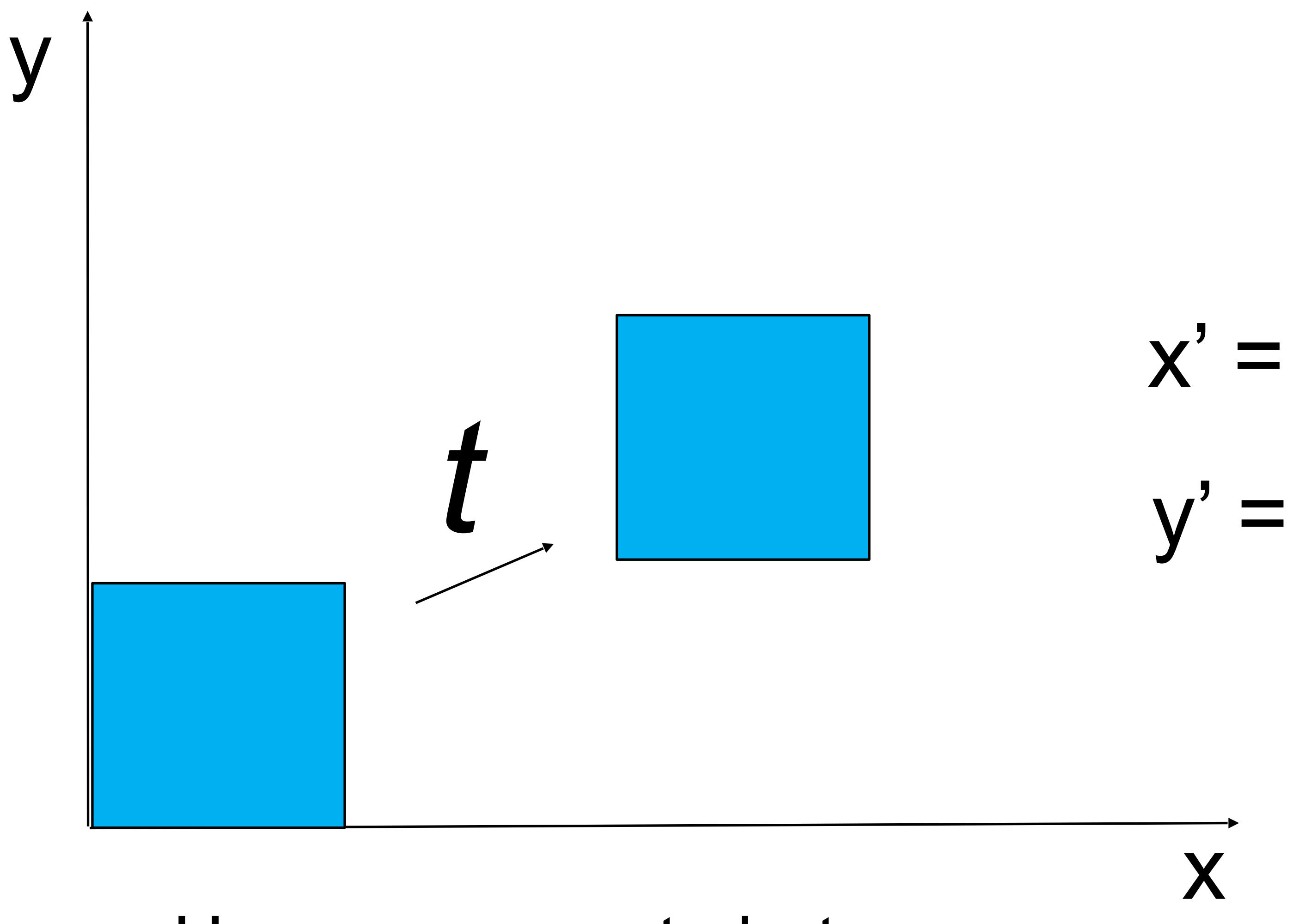
Heterogeneous

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$$\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{t}$$

2D Transformations

Translation



$x' = x + t_x$

$y' = y + t_y$

Heterogeneous

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$$x' = x + t$$

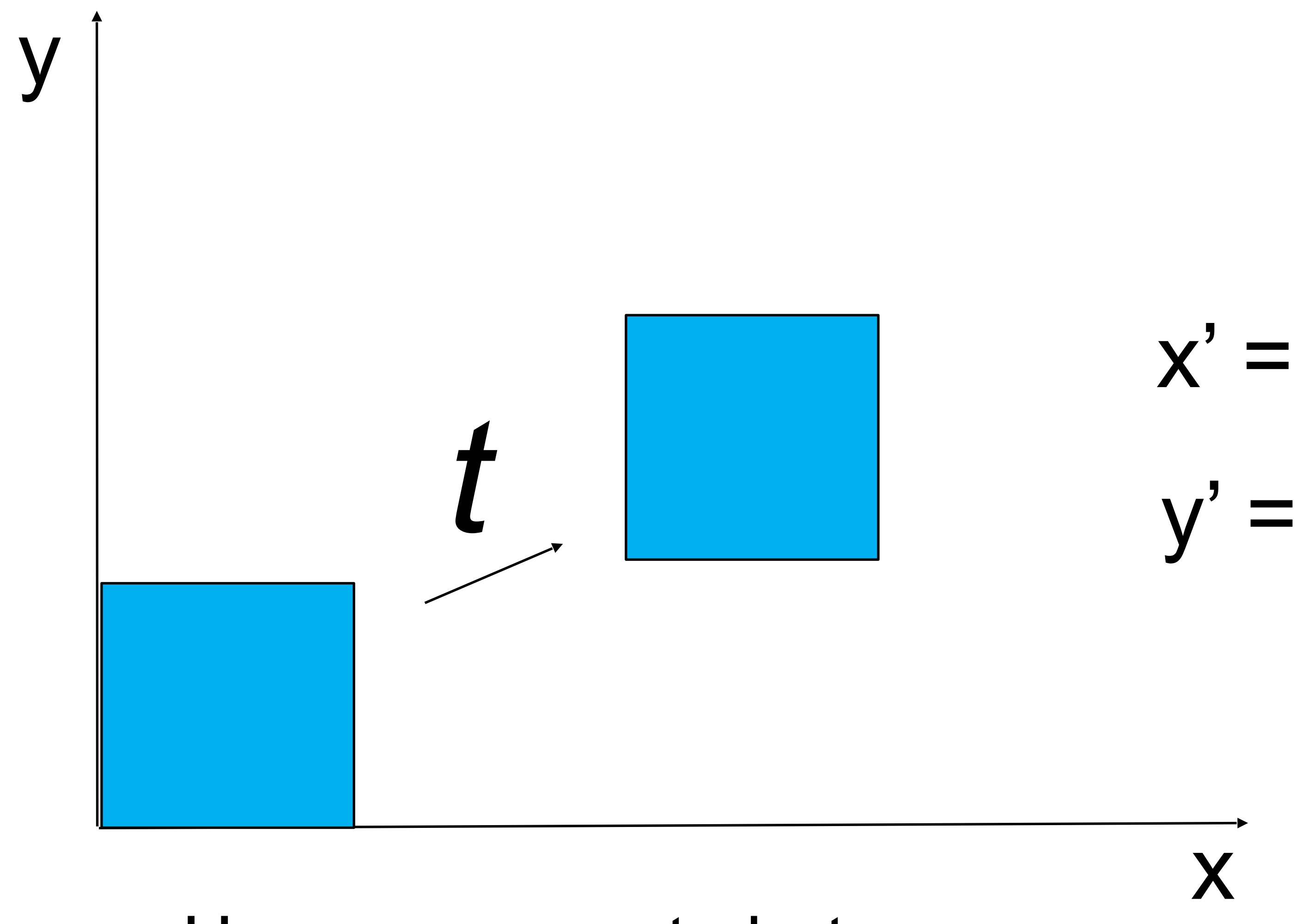
Homogeneous to heterogeneous

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$x' = \begin{bmatrix} I & t \end{bmatrix} \bar{x}$$

2D Transformations

Translation



$x' = x + t_x$

$y' = y + t_y$

Heterogeneous

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$$\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{t}$$

Homogeneous to heterogeneous

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\boldsymbol{x}' = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{t} \end{bmatrix} \bar{\boldsymbol{x}}$$

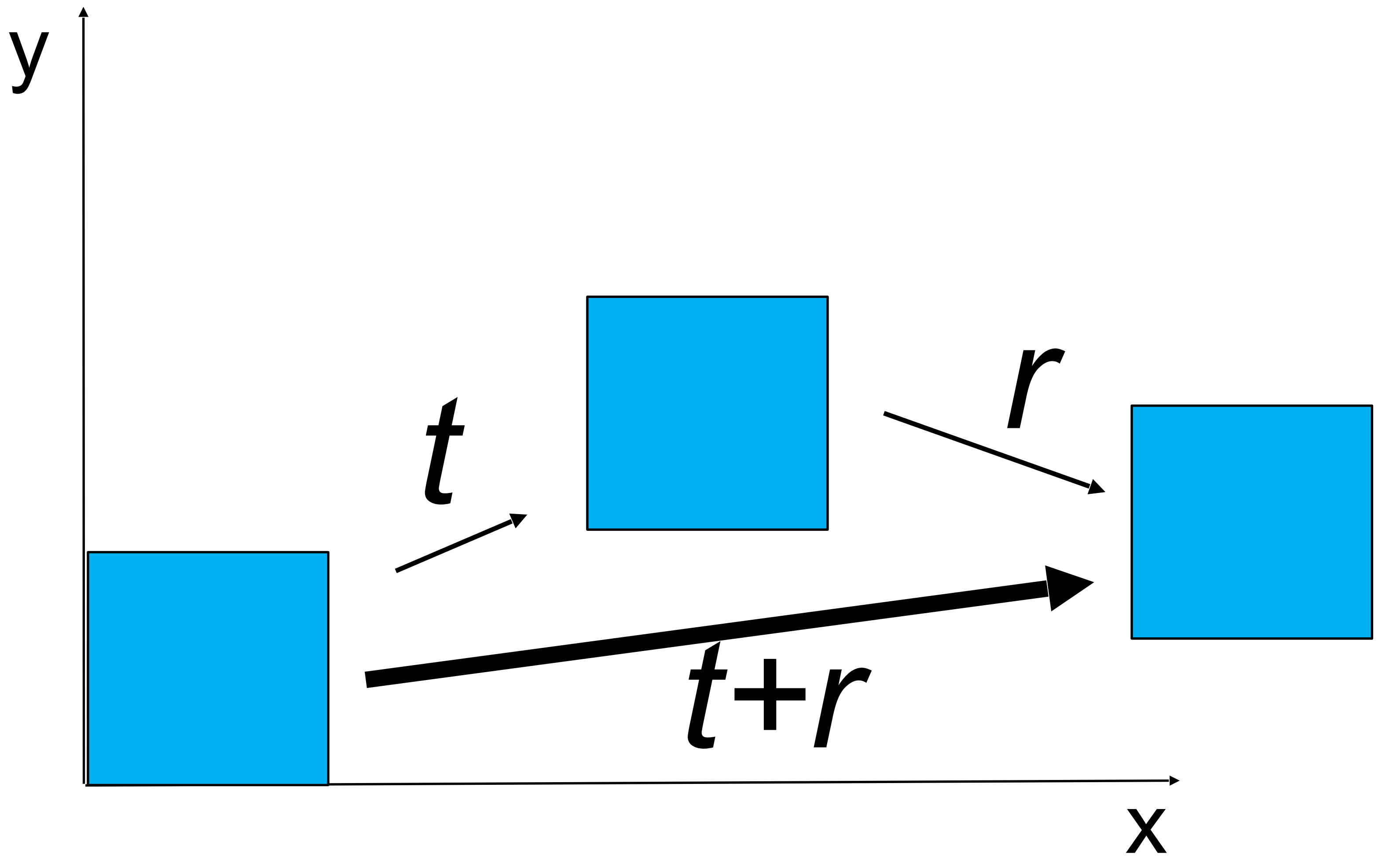
Homogeneous

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\bar{\boldsymbol{x}}' = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{t} \\ 0^T & 1 \end{bmatrix} \bar{\boldsymbol{x}}$$

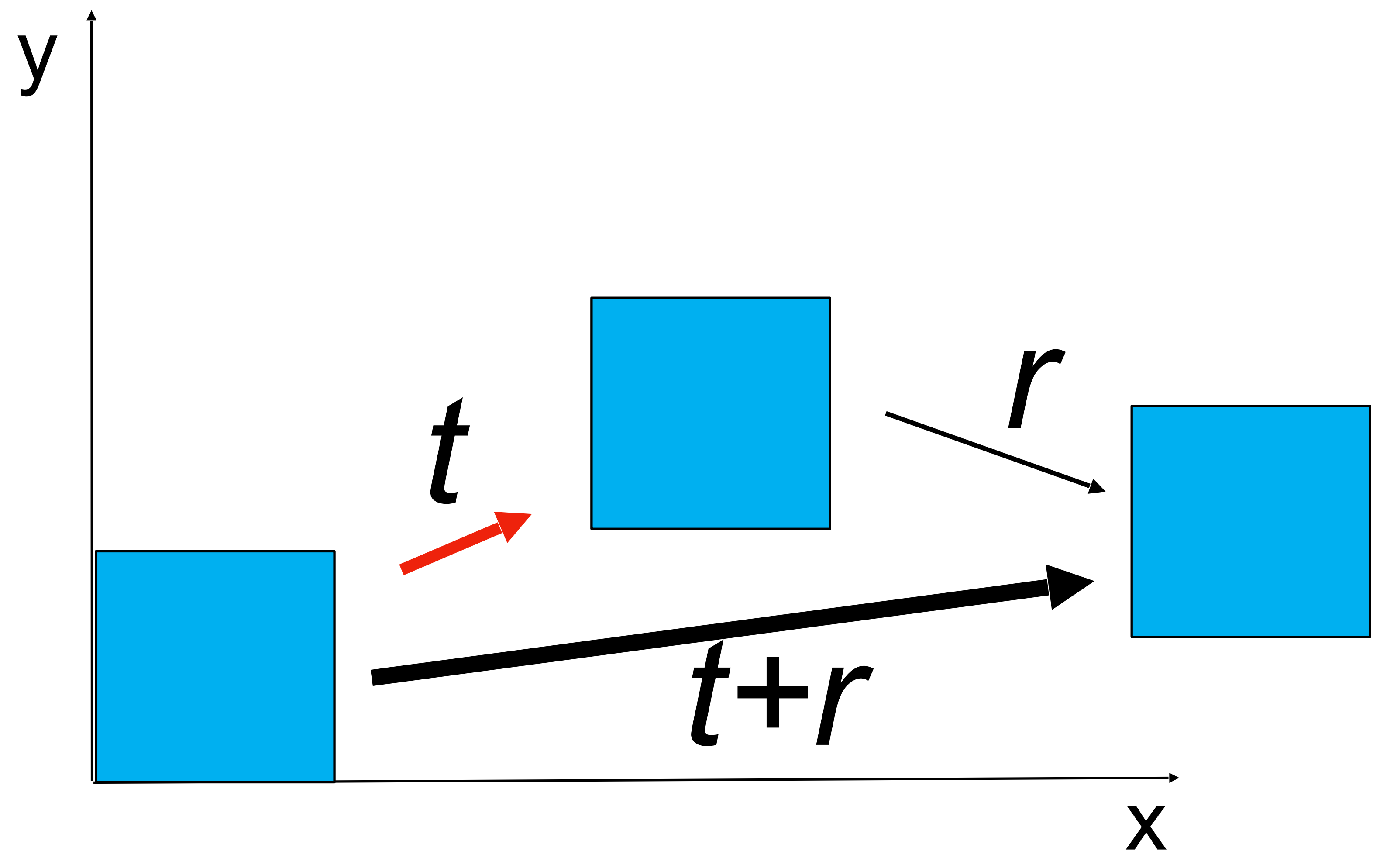
2D Transformations

Translation



2D Transformations

Translation

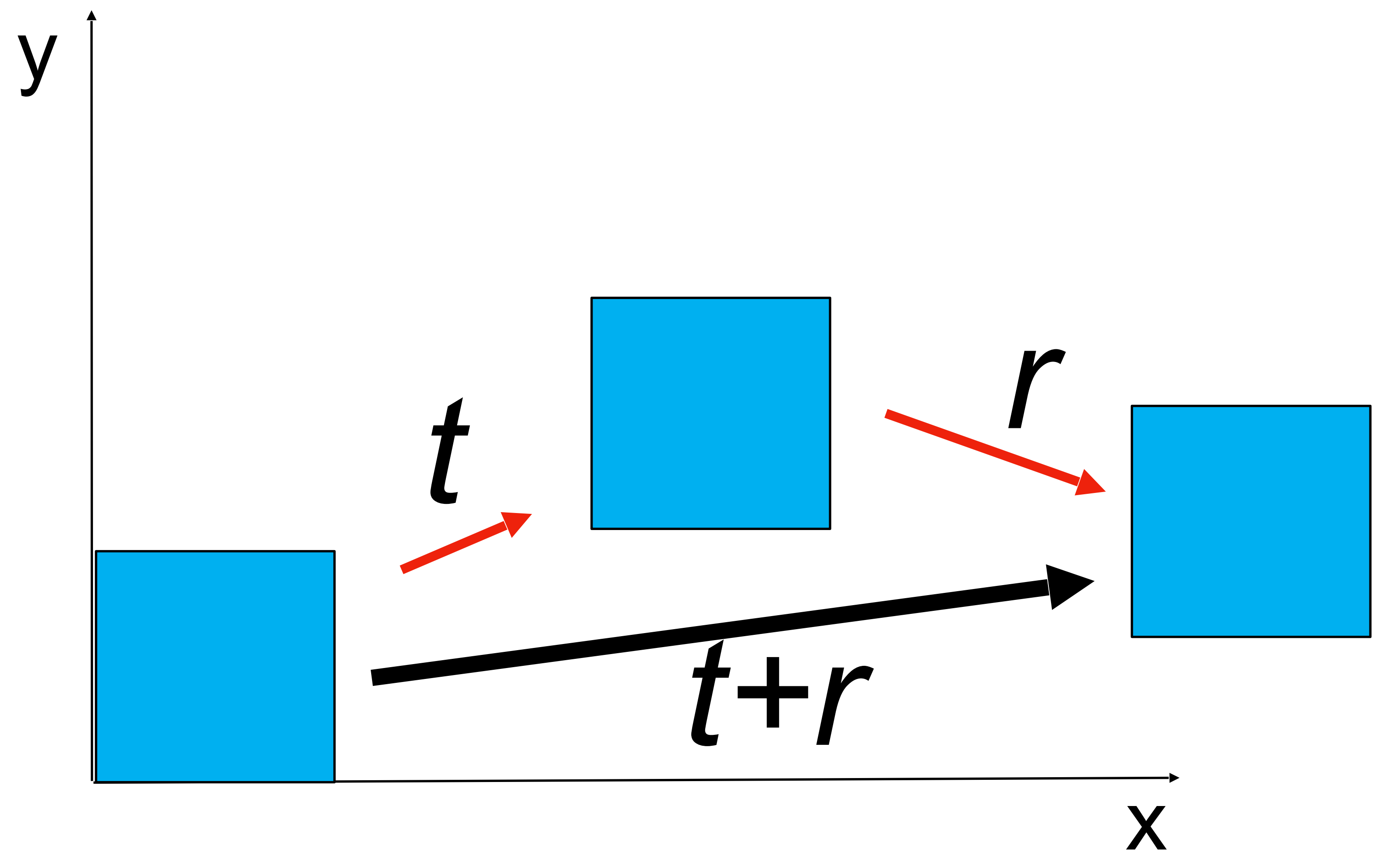


Now we can chain transformations

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

2D Transformations

Translation



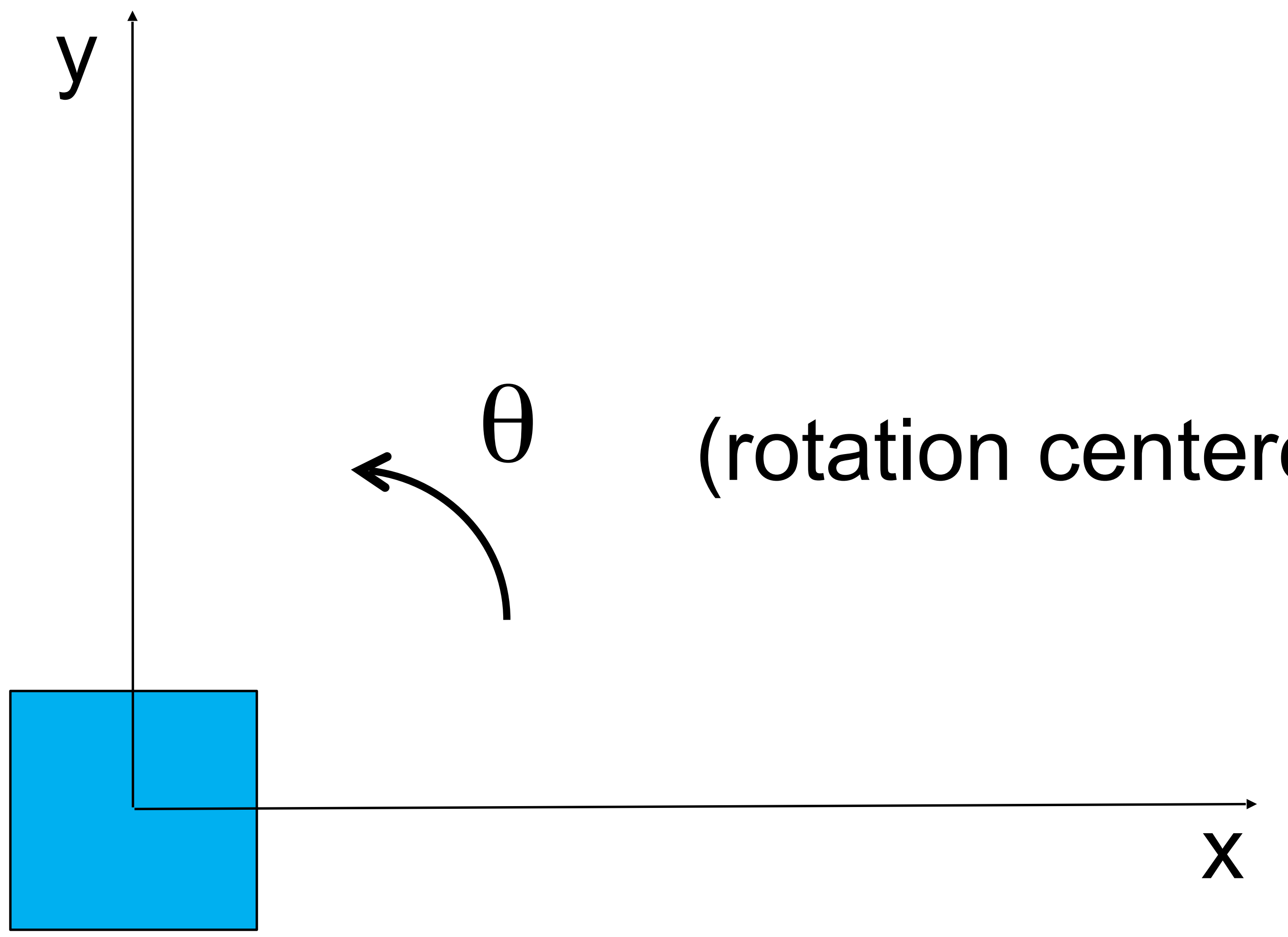
Now we can chain transformations

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & r_x \\ 0 & 1 & r_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$x' = RTx$$

