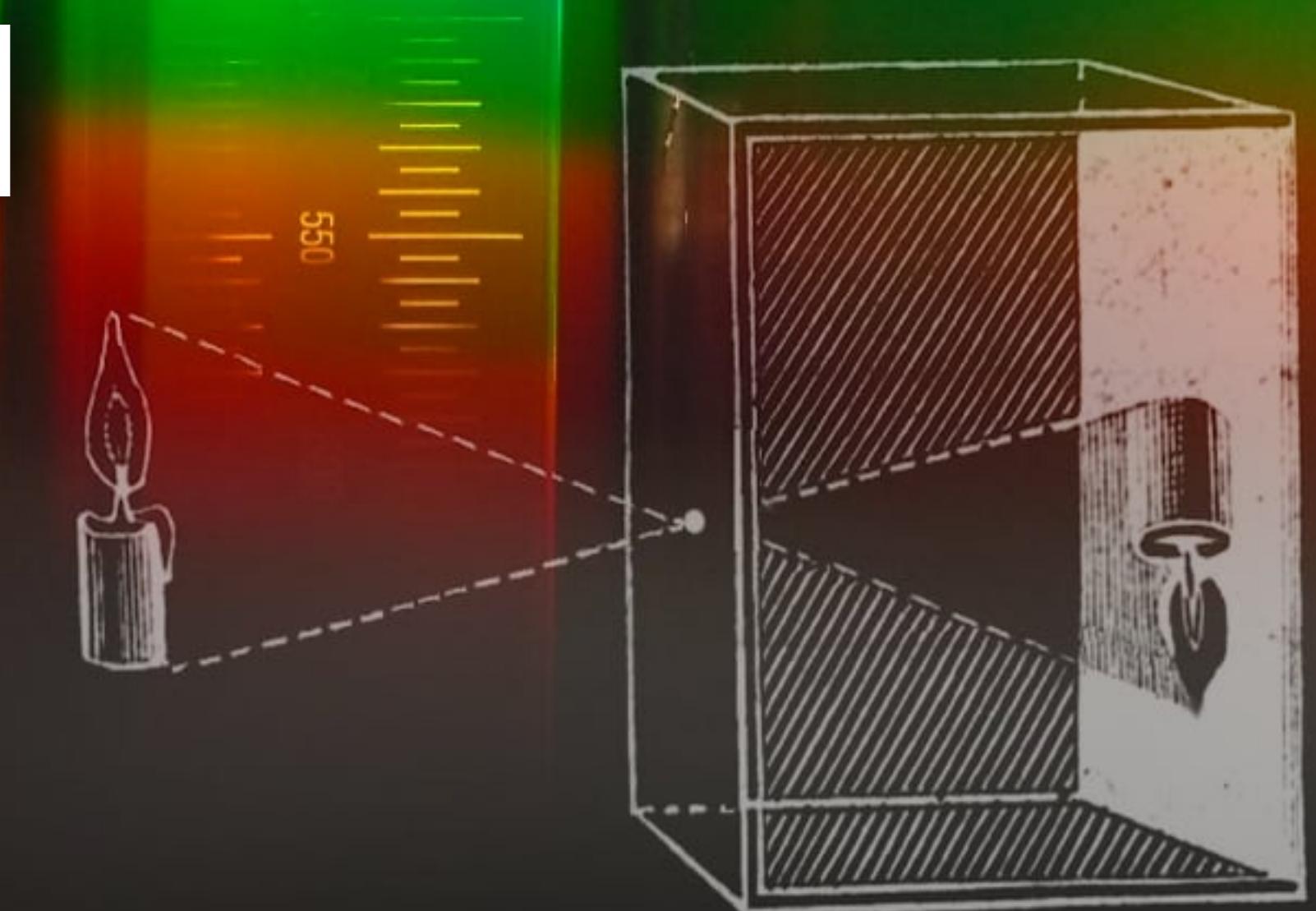


# IMAGE FORMATION



6.S980

# Other Admin Things

- **Course website is ground-truth and supersedes all other info!**
- **Please don't write me emails, write Piazza messages.**
- Find calendar to subscribe to on course website.
- MIT Lecture Capture system had a bug for first lecture - no recording will be available :/
- Slides posted.
- First assignment posted later today!

# Course Project Clarifications

# Course Project Clarifications

- Teams of 1-3.

# Course Project Clarifications

- Teams of 1-3.
- Any topic related to 3D computer vision is allowed. It has to have some 3D component - differentiable rendering, geometry, shape, depth, normals, pose estimation, correspondence, scene flow, ...

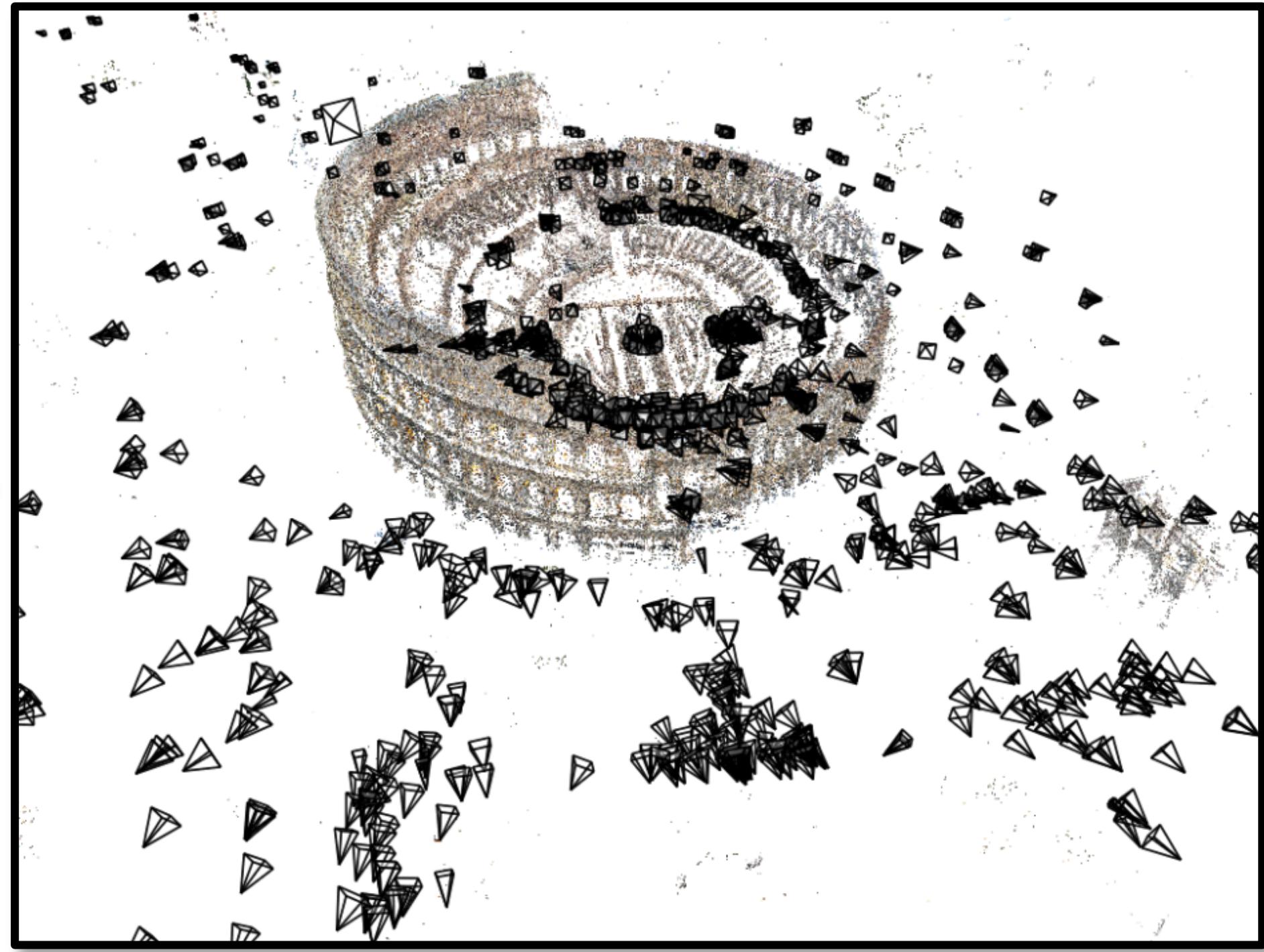
# Course Project Clarifications

- Teams of 1-3.
- Any topic related to 3D computer vision is allowed. It has to have some 3D component - differentiable rendering, geometry, shape, depth, normals, pose estimation, correspondence, scene flow, ...
- **Counts for 30% of final grade.**
  - Proposal (5%)
  - ~3-minute Mid-term Presentation (5%)
  - Final Report + Presentation (20%)
  - Late days for proposal, but not for final report.

# Course Project Clarifications

- Teams of 1-3.
- Any topic related to 3D computer vision is allowed. It has to have some 3D component - differentiable rendering, geometry, shape, depth, normals, pose estimation, correspondence, scene flow, ...
- Counts for 30% of final grade.
  - Proposal (5%)
  - ~3-minute Mid-term Presentation (5%)
  - Final Report + Presentation (20%)
  - Late days for proposal, but not for final report.
- See course website for up-to-date deadlines. Summary:
  - Proposal due Oct 18.
  - Project Mid-term Update Presentation (3 min per team) in Nov. 3rd lecture.
  - Final project presentations (~5-10 minutes per team) on Dec 8th and Dec 13th.

# Image Formation and Multi-View Geometry



Why?

We want to understand 3D world only from 2D observations (images). For that, we need to have a mathematical understanding of how they are connected.

What you'll learn.

Mathematical model of cameras. Reconstruct camera poses, approximate geometry, and camera parameters from 2D images of a scene.

# Some Slides adapted from...

- CMU 16-889: Learning for 3D Vision

**Prof. Shubham Tulsiani**

- CMU 16-385: Computer Vision

**Prof. Kris Kitani**

- MIT 6.819/6.869: Advances in Computer Vision,

**Profs. Bill Freeman, Phillip Isola, Antonio Torralba**

# What is a 3D scene?



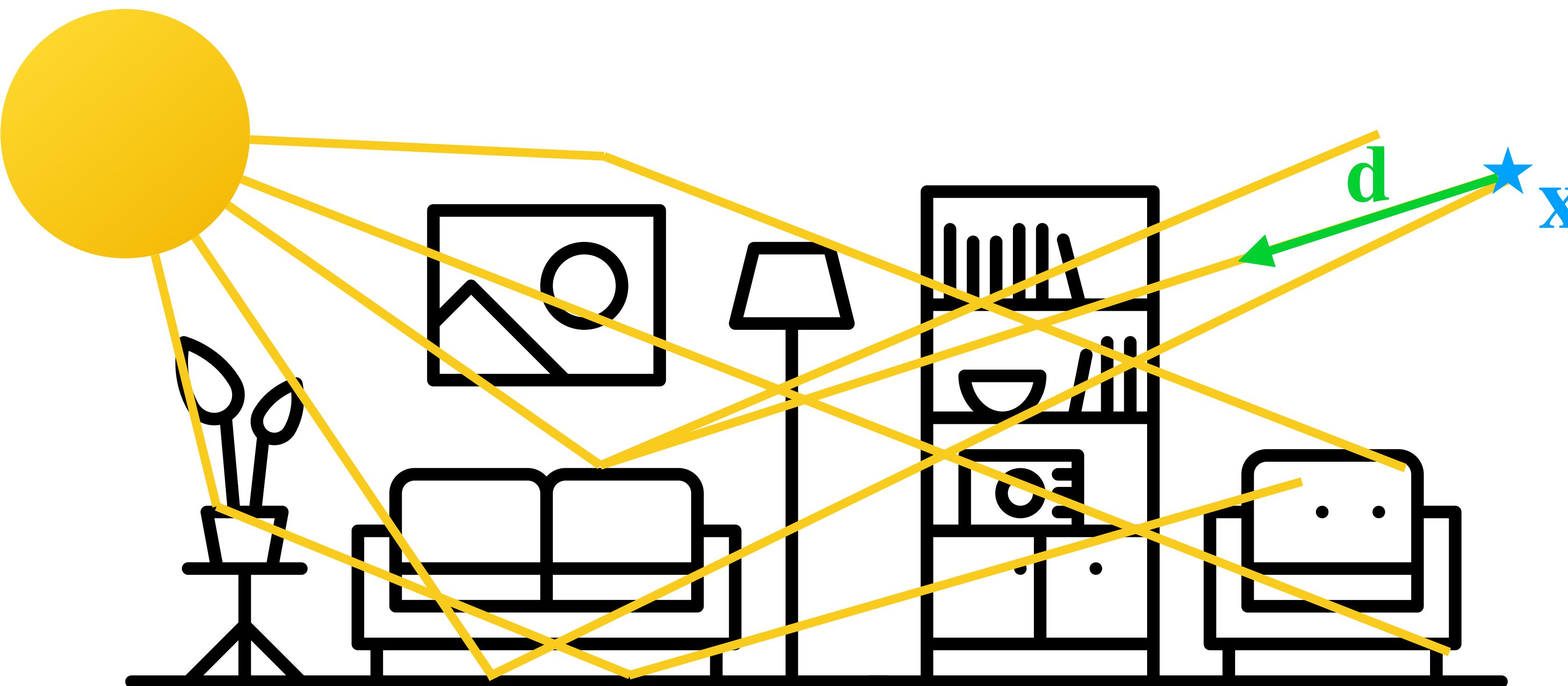
materials, light sources, 3D shape, color, weight, density, friction coefficients, etc

# How do we observe scenes?



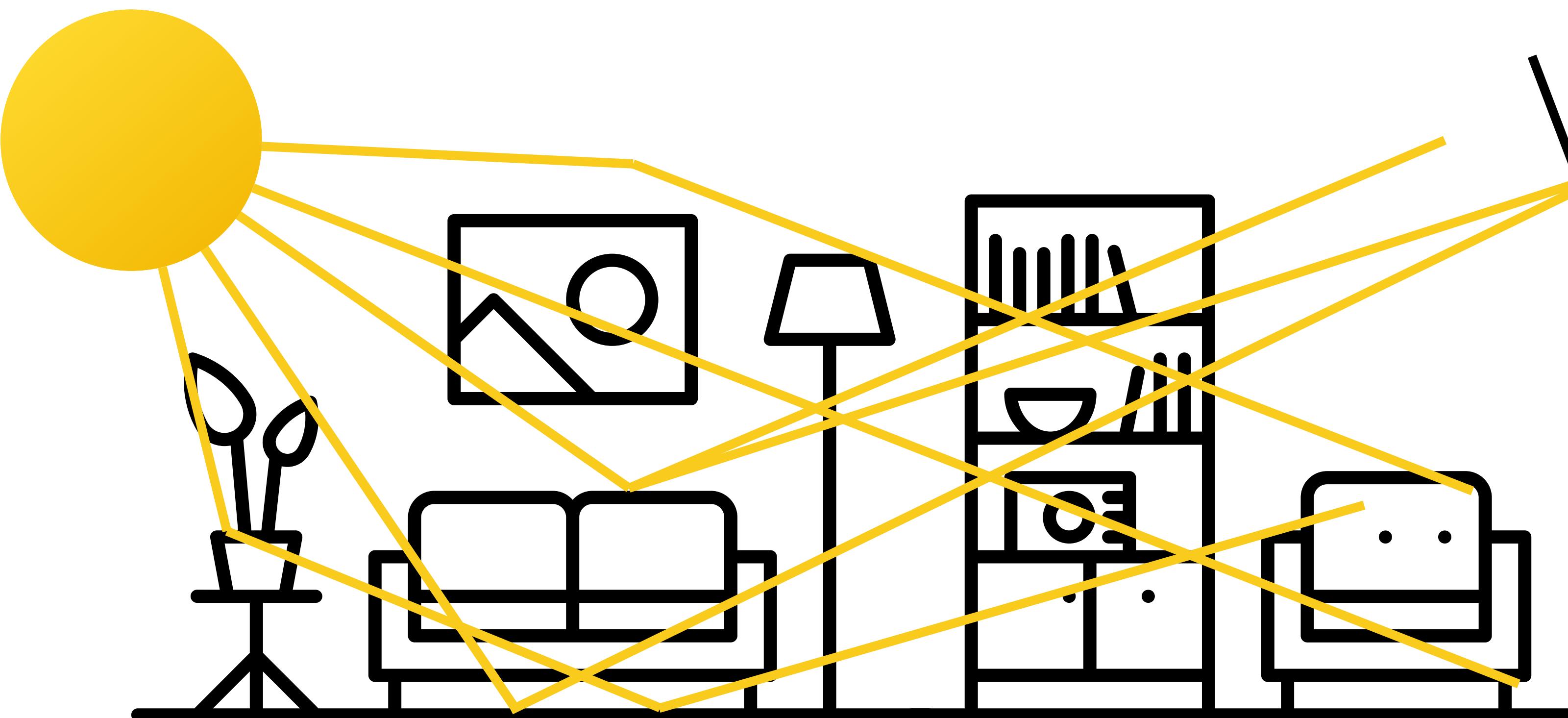
An eye (or a camera) observes a subset of all the light rays in a scene.

The Light Field: 3D coordinate plus ray direction is mapped to the color of that ray.

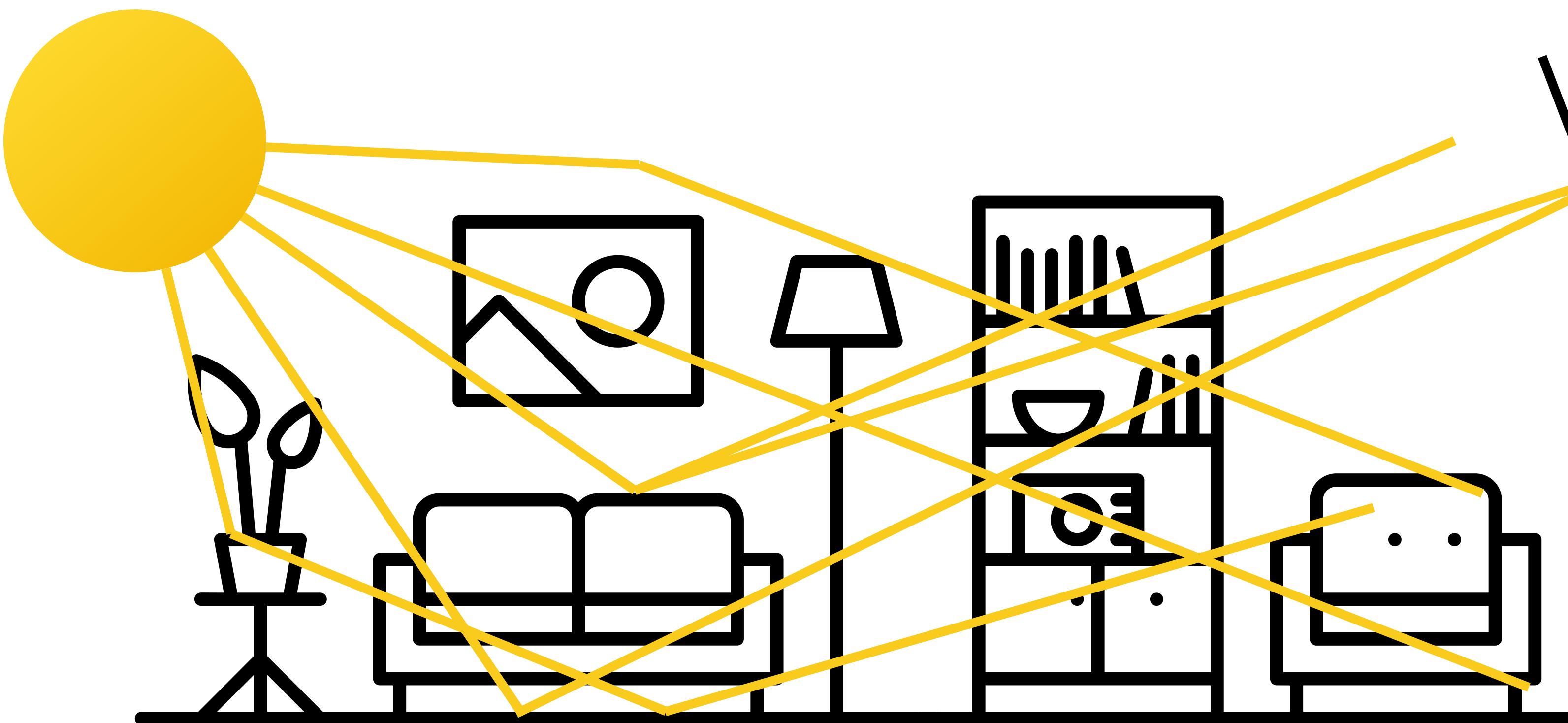


$$LF : \mathbb{R}^3 \times \mathbb{S}^2 \rightarrow \mathbb{R}^3, \quad LF(\mathbf{x}, \mathbf{d}) = \mathbf{c}$$

# Why don't we get an image if we hold up a piece of paper?

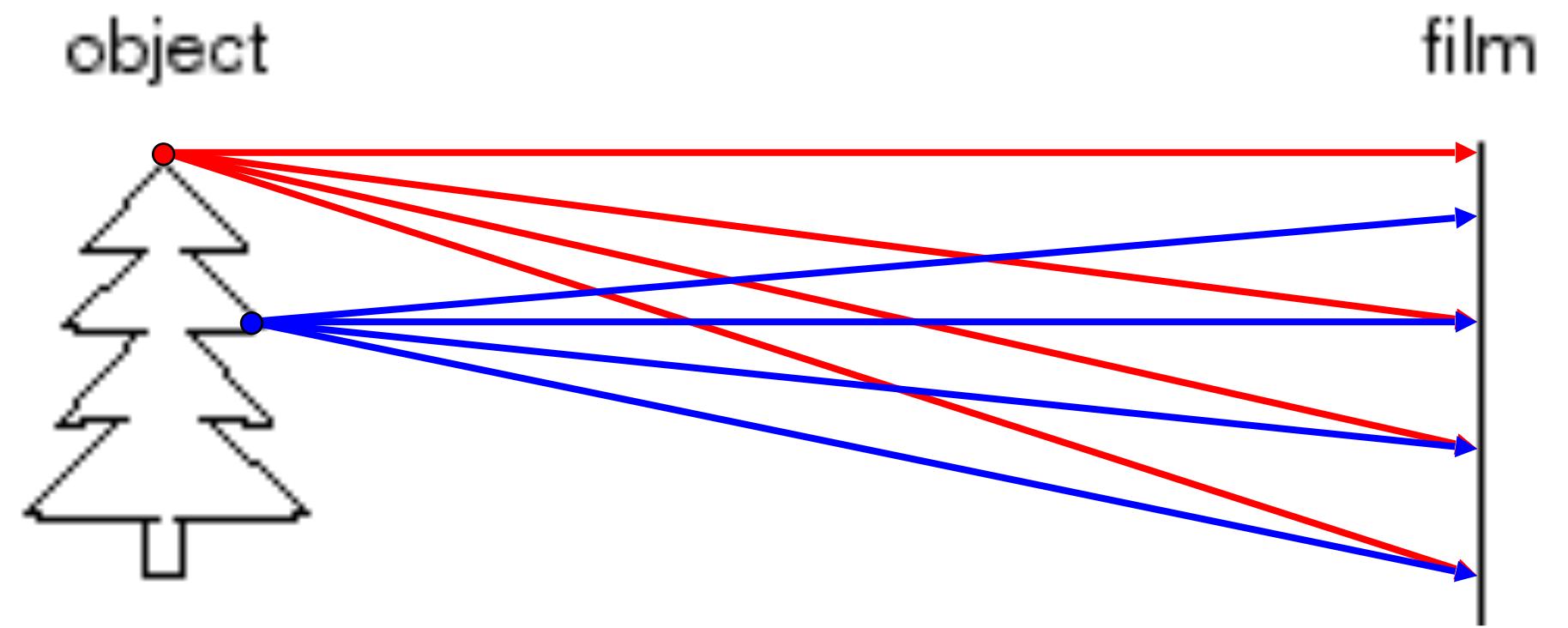


**Every “pixel” on the paper is the average of all the rays in the scene.**



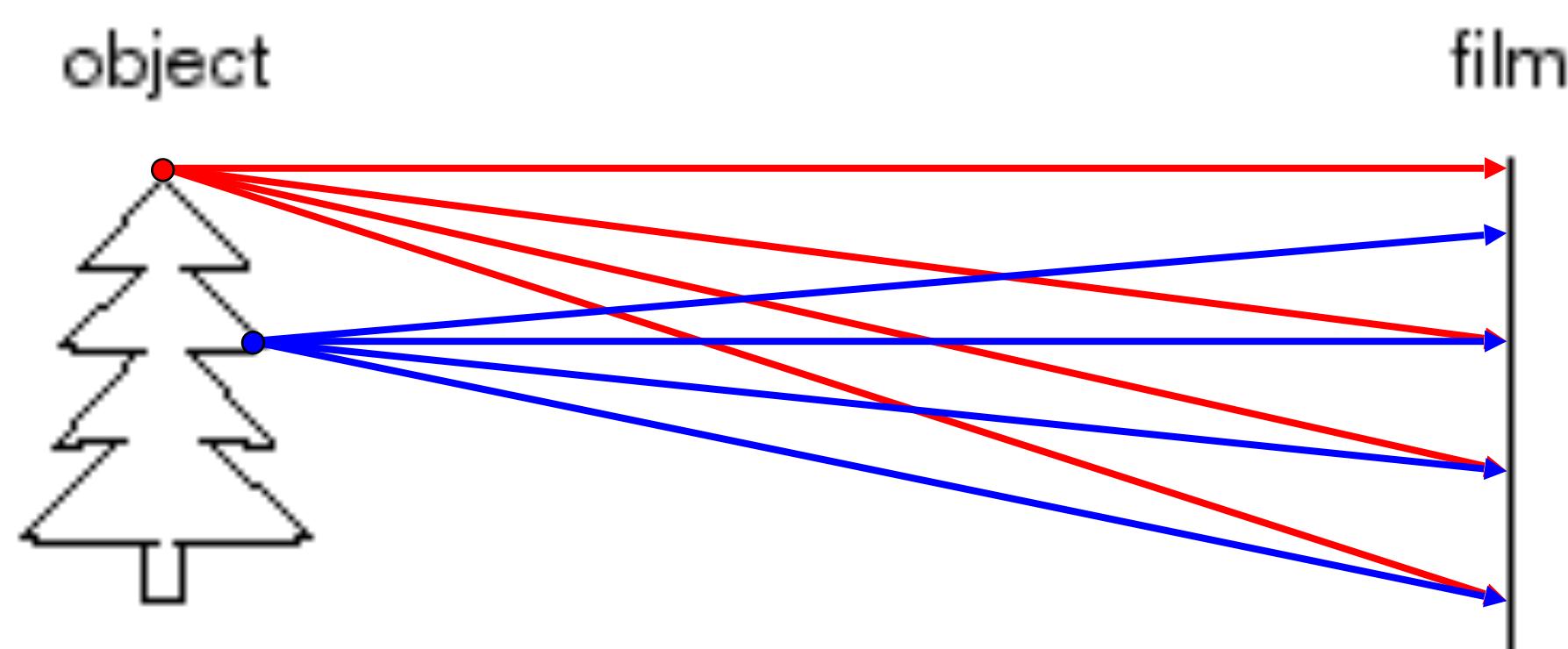
# Let's make a Camera!

# Let's make a Camera!

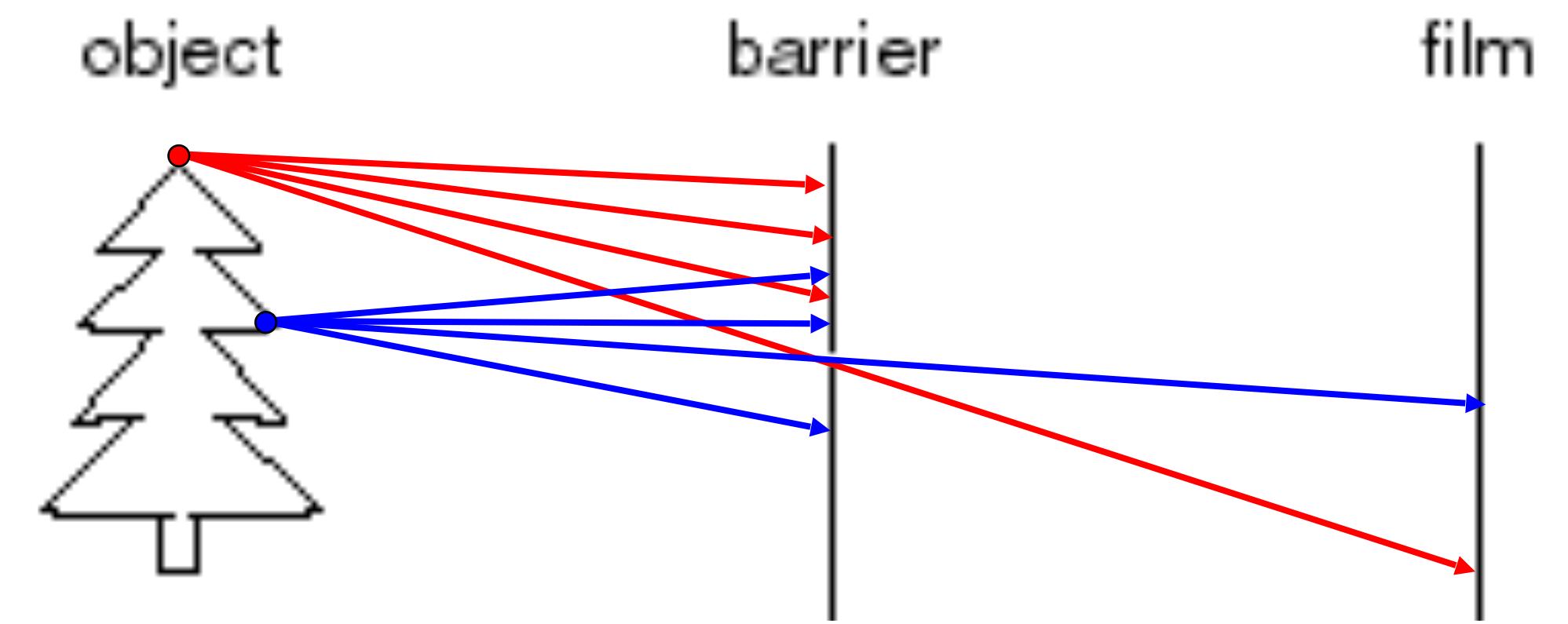


Idea 1: Put piece of film in front of an object

# Let's make a Camera!



Idea 1: Put piece of film in front of an object



Add a barrier to block most rays.

# Camera Obscura

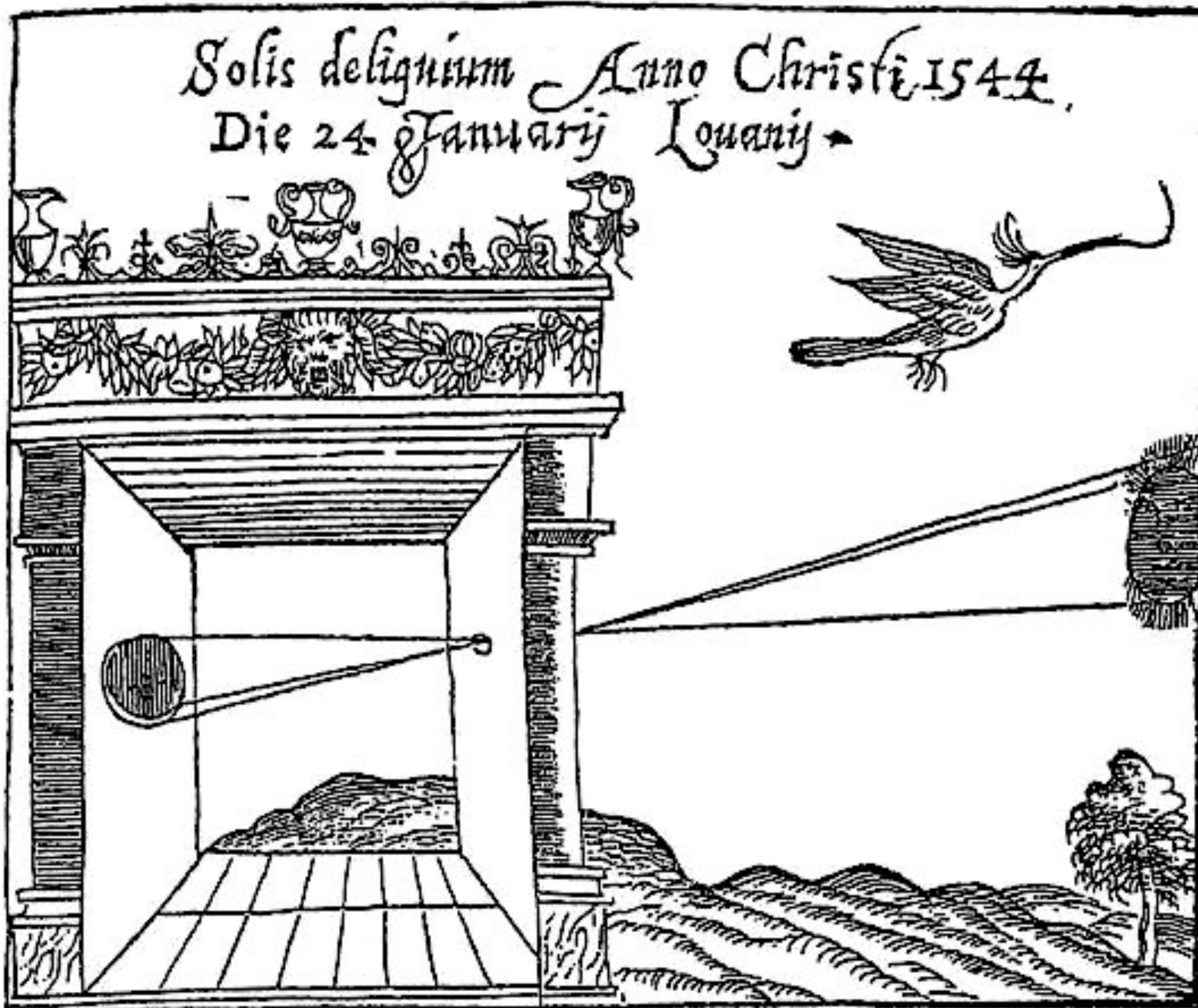
- Basic principle known to Mozi (470-390 BCE), Aristotle (384-322 BCE)
- Drawing aid for artists: described by Leonardo da Vinci (1452-1519)



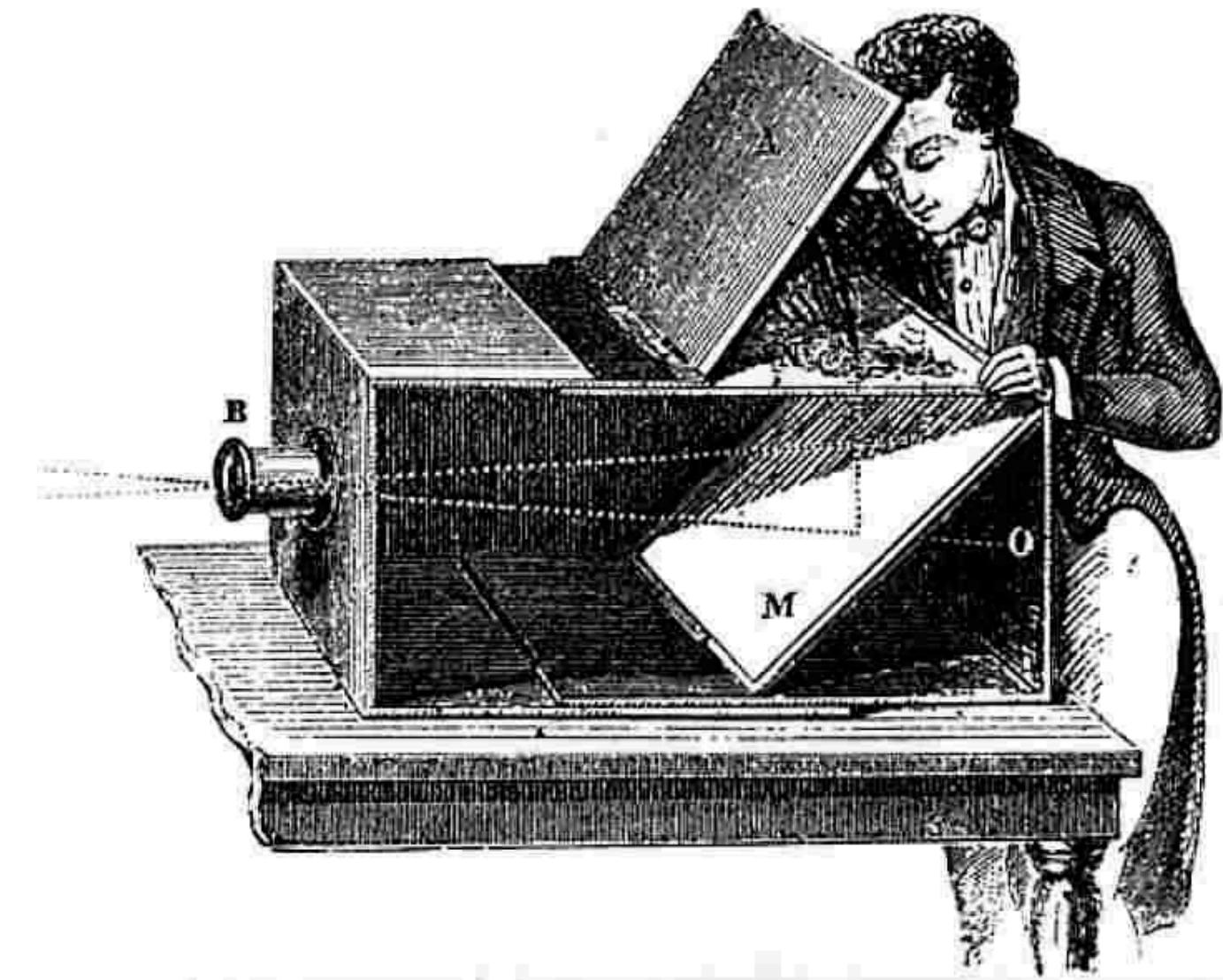
Gemma Frisius, 1558

# Camera Obscura

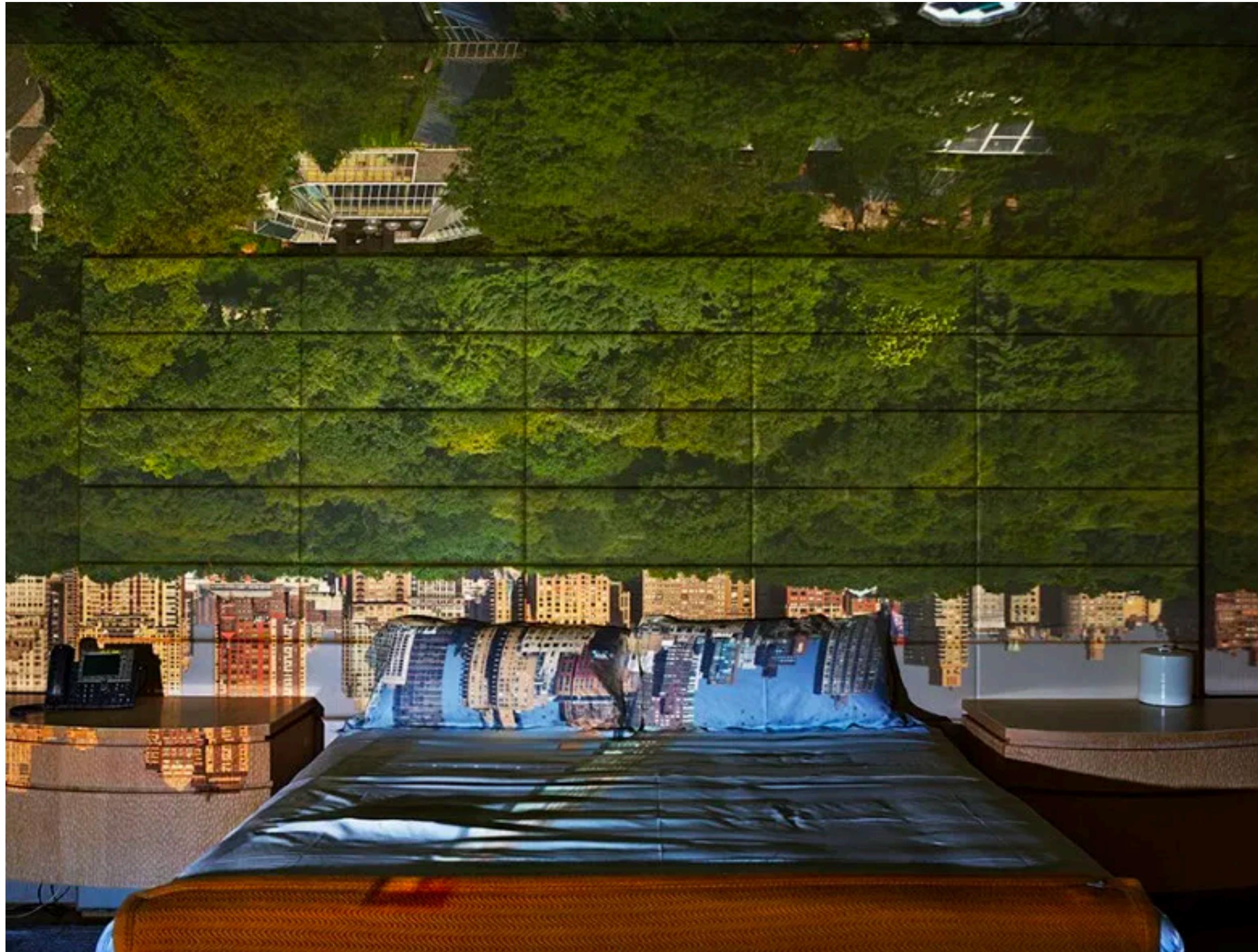
- Basic principle known to Mozi (470-390 BCE), Aristotle (384-322 BCE)
- Drawing aid for artists: described by Leonardo da Vinci (1452-1519)



Gemma Frisius, 1558



# Camera Obscura (*Dark Room*)



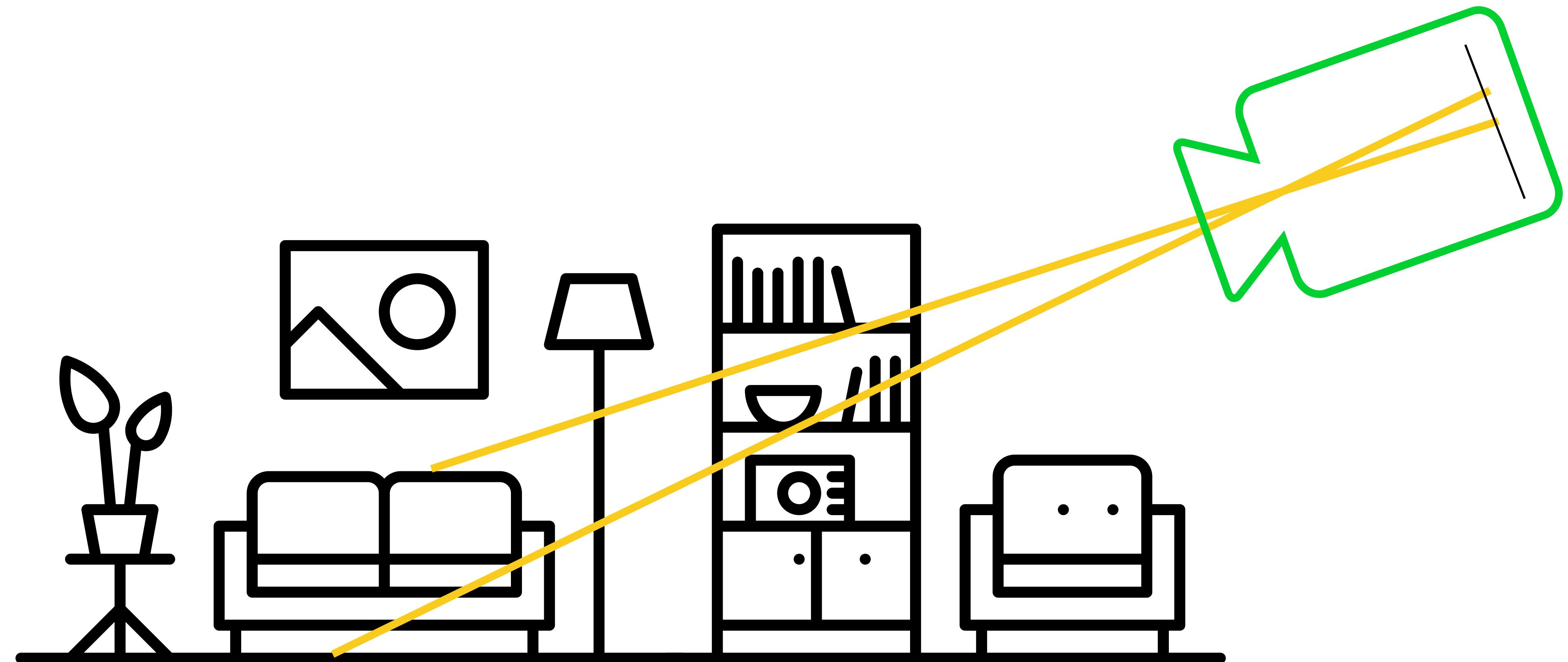
Camera Obscura: View of Central Park  
Looking West in Bedroom. Summer, 2018

After scouting rooms and reserving one for at least a day, Morell masks the windows except for the aperture. He controls three elements: the size of the hole, with a smaller one yielding a sharper but dimmer image; the length of the exposure, usually eight hours; and the distance from the hole to the surface on which the outside image falls and which he will photograph. He used 4 x 5 and 8 x 10 view cameras and lenses ranging from 75 to 150 mm.

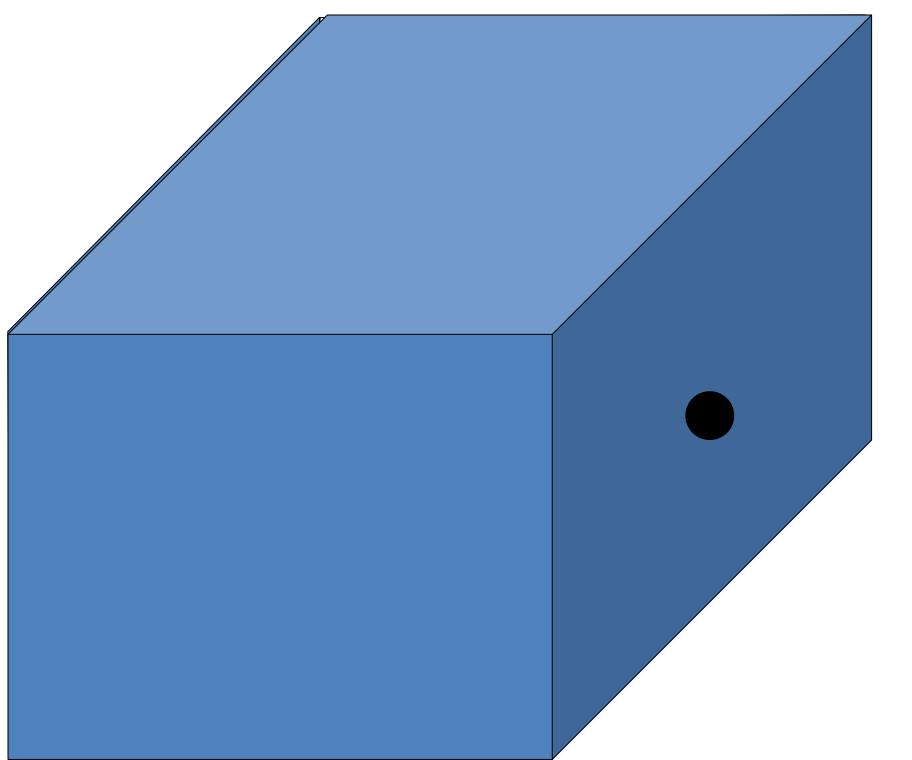
After he's done inside, it gets harder. "I leave the room and I am constantly checking the weather, I'm hoping the maid reads my note not to come in, I'm worrying that the sun will hit the plastic masking and it will fall down, or that I didn't trigger the lens."

[http://www.abelardomorell.net/  
project/camera-obscura/](http://www.abelardomorell.net/project/camera-obscura/)

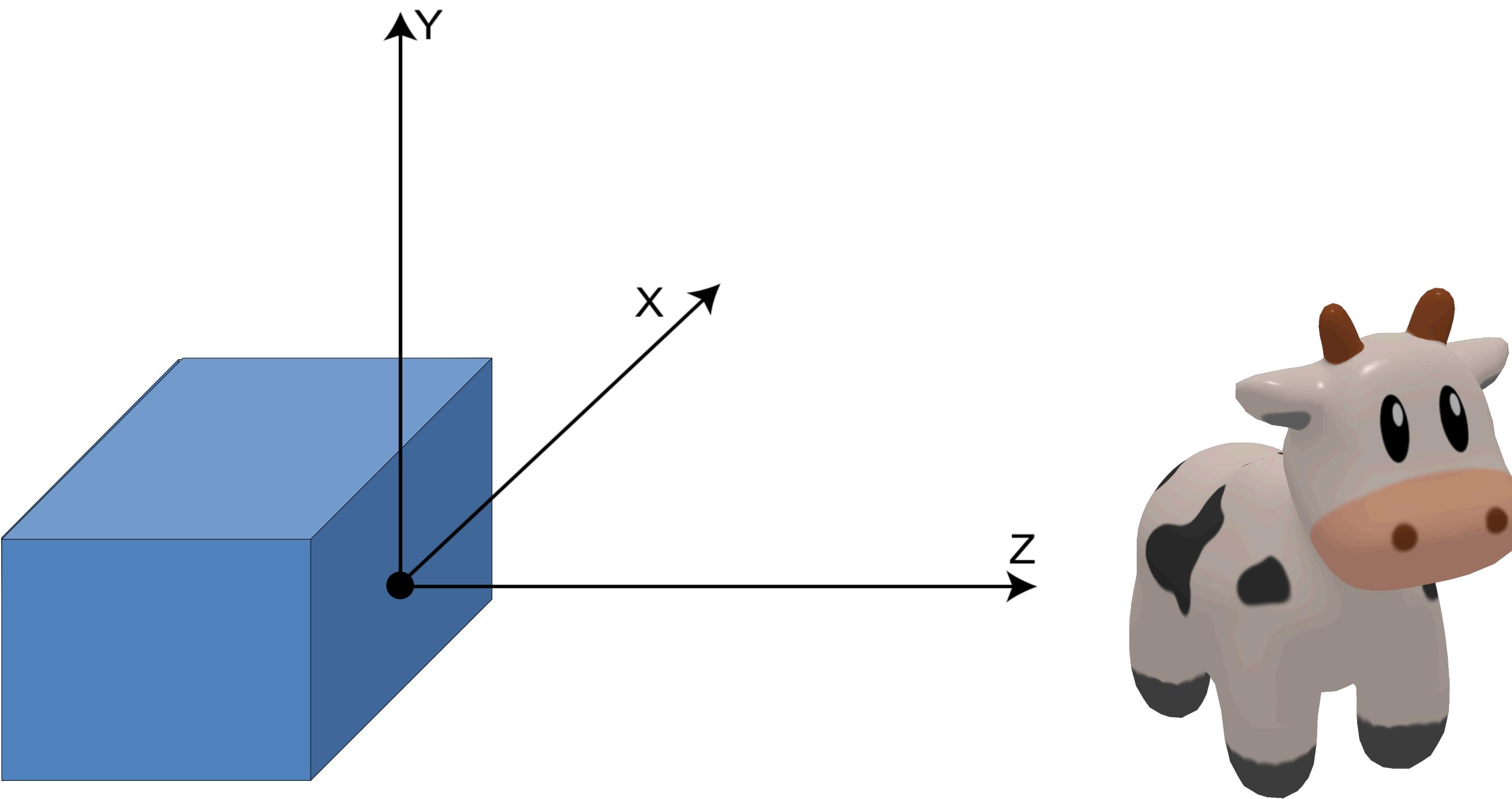
# Today: Model the camera, describe the rays.