2D Transformations

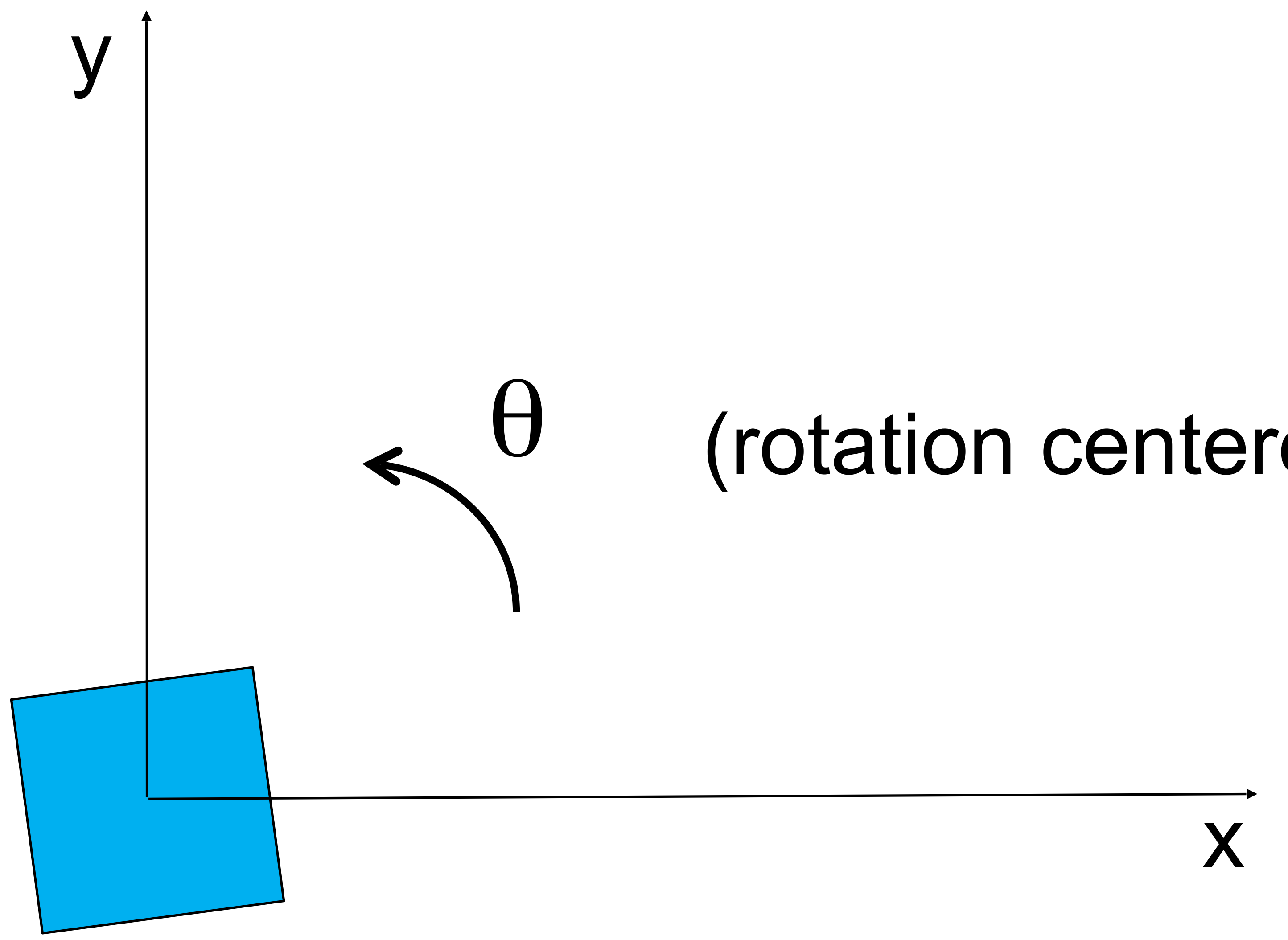
Rotation



(rotation centered on the origin)

2D Transformations

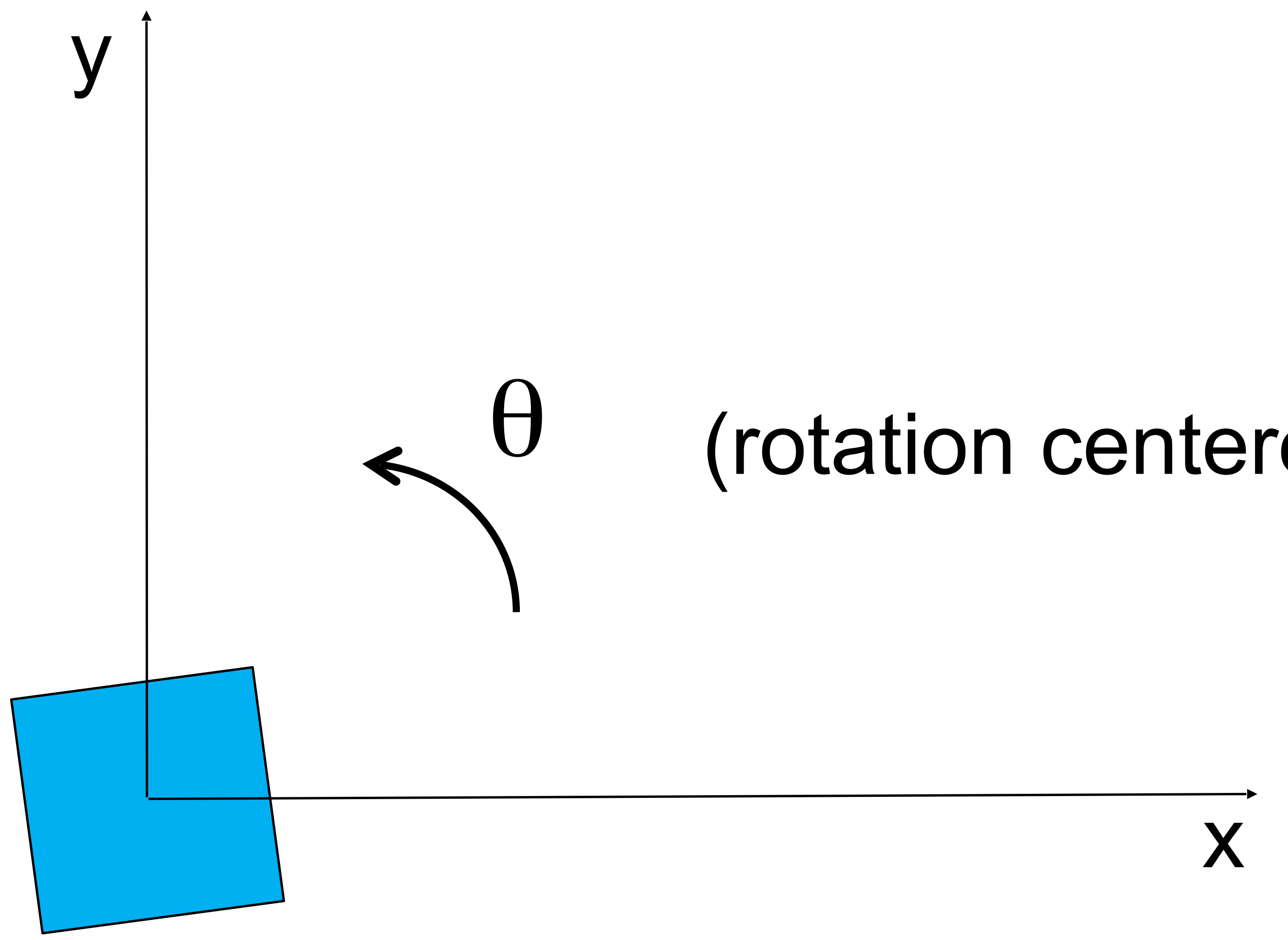
Rotation



(rotation centered on the origin)

2D Transformations

Rotation

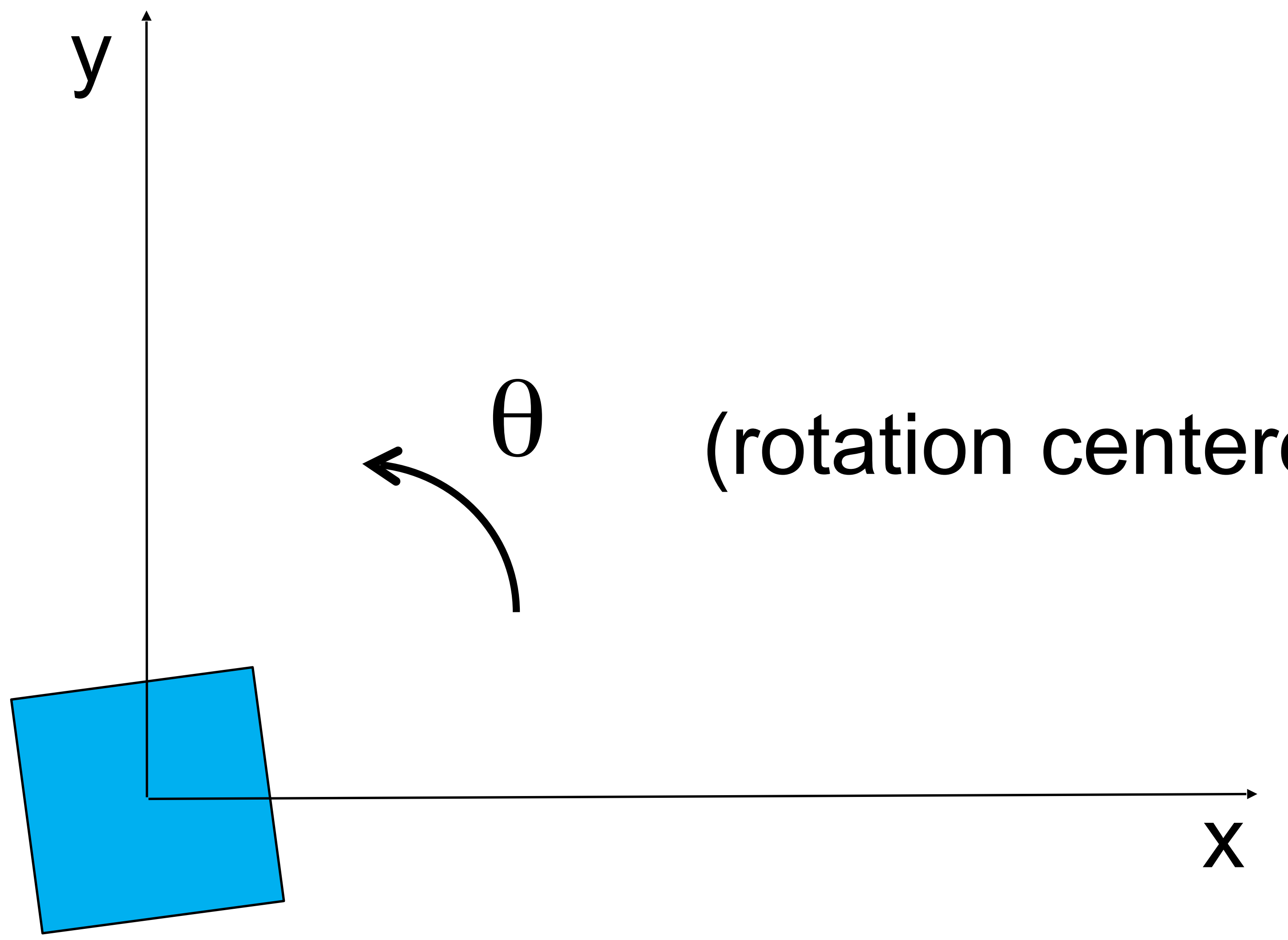


(rotation centered on the origin)

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} ? & ? & ? \\ ? & ? & ? \\ ? & ? & ? \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

2D Transformations

Rotation

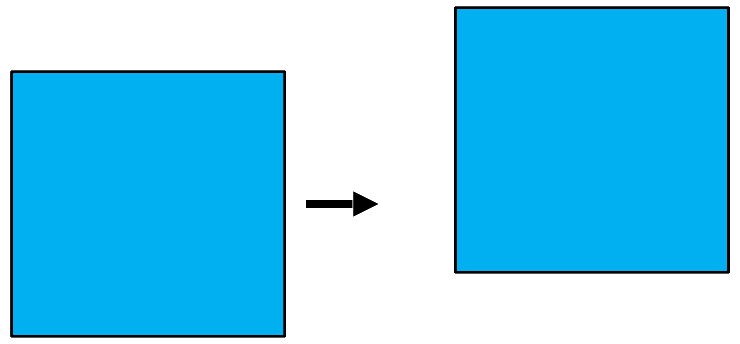


(rotation centered on the origin)

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

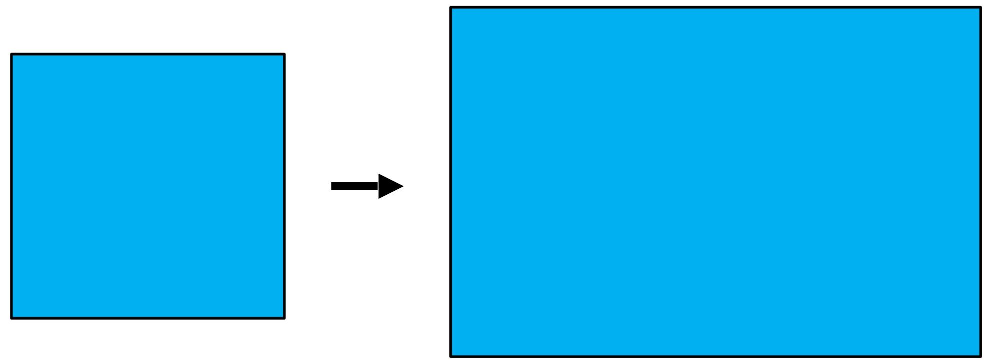
2D Transformations

Translation

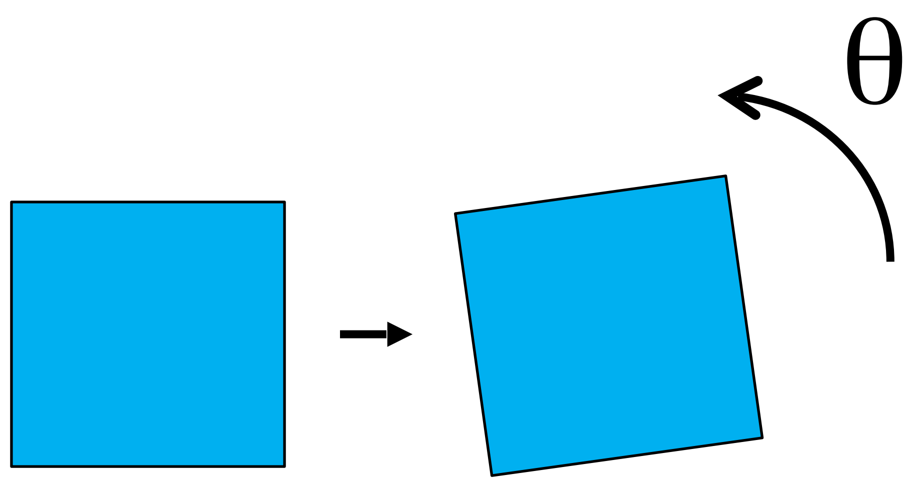


$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Scaling

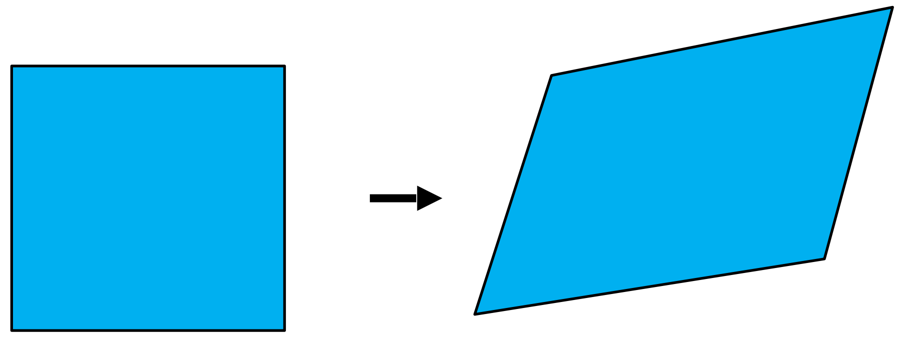


Rotation



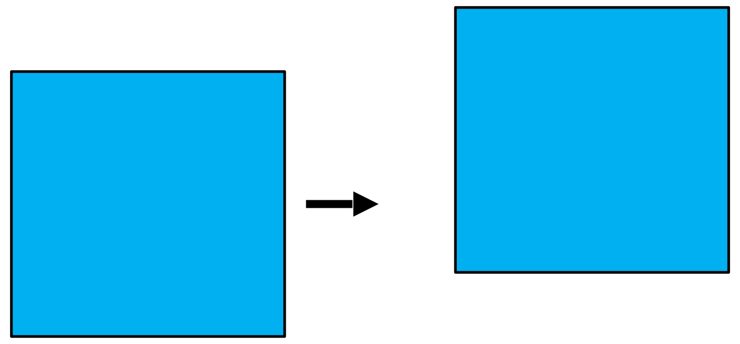
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Shearing



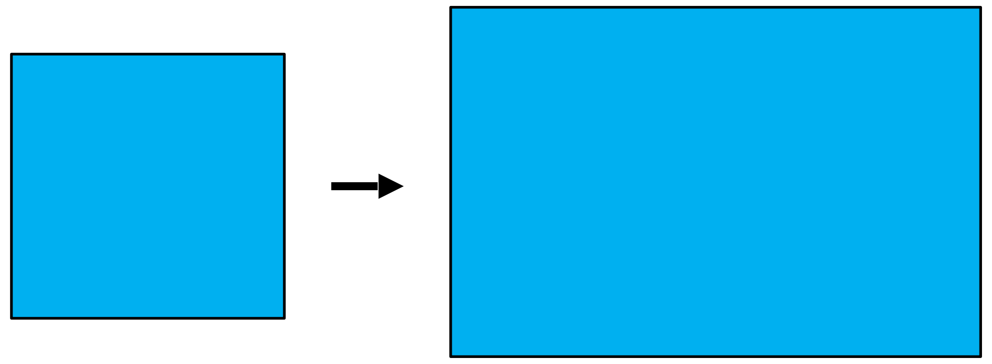
2D Transformations

Translation



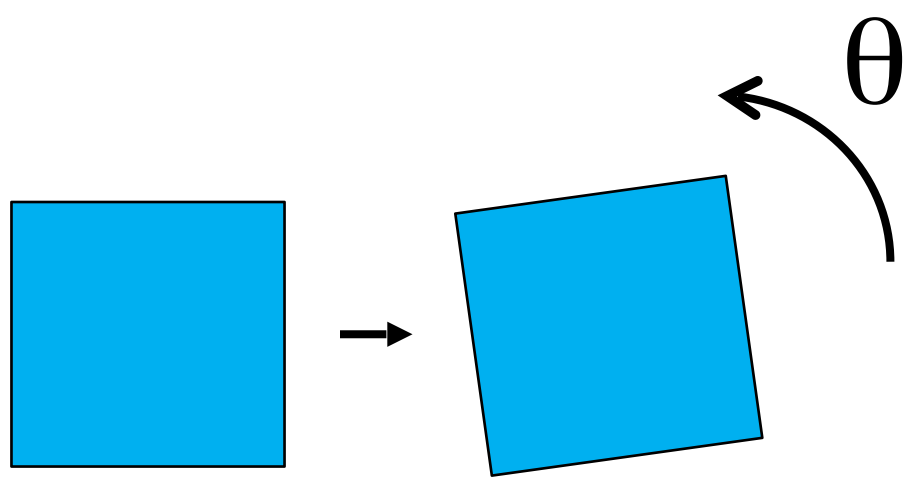
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Scaling