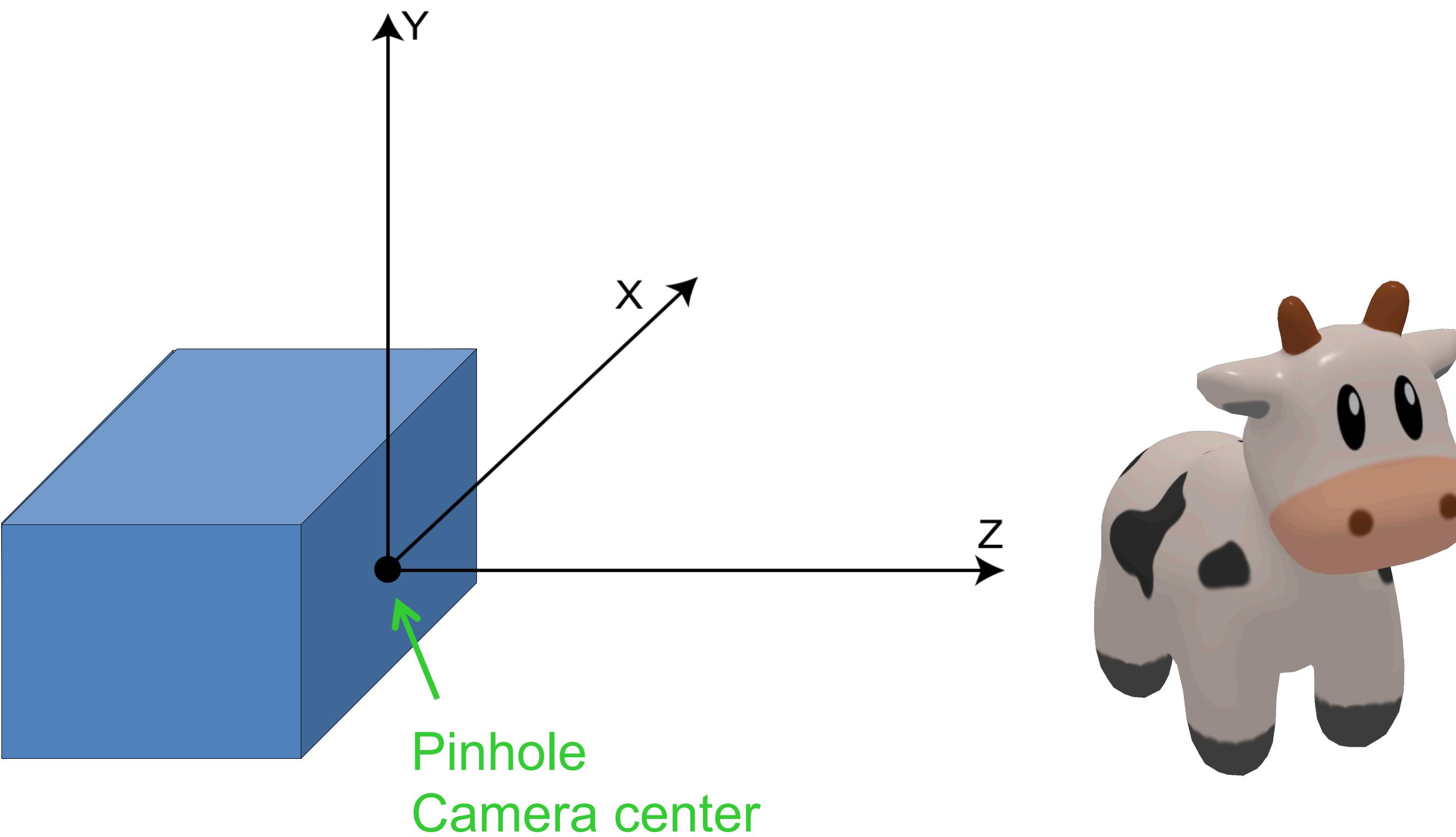
# Perspective projection



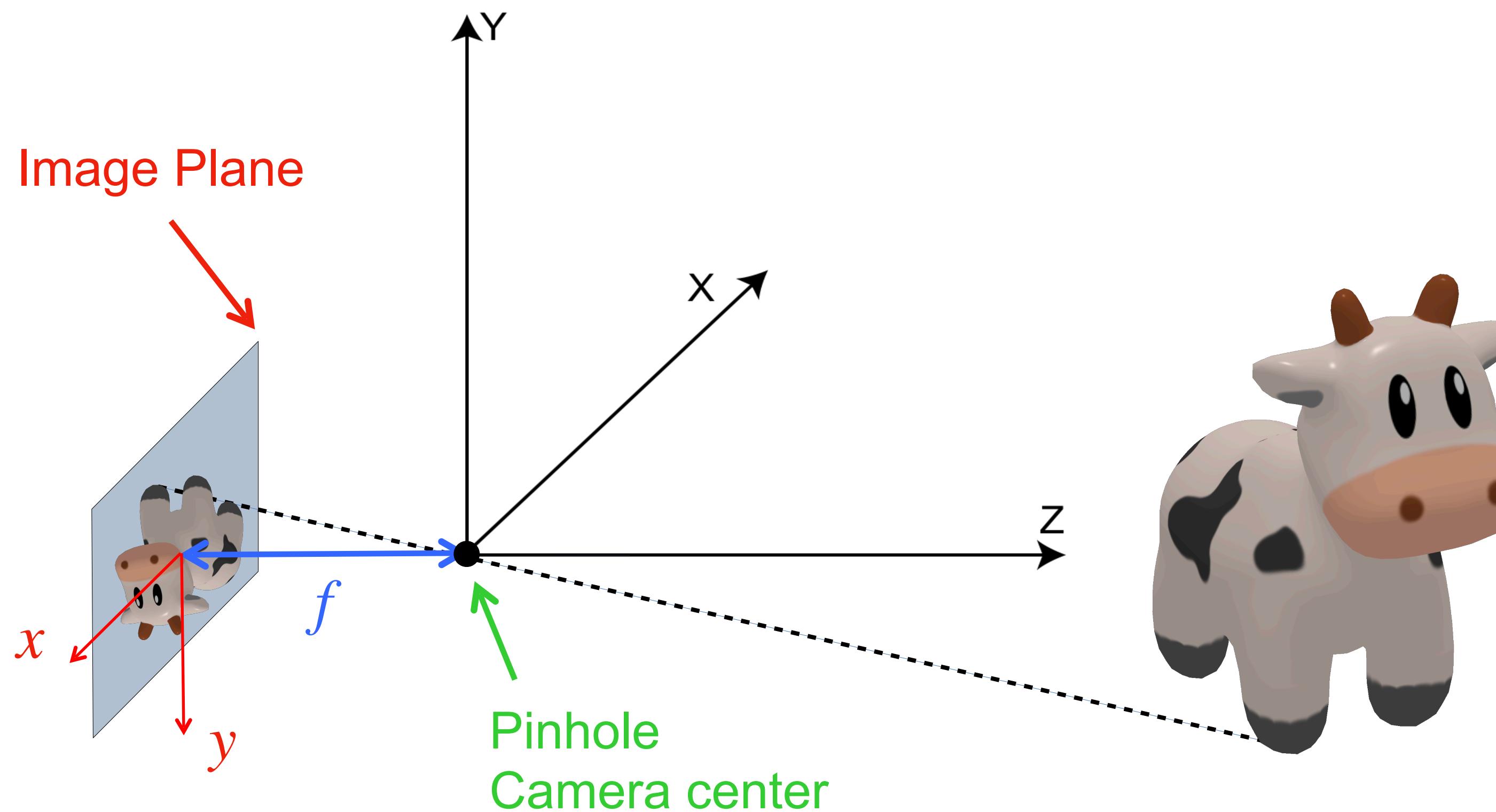
# Perspective projection



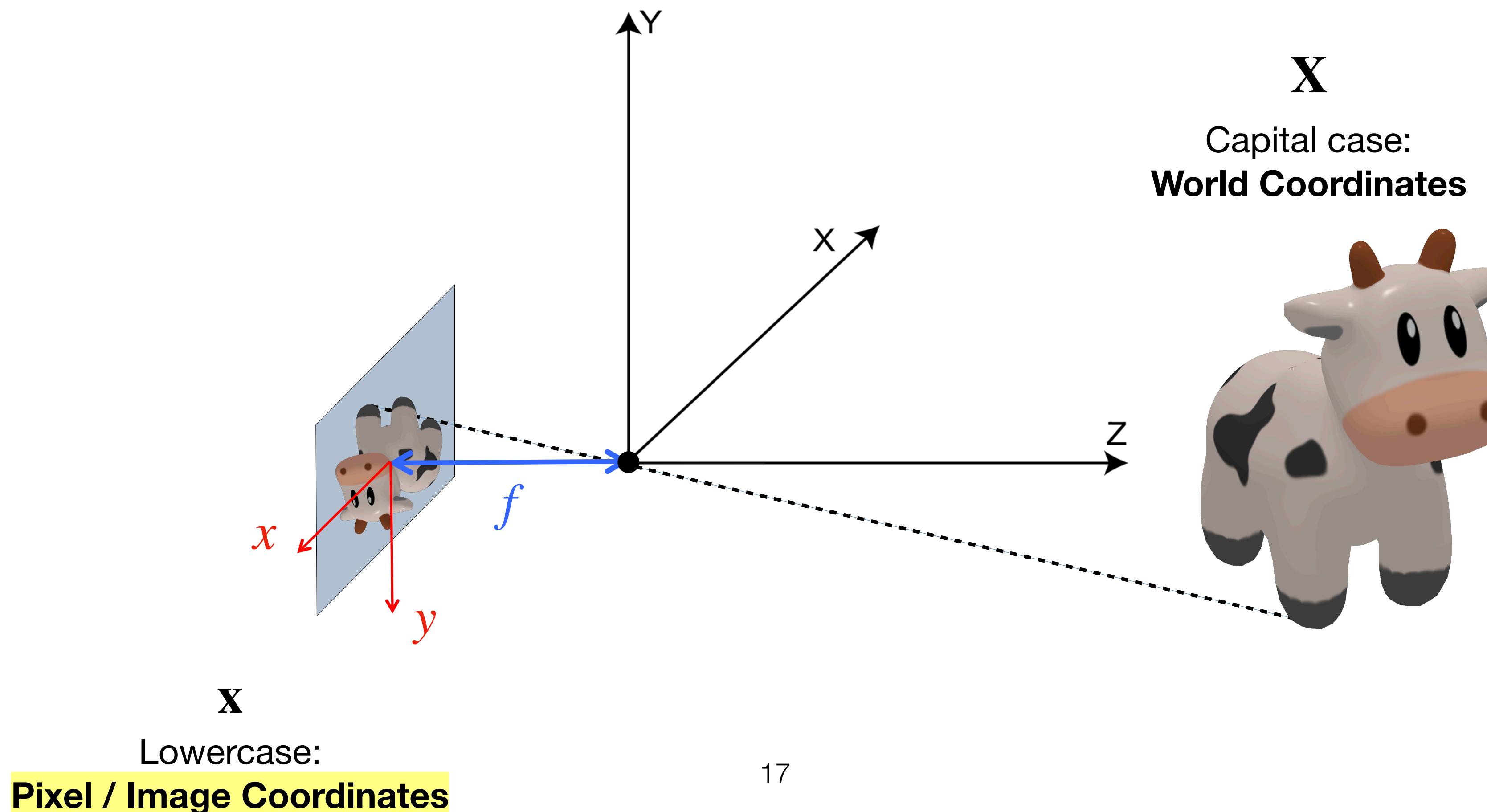
# Perspective projection



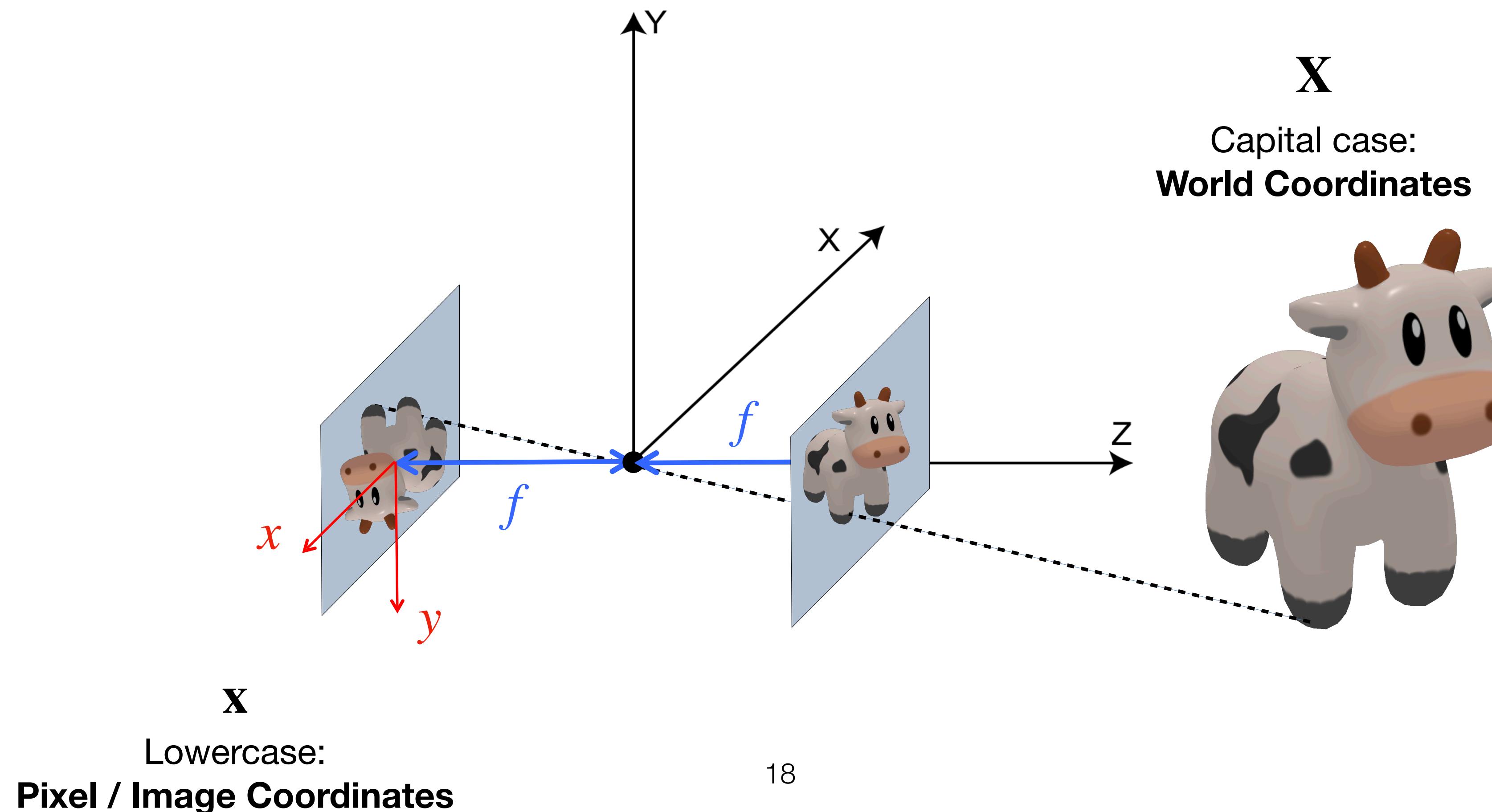
# Perspective projection



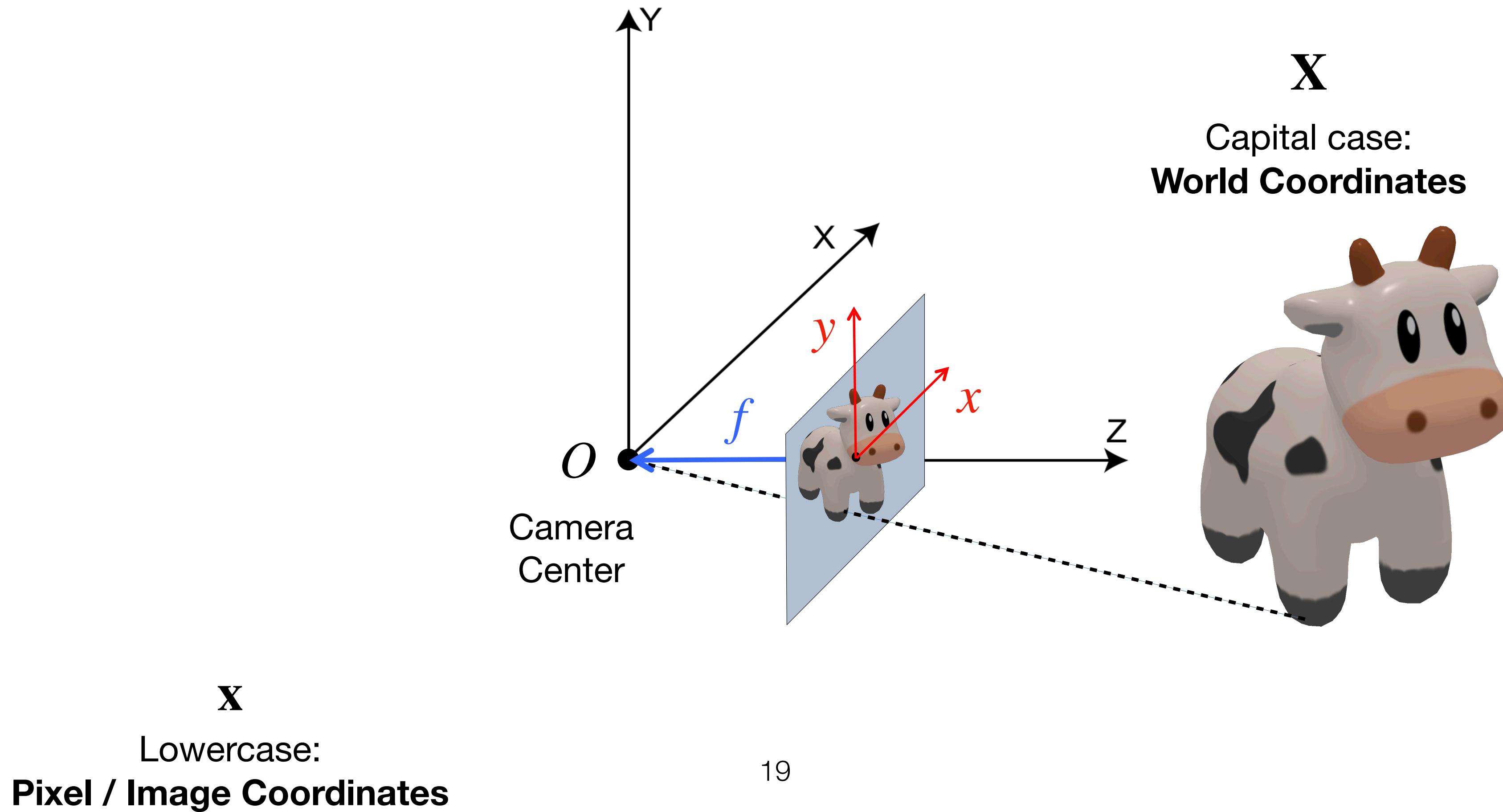
# Perspective projection



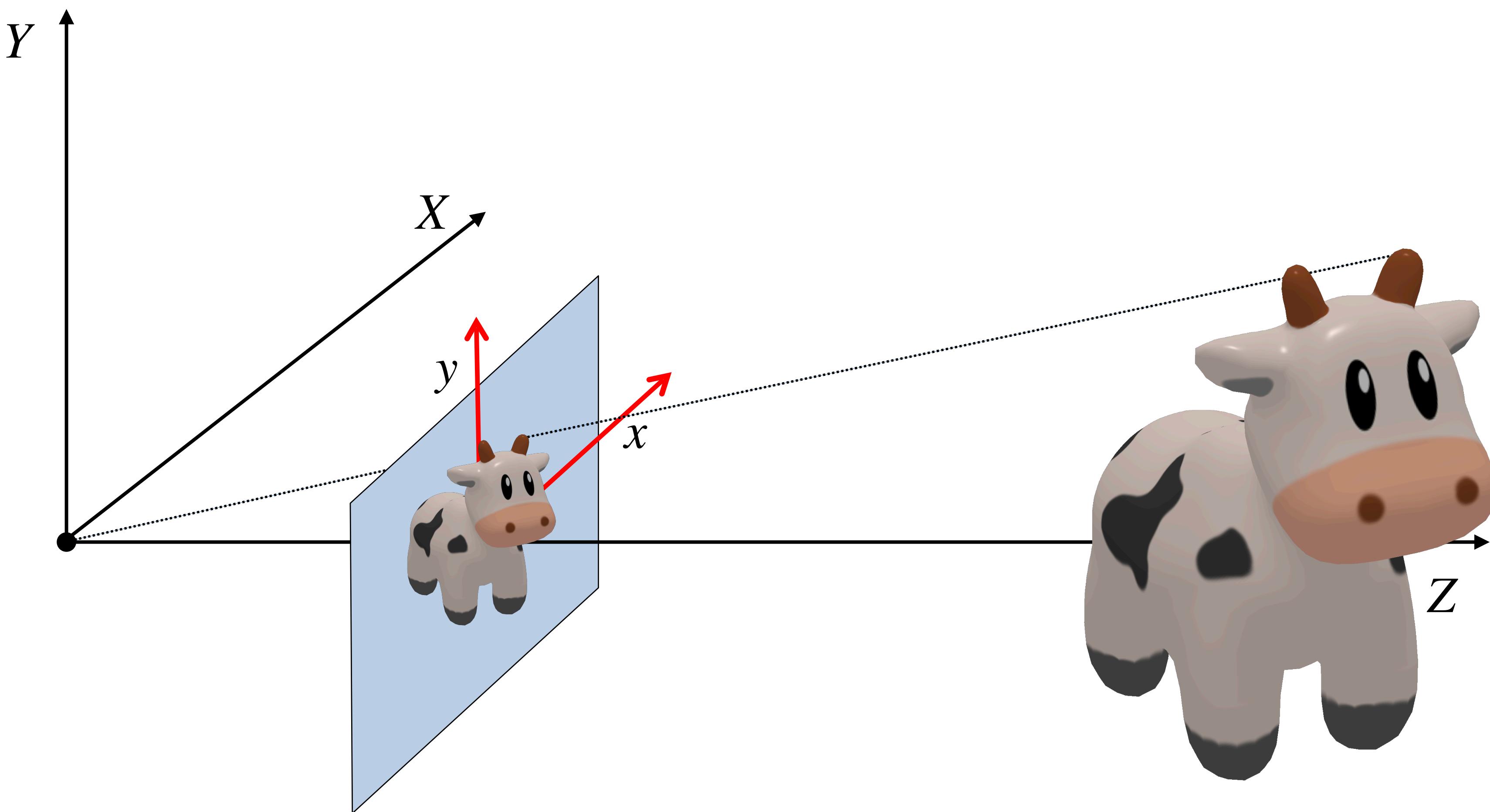
# Perspective projection



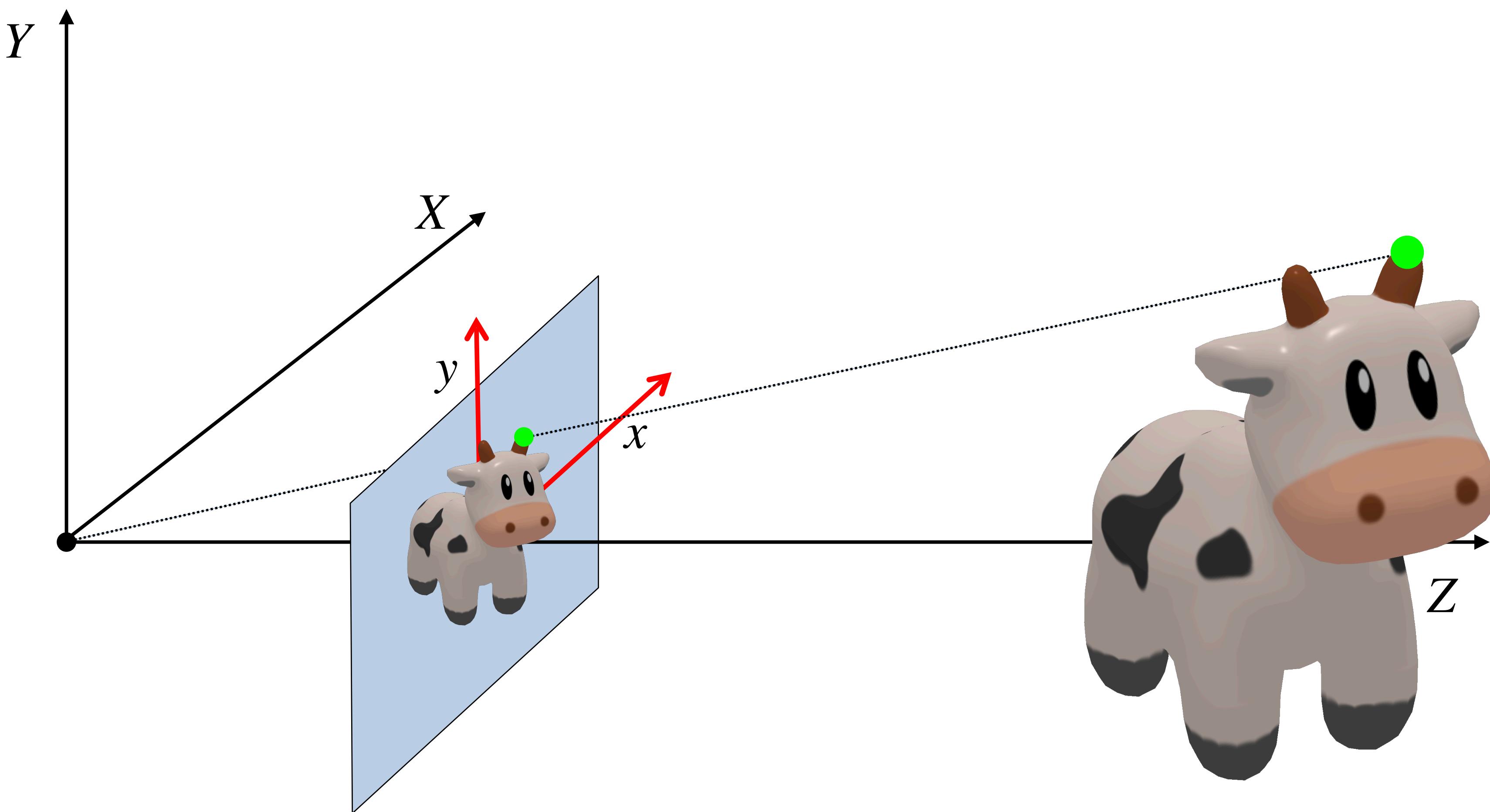
# Perspective projection



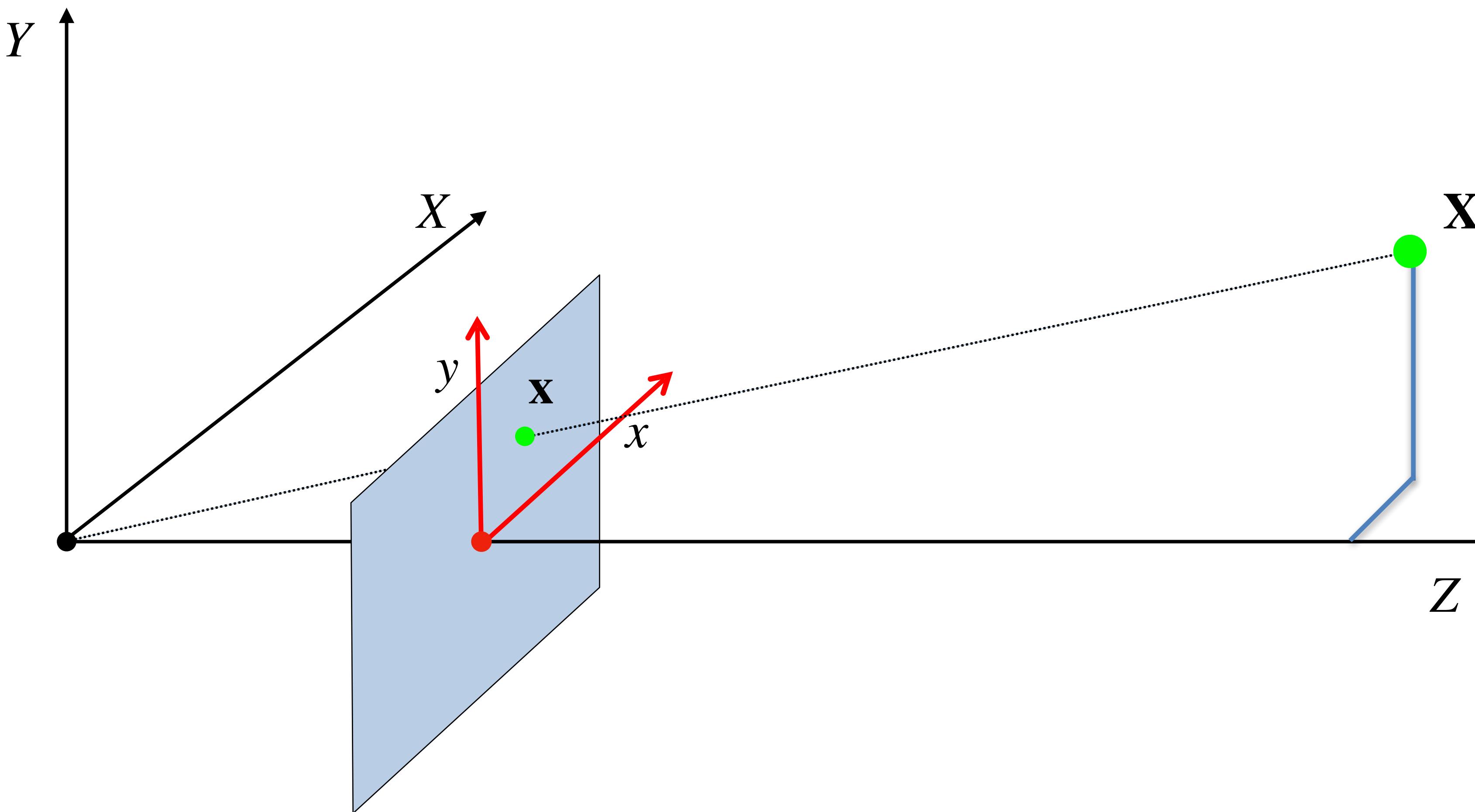
# Perspective projection



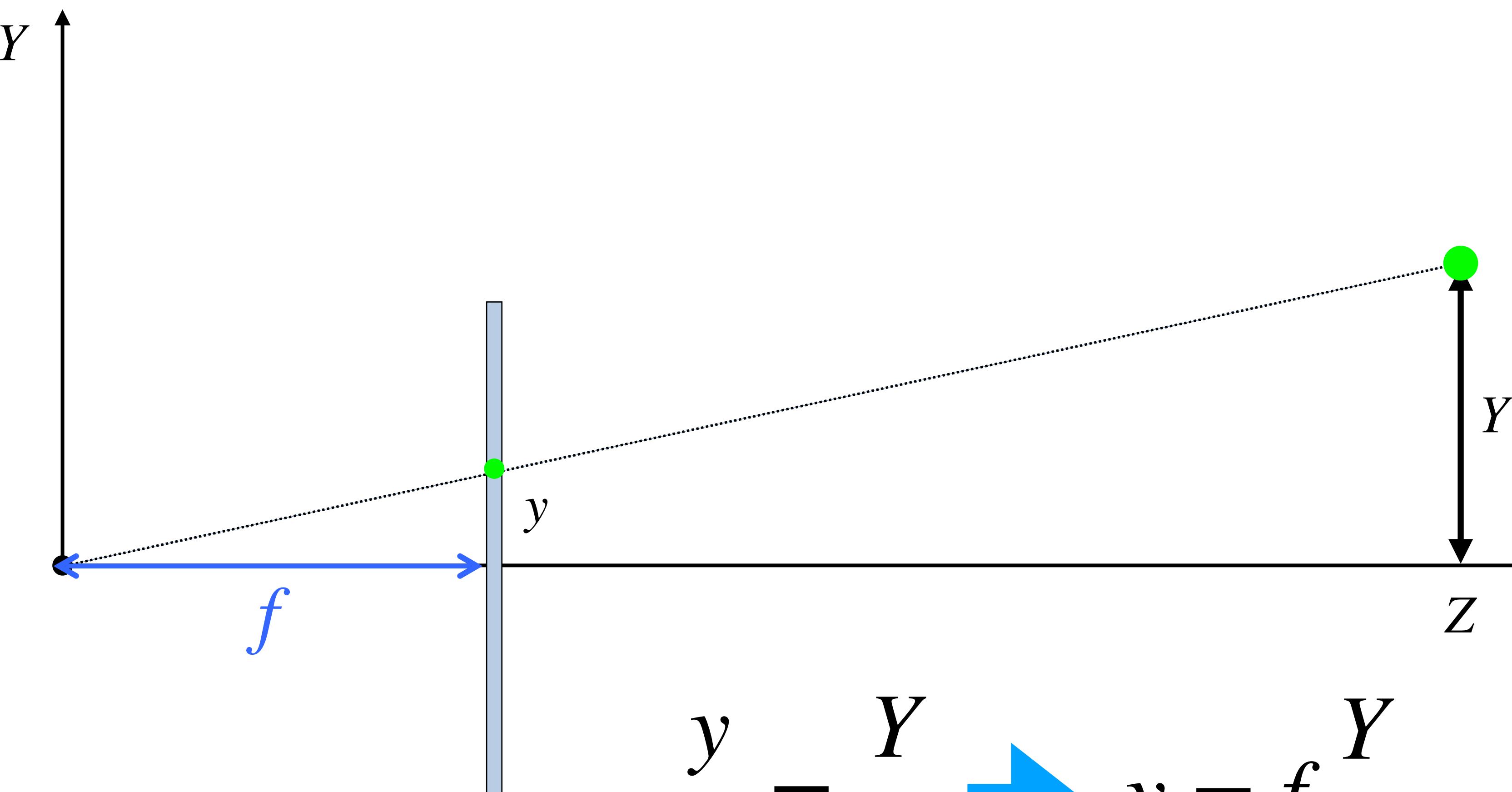
# Perspective projection



# Perspective projection

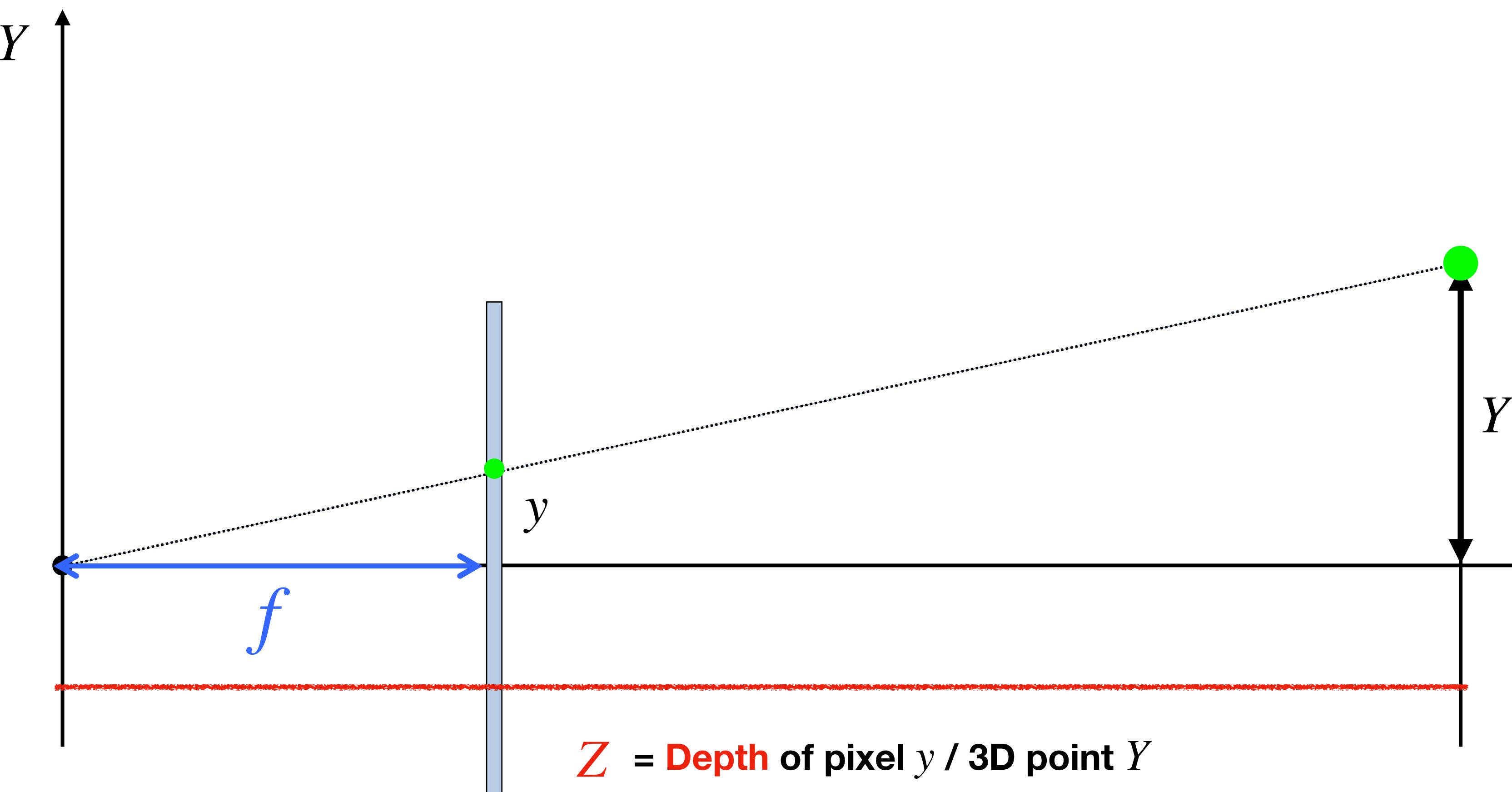


# Perspective projection



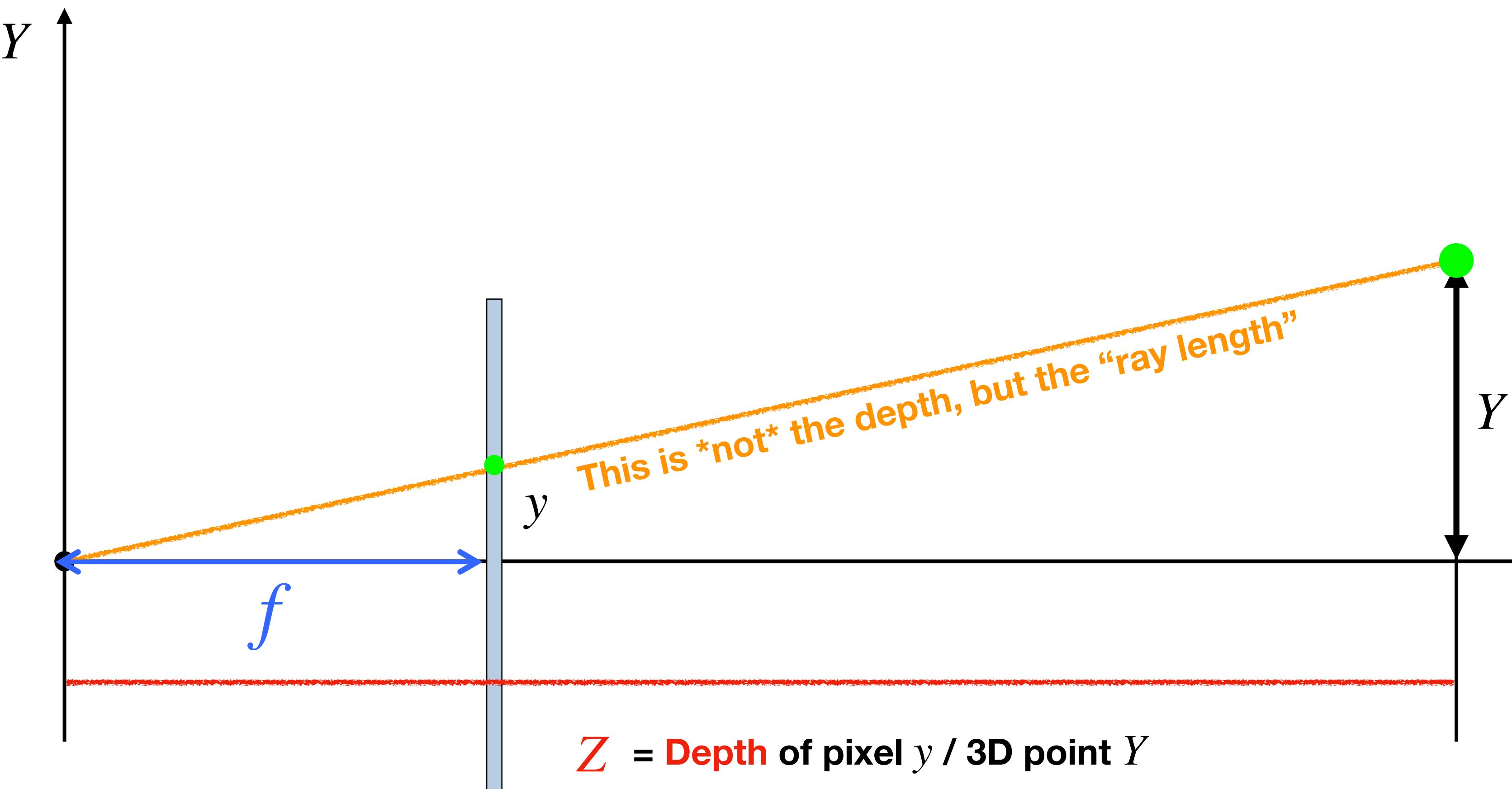
$$\frac{y}{f} = \frac{Y}{Z} \rightarrow y = f \frac{Y}{Z}$$

# Depth of a pixel / 3D point



$$y = f - \frac{Y}{Z}$$

# Depth vs. Ray Length



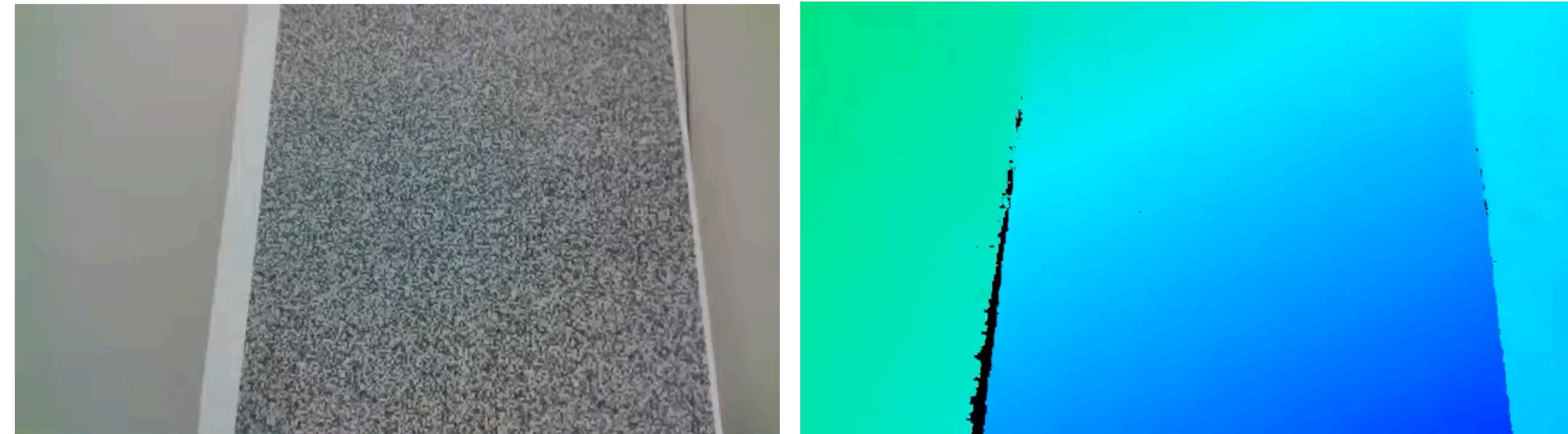
$$y = f \frac{Y}{Z}$$

# Depth Sensors



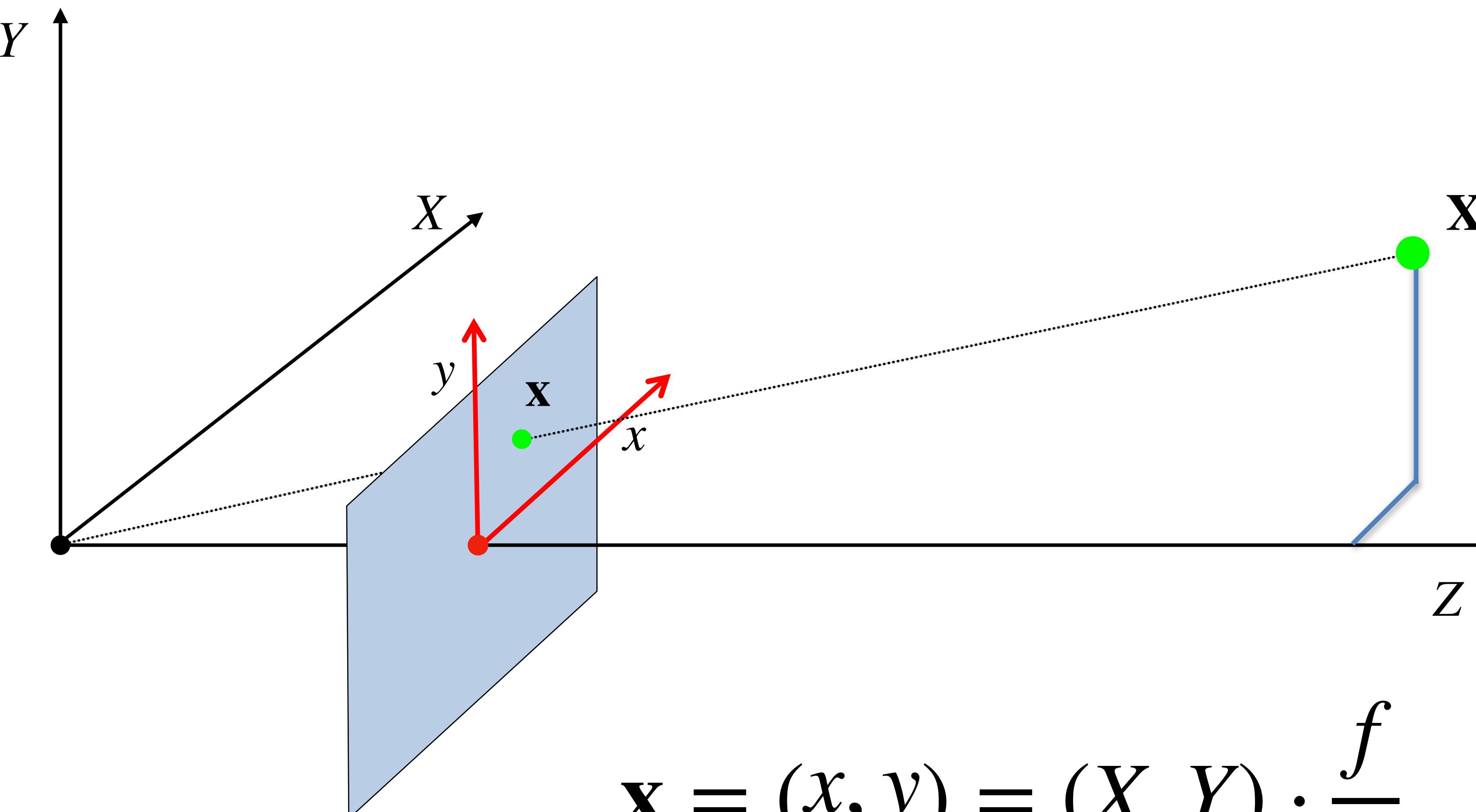
Output of Intel RealSense Camera

# Depth Sensors



Output of Intel RealSense Camera

# Perspective projection

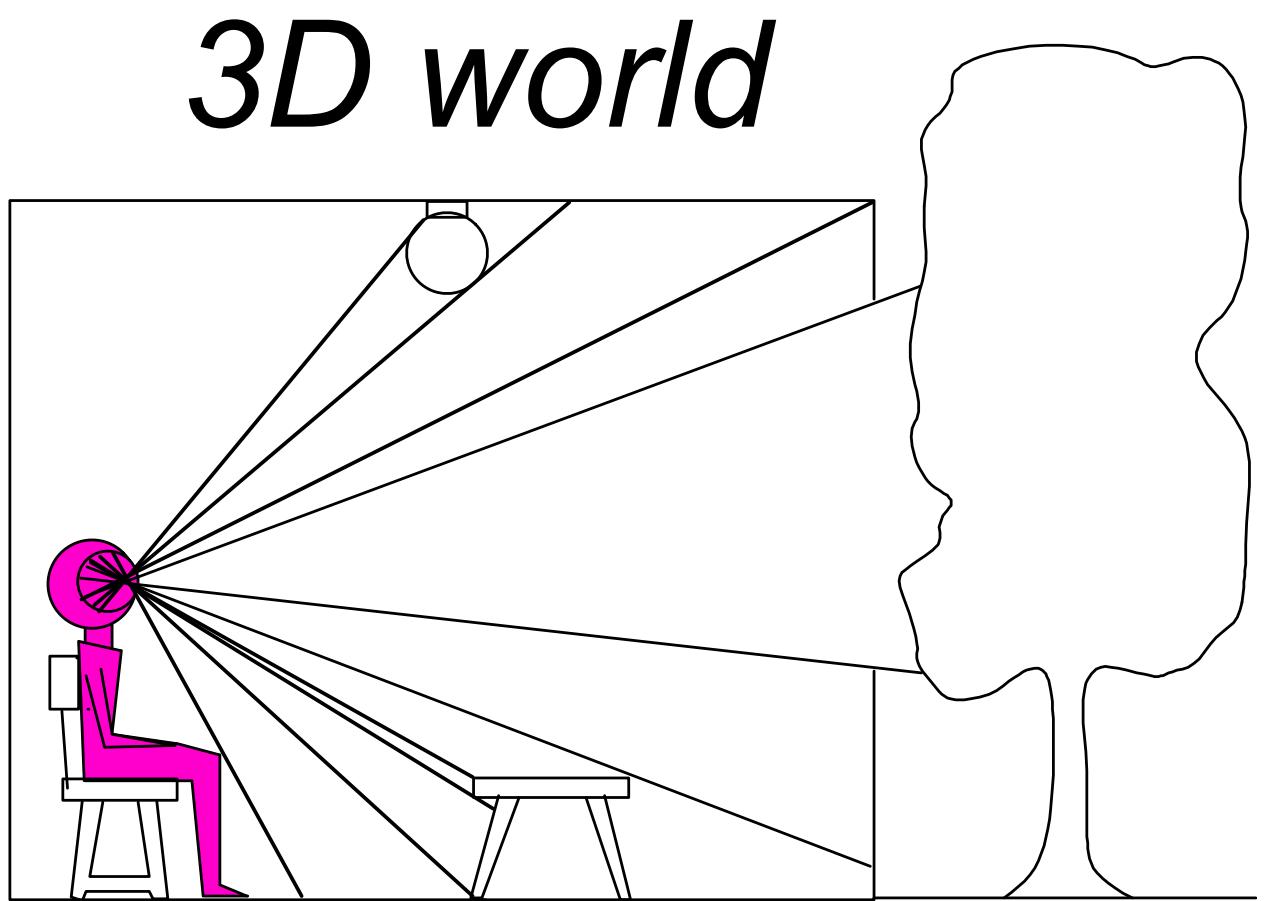


$$\mathbf{x} = (x, y) = (X, Y) \cdot \frac{f}{Z}$$

Image / Pixel  
Coordinates

World Coordinates

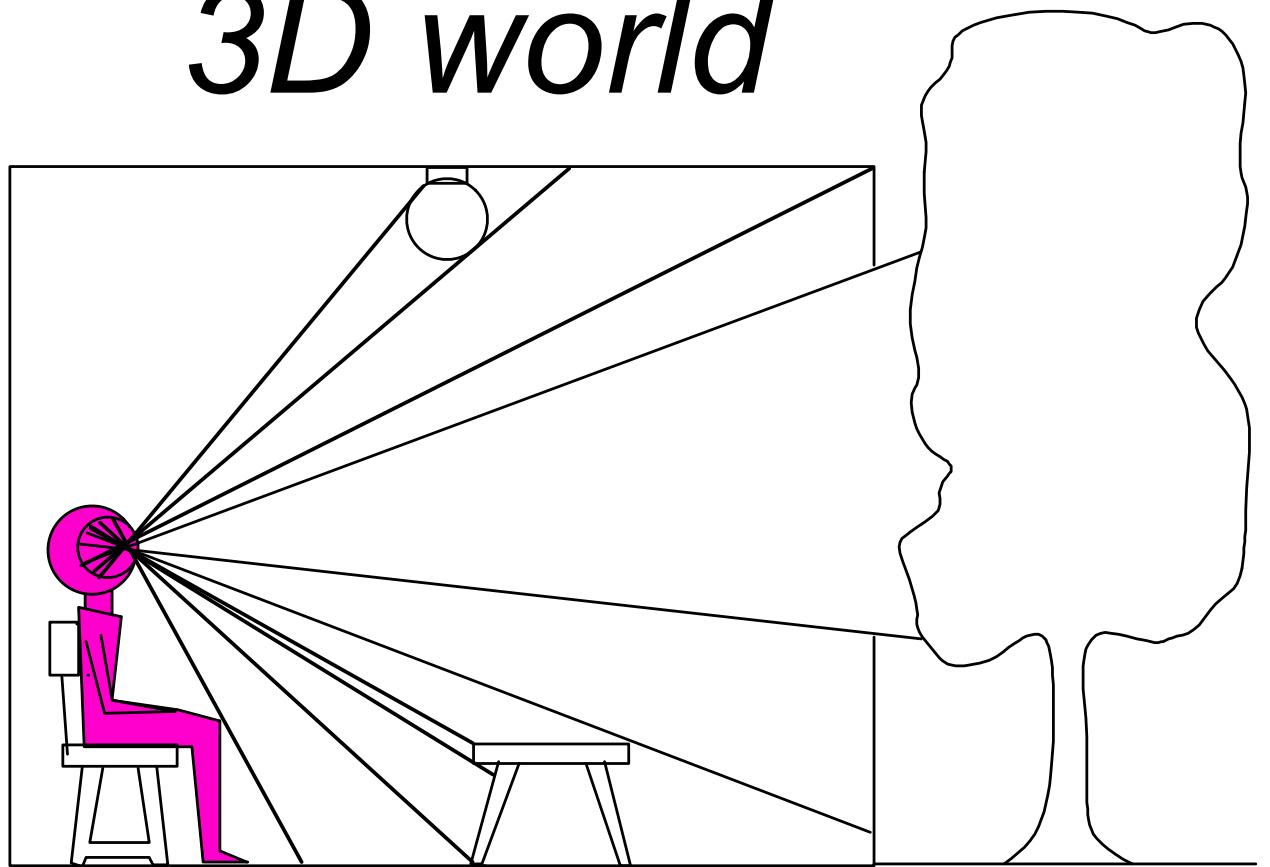
# Perspective Projection



Point of observation

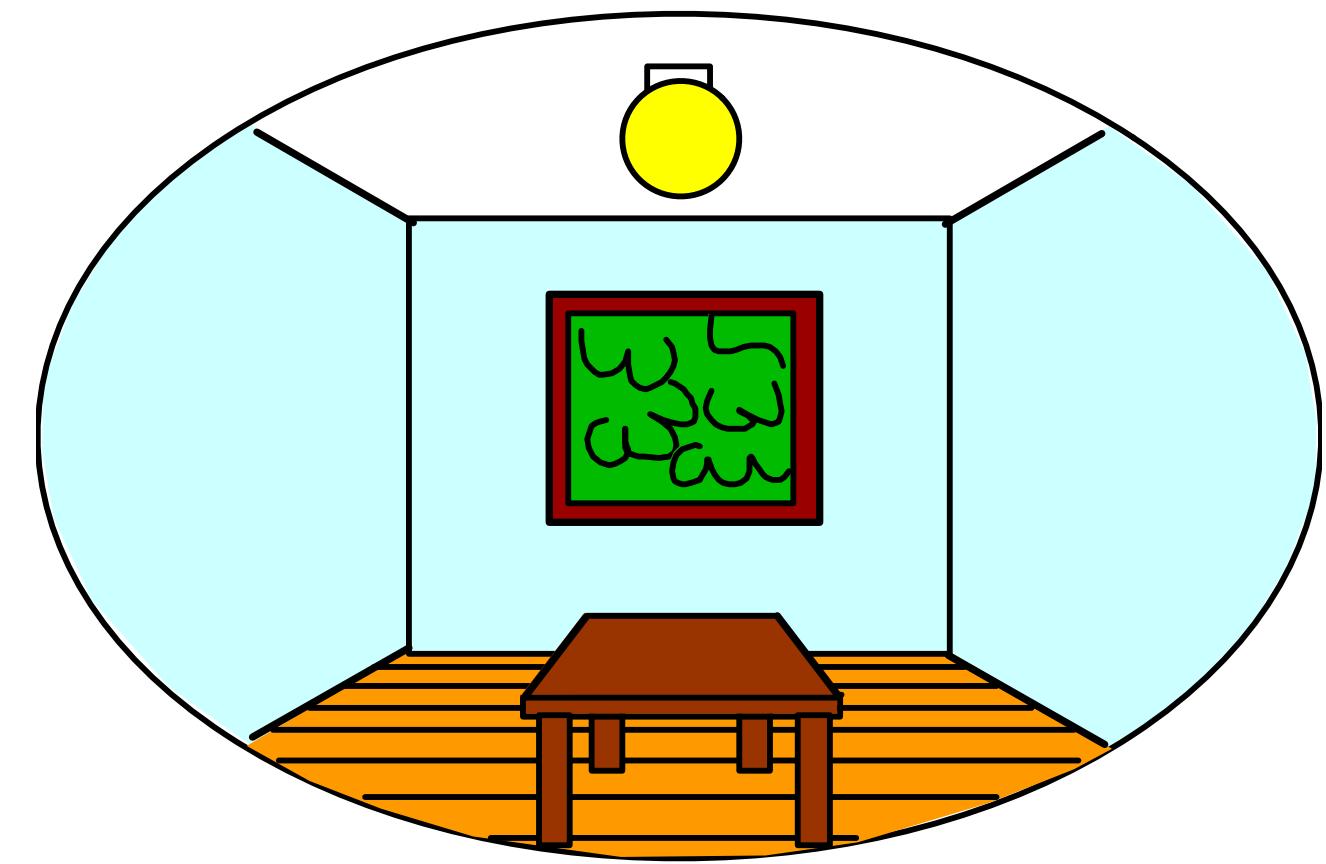
# Perspective Projection

*3D world*



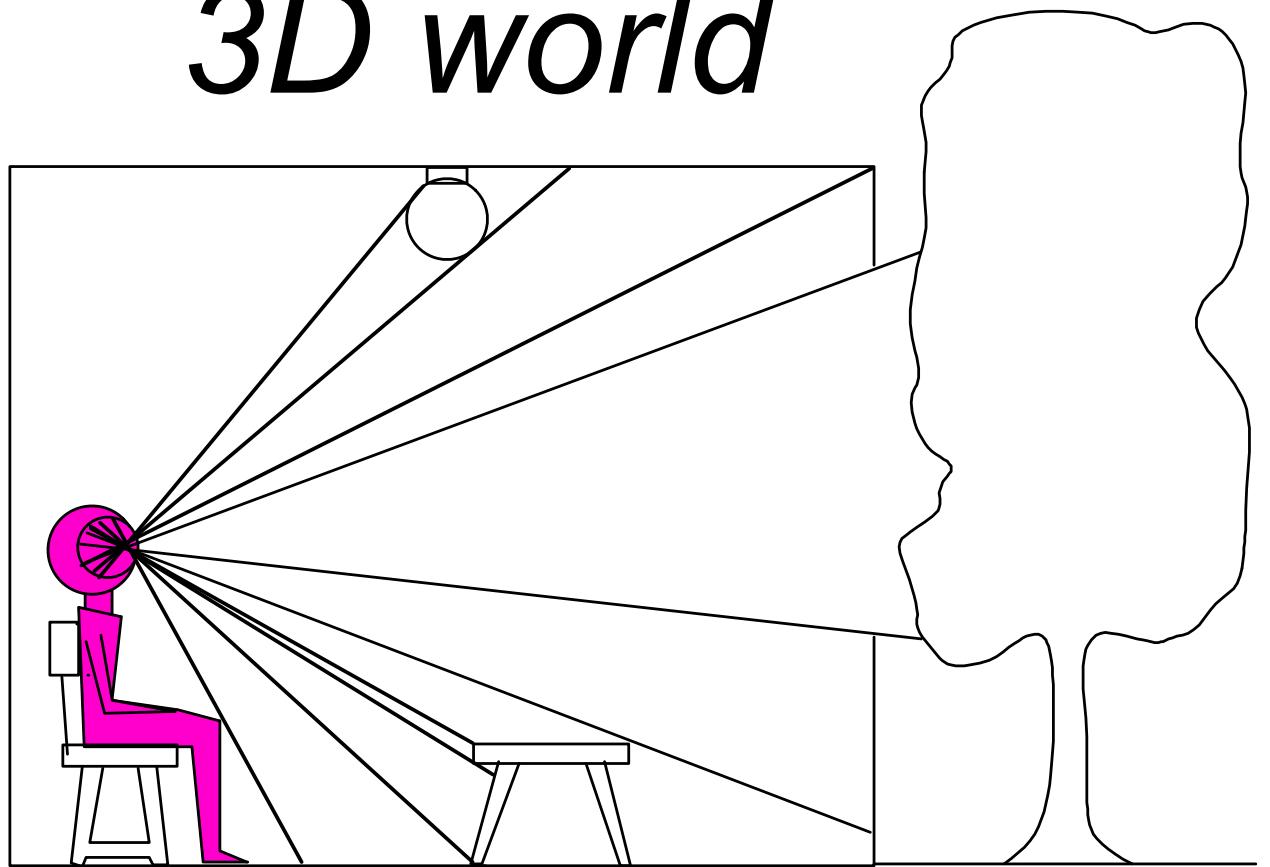
Point of observation

*2D image*



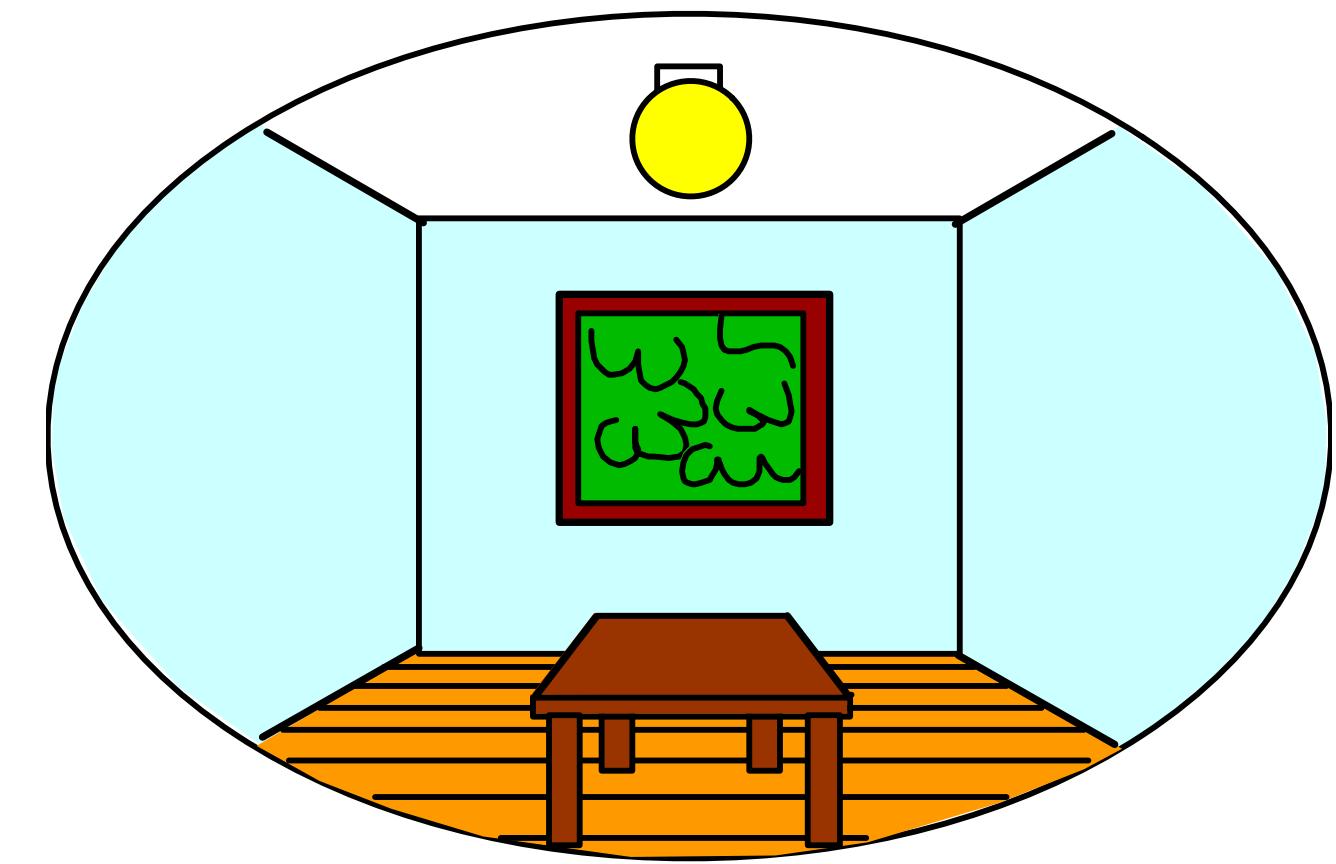
# Perspective Projection

*3D world*