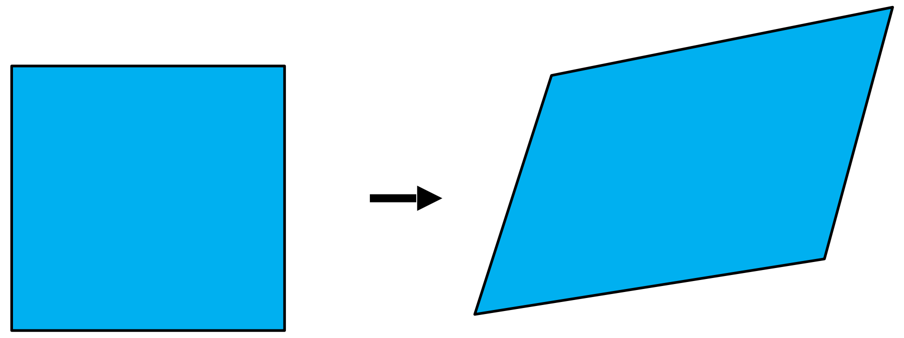
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Rotation



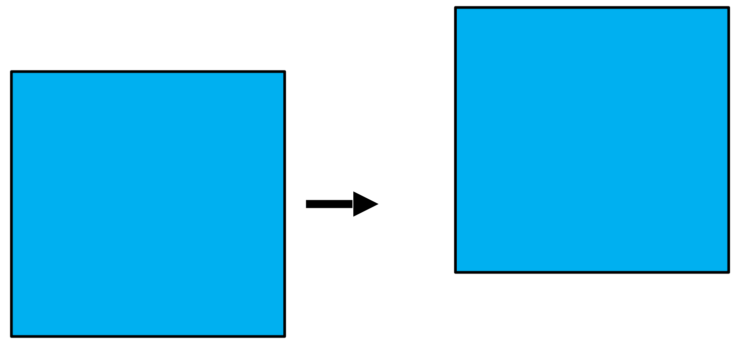
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Shearing



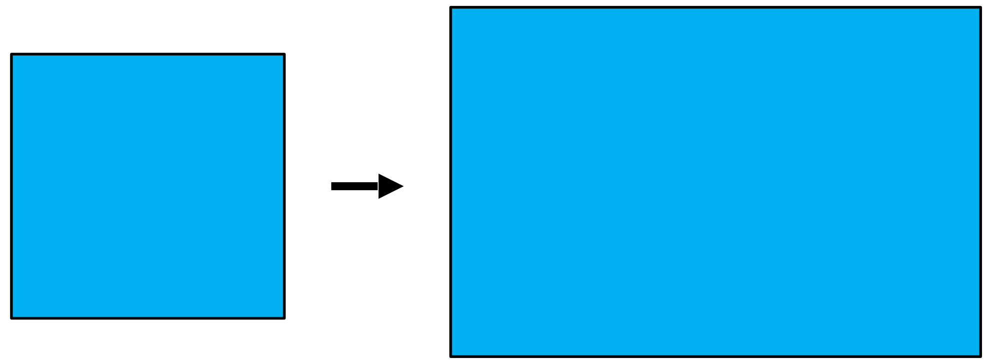
2D Transformations

Translation



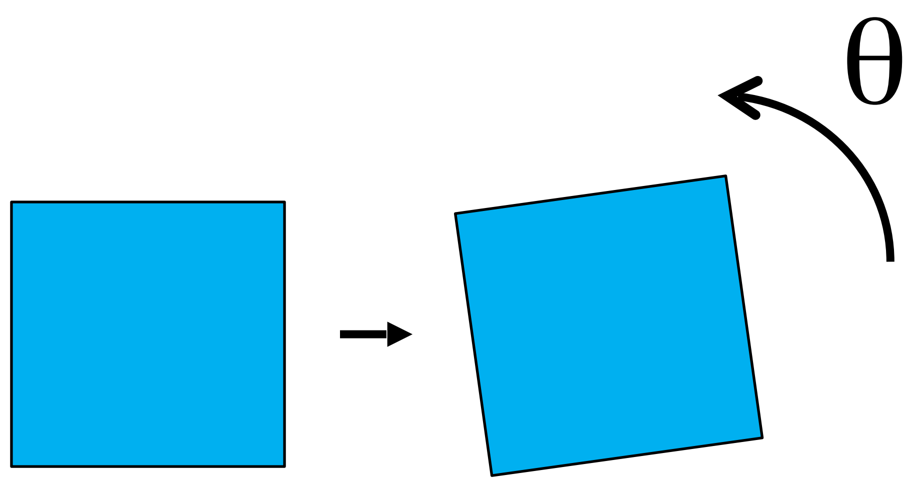
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Scaling



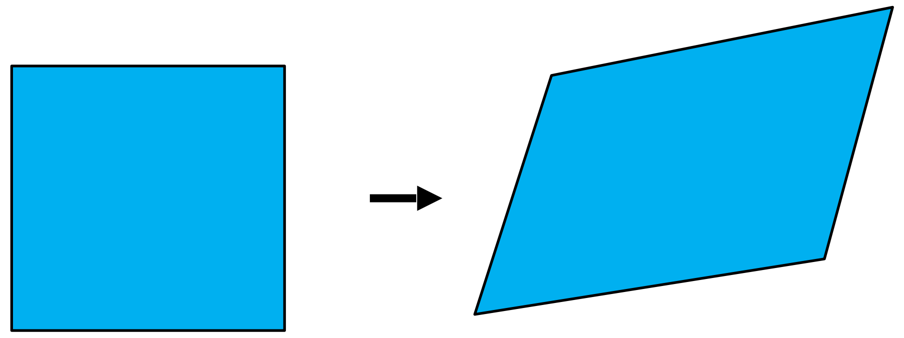
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Rotation



$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

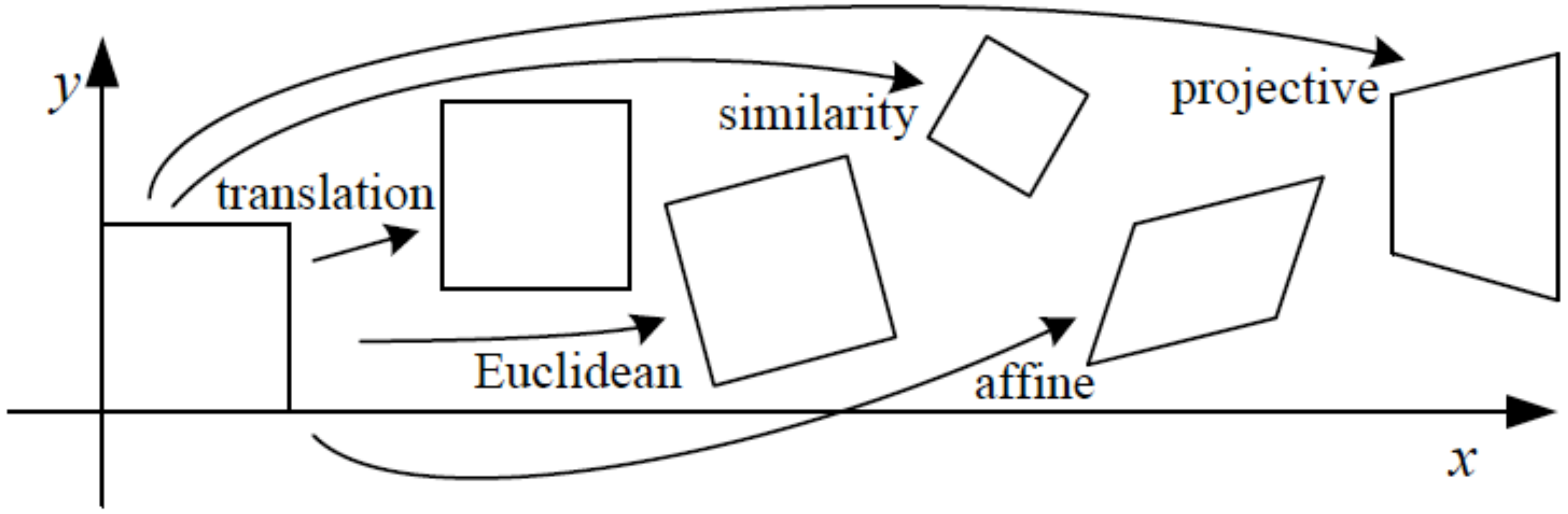
Shearing



$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & a_x & 0 \\ a_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

2D Transformations

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & a_x & 0 \\ a_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$



Euclidean = rotation and translation

Similarity = rotation, translation and uniform scaling ($s_x = s_y$)

Affine = rotation, translation, shearing and uniform scaling ($s_x = s_y$)

2D Transformations

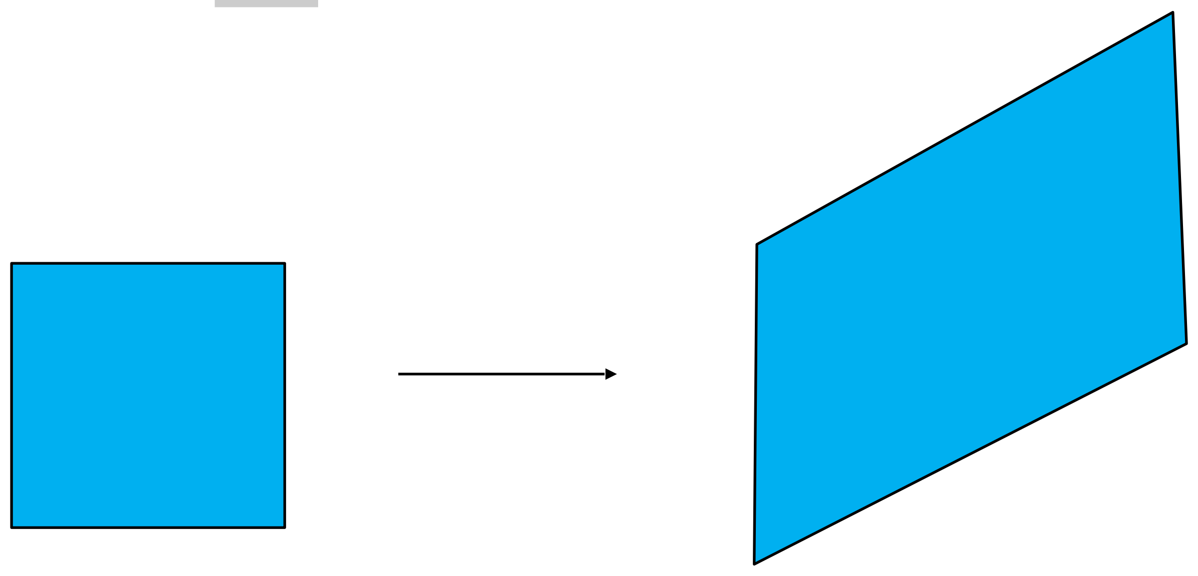
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & a_x & 0 \\ a_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Affine = rotation, translation, shearing and uniform scaling ($s_x = s_y$)

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

Properties:

- 6 degrees of freedom
- Parallel lines remain parallel



Homogeneous coordinates

2D

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

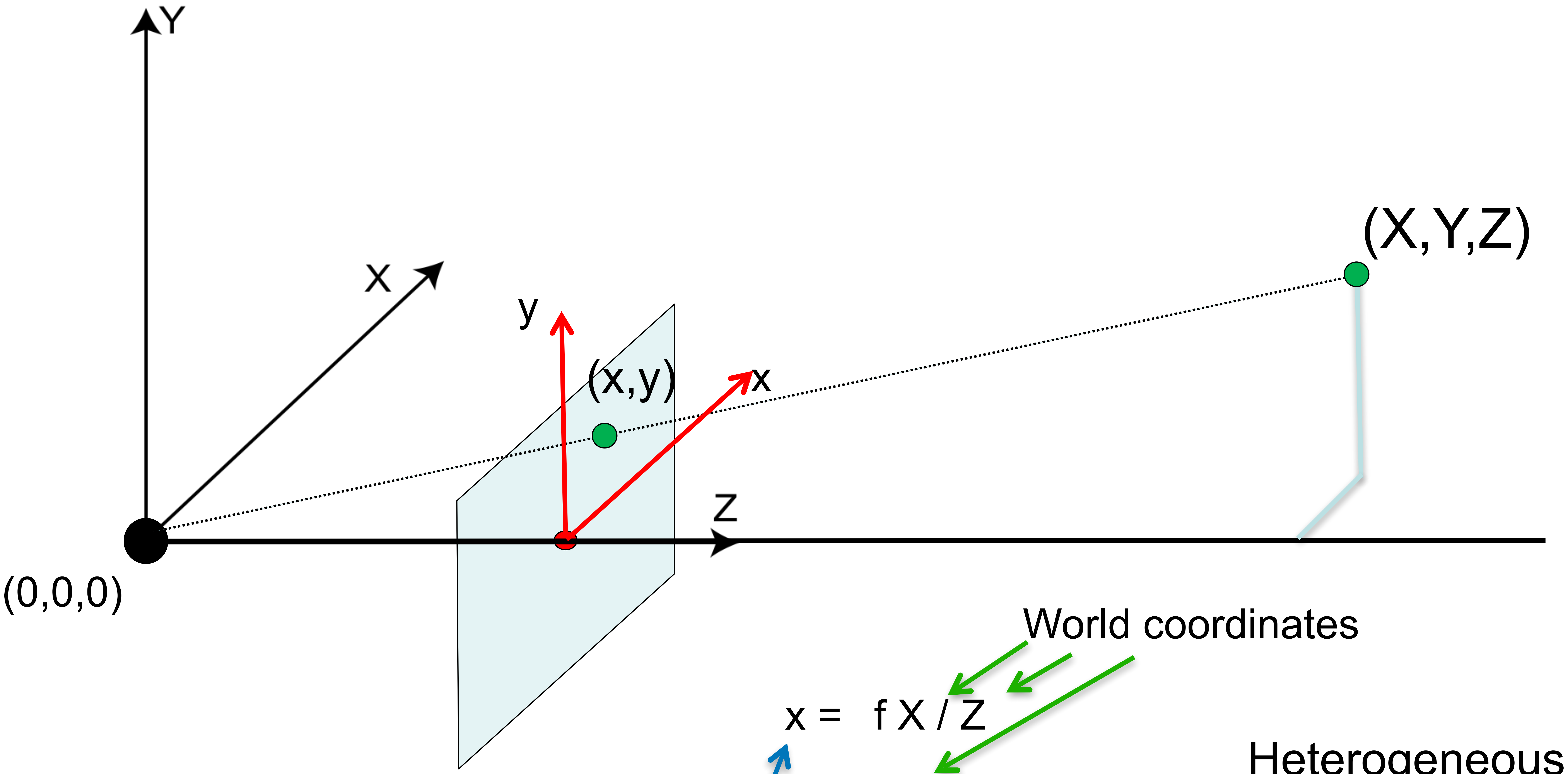
$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)$$

3D

$$(x, y, z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \Rightarrow (x/w, y/w, z/w)$$

Perspective projection



World coordinates

$x = f X / Z$

$y = f Y / Z$

image coordinates

Heterogeneous coordinates

Perspective projection

Heterogeneous coordinates

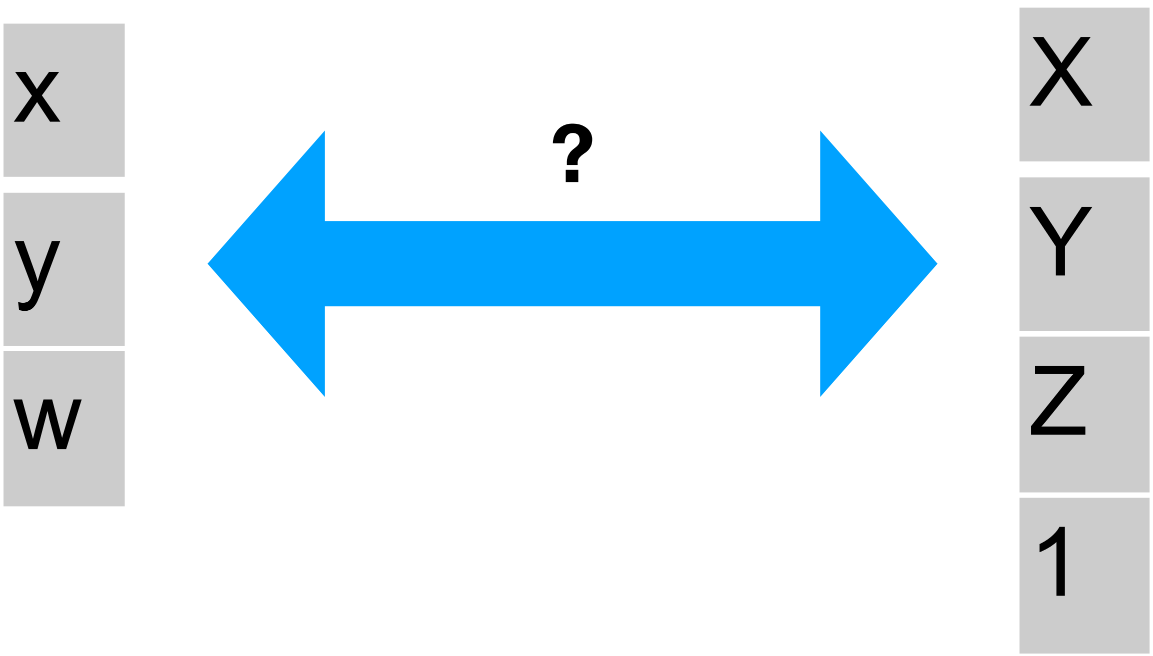
World coordinates

$$x = f X / Z$$

$$y = f Y / Z$$

image coordinates

Homogeneous coordinates



Perspective projection

Heterogeneous coordinates

World coordinates

x = f X / Z

y = f Y / Z

image coordinates

Homogeneous coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Perspective projection

Heterogeneous coordinates

World coordinates

$$x = f X / Z$$

$$y = f Y / Z$$

image coordinates

Homogeneous coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f & 0 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Perspective projection

Heterogeneous coordinates

World coordinates

$$x = f X / Z$$

$$y = f Y / Z$$

image coordinates

Homogeneous coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f & 0 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z/f \end{bmatrix}$$

Perspective projection

Heterogeneous coordinates

World coordinates

$x = f X / Z$

$y = f Y / Z$

image coordinates

Homogeneous coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f & 0 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z/f \end{bmatrix}$$

Going back to heterogeneous coordinates:

$$\begin{bmatrix} X \\ Y \\ Z/f \end{bmatrix} \longrightarrow (f\, X/Z,\ f\, Y/Z)$$

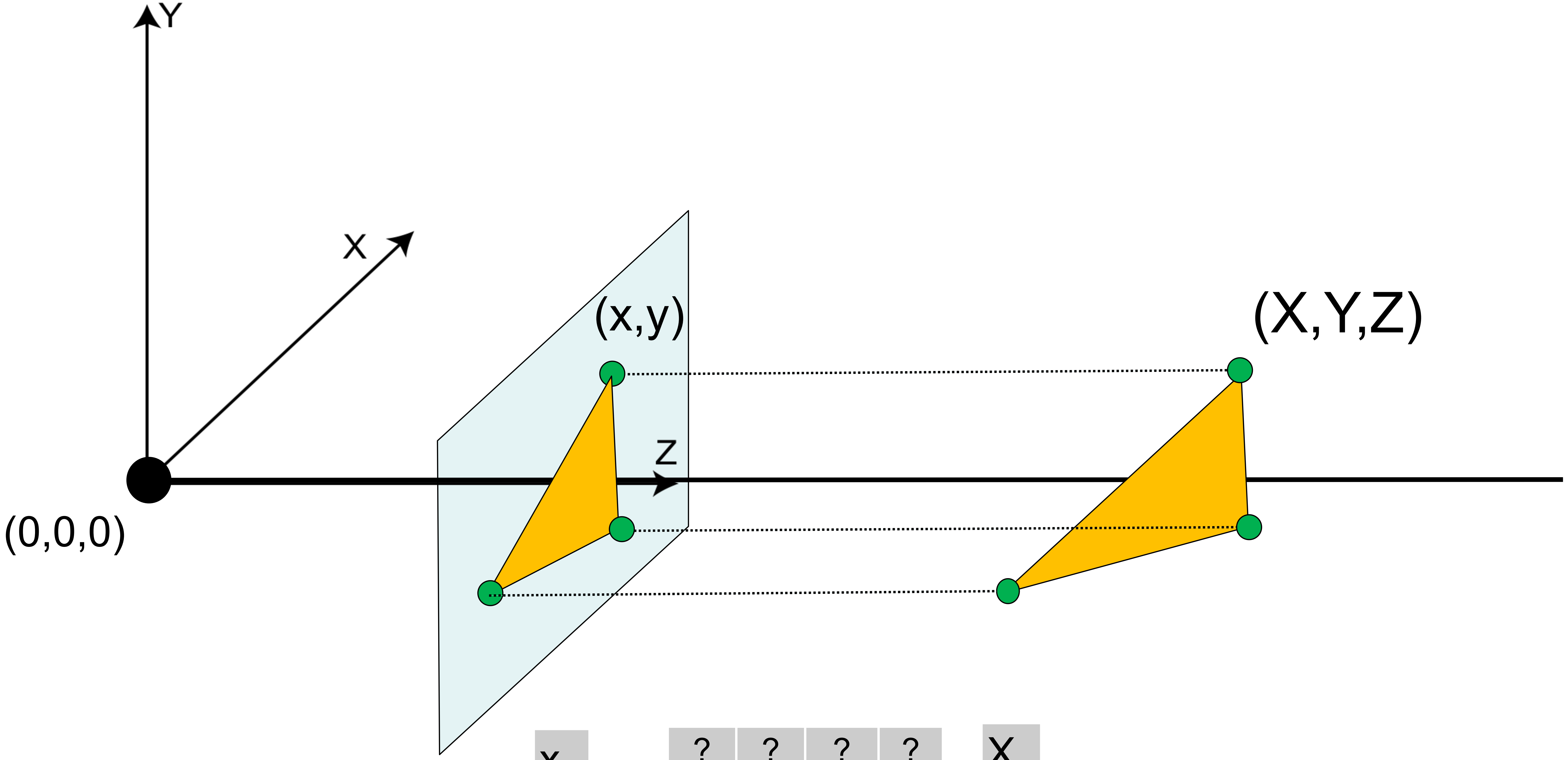
Perspective projection

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f & 0 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z/f \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} fX \\ fY \\ Z \end{bmatrix}$$