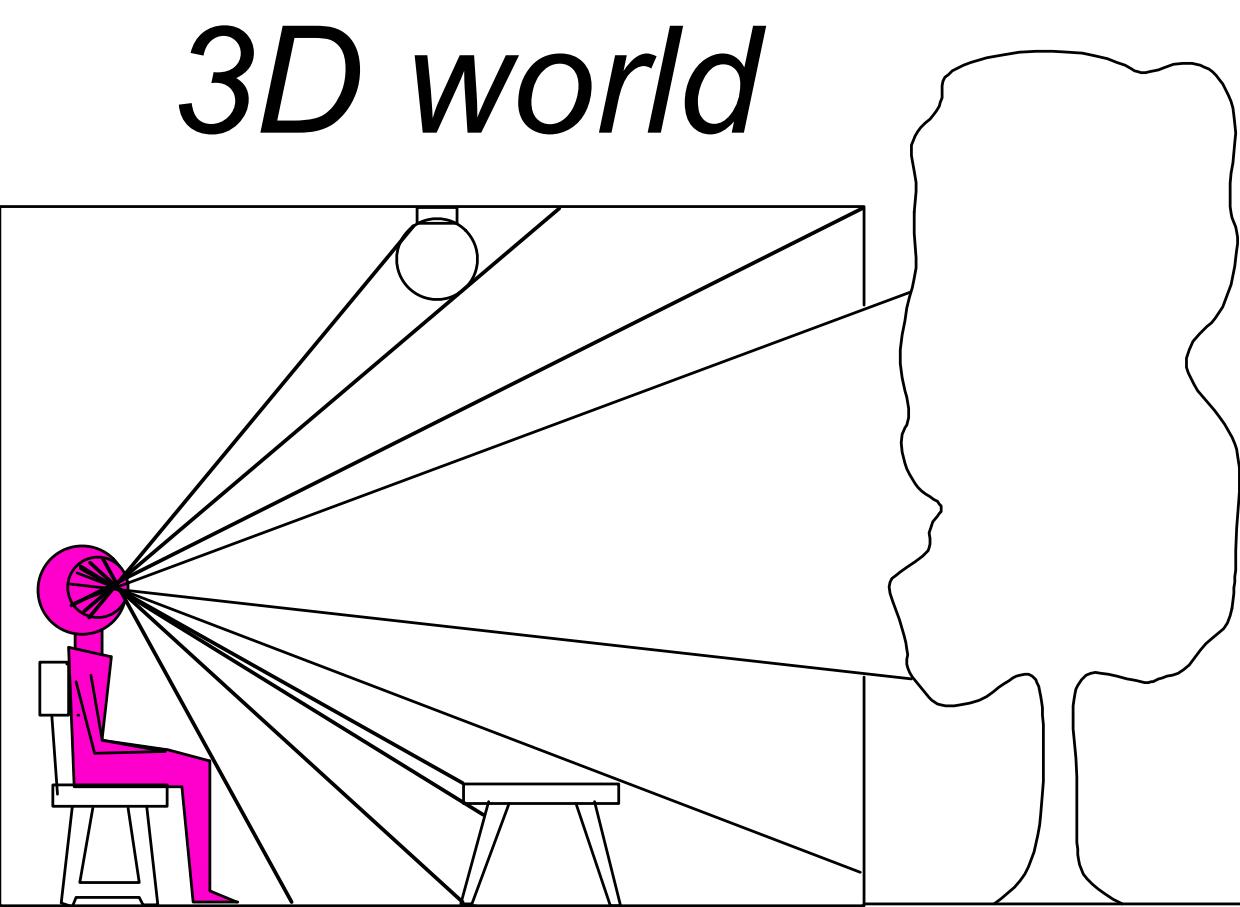
Point of observation

*2D image*

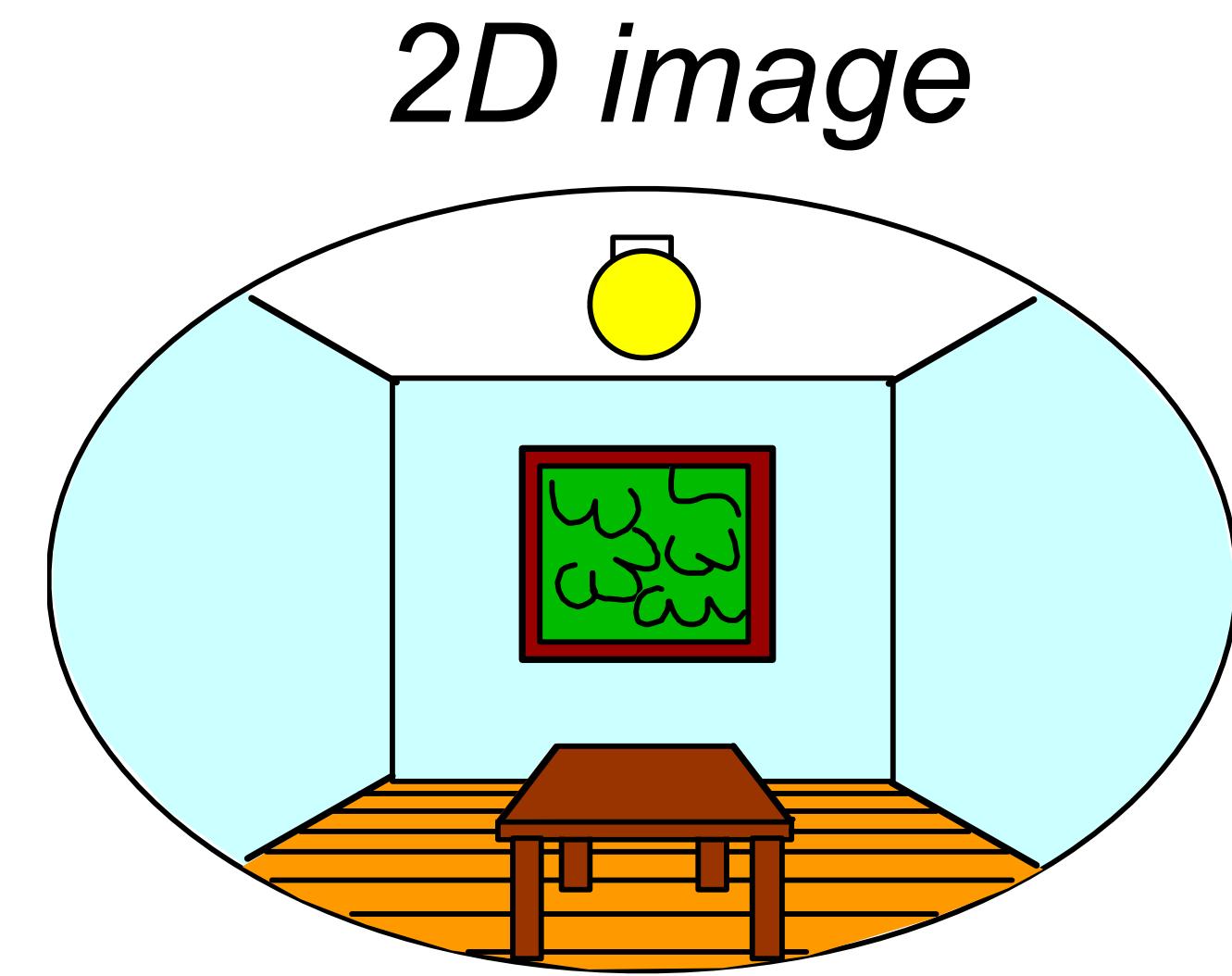


What properties of the world are preserved?

# Perspective Projection



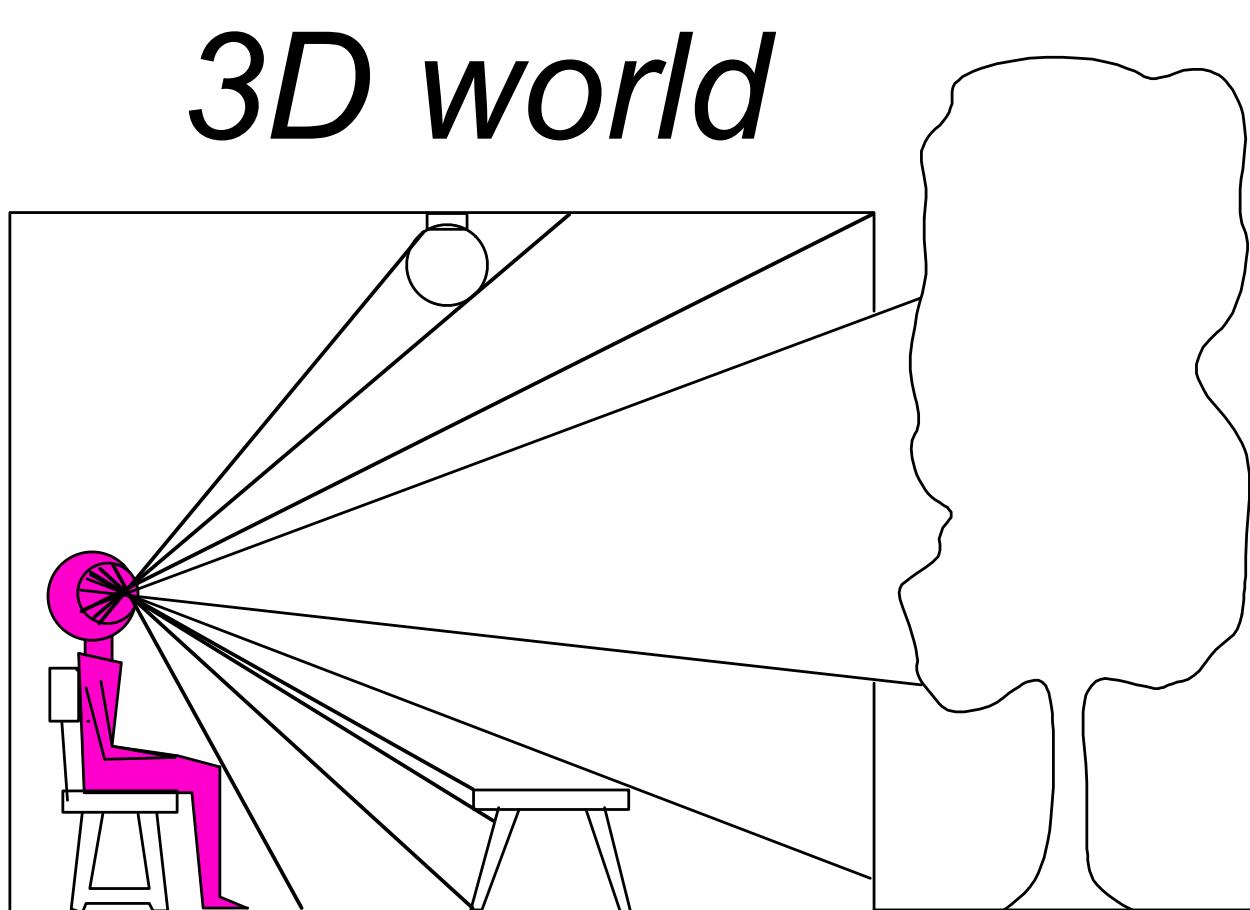
Point of observation



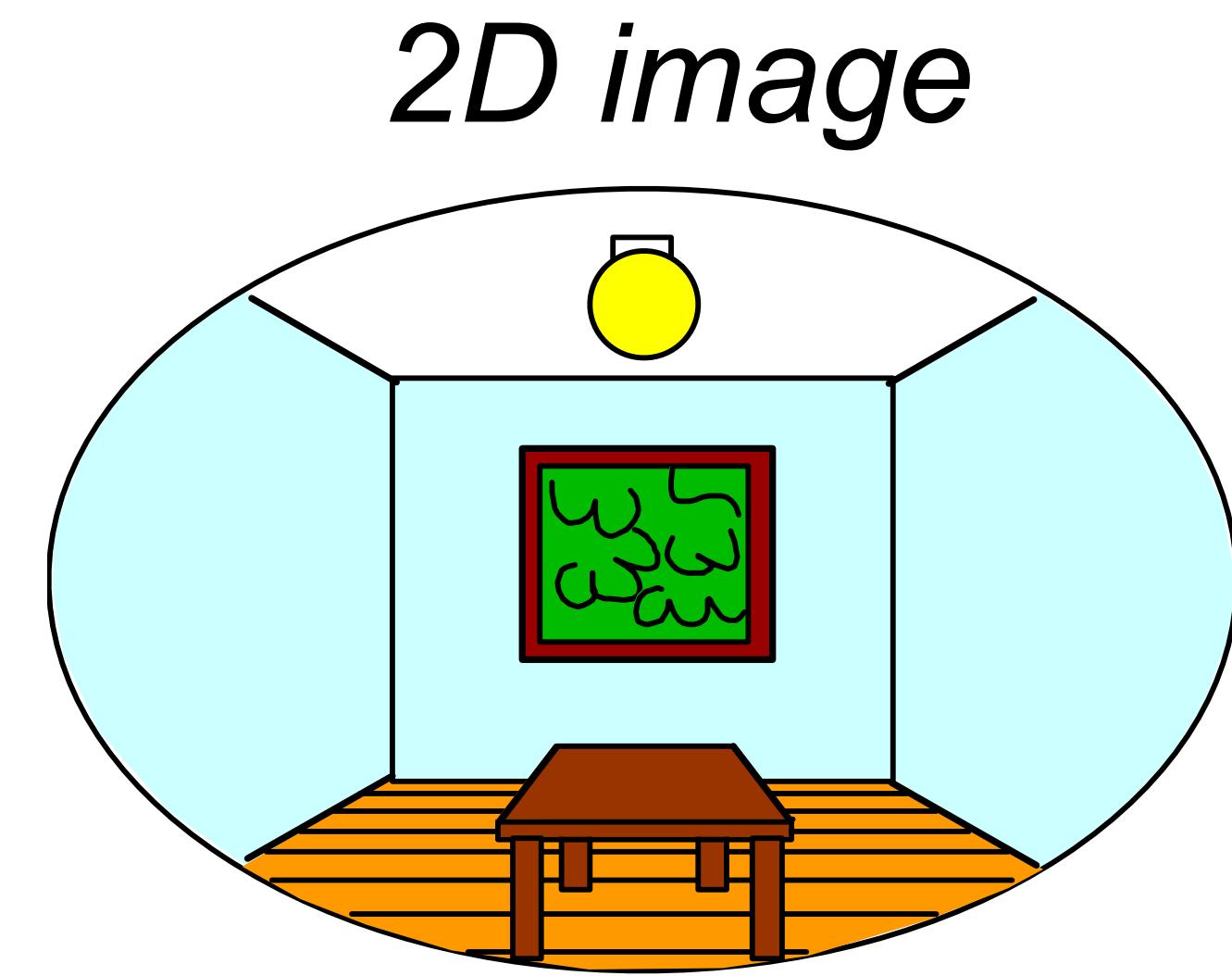
What properties of the world are preserved?

- Straight lines, incidence

# Perspective Projection



Point of observation

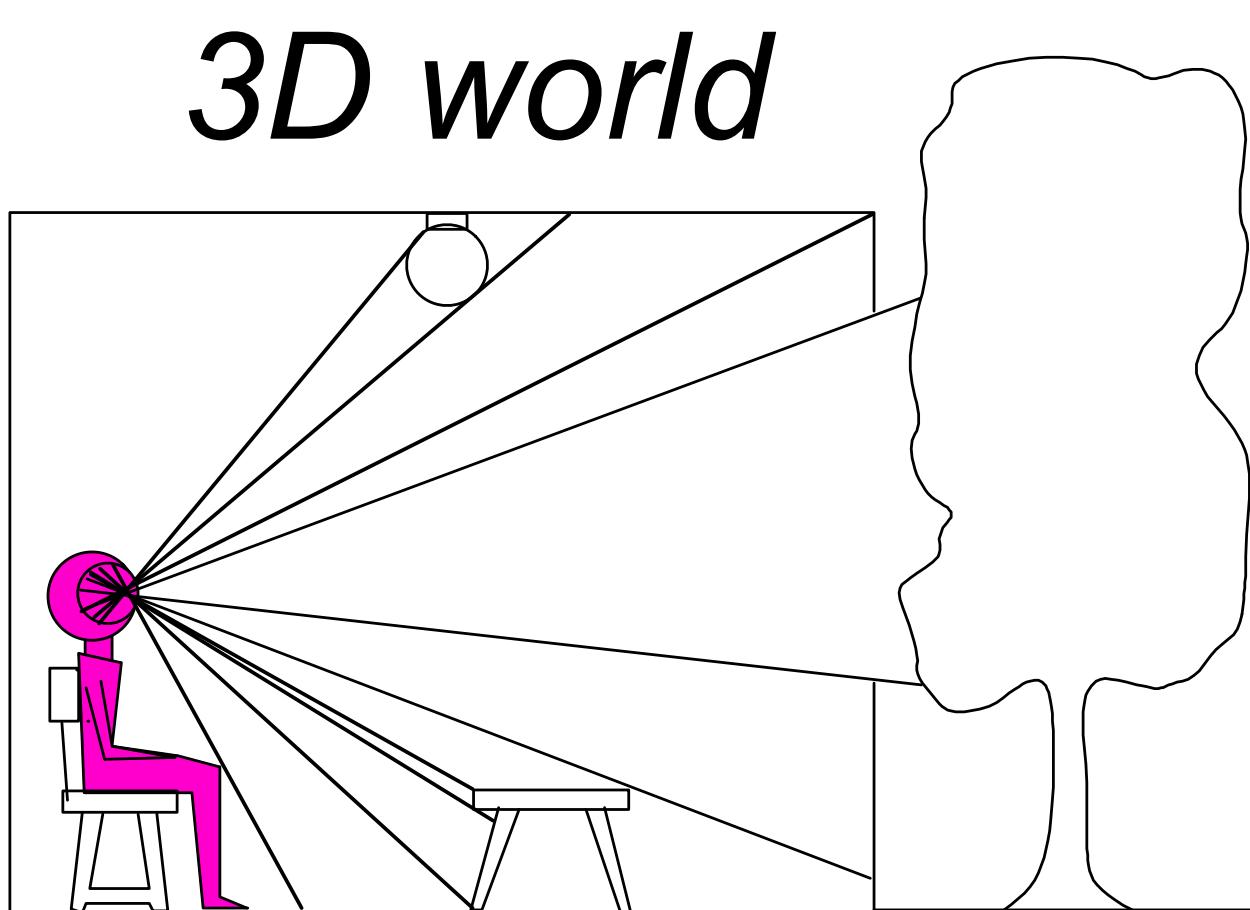


What properties of the world are preserved?

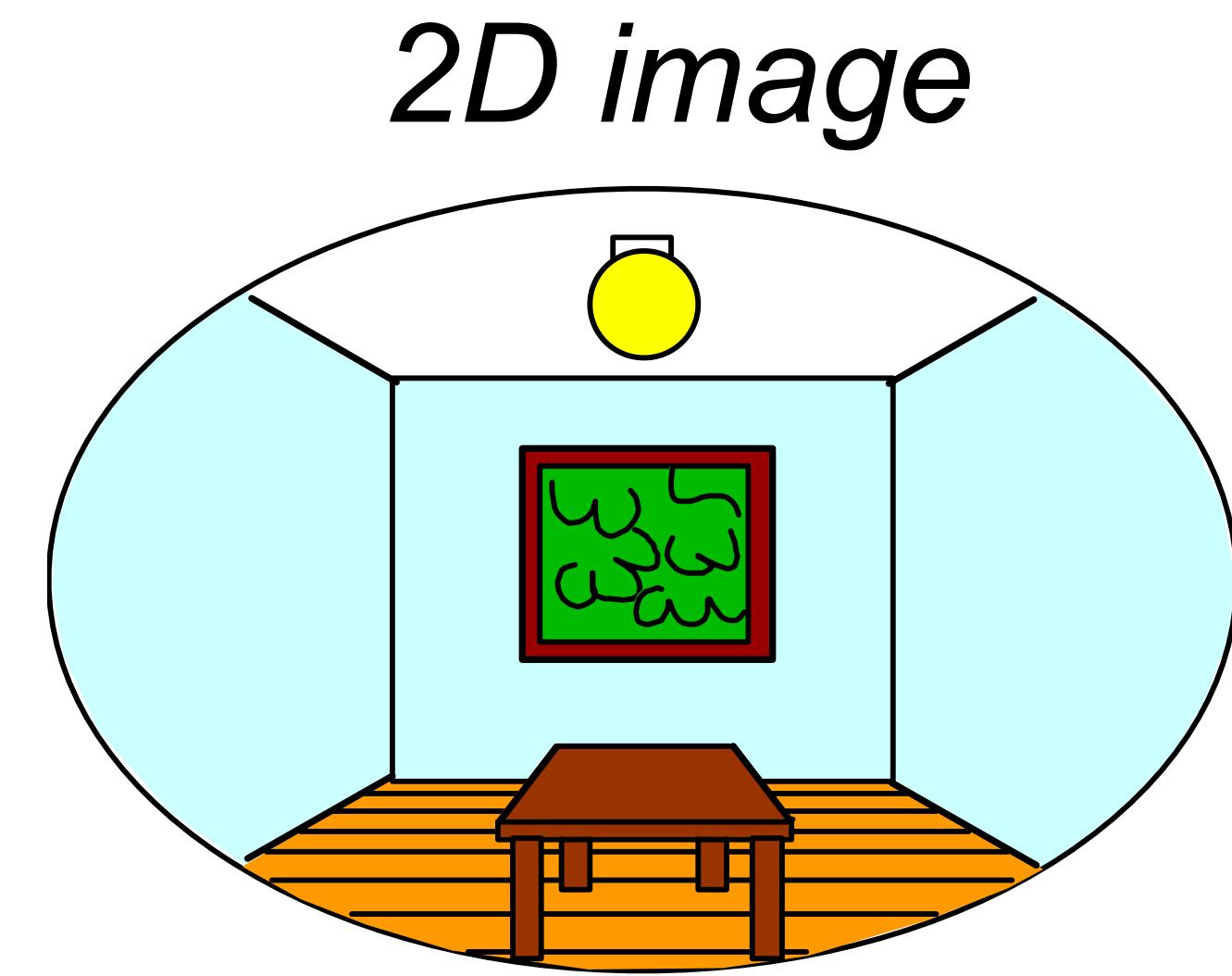
- Straight lines, incidence

What properties are not preserved?

# Perspective Projection



Point of observation



What properties of the world are preserved?

- Straight lines, incidence

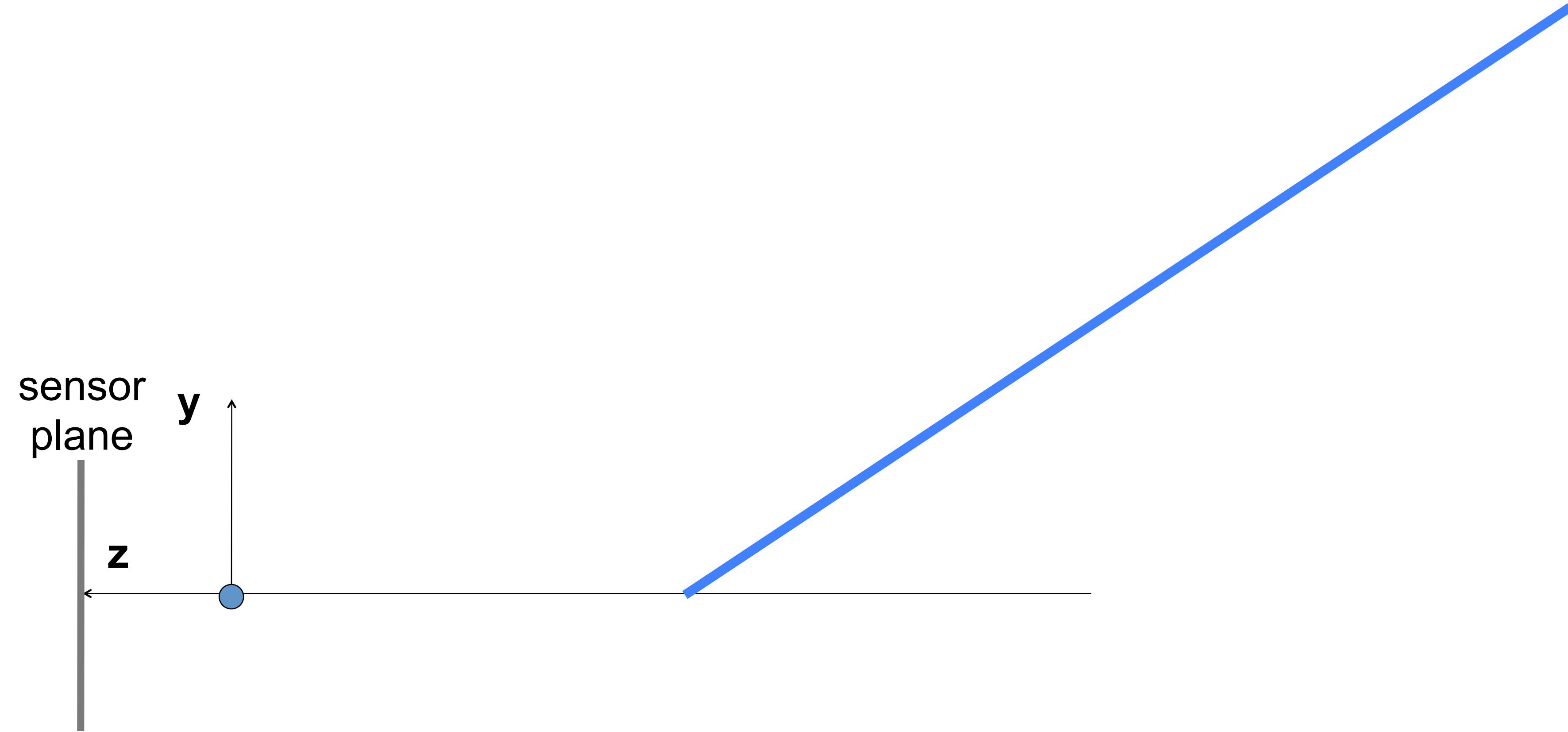
What properties are not preserved?

- Angles, lengths

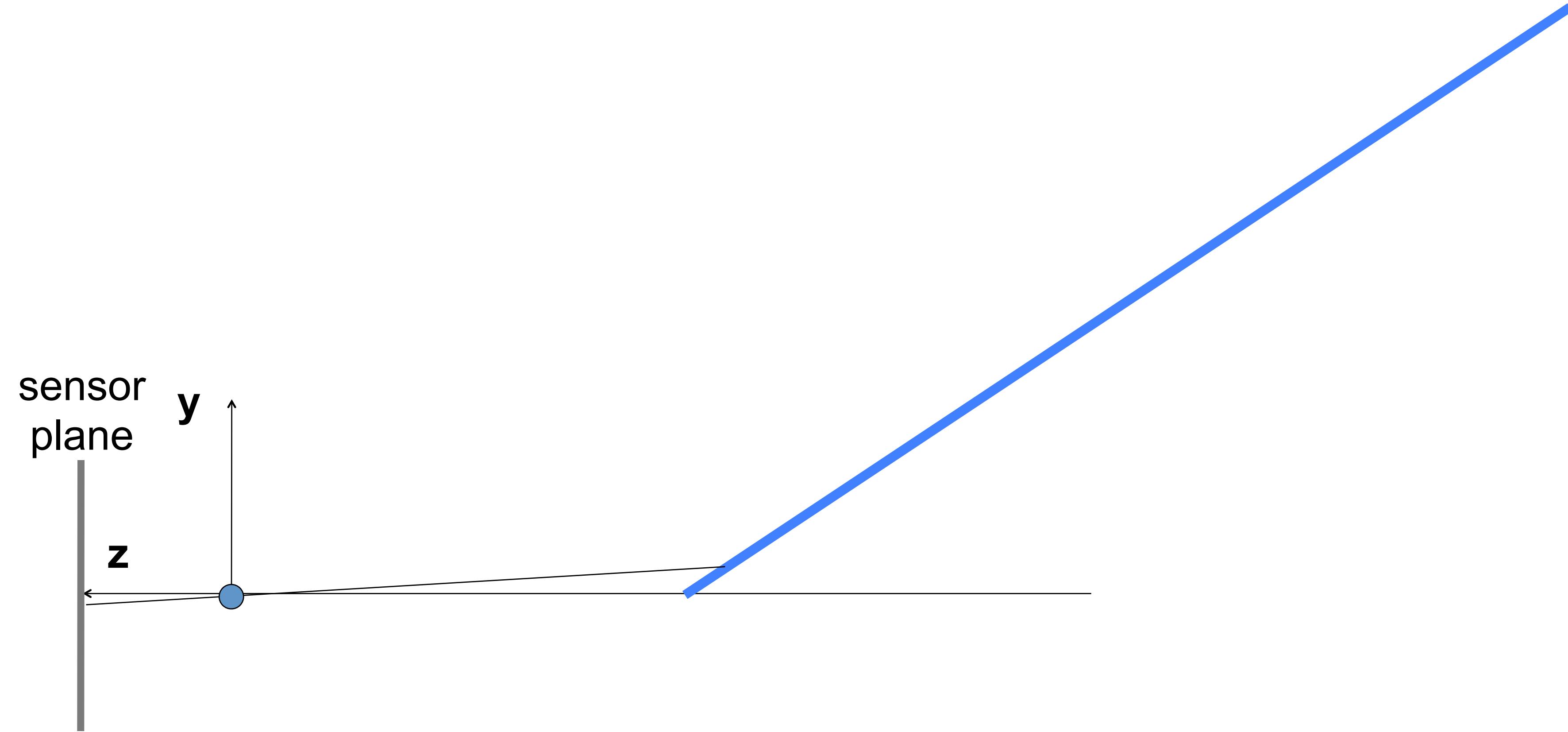
# Vanishing point



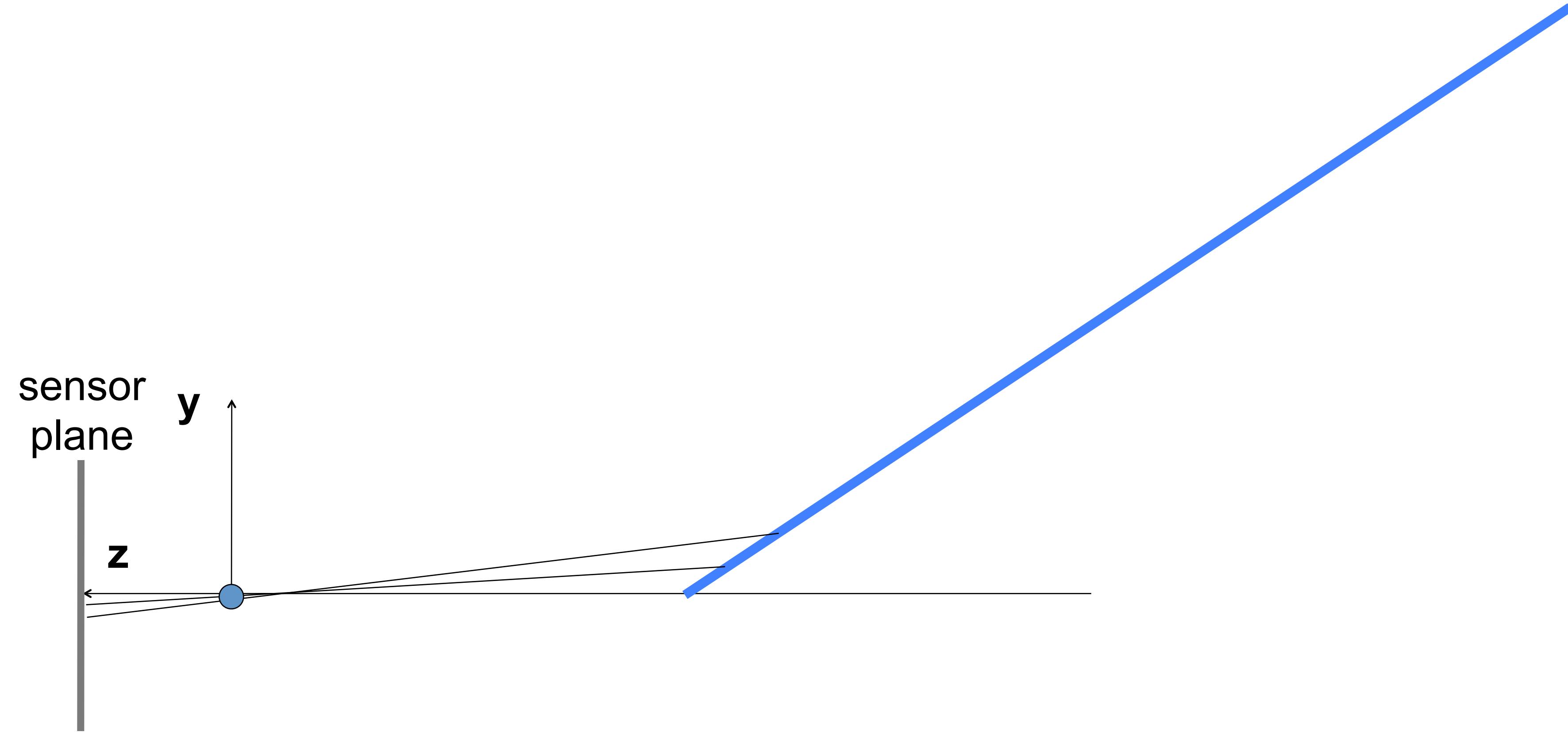
# Vanishing point



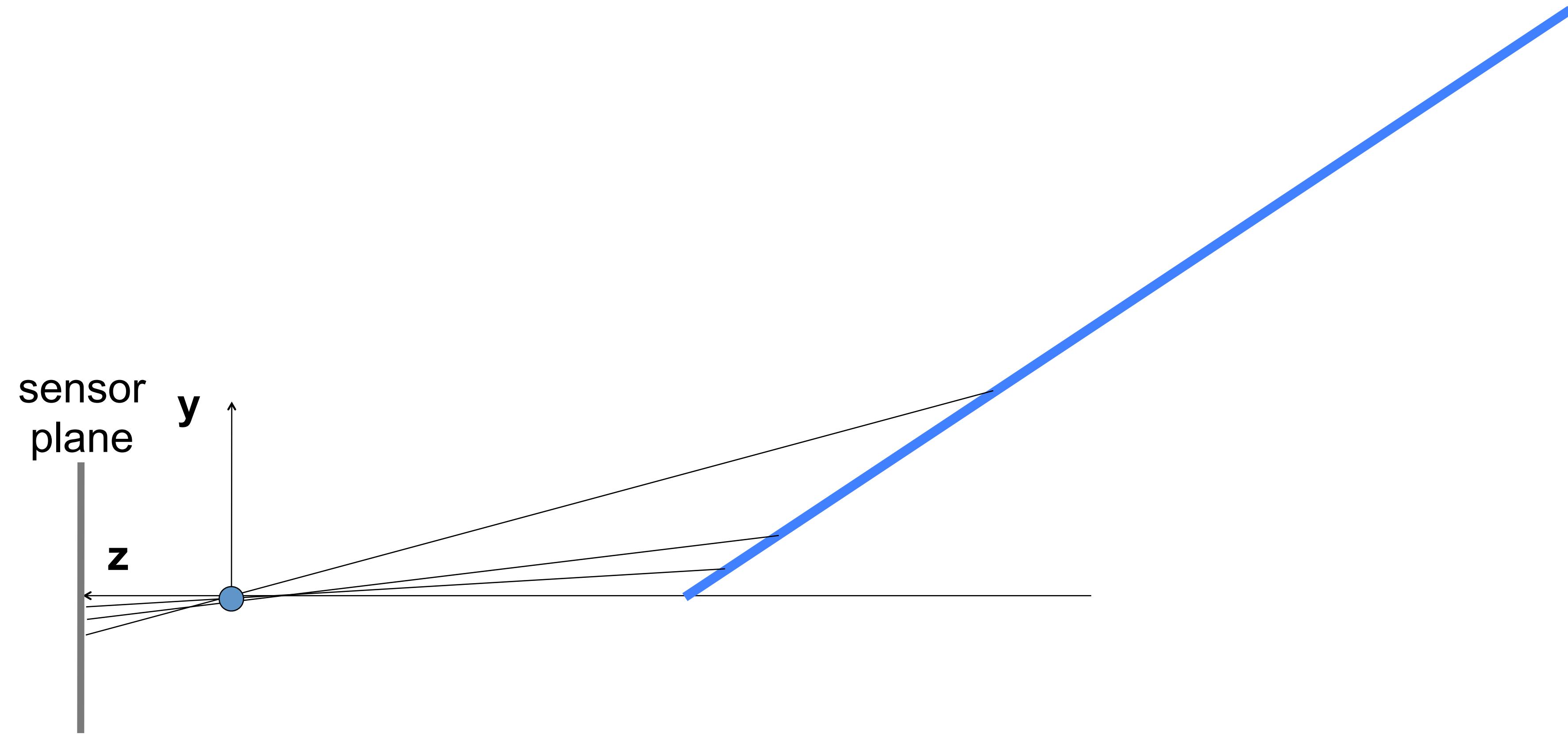
# Vanishing point



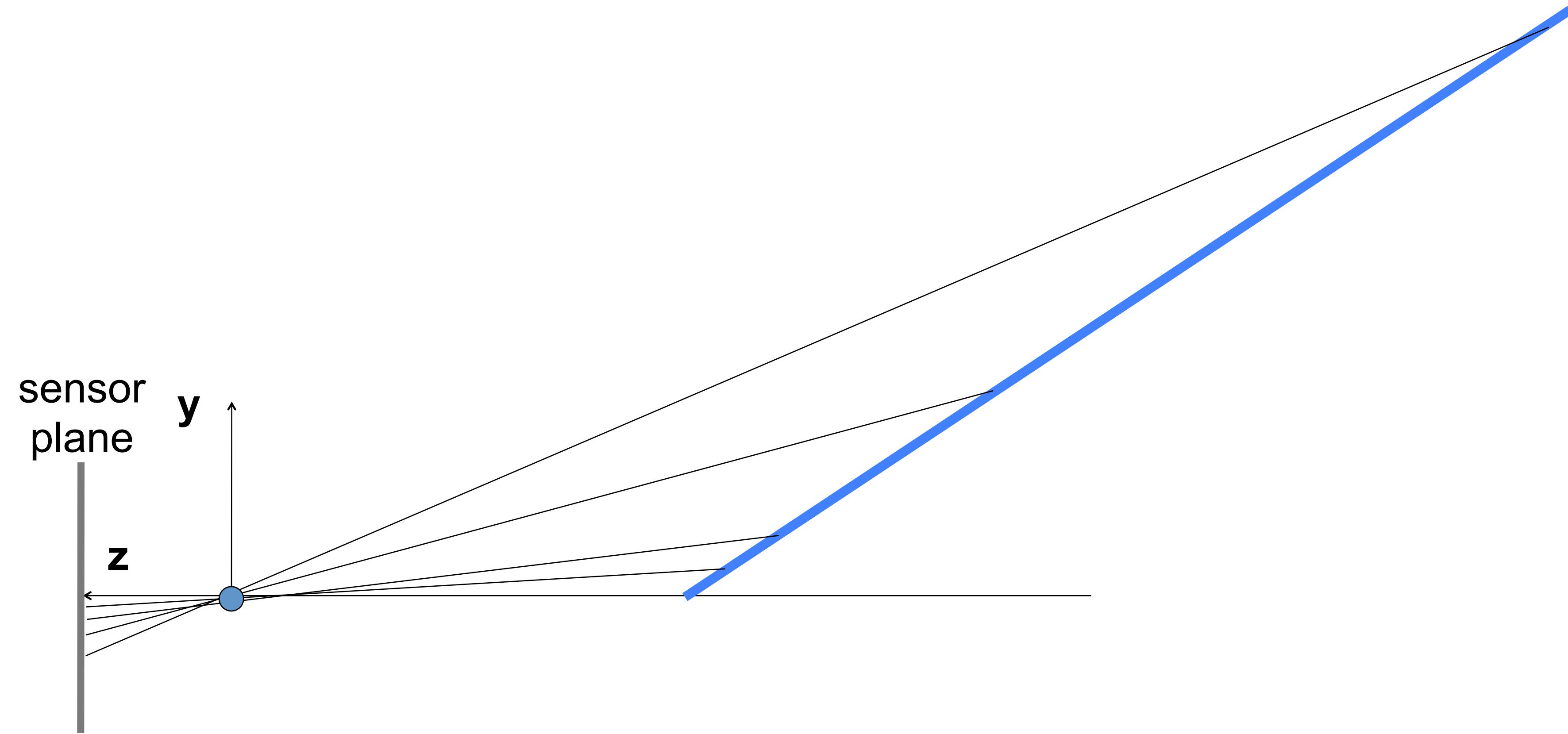
# Vanishing point



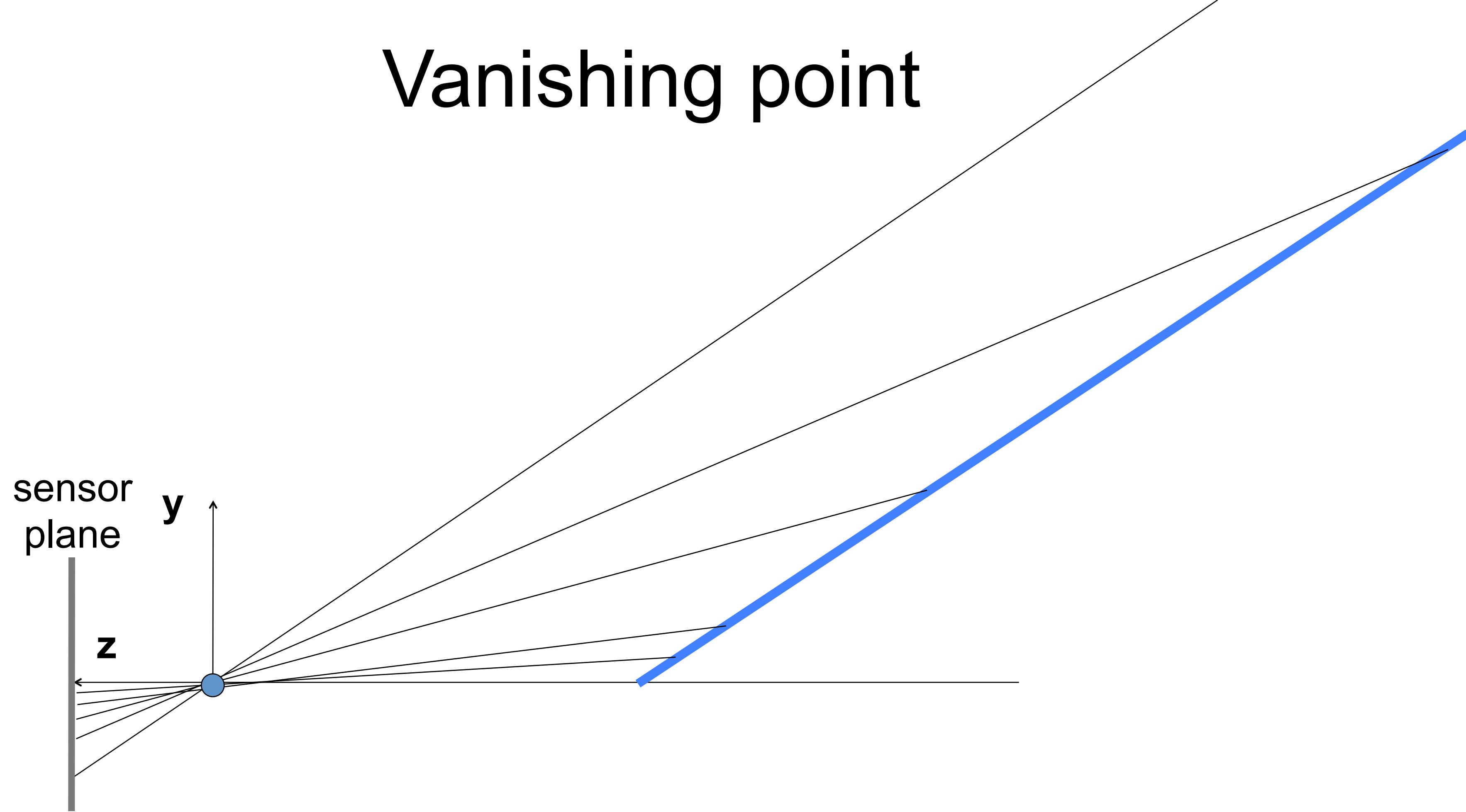
# Vanishing point



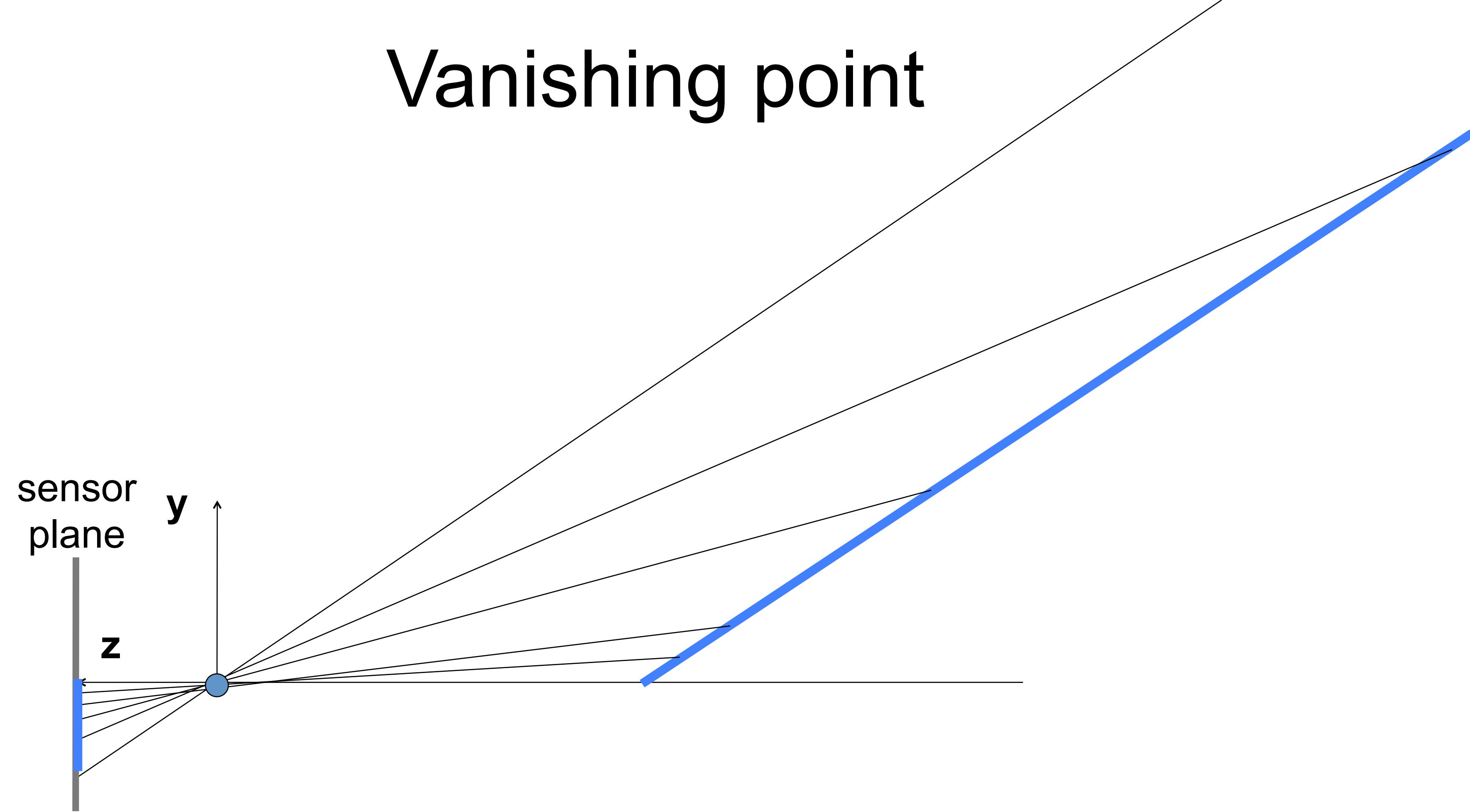
# Vanishing point



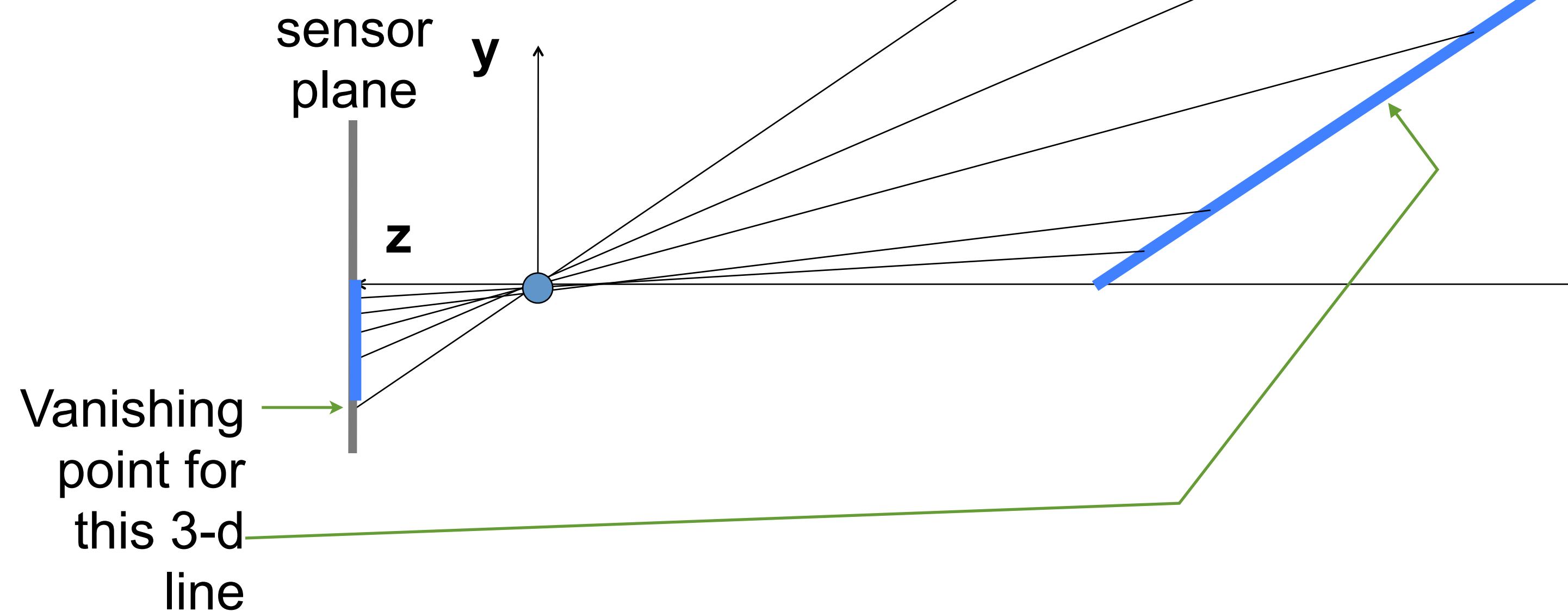
# Vanishing point



# Vanishing point



# Vanishing point



## Line in 3-space

$$X(t) = X_0 + at$$

$$Y(t) = Y_0 + bt$$

$$Z(t) = Z_0 + ct$$

$$x(t) = \frac{fX}{Z} = \frac{fX_0 + fat}{Z_0 + ct}$$

$$y(t) = \frac{fY}{Z} = \frac{fY_0 + fbt}{Z_0 + ct}$$

In the limit as  $t \rightarrow \pm\infty$   
we have (for  $c \neq 0$ ):



$$x(t \rightarrow \infty) \rightarrow \frac{fa}{c}$$

$$y(t \rightarrow \infty) \rightarrow \frac{fb}{c}$$

## Line in 3-space

$$X(t) = X_0 + at$$

$$Y(t) = Y_0 + bt$$

$$Z(t) = Z_0 + ct$$

In the limit as  $t \rightarrow \pm\infty$   
we have (for  $c \neq 0$ ):

## Perspective projection of that line

$$x(t) = \frac{fX}{Z} = \frac{fX_0 + fat}{Z_0 + ct}$$

$$y(t) = \frac{fY}{Z} = \frac{fY_0 + fbt}{Z_0 + ct}$$



$$x(t \rightarrow \infty) \rightarrow \frac{fa}{c}$$

$$y(t \rightarrow \infty) \rightarrow \frac{fb}{c}$$

## Line in 3-space

$$X(t) = X_0 + at$$

$$Y(t) = Y_0 + bt$$

$$Z(t) = Z_0 + ct$$

## Perspective projection of that line

$$x(t) = \frac{fX}{Z} = \frac{fX_0 + fat}{Z_0 + ct}$$

$$y(t) = \frac{fY}{Z} = \frac{fY_0 + fbt}{Z_0 + ct}$$

In the limit as  $t \rightarrow \pm\infty$   
we have (for  $c \neq 0$ ):



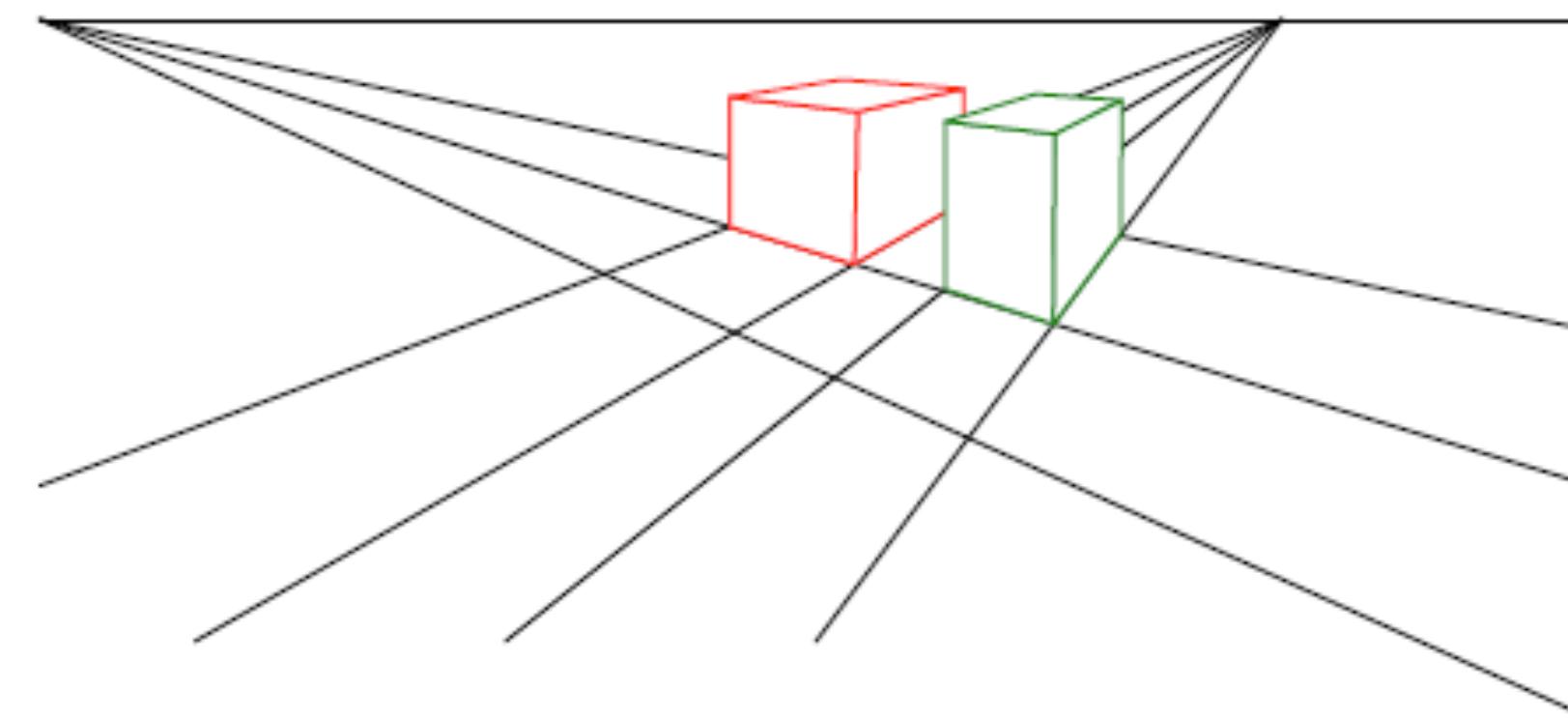
$$x(t \rightarrow \infty) \rightarrow \frac{fa}{c}$$

$$y(t \rightarrow \infty) \rightarrow \frac{fb}{c}$$

This tells us that any set of parallel lines (same  $a, b, c$  parameters) project to the same point (called the vanishing point).

# Vanishing points

- Each set of parallel lines (=direction) meets at a different point
  - The *vanishing point* for this direction
- Sets of parallel lines on the same plane lead to *colinear* vanishing points.
  - The line is called the *horizon* for that plane



# Perspective Projection



# Perspective Projection

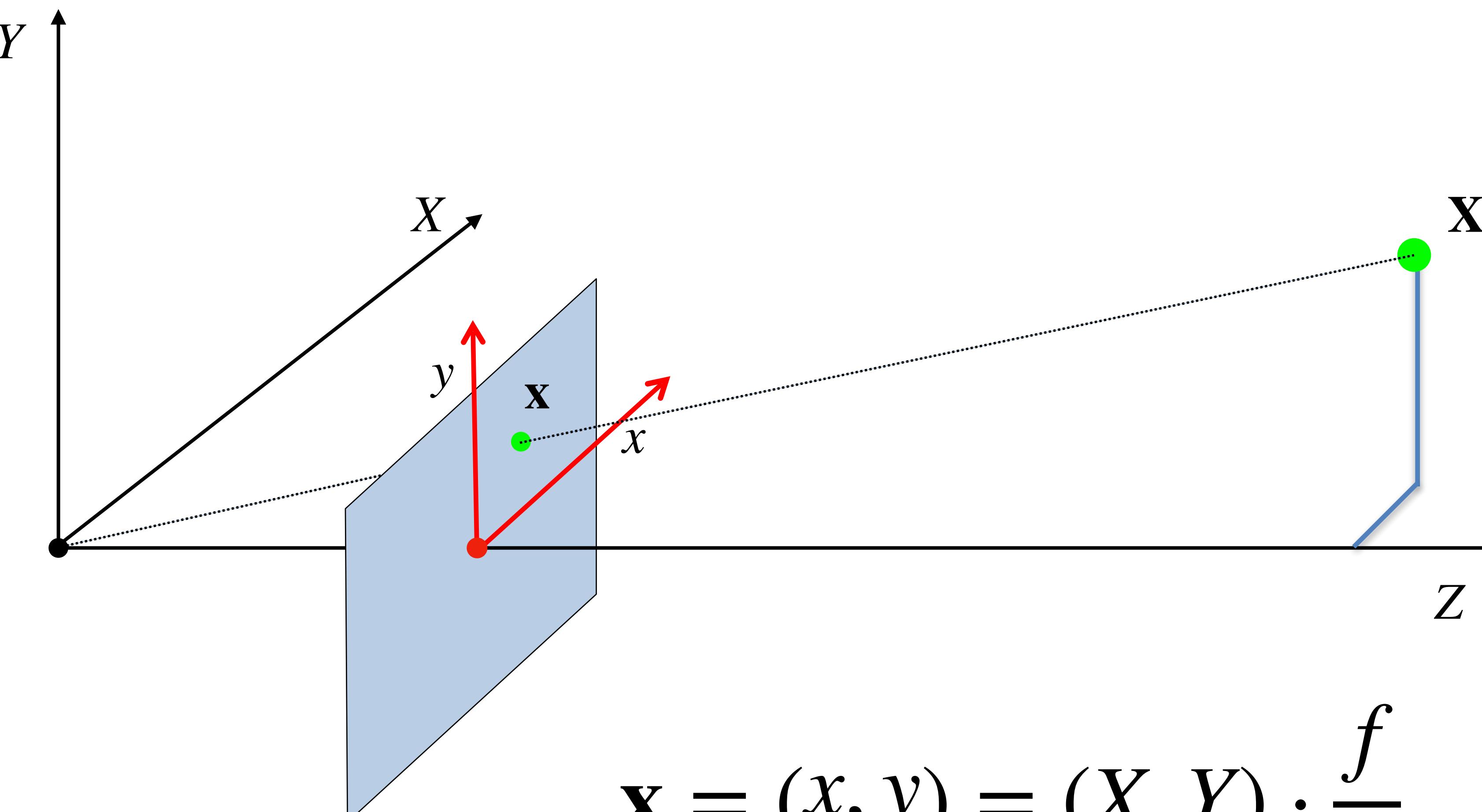


# Perspective Projection



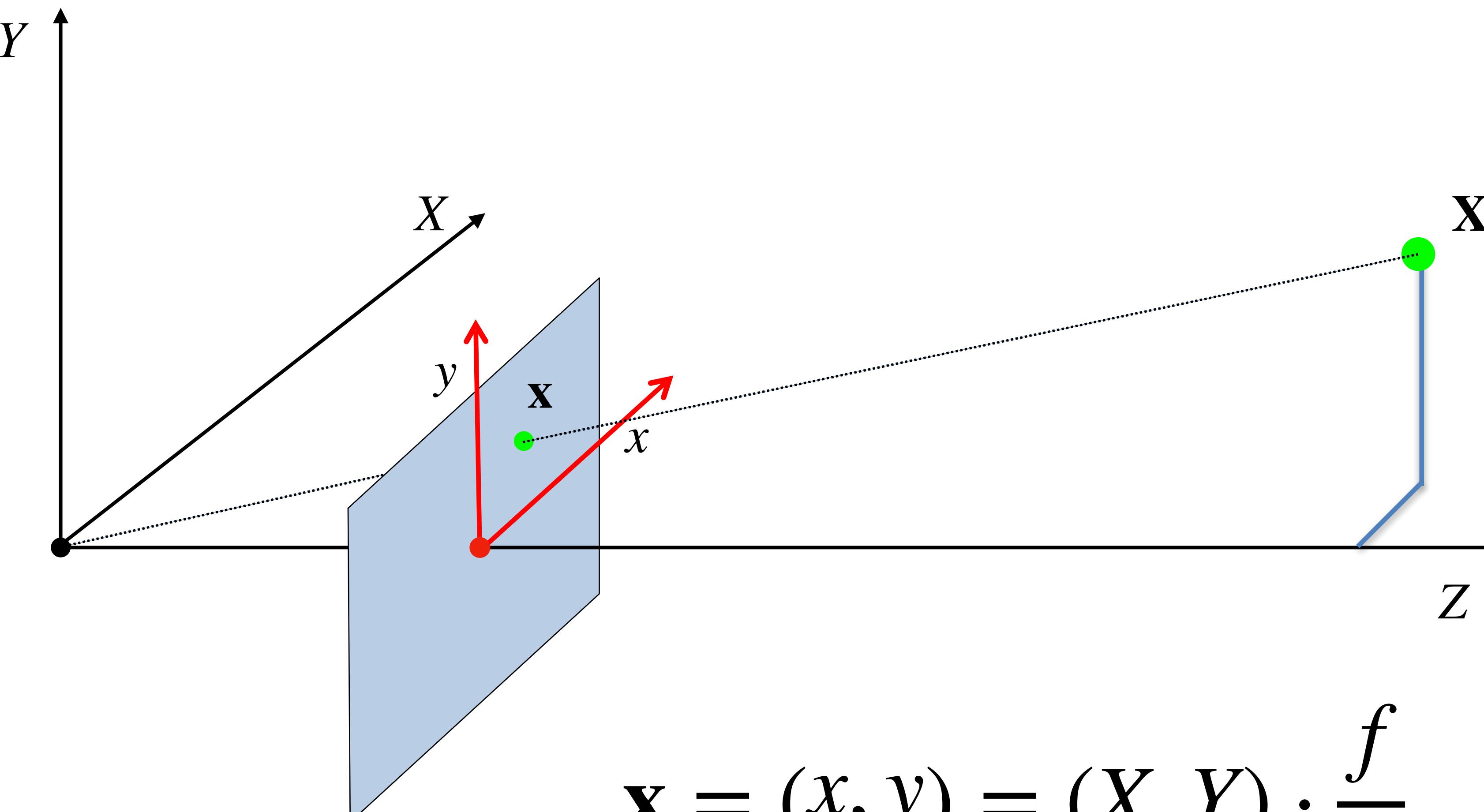
# Questions?