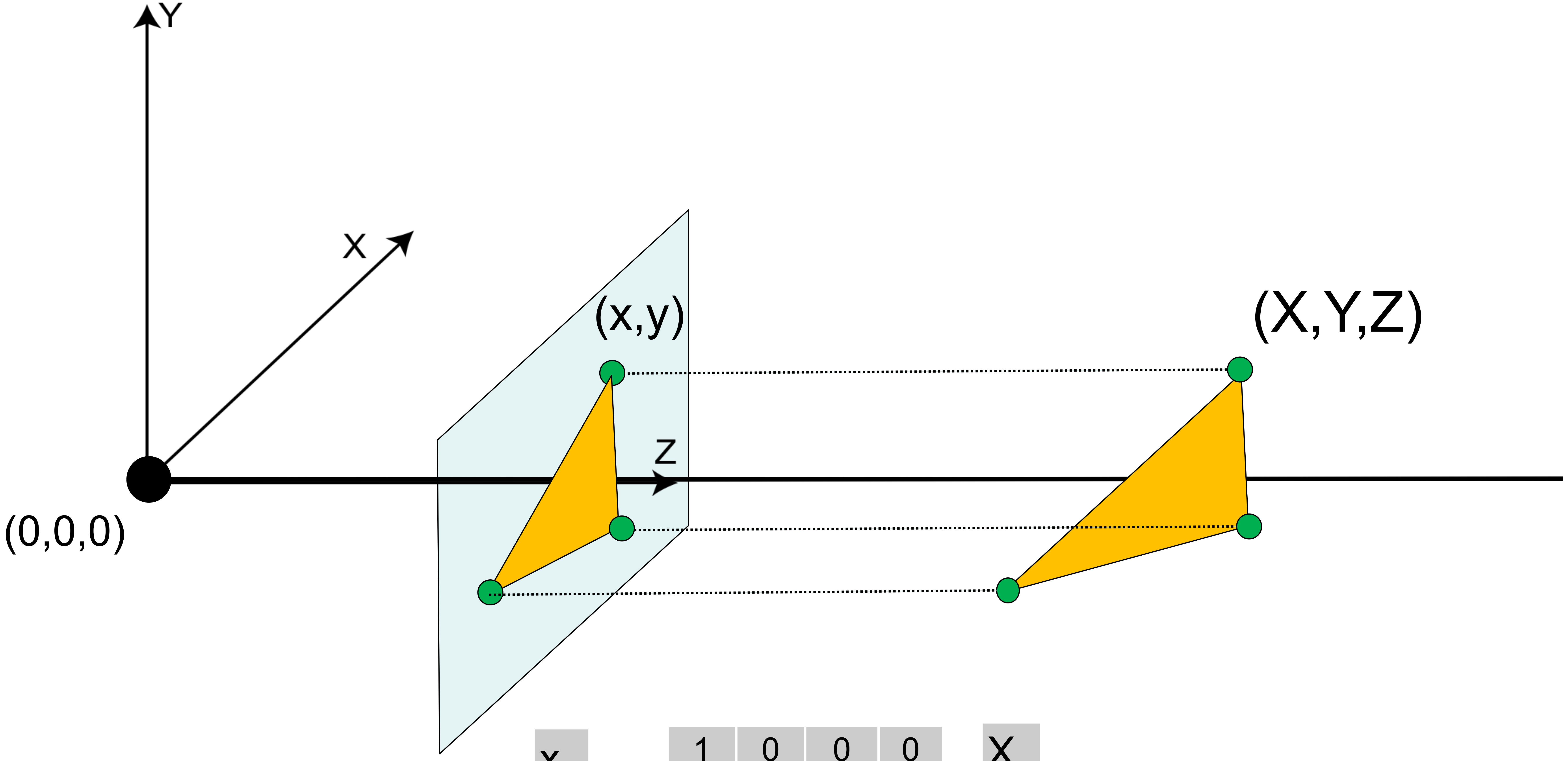
$\rightarrow$ (f X/Z, f Y/Z)

Orthographic (parallel) projection



$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Orthographic (parallel) projection



$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Camera parameters

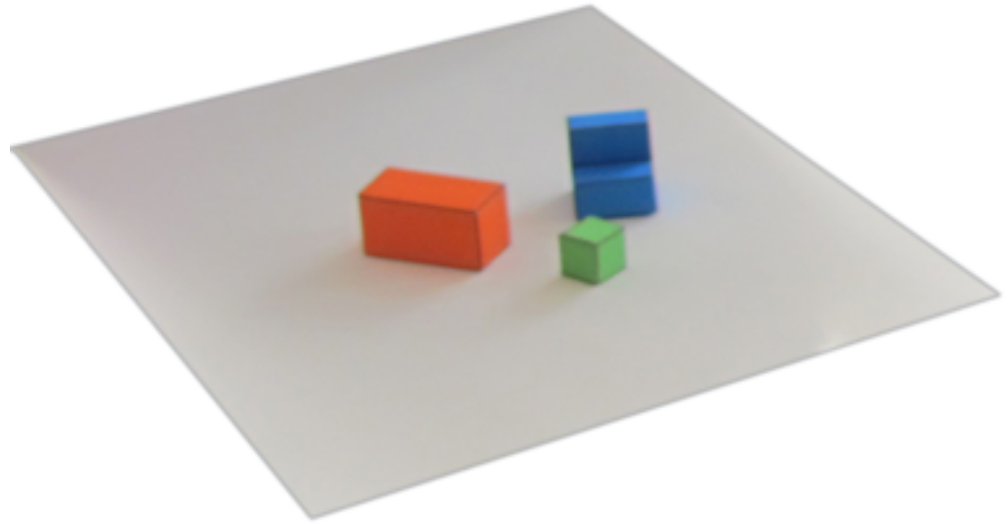
(x, y)

2D image

?

3D world

(X, Y, Z)

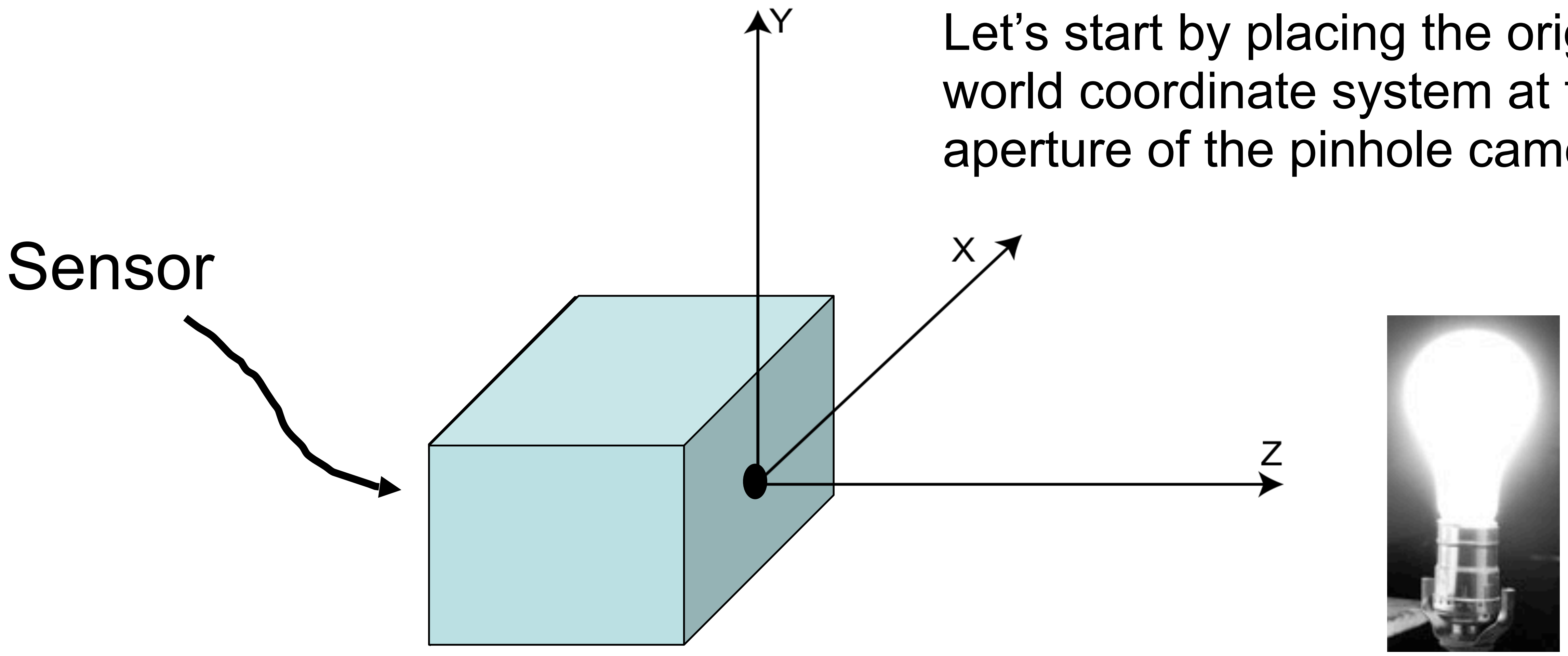


$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Pixels

World coordinates

Camera parameters

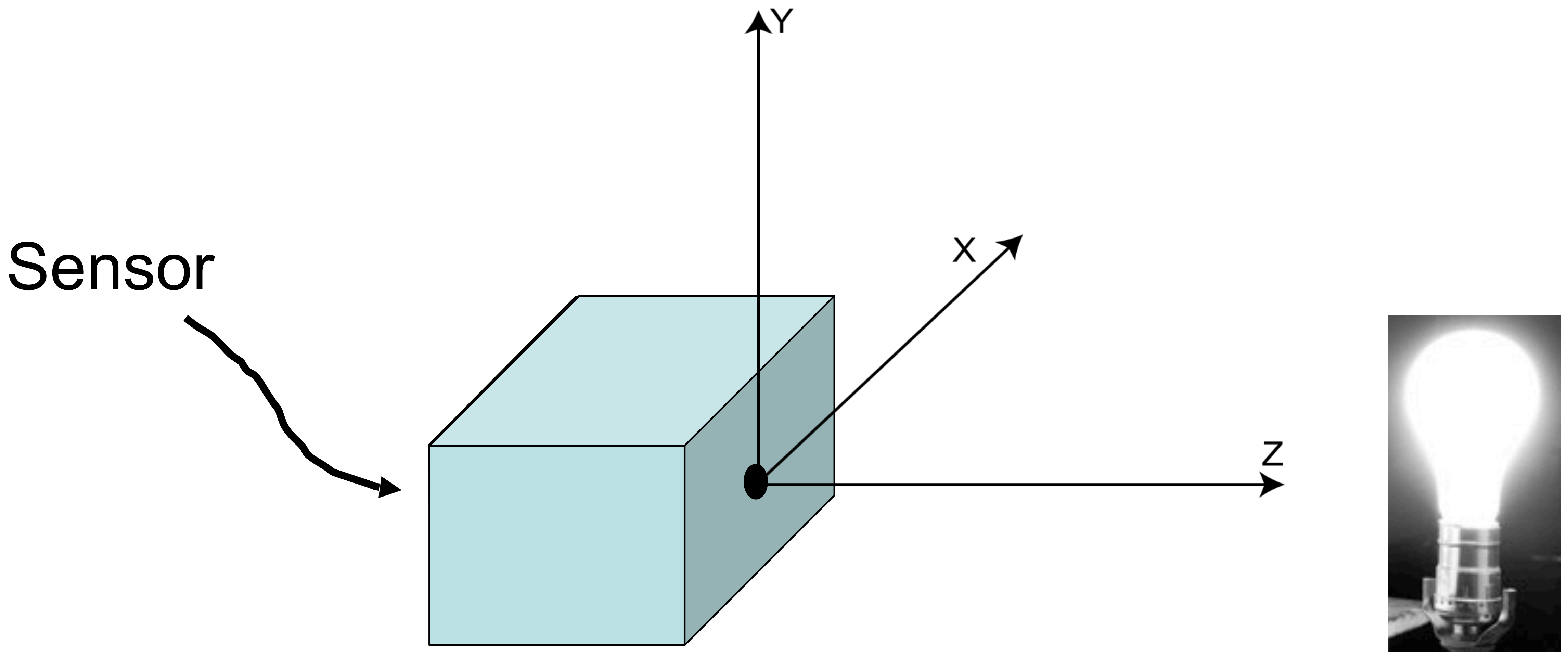


Let's start by placing the origin of the world coordinate system at the aperture of the pinhole camera

For the pinhole camera:

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} fX \\ fY \\ Z \end{bmatrix} \longrightarrow (f\, X/Z,\ f\, Y/Z)$$

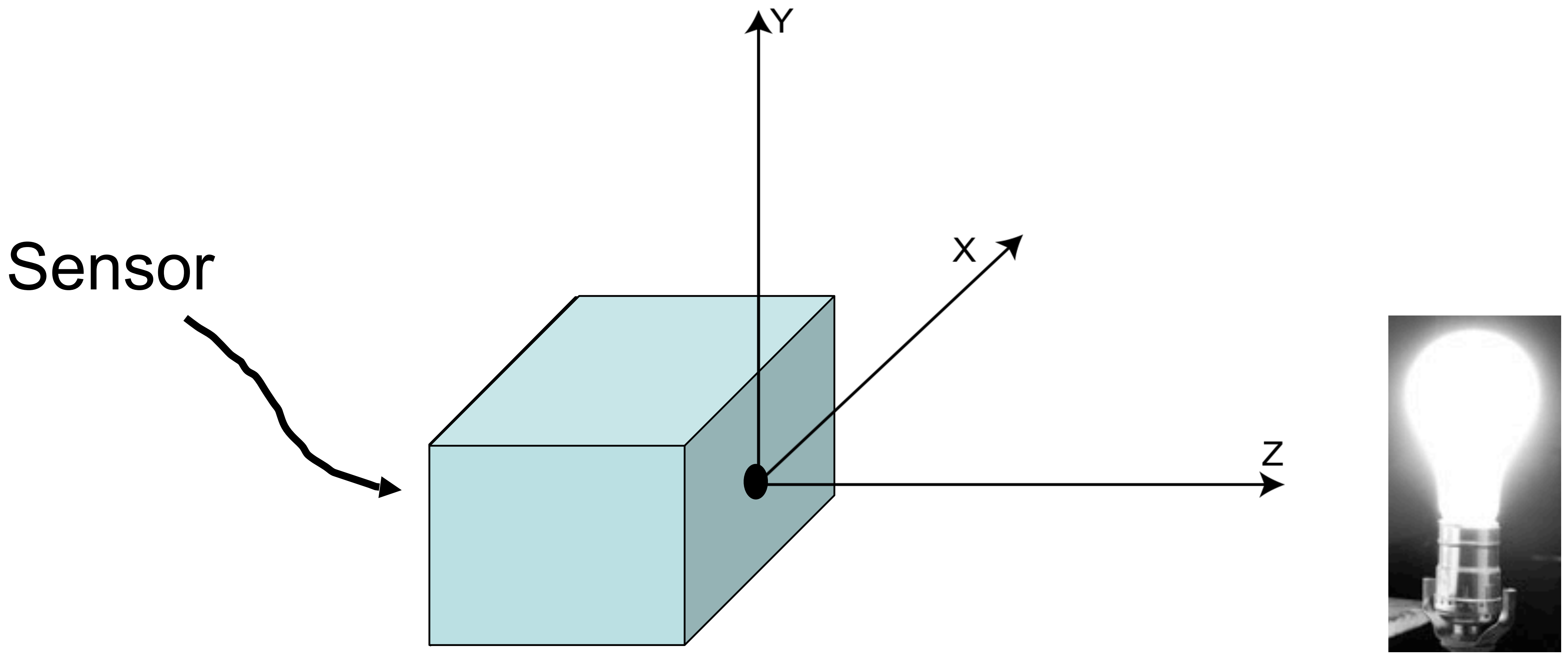
Camera parameters



When changing to pixels, there will be an arbitrary scaling:

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} aX \\ aY \\ Z \end{bmatrix} \longrightarrow (a\, X/Z,\ a\, Y/Z)$$

Camera parameters

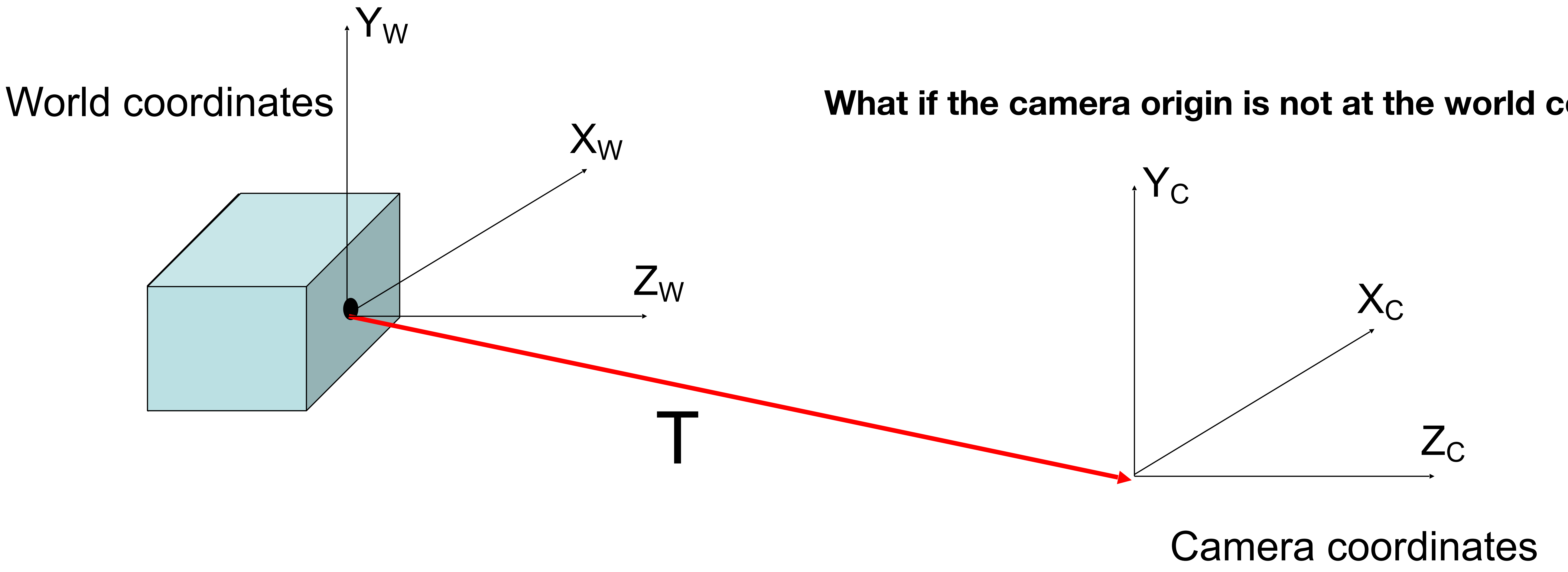


if pixels are rectangular

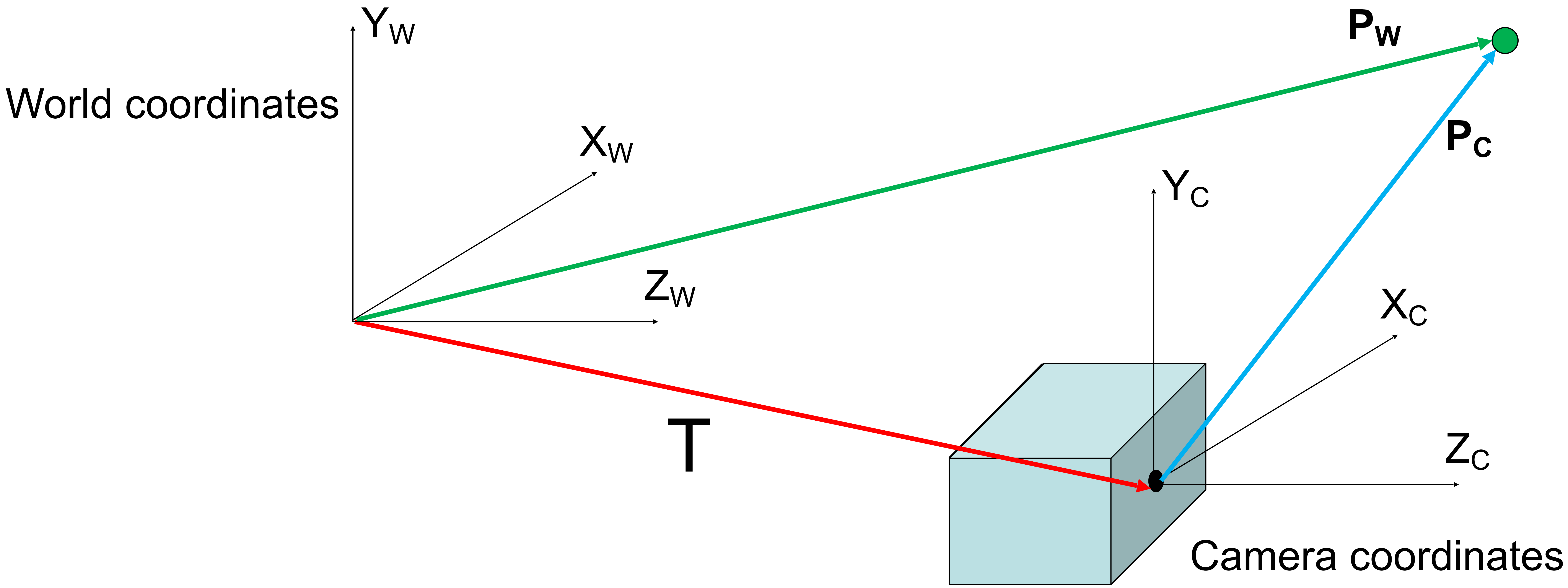
$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} aX \\ bY \\ Z \end{bmatrix} \longrightarrow (a\ X/Z,\ b\ Y/Z)$$

Camera parameters

What if the camera origin is not at the world coordinates origin?