# Perspective projection



$$\mathbf{x} = (x, y) = (X, Y) \cdot \frac{f}{Z}$$

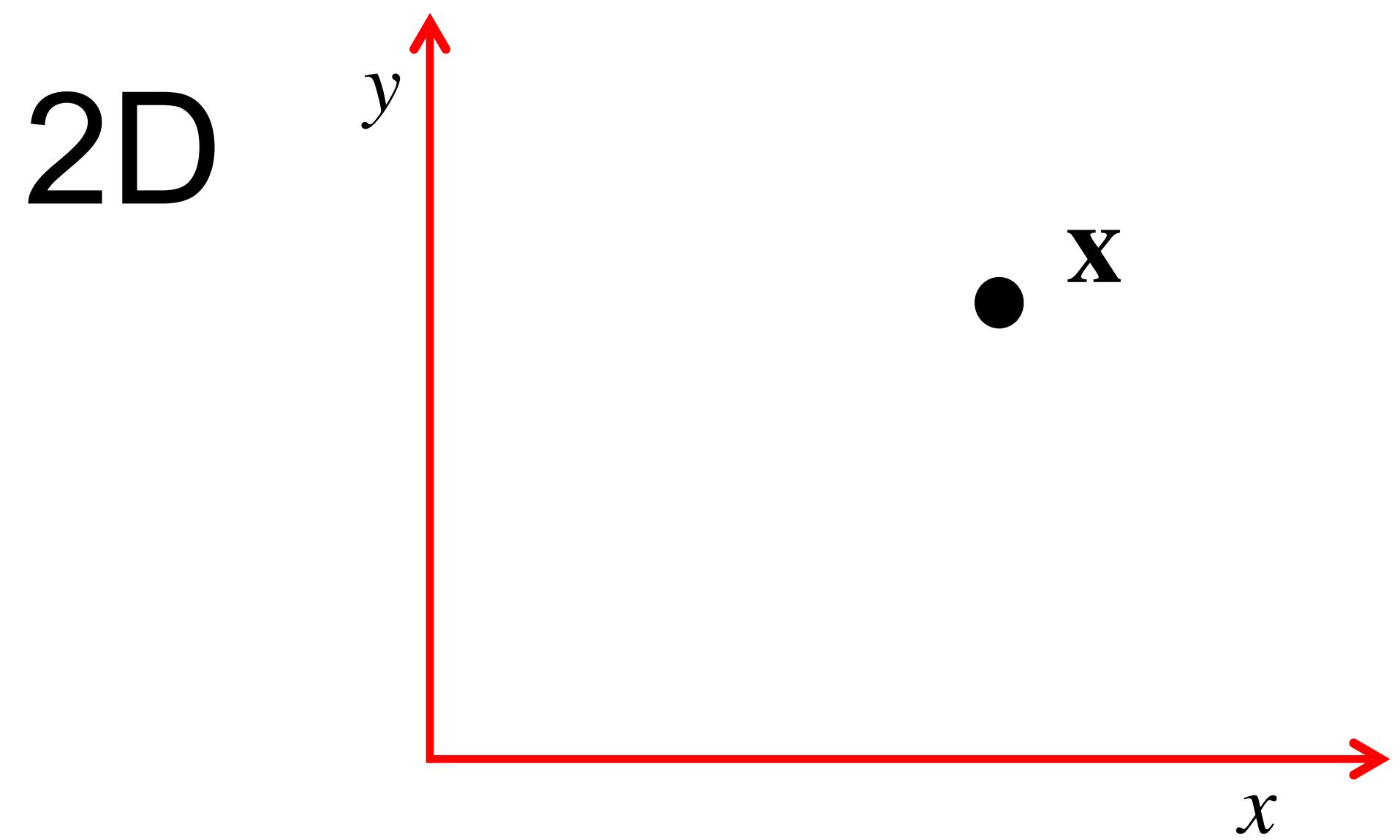
# Perspective projection



$$\mathbf{x} = (x, y) = (X, Y) \cdot \frac{f}{Z}$$

This is awkward... Not a linear operator.

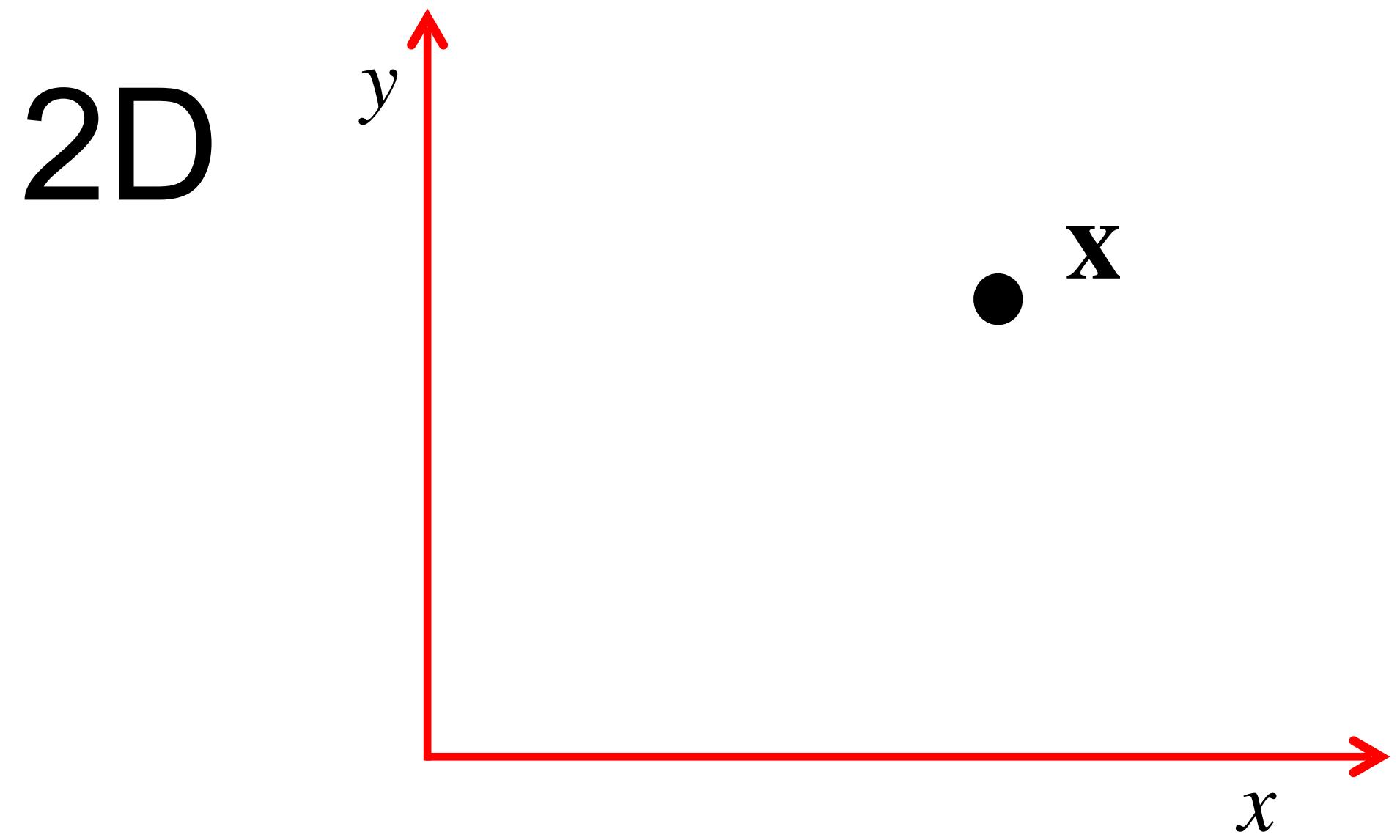
# Homogeneous coordinates



$$\mathbf{x} = (x, y) \rightarrow \tilde{\mathbf{x}} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

heterogeneous coordinates      homogeneous coordinates

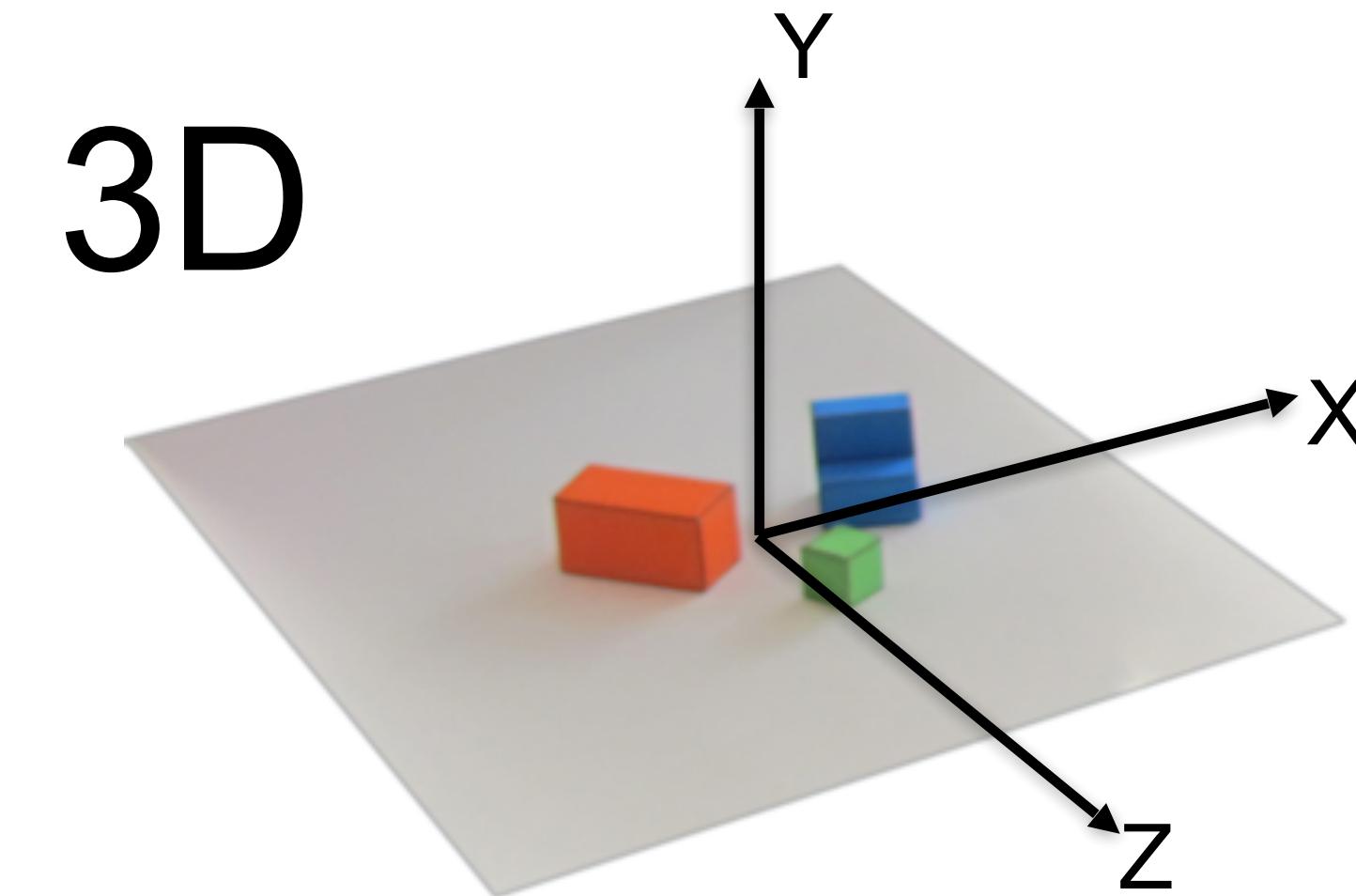
# Homogeneous coordinates



$$\mathbf{x} = (x, y) \rightarrow \tilde{\mathbf{x}} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

heterogeneous  
coordinates

homogeneous  
coordinates



$$\mathbf{X} = (X, Y, Z) \rightarrow \tilde{\mathbf{X}} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

From heterogeneous to homogeneous:

$$\mathbf{x} = (x, y) \quad \rightarrow \quad \tilde{\mathbf{x}} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} w \cdot x \\ w \cdot y \\ w \cdot 1 \end{bmatrix}$$

From heterogeneous to homogeneous:

$$\mathbf{x} = (x, y) \quad \rightarrow \quad \tilde{\mathbf{x}} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} w \cdot x \\ w \cdot y \\ w \cdot 1 \end{bmatrix}$$

From homogeneous to heterogeneous:

$$\tilde{\mathbf{x}} = \begin{bmatrix} x \\ y \\ w \end{bmatrix} \quad \rightarrow \quad \mathbf{x} = (x/w, y/w)$$

# Homogeneous coordinates

## 2D

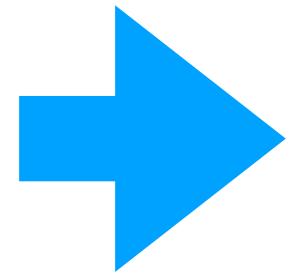
$$\begin{array}{ccc} \mathbb{R}^2 & & \mathbb{P}^2 \\ (x, y) & \xrightarrow{\hspace{1cm}} & \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \end{array}$$

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \xrightarrow{\hspace{1cm}} (x/w, y/w)$$

# Homogeneous coordinates

2D

$$\mathbb{R}^2$$
$$(x, y)$$



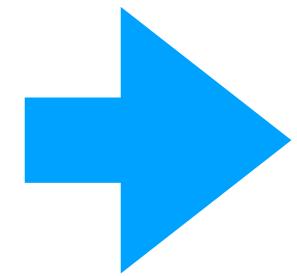
$$\mathbb{P}^2$$
$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

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

3D

$$\mathbb{R}^3$$

$$(X, Y, Z)$$



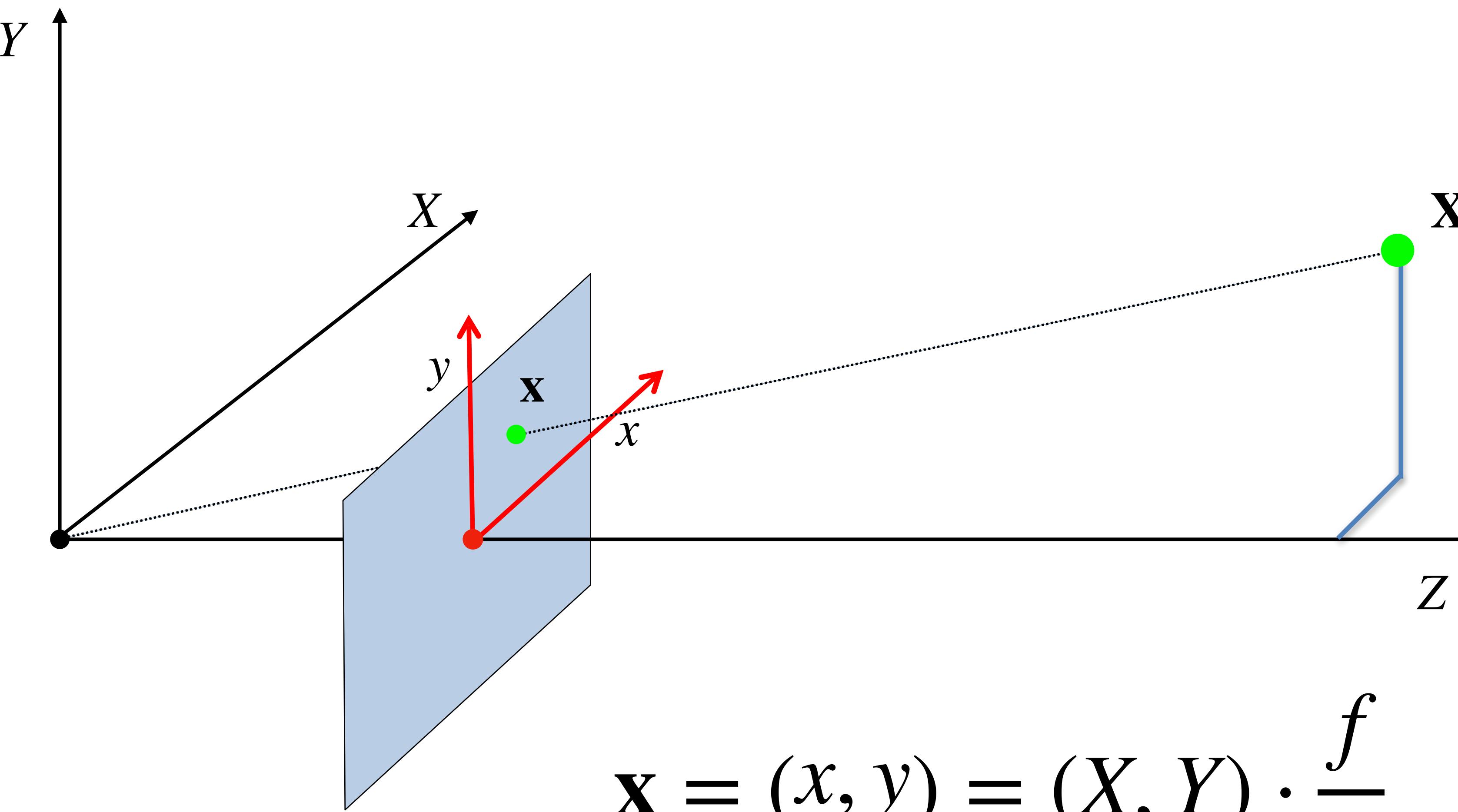
$$\mathbb{P}^3$$

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} X \\ Y \\ Z \\ W \end{bmatrix} \rightarrow (X/W, Y/W, Z/W)$$

# Questions?

# Perspective projection



$$\mathbf{x} = (x, y) = (X, Y) \cdot \frac{f}{Z}$$

This is awkward... Not a linear operator.

# Perspective projection

**Heterogeneous coordinates**

$$\mathbf{x} = (x, y) = (X, Y) \cdot \frac{1}{Z}$$

Image / Pixel  
Coordinates

World Coordinates

**Homogeneous coordinates**

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

Image / Pixel  
Coordinates      World Coordinates

# Perspective projection

**Heterogeneous coordinates**

$$\mathbf{x} = (x, y) = (X, Y) \cdot \frac{1}{Z}$$

World Coordinates

Image / Pixel  
Coordinates

**Homogeneous coordinates**

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

Image / Pixel  
Coordinates

World  
coords.

# Perspective projection

**Heterogeneous coordinates**

$$\mathbf{x} = (x, y) = (X, Y) \cdot \frac{1}{Z}$$

World Coordinates

Image / Pixel  
Coordinates

**Homogeneous coordinates**

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

Image / Pixel  
Coordinates

Projection Matrix

World  
coords.

# Perspective projection

**Heterogeneous coordinates**

$$\mathbf{x} = (x, y) = (X, Y) \cdot \frac{1}{Z}$$

Image / Pixel  
Coordinates

World Coordinates

**Homogeneous coordinates**

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

Image / Pixel  
Coordinates

Projection Matrix

World  
coords.

# Perspective projection

**Homogeneous coordinates**

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

Image / Pixel  
Coordinates      Projection Matrix      World  
    coords.

# Perspective projection

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

Image / Pixel  
Coordinates      Projection Matrix      World  
coords.

$$\tilde{\mathbf{x}} = \begin{bmatrix} X \\ Y \\ Z/f \end{bmatrix}$$

# Perspective projection

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

Image / Pixel  
Coordinates      Projection Matrix      World  
coords.

$$\tilde{\mathbf{x}} = \begin{bmatrix} X \\ Y \\ Z/f \end{bmatrix} \rightarrow \mathbf{x} = \left( f \frac{X}{Z}, f \frac{Y}{Z} \right)$$

# Perspective projection

$$\tilde{\mathbf{x}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f & 0 \end{bmatrix} \tilde{\mathbf{x}} = \begin{bmatrix} X \\ Y \\ Z/f \end{bmatrix} \rightarrow \mathbf{x} = \left( f \frac{X}{Z}, f \frac{Y}{Z} \right)$$

# Perspective projection

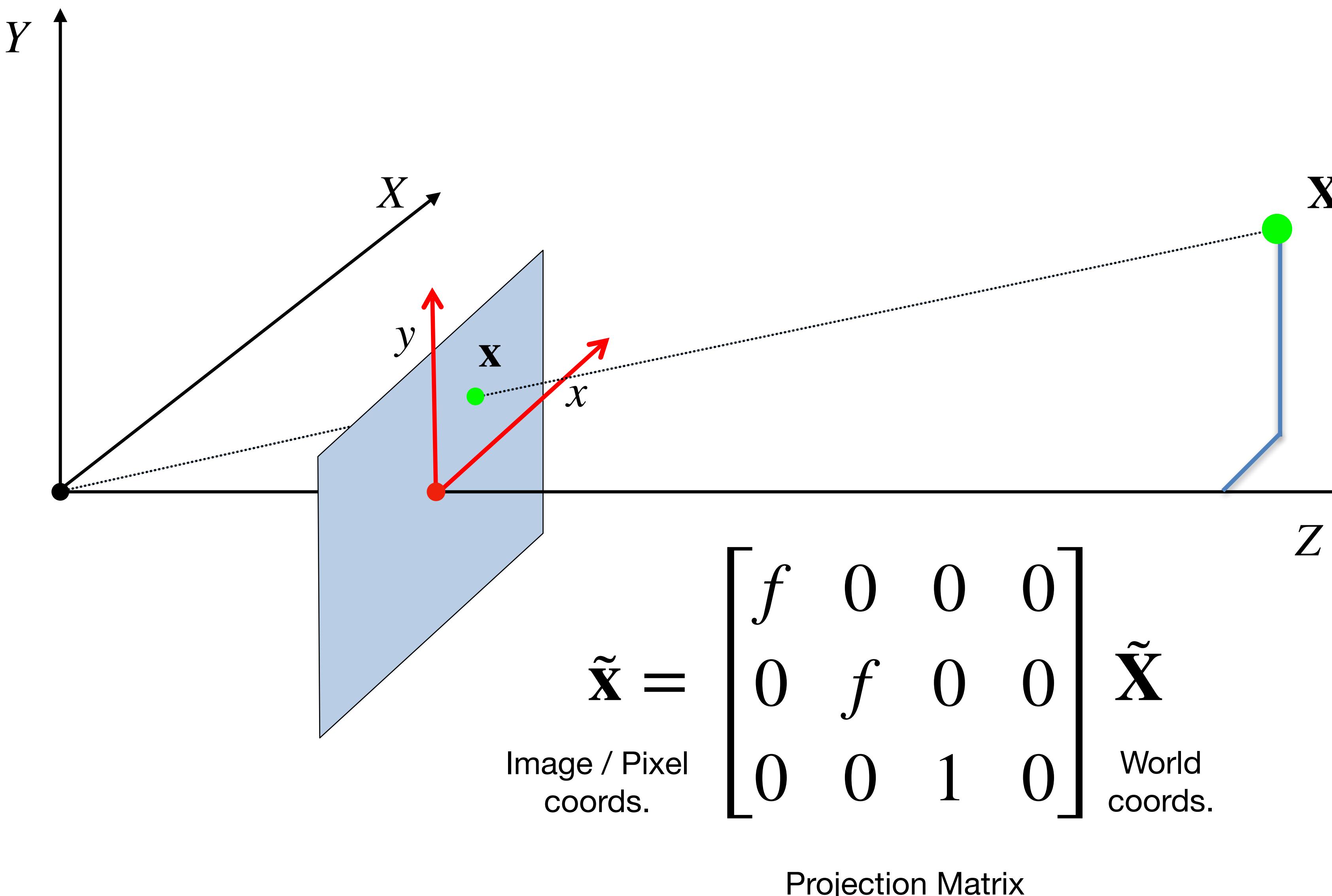
$$\tilde{\mathbf{x}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f & 0 \end{bmatrix} \tilde{\mathbf{X}} = \begin{bmatrix} X \\ Y \\ Z/f \end{bmatrix} \rightarrow \mathbf{x} = \left( f \frac{X}{Z}, f \frac{Y}{Z} \right)$$

$$\tilde{\mathbf{x}} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tilde{\mathbf{X}} = \begin{bmatrix} f \cdot X \\ f \cdot Y \\ Z \end{bmatrix} \rightarrow \mathbf{x} = \left( f \frac{X}{Z}, f \frac{Y}{Z} \right)$$

# Perspective projection

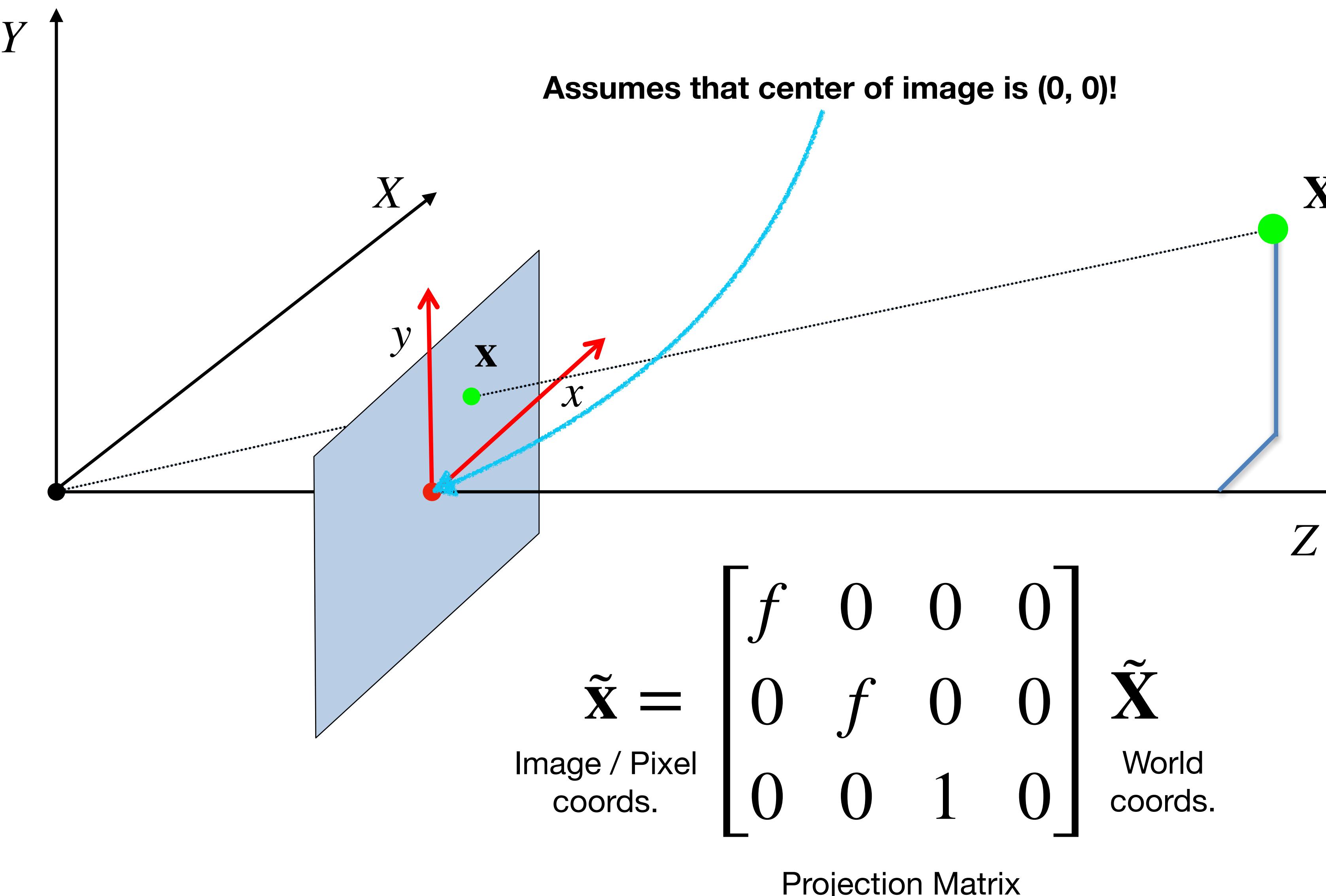
$$\tilde{\mathbf{x}} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tilde{\mathbf{X}}$$