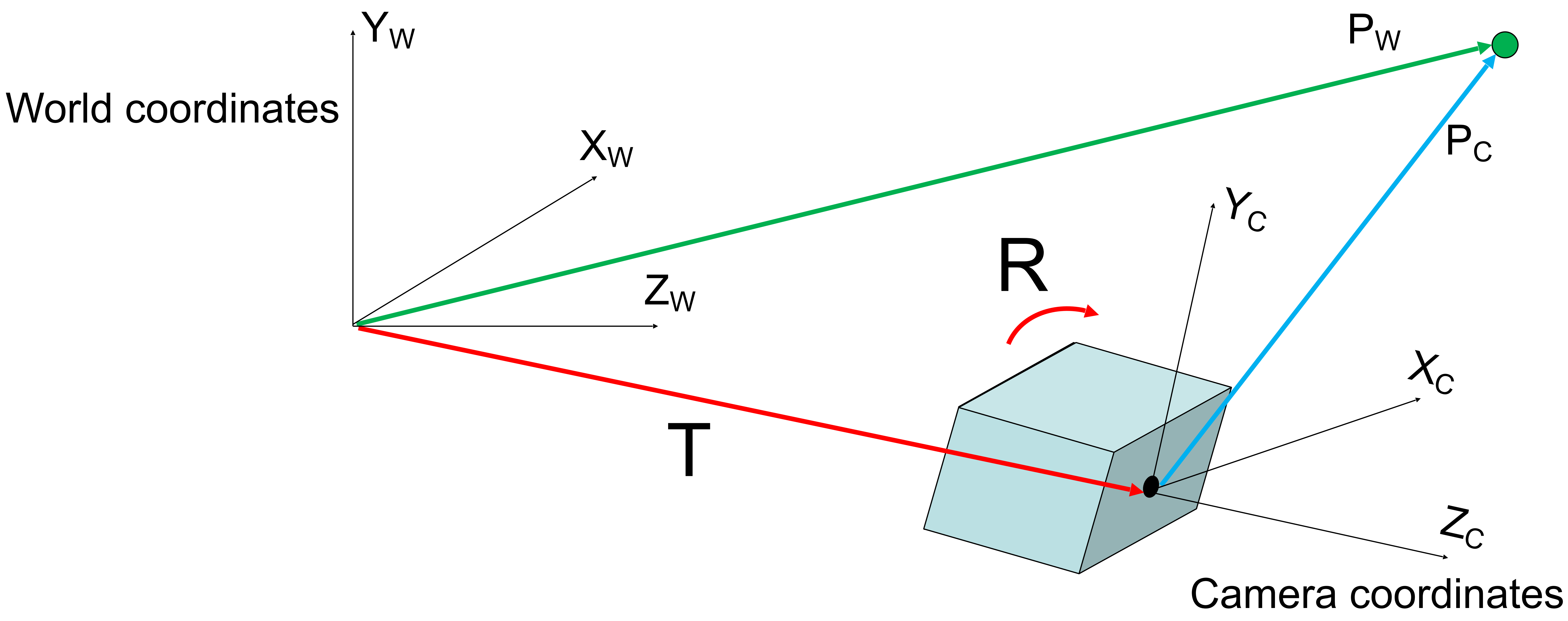
Camera parameters



In heterogeneous coordinates:

$$P_C = P_W - T$$

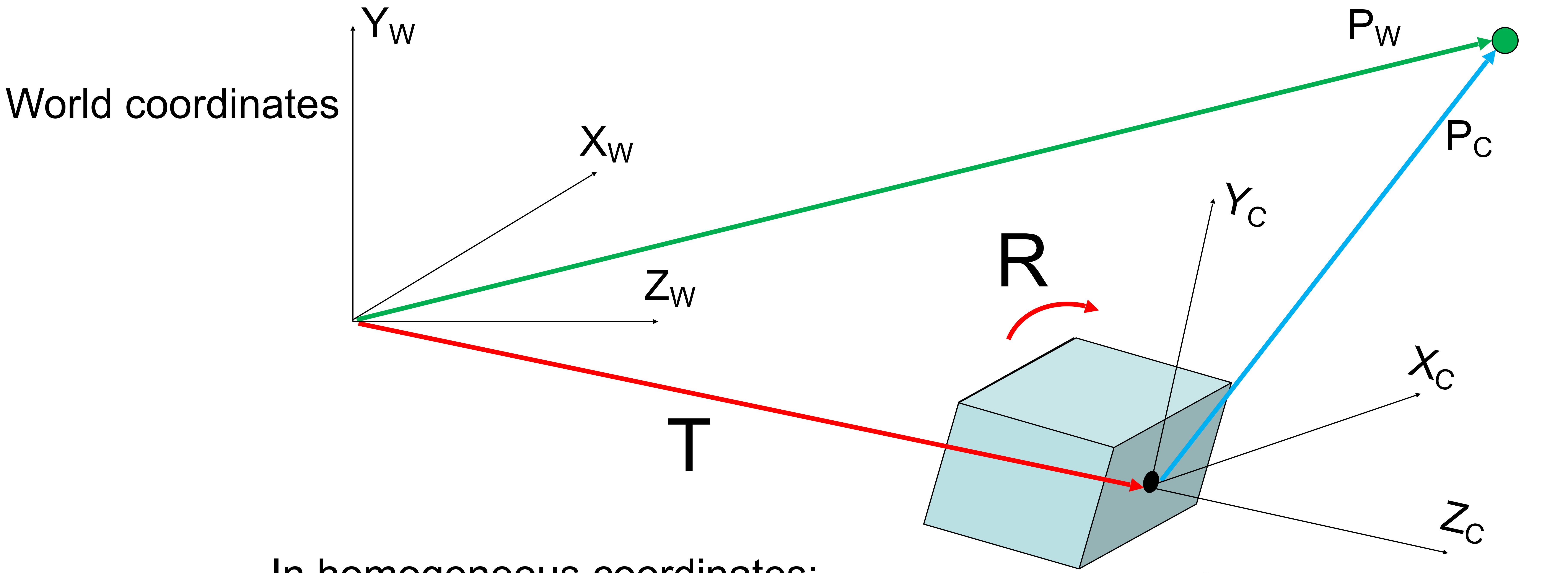
Camera parameters



In heterogeneous coordinates:

$$P_C = R(P_W - T)$$

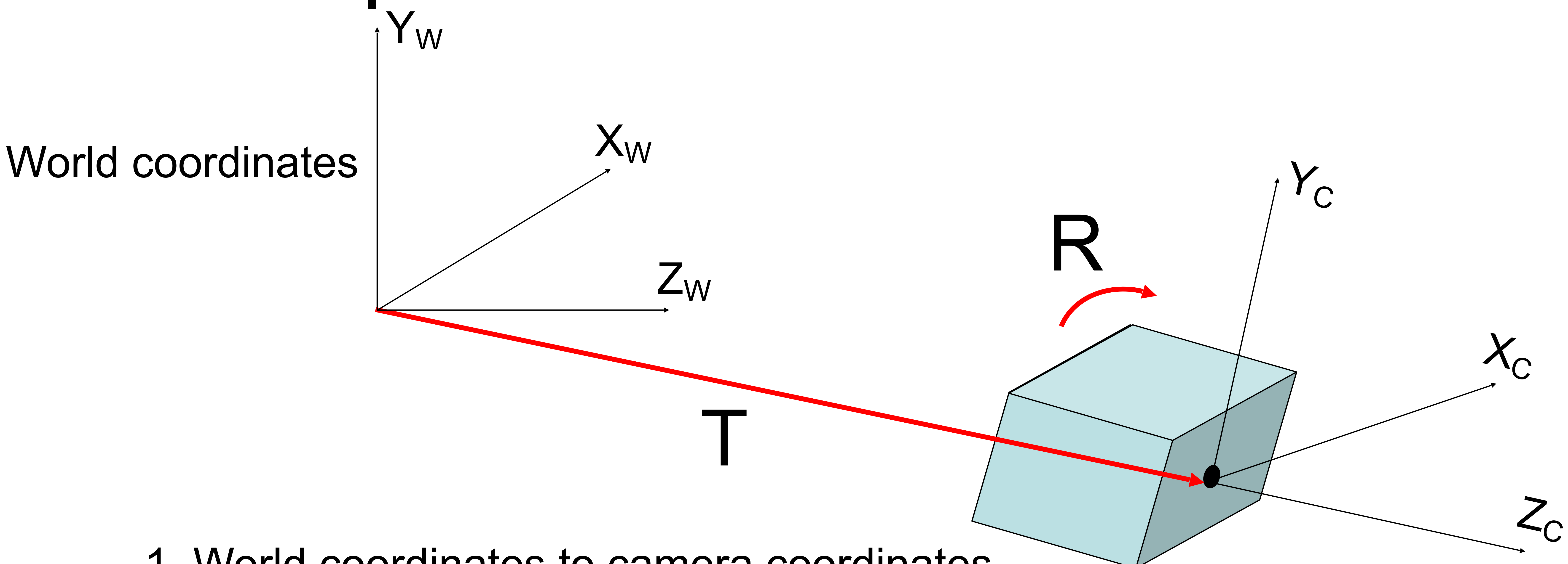
Camera parameters



In homogeneous coordinates:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} \underset{[3\times3]}{\mathbf{R}} & \underset{[3\times1]}{\mathbf{-RT}} \\ \underset{[1\times3]}{\mathbf{0}} & \underset{[1\times1]}{1} \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

Camera parameters



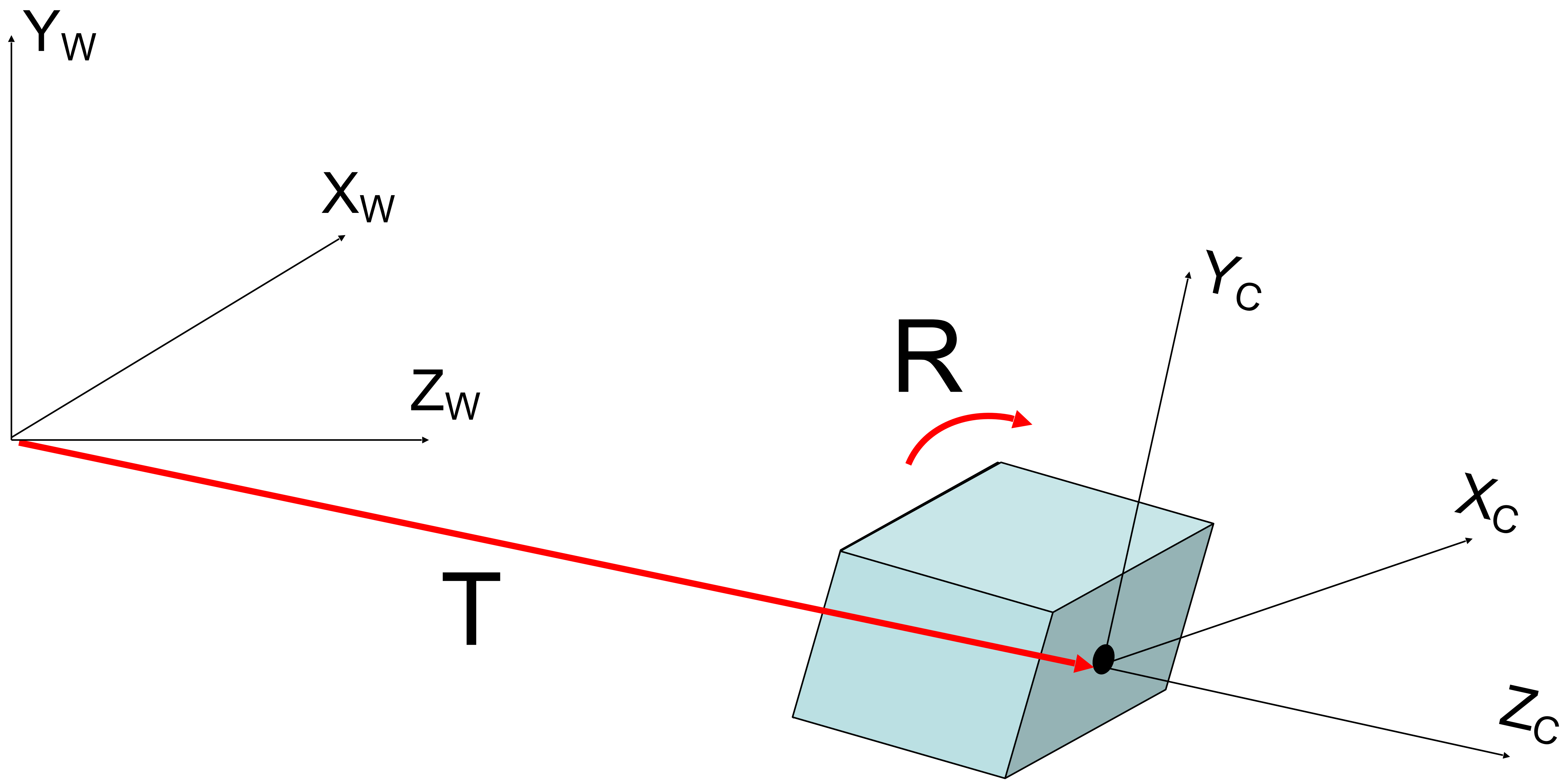
1. World coordinates to camera coordinates

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{-RT} \\ \mathbf{0} & 1 \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

2. Camera coordinates to image coordinates (square pixels)

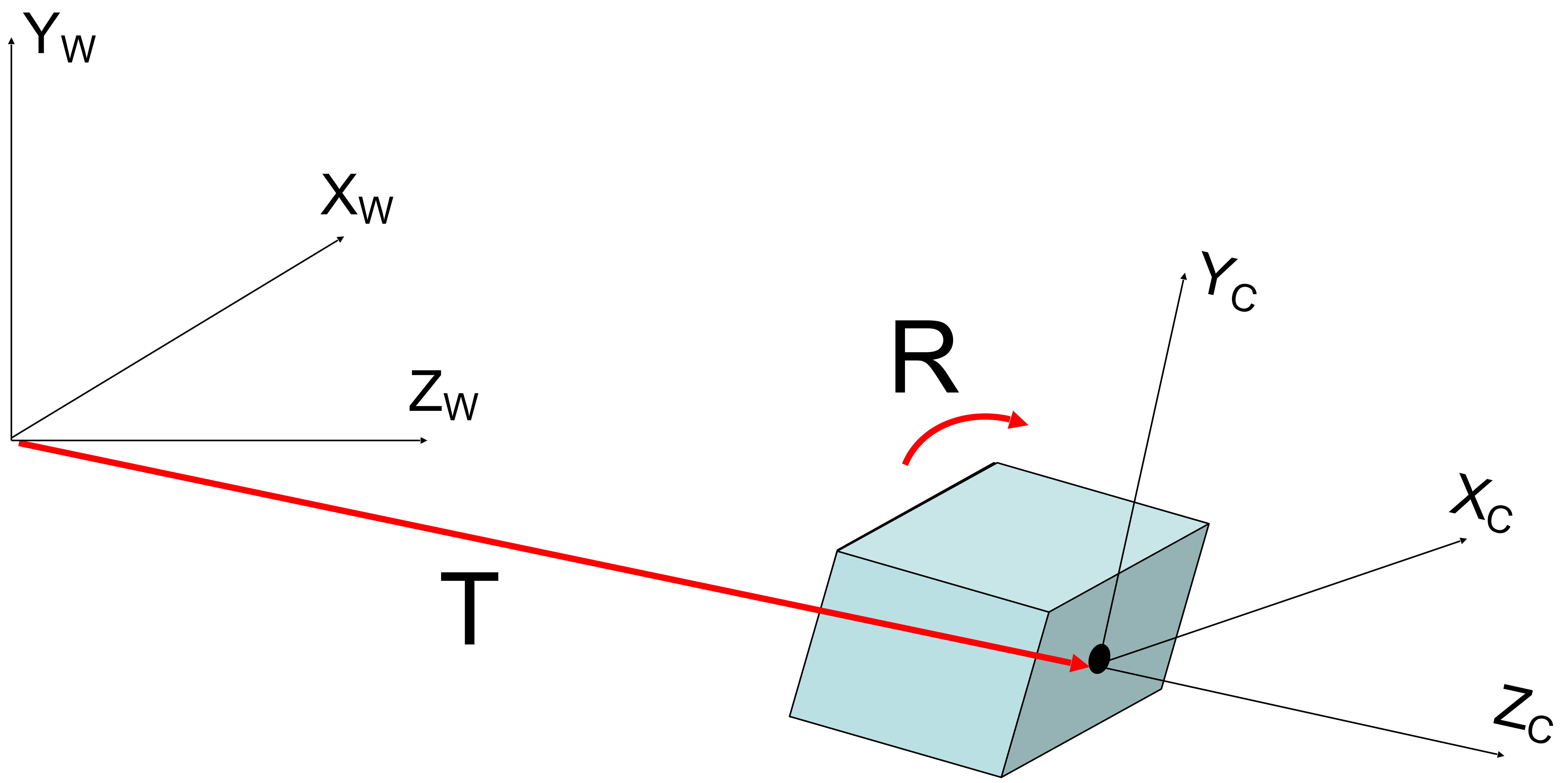
$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix}$$

Camera parameters



$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

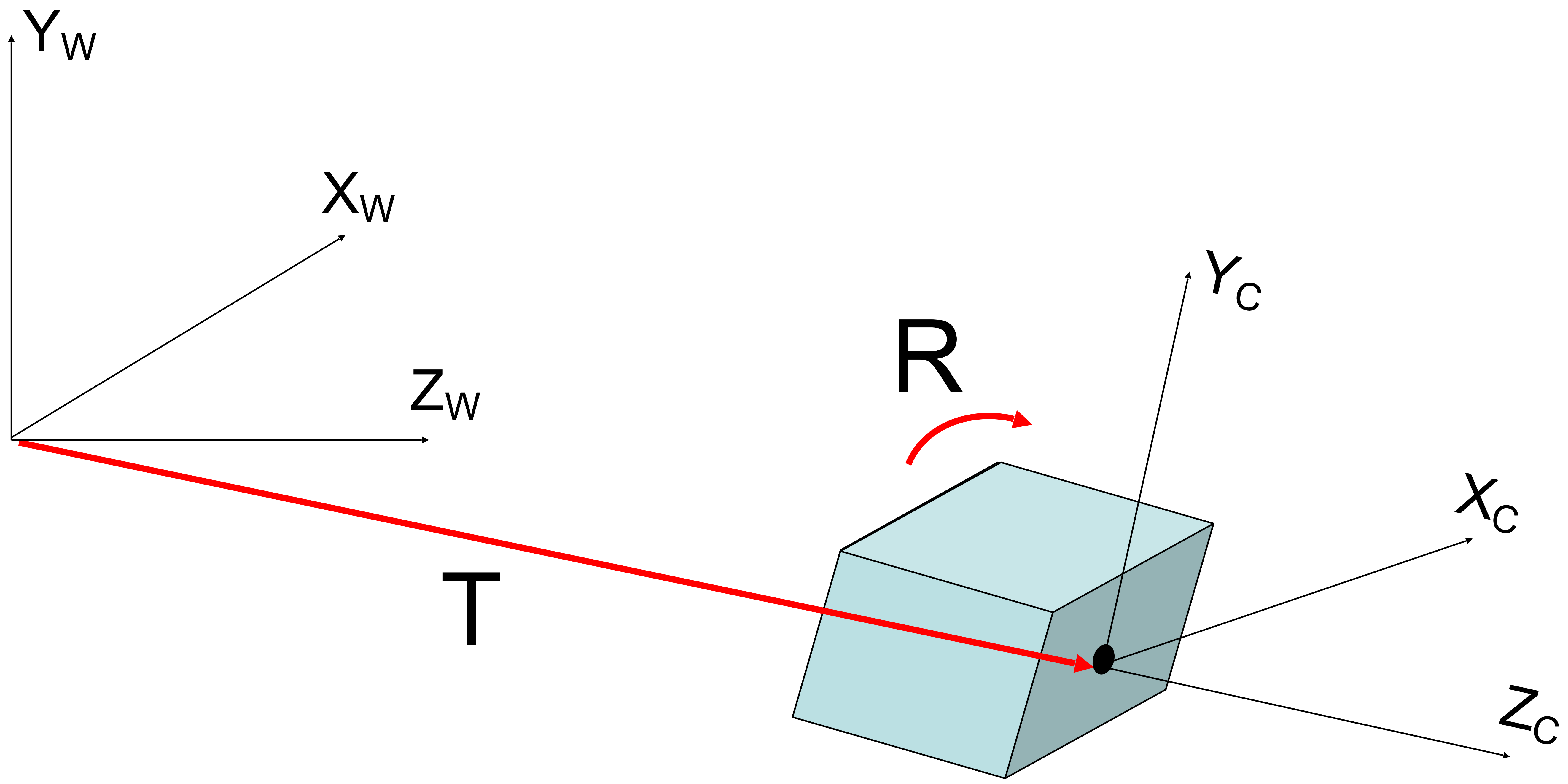
Camera parameters



[3x4] [4x4]

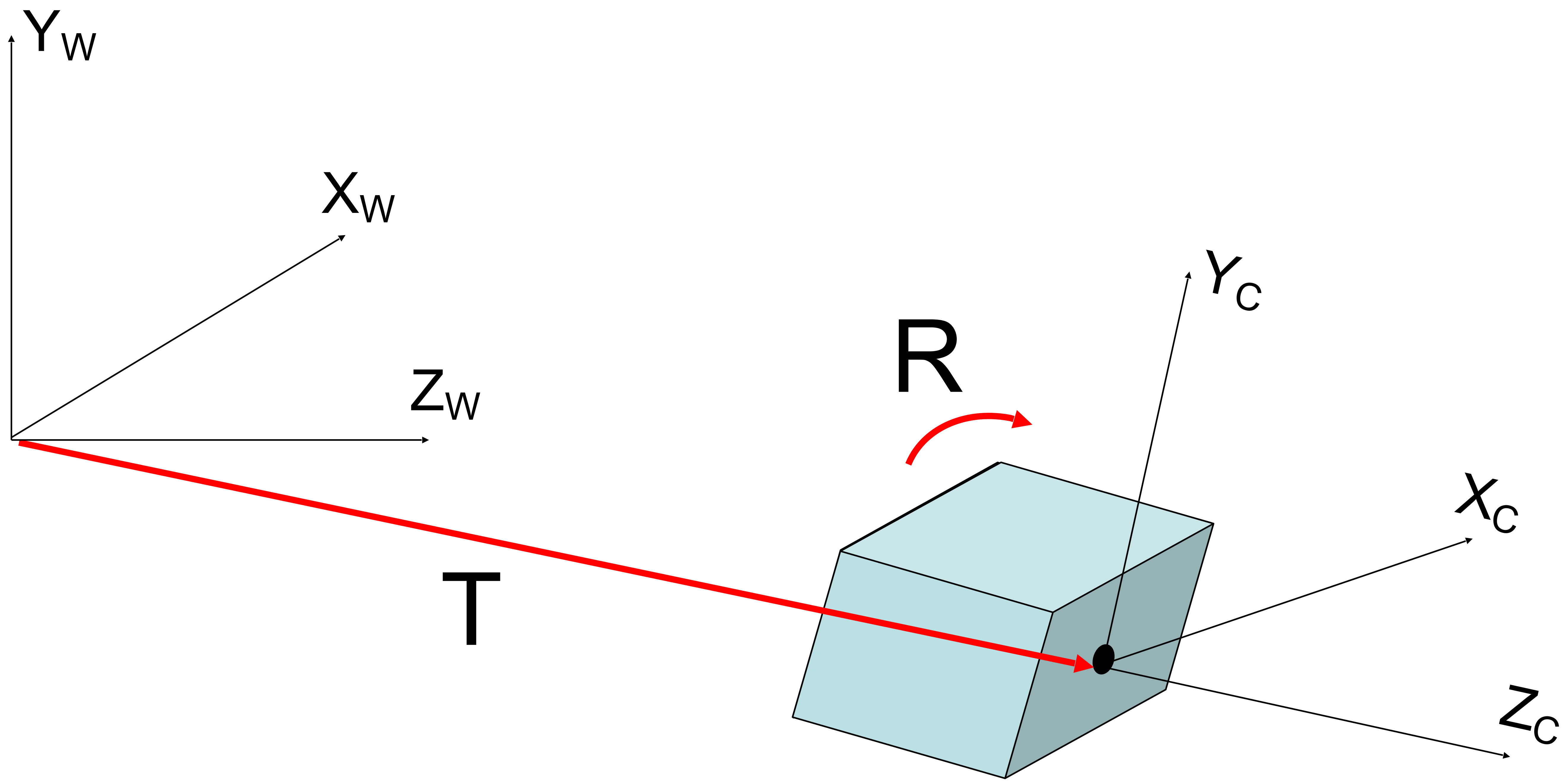
$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{R} & \mathbf{-RT} \\ \mathbf{0} & 1 \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

Camera parameters



$$\underset{}{\begin{bmatrix} x \\ y \\ w \end{bmatrix}} = \underset{[3x3]}{\begin{bmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & 1 \end{bmatrix}} \cdot \underset{[3x4]}{\begin{bmatrix} \mathbf{R} & \mathbf{-RT} \end{bmatrix}} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

Camera parameters

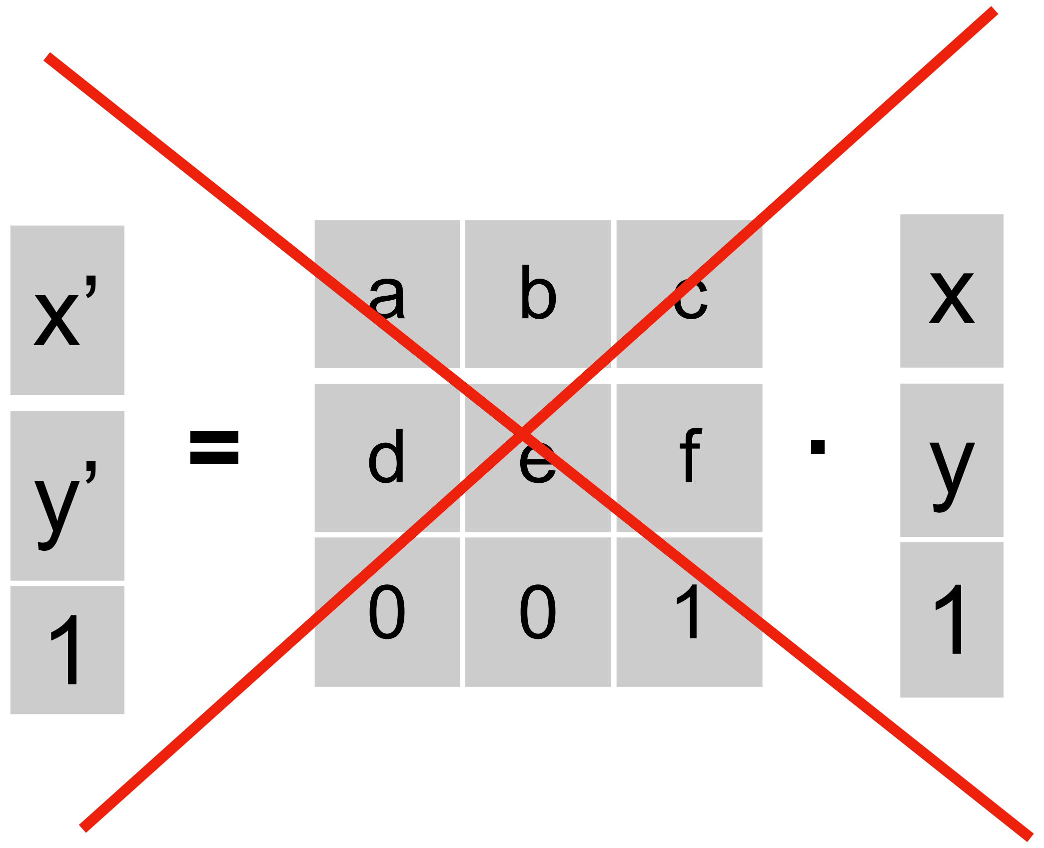


$$\underbrace{\begin{bmatrix} x \\ y \\ w \end{bmatrix}}_{} = \underbrace{\overset{[3x3]}{\begin{bmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & 1 \end{bmatrix}}}_{\text{Intrinsic parameters}} \cdot \underbrace{\overset{[3x3]}{\begin{bmatrix} & & \\ & \mathbf{R} & \\ & & \end{bmatrix}} \cdot \overset{[3x4]}{\begin{bmatrix} & \mathbf{I} & & \mathbf{-T} \end{bmatrix}}}_{\text{Extrinsic parameters}} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

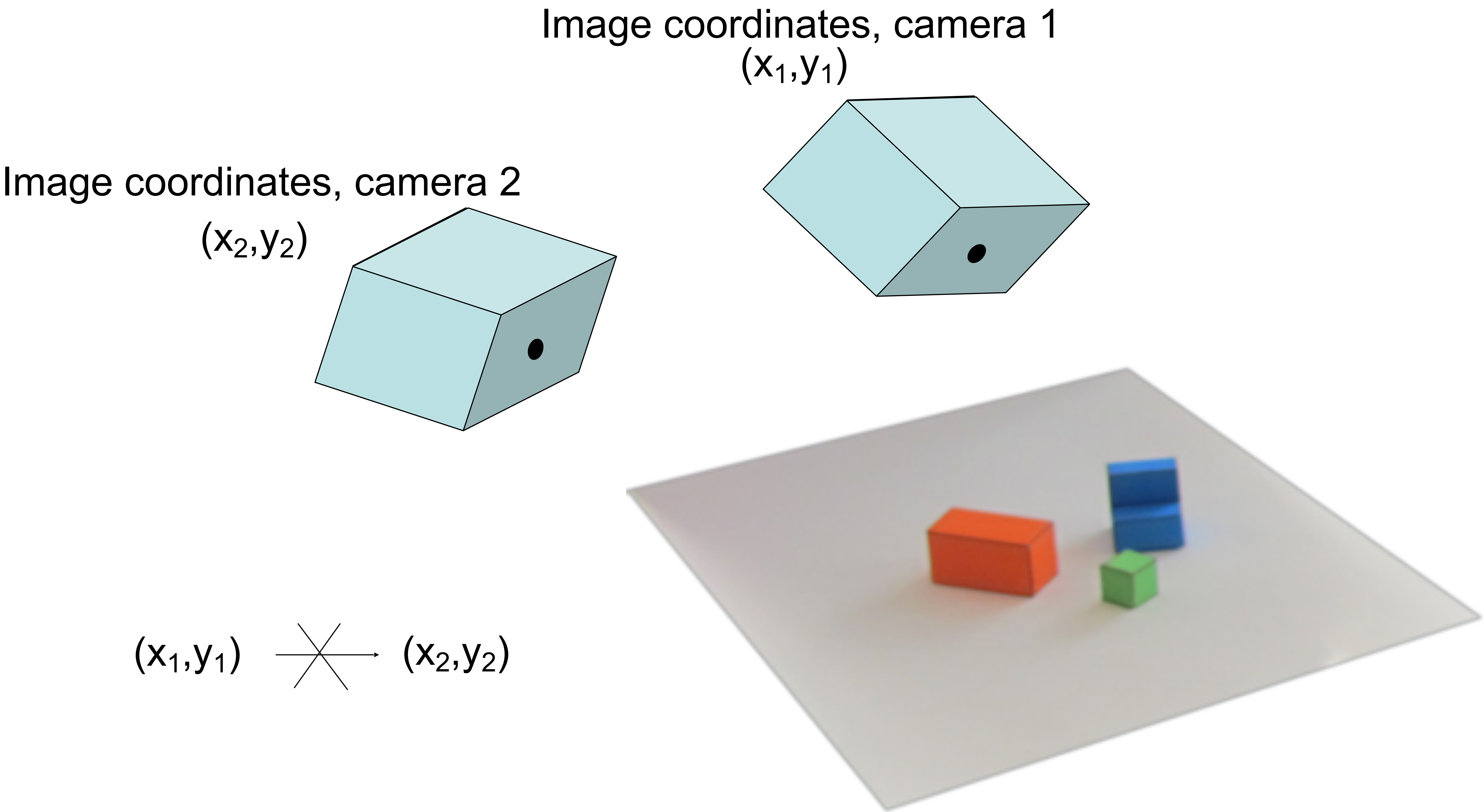
Making image panoramas

Example: two pictures taken by rotating the camera:

If we try to build a panorama by overlapping them:



Mapping one camera into another

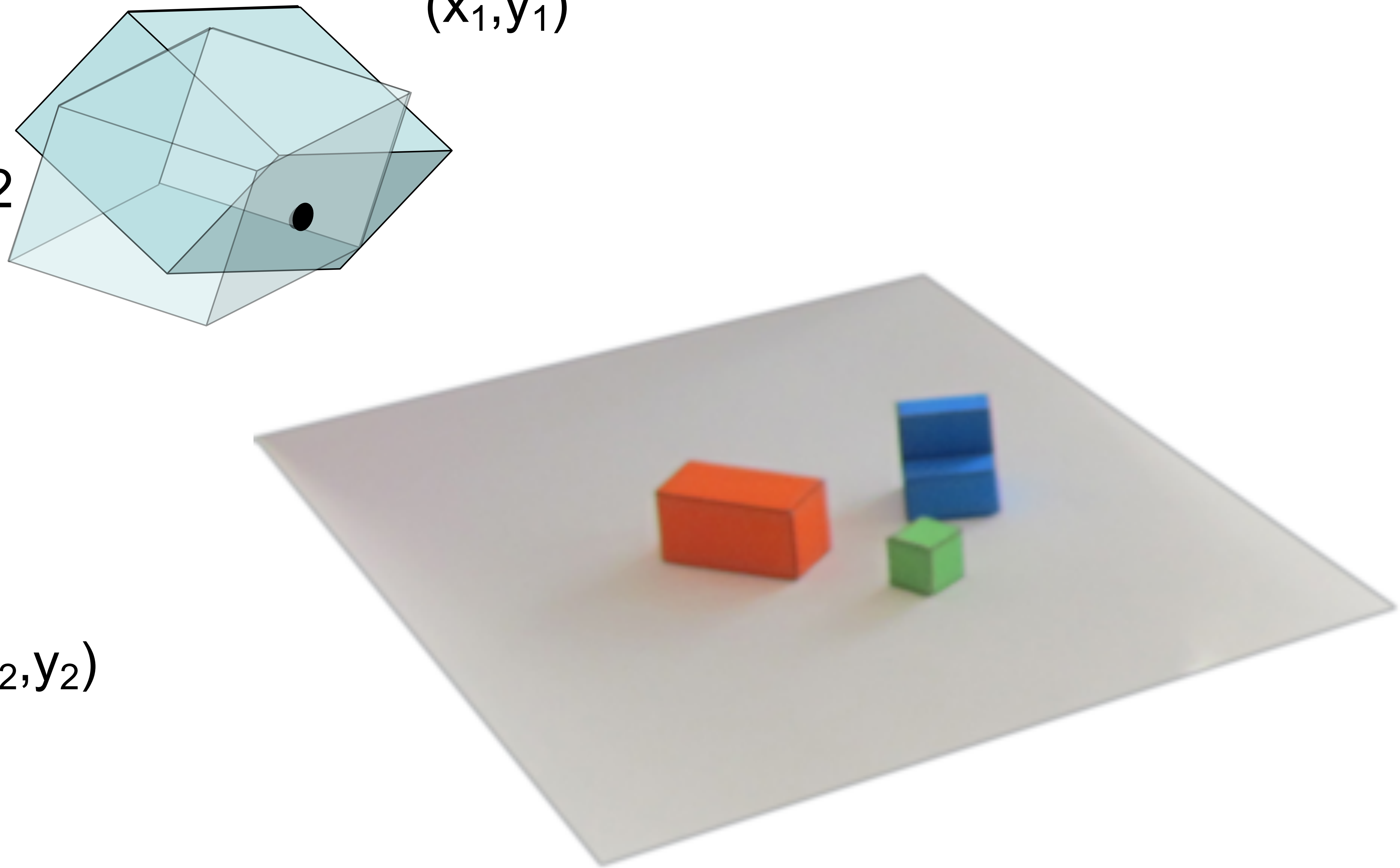


In general, we can not find a transformation from x_1 to x_2. It requires knowing the 3D coordinates of each corresponding point.
(The general mapping has to depend on 3D shape, otherwise we would learn no information from the 2nd image of a stereo camera!)

Mapping one camera into another

Image coordinates, camera 1
(x_1,y_1)

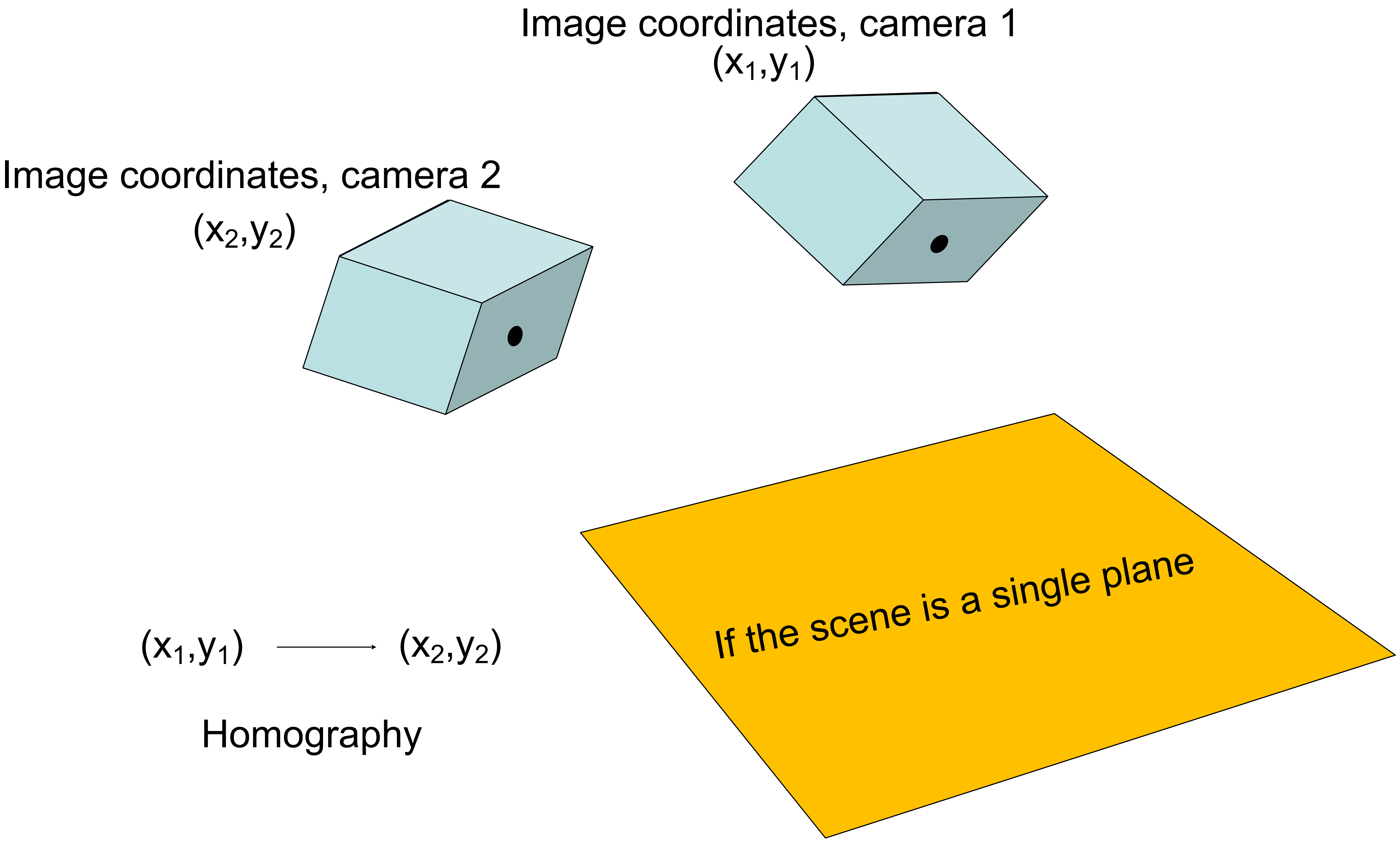
Image coordinates, camera 2
(x_2,y_2)



$(x_1,y_1) \longrightarrow (x_2,y_2)$

Homography

Mapping one camera into another



$(x_1,y_1) \longrightarrow (x_2,y_2)$

Homography

Homography

What happens if we allow 9 degrees of freedom?

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

(note that only 8 DOF are relevant here)

The homography allow mapping one camera into another when:'

- scene is planar
- both cameras only differ by a rotation (no translation)

Homography

Example: two pictures taken by rotating the camera:

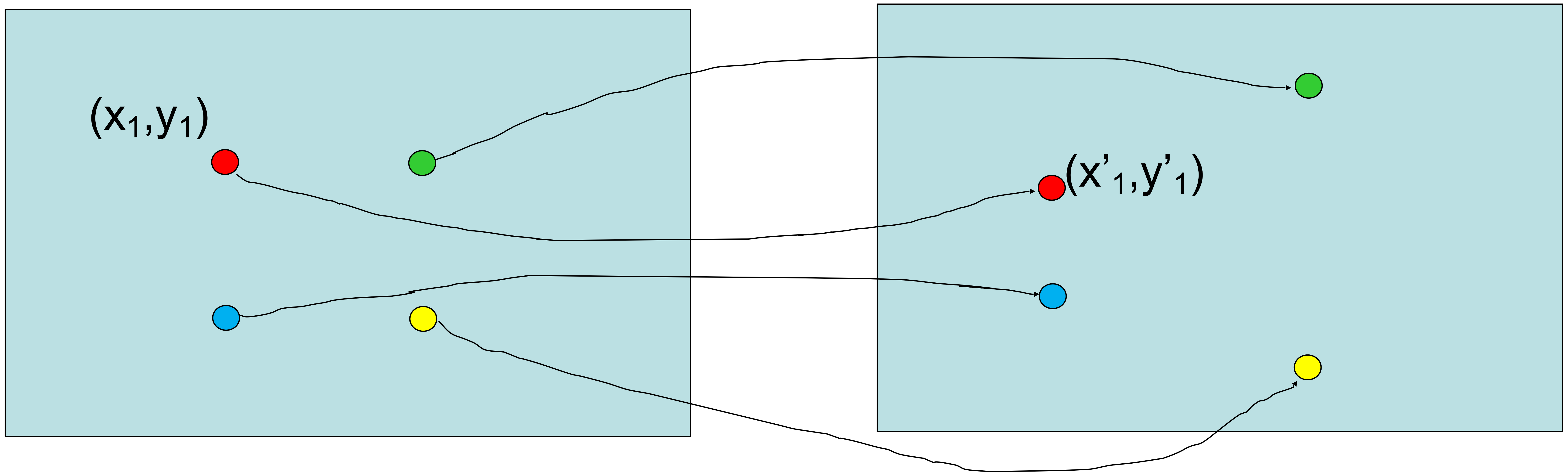
If we try to build a panorama by overlapping them:

Homography

Example: two pictures taken by rotating the camera:

With an homography you can map both images into a single camera:

Homography



$$\begin{bmatrix} x_1' \\ y_1' \\ w_1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

(note that only 8 DOF are relevant here)

Homography

$$\begin{bmatrix} x_1' \\ y_1' \\ w_1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

Going to heterogeneous coordinates:

$$x_1' = \frac{ax_1 + by_1 + c}{gx_1 + hy_1 + i}$$

$$y_1' = \frac{dx_1 + ey_1 + f}{gx_1 + hy_1 + i}$$

Re-arranging the terms:

$$gx_1x'_1 + hy_1x'_1 + ix_1 = ax_1 + by_1 + c$$

$$gx_1y'_1 + hy_1y'_1 + ix_1 = dx_1 + ey_1 + f$$

Homography

$$g x_1 x'_1 + h y_1 x'_1 + i x_1 = a x_1 + b y_1 + c$$

$$g x_1 y'_1 + h y_1 y'_1 + i x_1 = d x_1 + e y_1 + f$$

Re-arranging the terms:

$$g x_1 x'_1 + h y_1 x'_1 + i x_1 - a x_1 - b y_1 - c = 0$$

$$g x_1 y'_1 + h y_1 y'_1 + i x_1 - d x_1 - e y_1 - f = 0$$

In matrix form

$$\begin{bmatrix} -x_1 & -y_1 & -1 & 0 & 0 & 0 & x_1 x'_1 & y_1 x'_1 & x_1 \\ 0 & 0 & 0 & -x_1 & -y_1 & -1 & x_1 x'_1 & y_1 x'_1 & x_1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \\ i \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Homography

With multiple corresponding points:

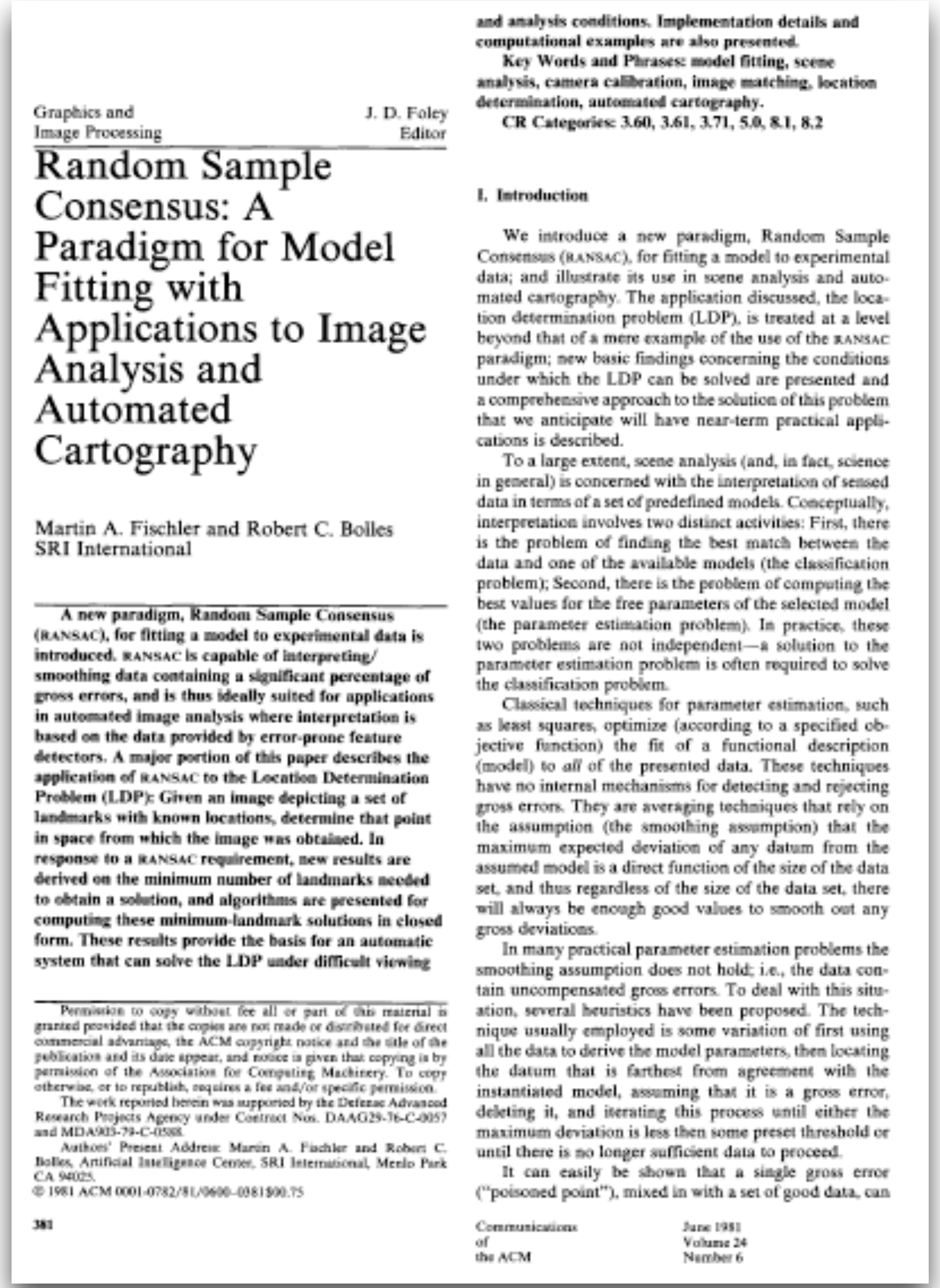
$$\begin{bmatrix} -x_1 & -y_1 & -1 & 0 & 0 & 0 & x_1x'_1 & y_1x'_1 & x_1 \\ 0 & 0 & 0 & -x_1 & -y_1 & -1 & x_1x'_1 & y_1x'_1 & x_1 \\ & & & & \vdots & & & & \\ -x_N & -y_N & -1 & 0 & 0 & 0 & x_Nx'_N & y_Nx'_N & x_N \\ 0 & 0 & 0 & -x_N & -y_N & -1 & x_Nx'_N & y_Nx'_N & x_N \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \\ i \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ . \\ . \\ . \\ 0 \end{bmatrix}$$

$$A\,h = 0$$

Compute SVD of A, and take the eigenvector with the smallest eigenvalue

Ransac

- M. A. Fischler, R. C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Comm. of the ACM, Vol 24, pp 381-395, 1981.



Graphics and Image Processing — J. D. Foley, Editor

Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography

Martin A. Fischler and Robert C. Bolles
SRI International

A new paradigm, Random Sample Consensus (RANSAC), for fitting a model to experimental data is introduced. RANSAC is capable of interpreting/smoothing data containing a significant percentage of gross errors, and is thus ideally suited for applications in automated image analysis where interpretation is based on the data provided by error-prone feature detectors. A major portion of this paper describes the application of RANSAC to the Location Determination Problem (LDP): Given an image depicting a set of landmarks with known locations, determine that point in space from which the image was obtained. In response to a RANSAC requirement, new results are derived on the minimum number of landmarks needed to obtain a solution, and algorithms are presented for computing these minimum-landmark solutions in closed form. These results provide the basis for an automatic system that can solve the LDP under difficult viewing and analysis conditions. Implementation details and computational examples are also presented.

Key Words and Phrases: model fitting, scene analysis, camera calibration, image matching, location determination, automated cartography.
CR Categories: 3.60, 3.61, 3.71, 5.0, 8.1, 8.2

The work reported herein was supported by the Defense Advanced Research Projects Agency under Contract Nos. DAAG29-76-C-0057 and MDA903-79-C-0588.
Authors' Present Address: Martin A. Fischler and Robert C. Bolles, Artificial Intelligence Center, SRI International, Menlo Park CA 94025.

I. Introduction

We introduce a new paradigm, Random Sample Consensus (RANSAC), for fitting a model to experimental data; and illustrate its use in scene analysis and automated cartography. The application discussed, the location determination problem (LDP), is treated at a level beyond that of a mere example of the use of the RANSAC paradigm; new basic findings concerning the conditions under which the LDP can be solved are presented and a comprehensive approach to the solution of this problem that we anticipate will have near-term practical applications is described.

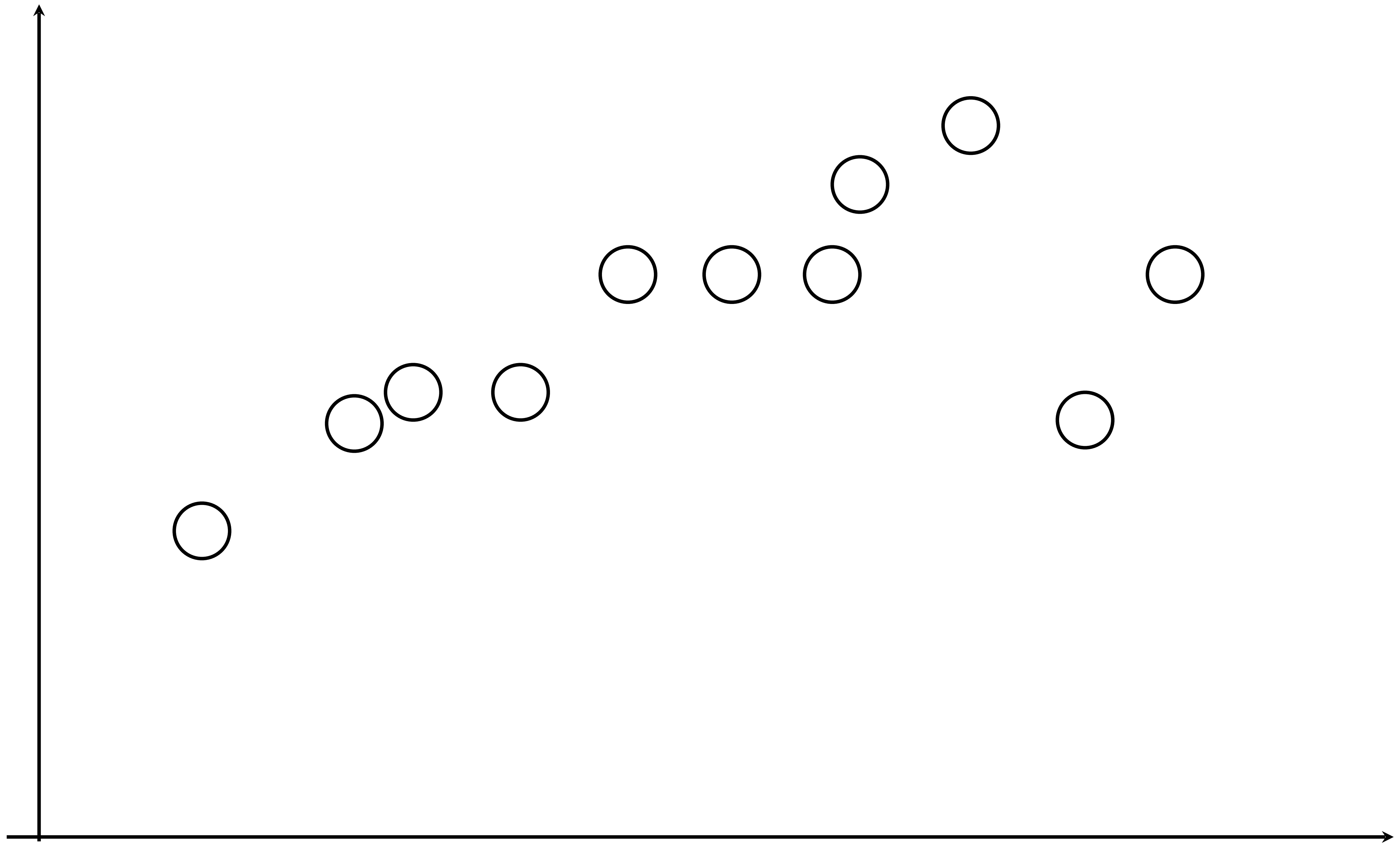
To a large extent, scene analysis (and, in fact, science in general) is concerned with the interpretation of sensed data in terms of a set of predefined models. Conceptually, interpretation involves two distinct activities: First, there is the problem of finding the best match between the data and one of the available models (the classification problem); Second, there is the problem of computing the best values for the free parameters of the selected model (the parameter estimation problem). In practice, these two problems are not independent—a solution to the parameter estimation problem is often required to solve the classification problem.

Classical techniques for parameter estimation, such as least squares, optimize (according to a specified objective function) the fit of a functional description (model) to *all* of the presented data. These techniques have no internal mechanisms for detecting and rejecting gross errors. They are averaging techniques that rely on the assumption (the smoothing assumption) that the maximum expected deviation of any datum from the assumed model is a direct function of the size of the data set, and thus regardless of the size of the data set, there will always be enough good values to smooth out any gross deviations.

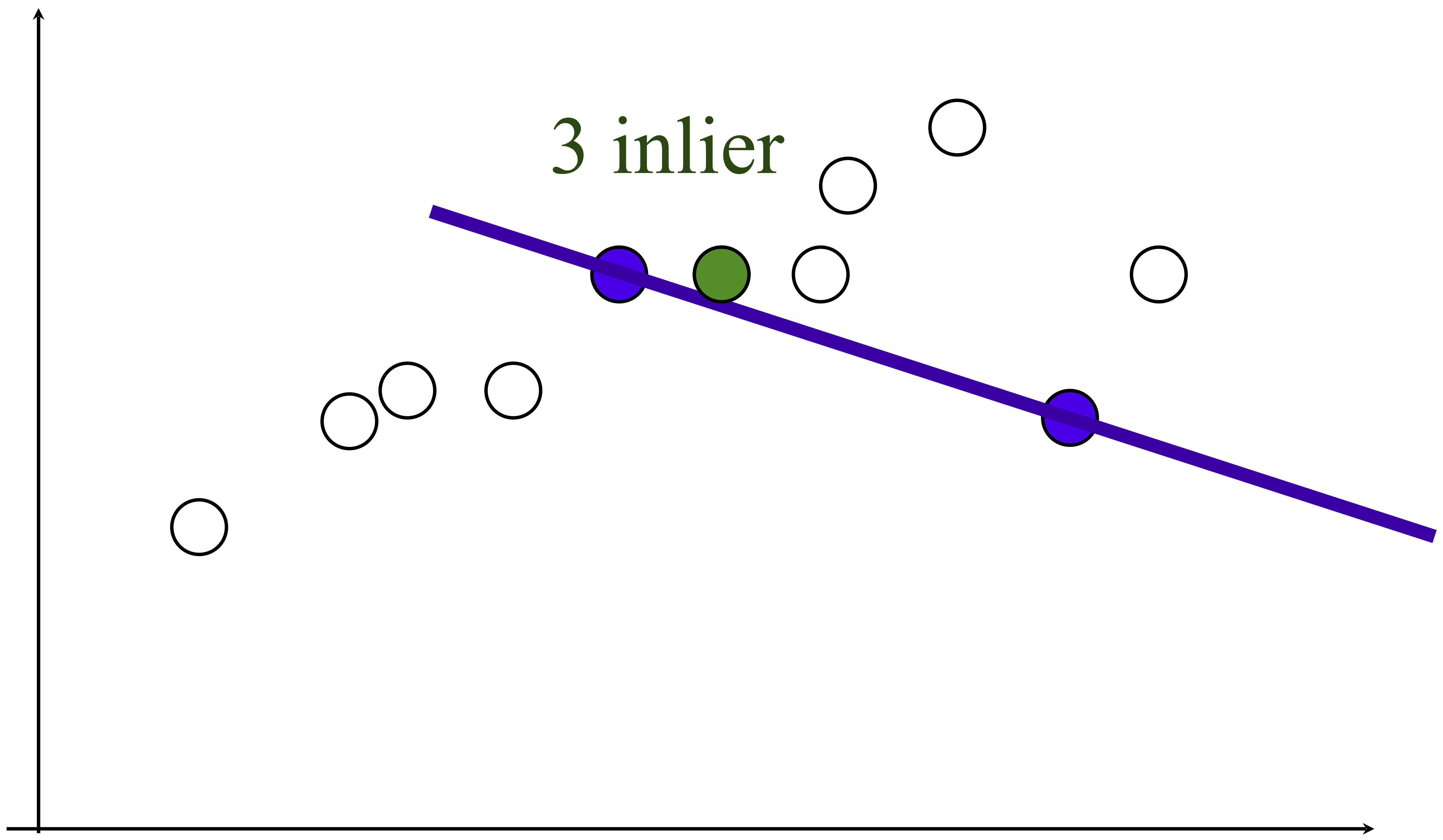
In many practical parameter estimation problems the smoothing assumption does not hold; i.e., the data contain uncompensated gross errors. To deal with this situation, several heuristics have been proposed. The technique usually employed is some variation of first using all the data to derive the model parameters, then locating the datum that is farthest from agreement with the instantiated model, assuming that it is a gross error, deleting it, and iterating this process until either the maximum deviation is less then some preset threshold or until there is no longer sufficient data to proceed.

It can easily be shown that a single gross error ("poisoned point"), mixed in with a set of good data, can

Simple example: fit a line

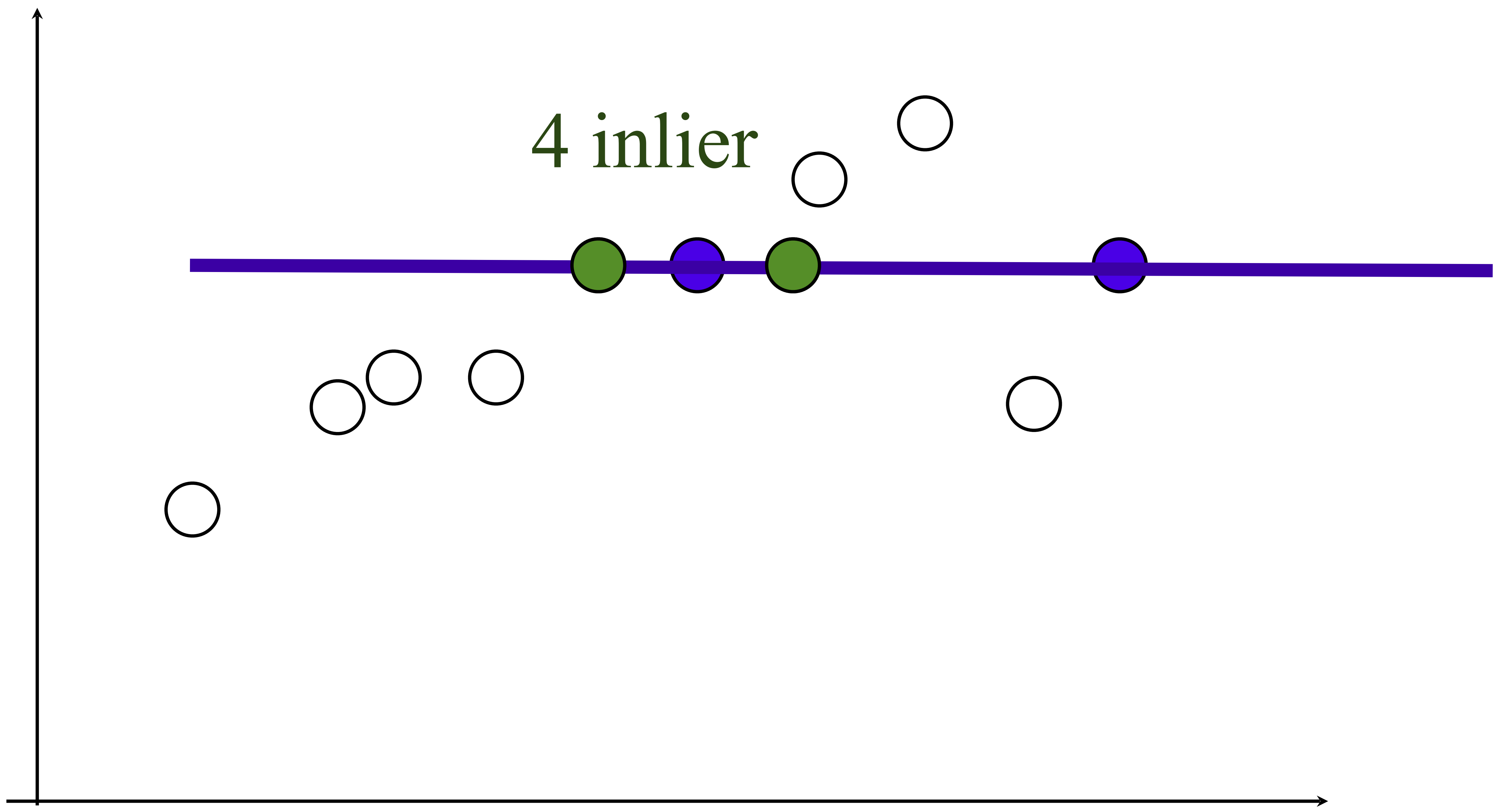


Simple example: fit a line



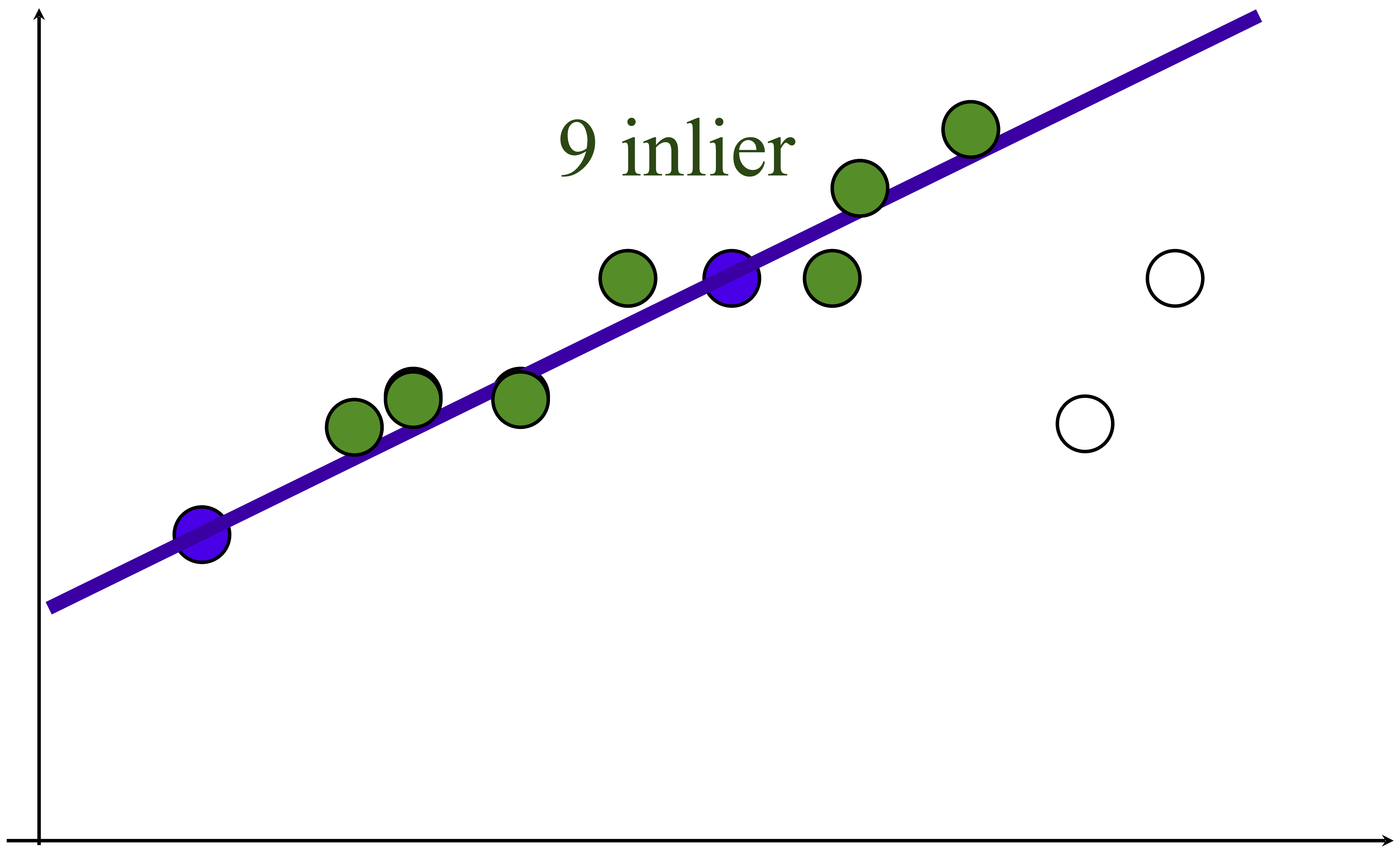
- Pick 2 points
- Fit line
- Count inliers

Simple example: fit a line



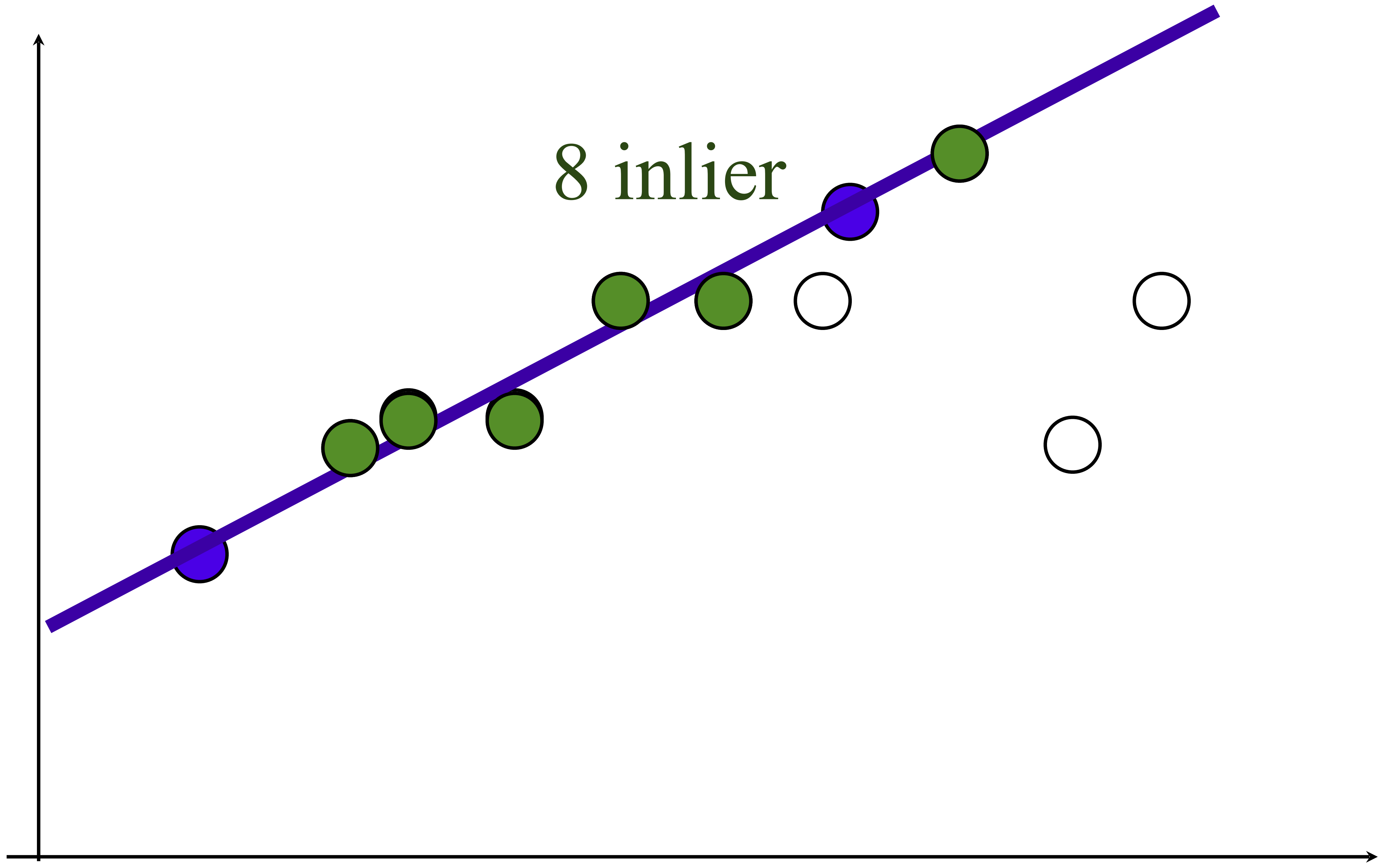
- Pick 2 points
- Fit line
- Count inliers

Simple example: fit a line



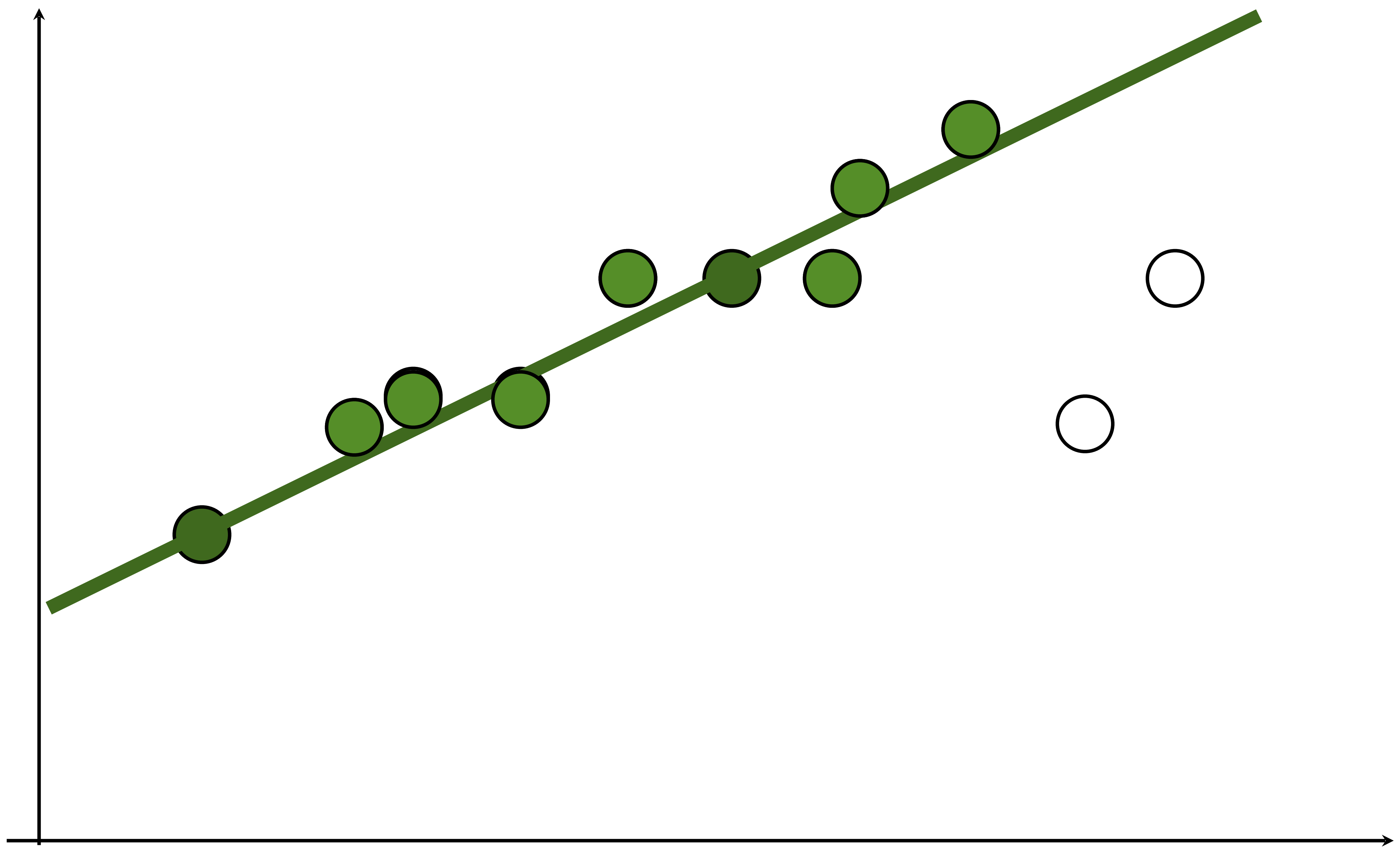
- Pick 2 points
- Fit line
- Count inliers

Simple example: fit a line



- Pick 2 points
- Fit line
- Count inliers

Simple example: fit a line



- Use biggest set of inliers
- Do least-square fit

RANSAC for estimating homography

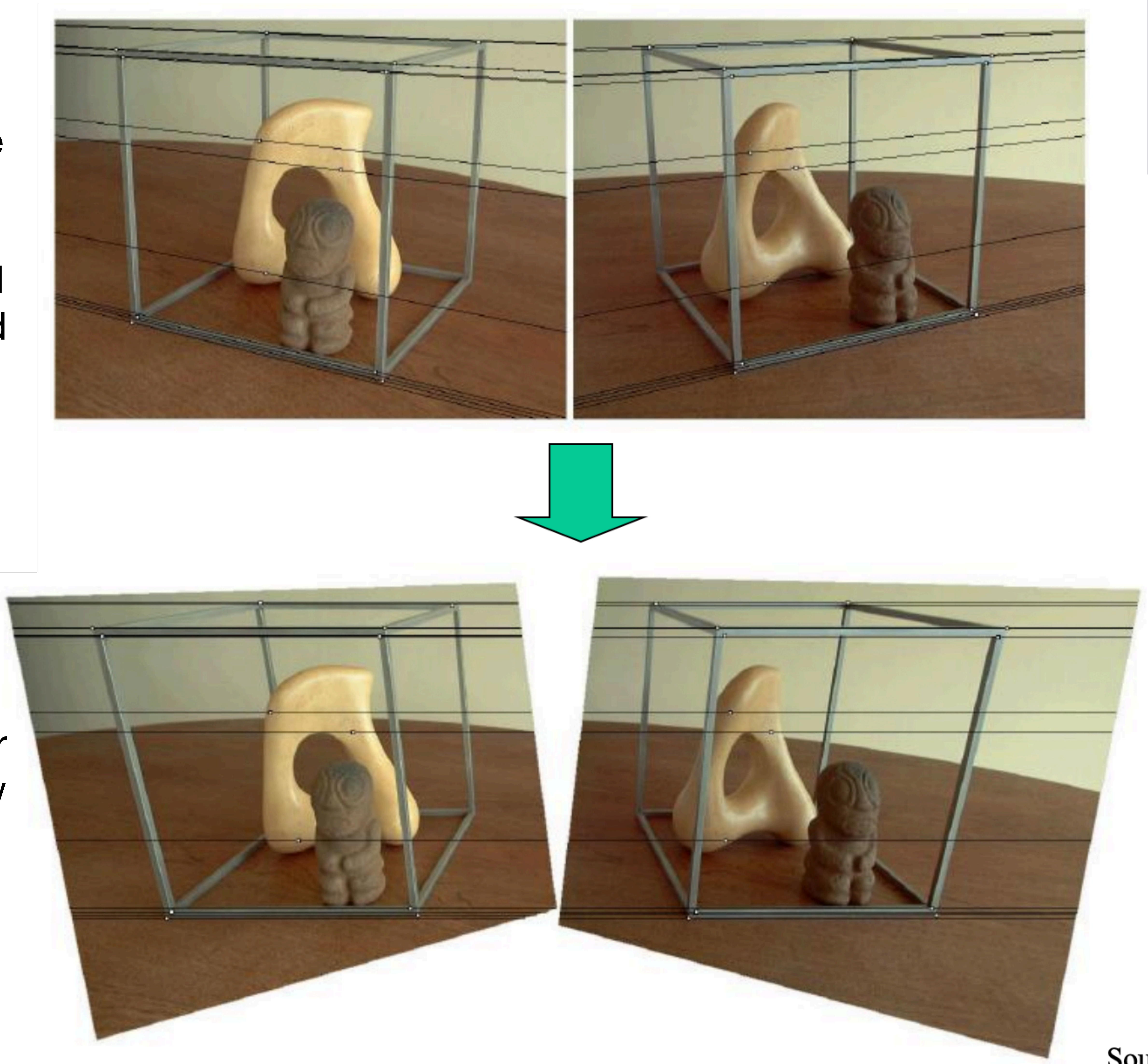
RANSAC loop:

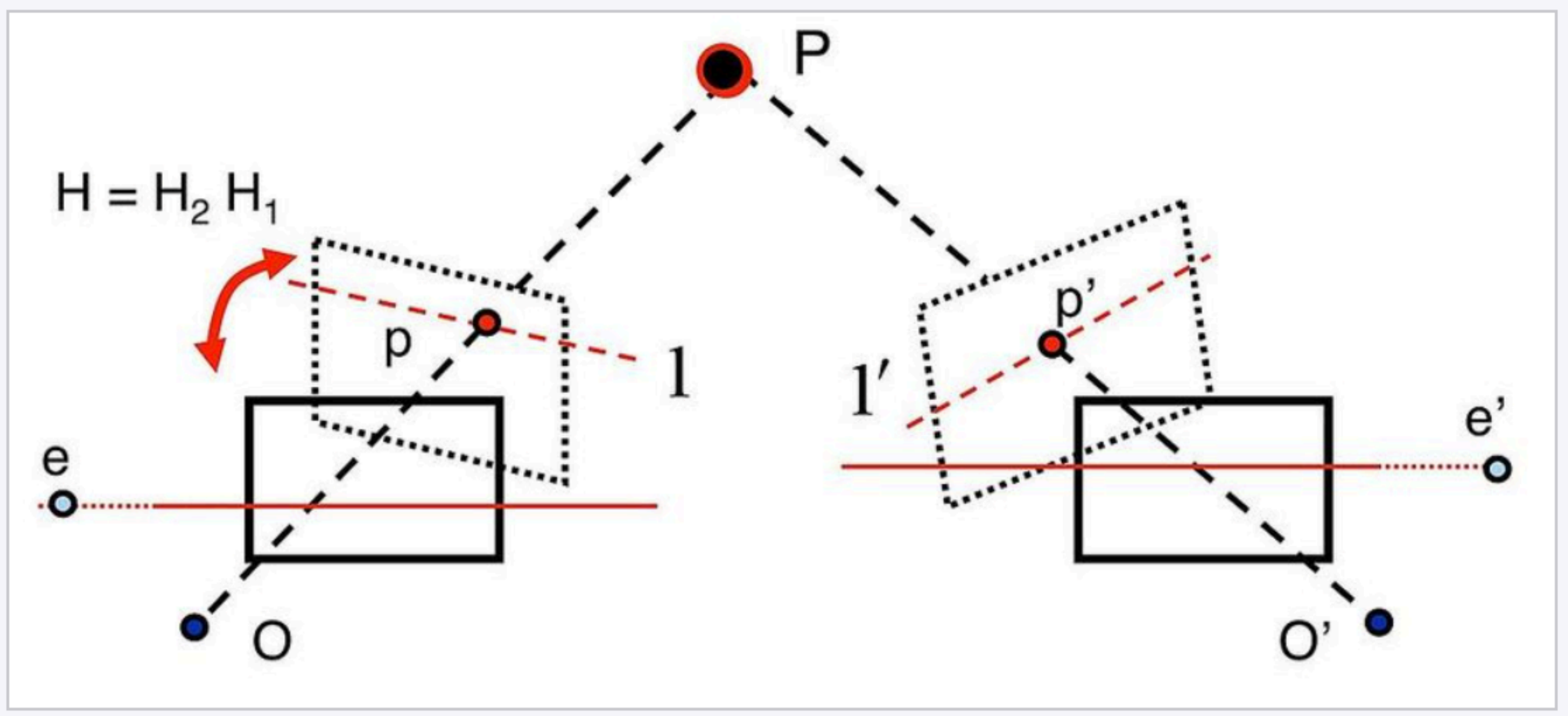
1. Select four feature pairs (at random)
2. Compute homography H (exact)
3. Compute *inliers* where $||p_i', \boldsymbol{H}\, p_i|| < \varepsilon$
4. Keep largest set of inliers
5. Re-compute least-squares H estimate using all of the inliers

Image rectification, a pre-processing for stereo

The images from a stereo pair can be transformed to appear as they would have had the cameras in the stereo rig been rotated to have coplanar sensors and their epipolar lines oriented along pixel scan lines.

This transformation just involves virtual camera rotations and thus is a homography. Image rectification is a common pre-processing step for stereo, since the search for matching features can now be along horizontal scan lines.





Model used for image rectification example.

from Wikipedia

Source: Alyosha Efros

summary / recap:

- Stereo and how it works
- Homogeneous coordinates for clean description of the geometry
- Intrinsic and extrinsic camera parameters
- Homographies for image stitching, for image rectification, etc.
- Ransac for fitting parameterized models such as homographies.