# Perspective projection

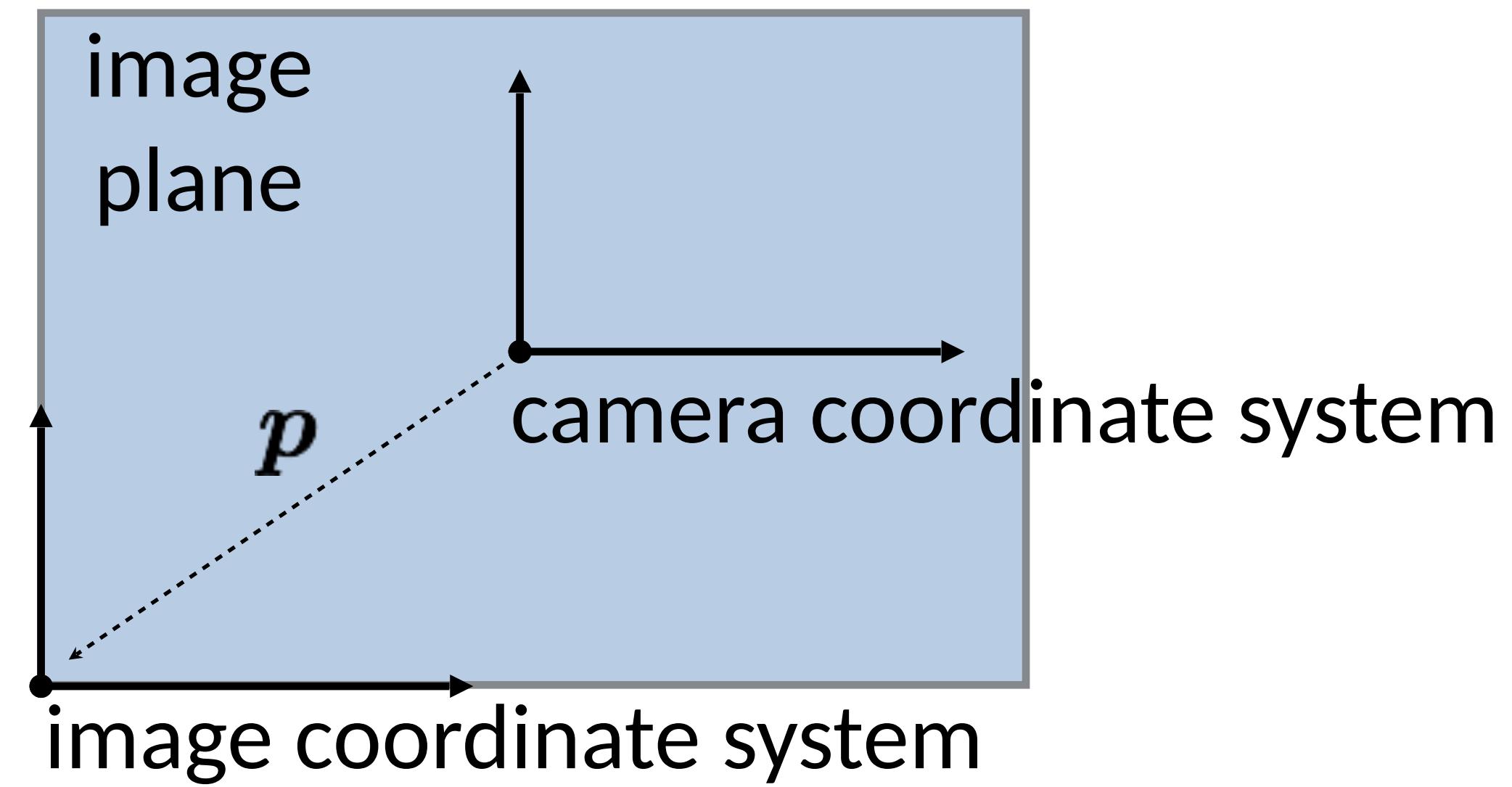


# Questions?

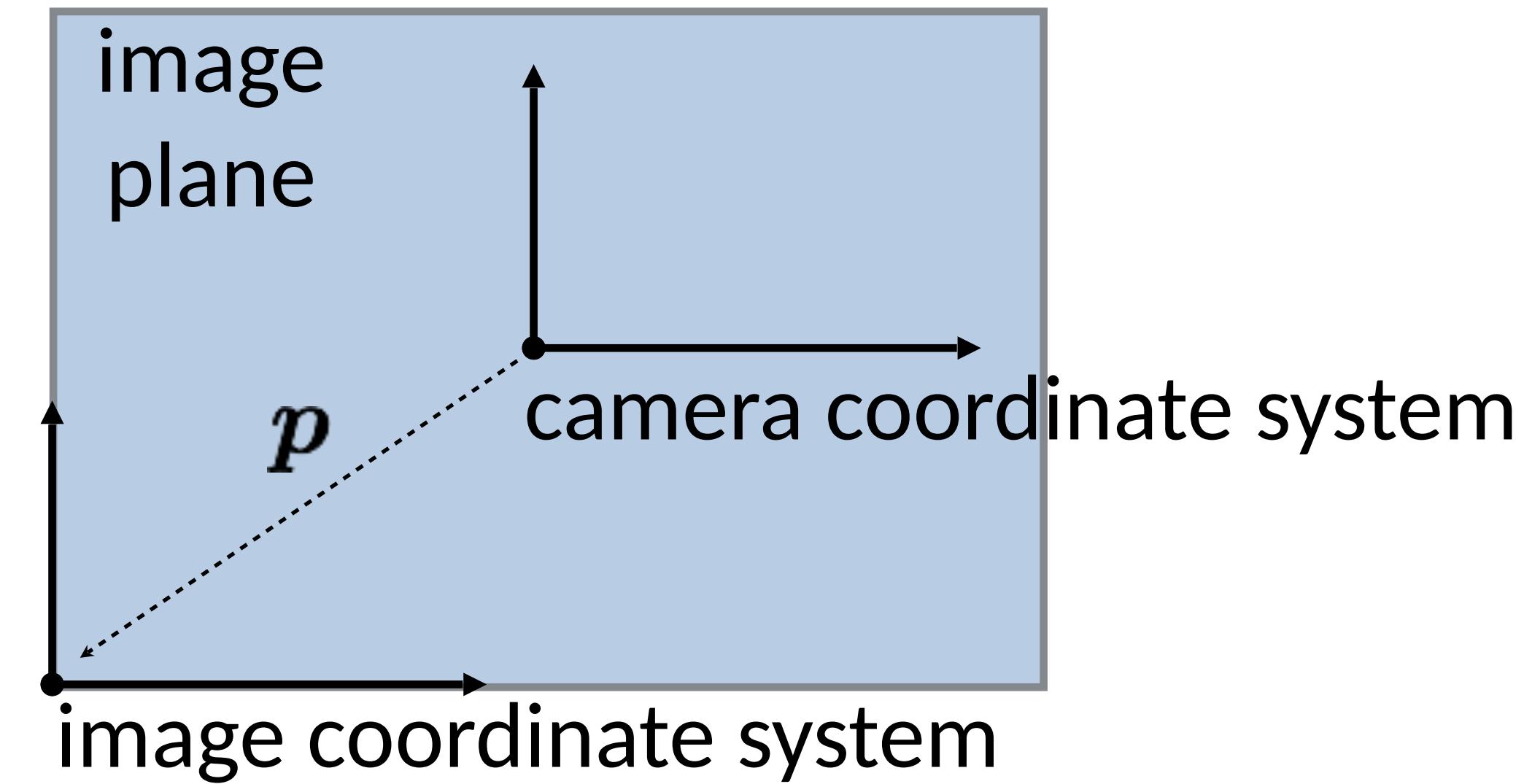
# Perspective projection



# More General Case: Arbitrary Image Centre

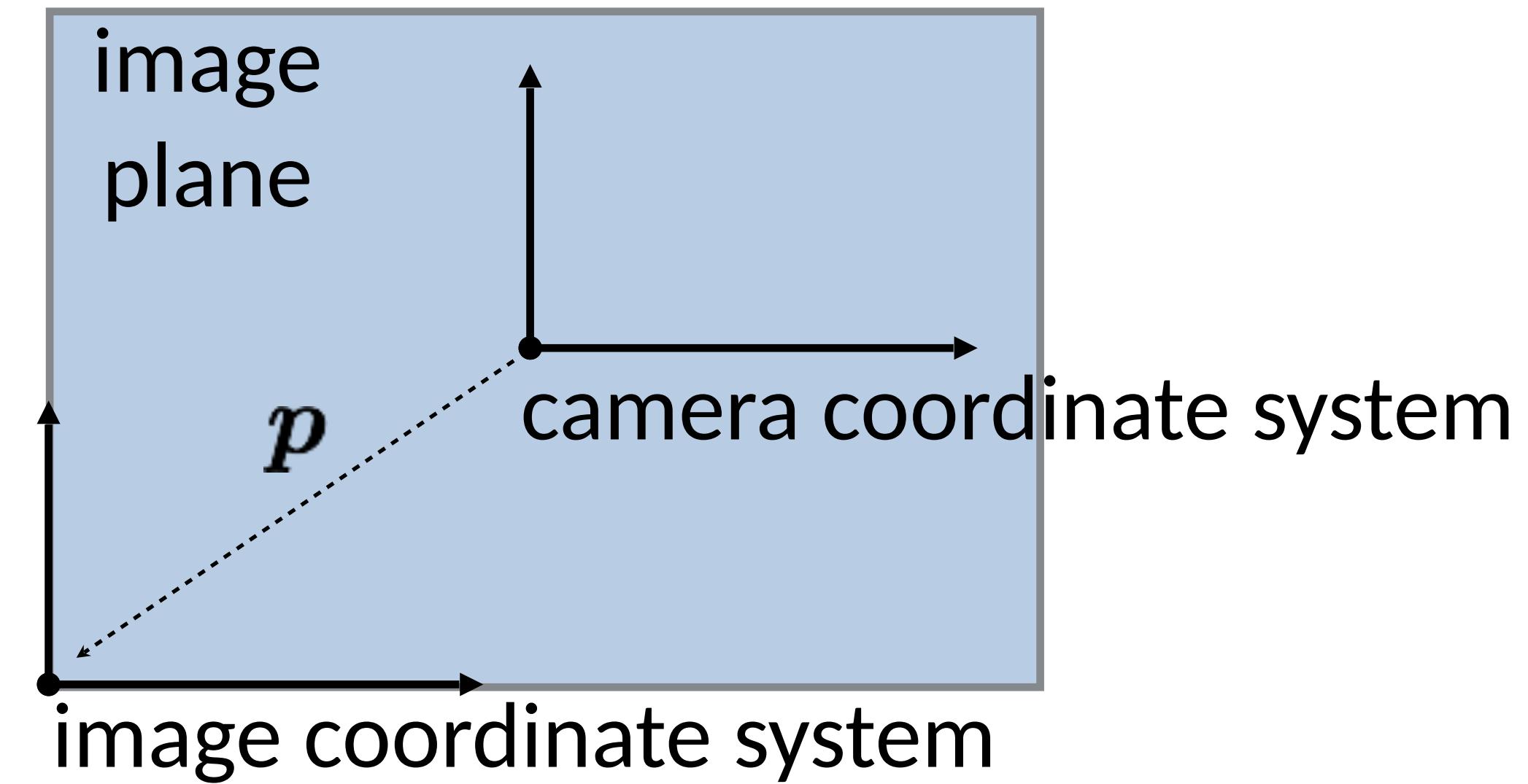


# More General Case: Arbitrary Image Centre



$$\begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

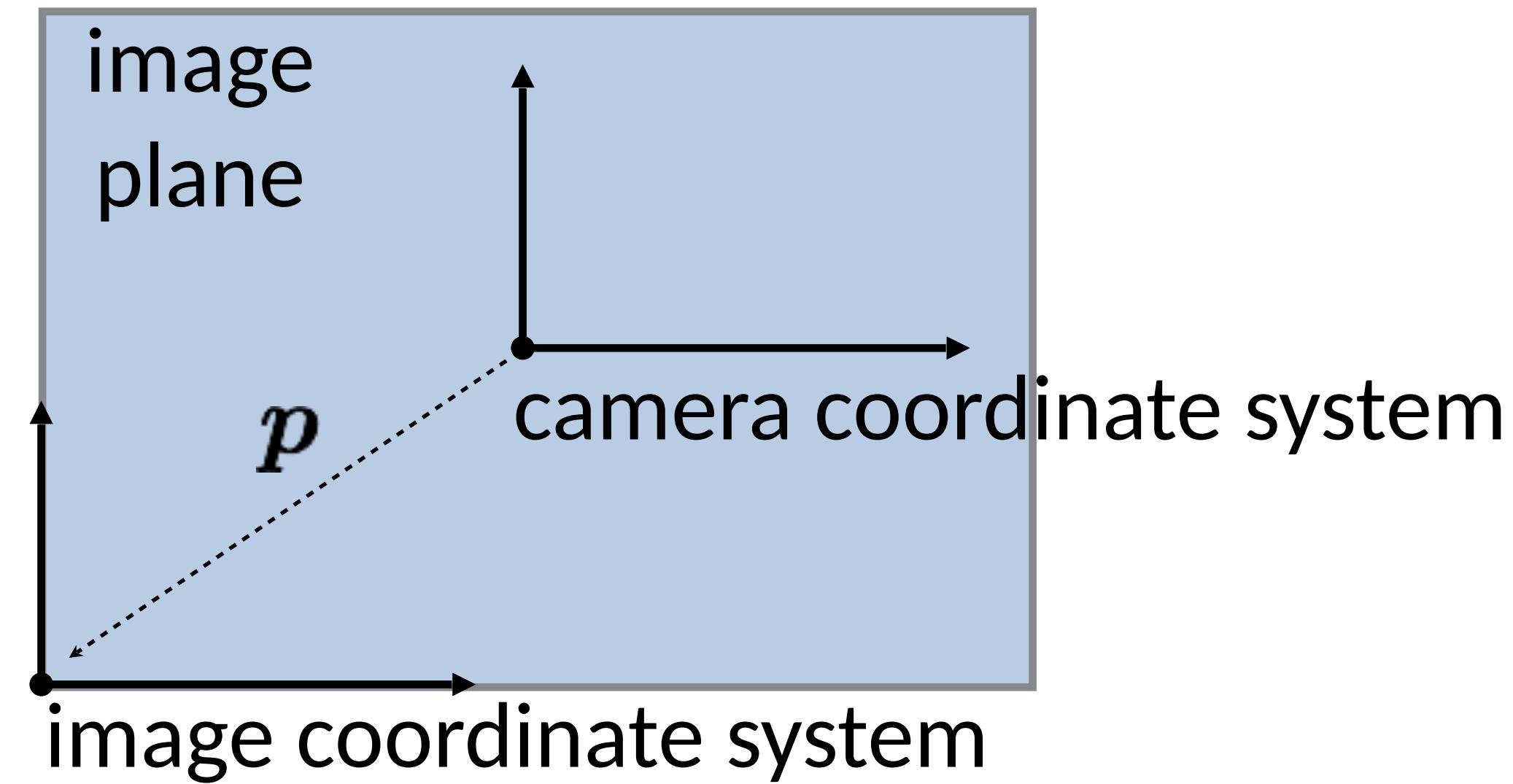
# More General Case: Arbitrary Image Centre



How does the projection matrix change?

$$\begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

# More General Case: Arbitrary Image Centre



How does the projection matrix change?

$$\begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

shift vector  
transforming  
camera origin to  
image origin

# Decomposing the Projection Matrix

We can decompose the projection matrix like this:

$$\begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

# Decomposing the Projection Matrix

We can decompose the projection matrix like this:

$$\begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

What does each part of the matrix represent?

# Decomposing the Projection Matrix

We can decompose the projection matrix like this:

$$\begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$



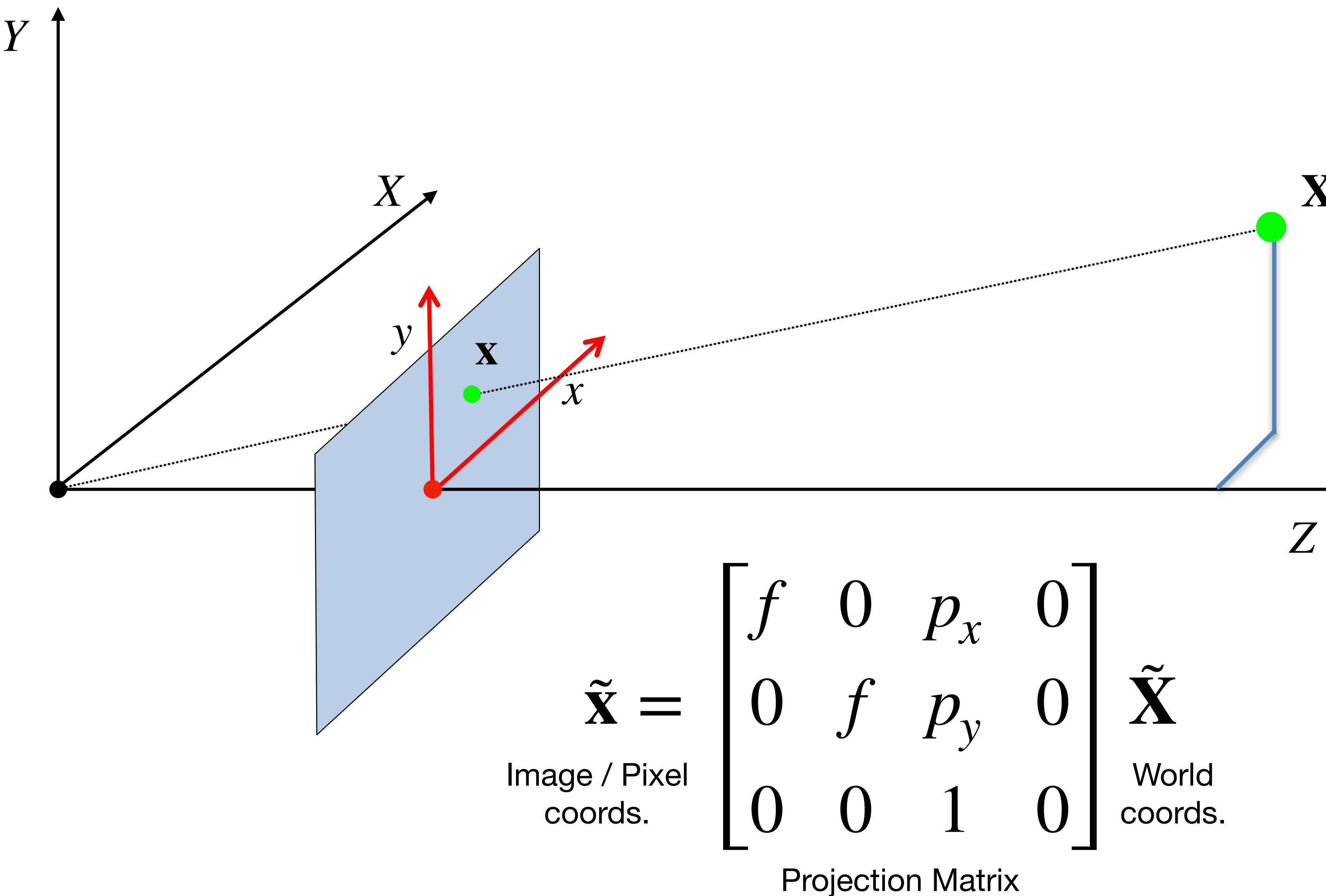
(homogeneous) transformation  
from 2D to 2D, accounting for non  
unit focal length and origin shift

(homogeneous) perspective projection  
from 3D to 2D, assuming image plane at  
 $z = 1$  and shared camera/image origin

Also written as:  $\mathbf{K} [I | 0]$

$$\text{where } \mathbf{K} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

# Perspective projection

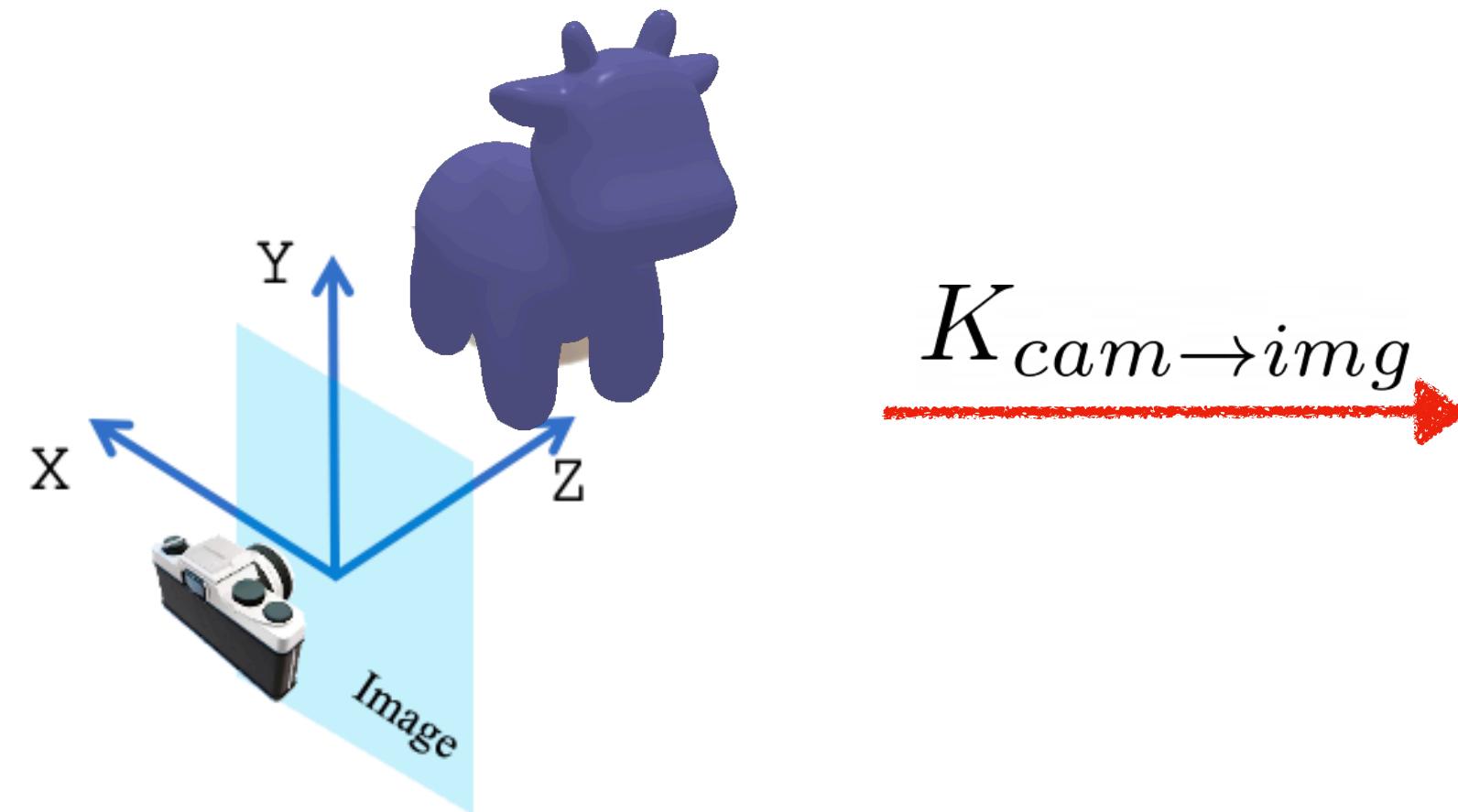


# In practice: Decoupling Projection from Image Size

$$K \equiv K_{img \rightarrow pix} \ K_{cam \rightarrow img}$$

# In practice: Decoupling Projection from Image Size

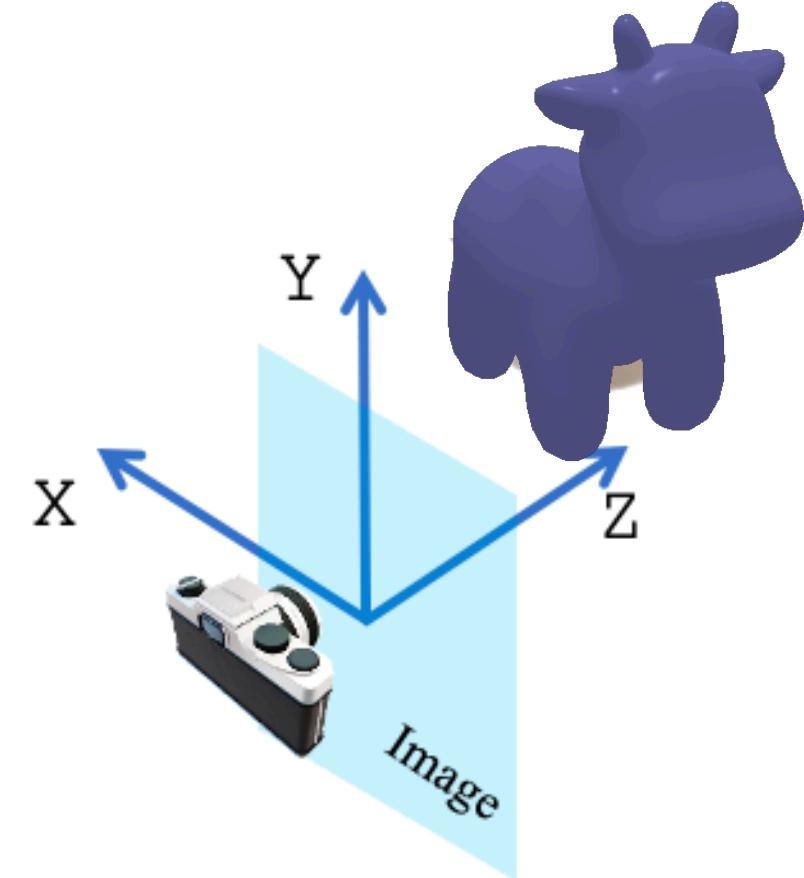
$$K \equiv K_{img \rightarrow pix} \ K_{cam \rightarrow img}$$



$\mathbb{P}^3$   
3D in Camera  
Frame

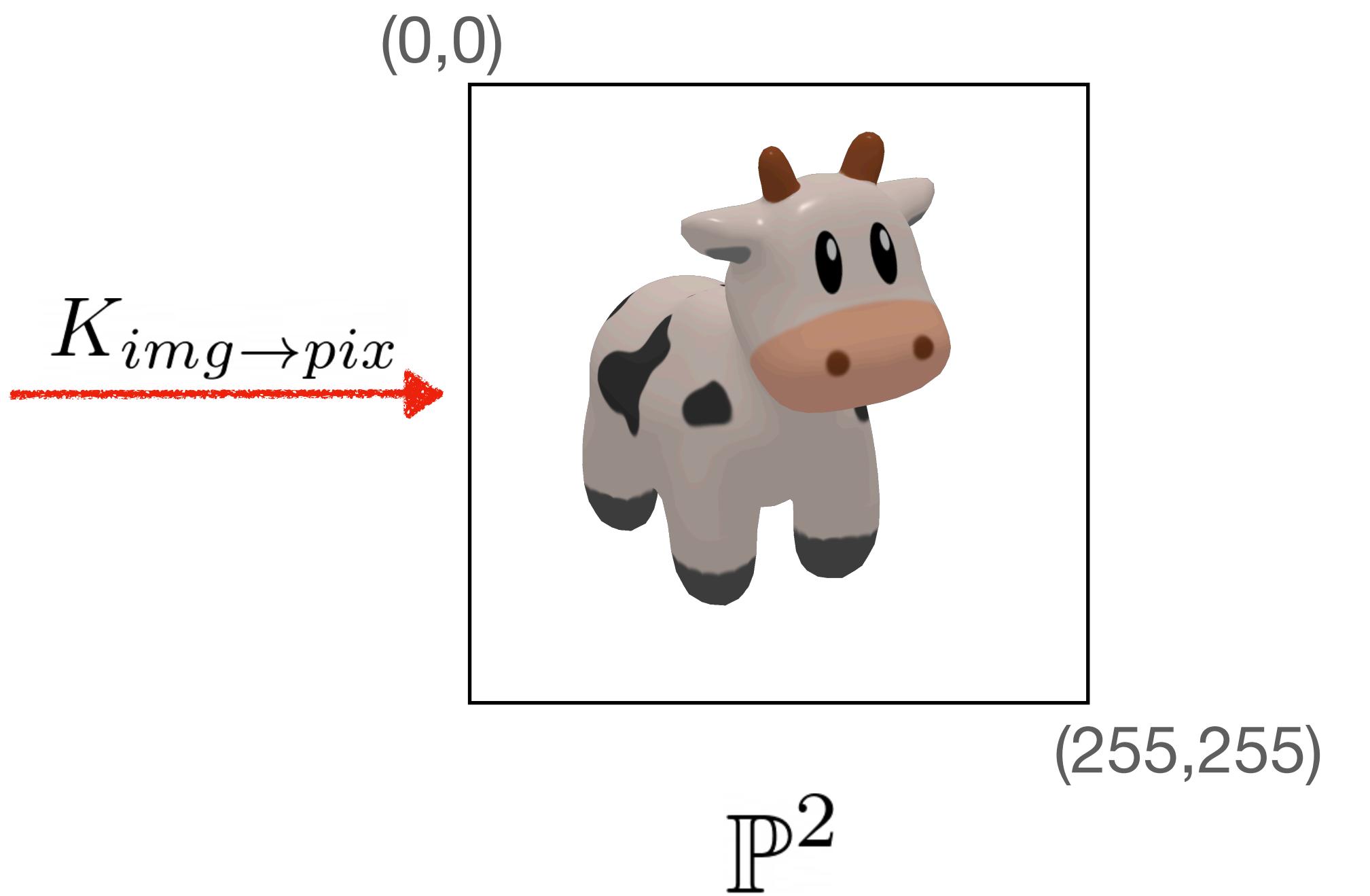
# In practice: Decoupling Projection from Image Size

$$K \equiv K_{img \rightarrow pix} \ K_{cam \rightarrow img}$$



$\mathbb{P}^3$   
3D in Camera  
Frame

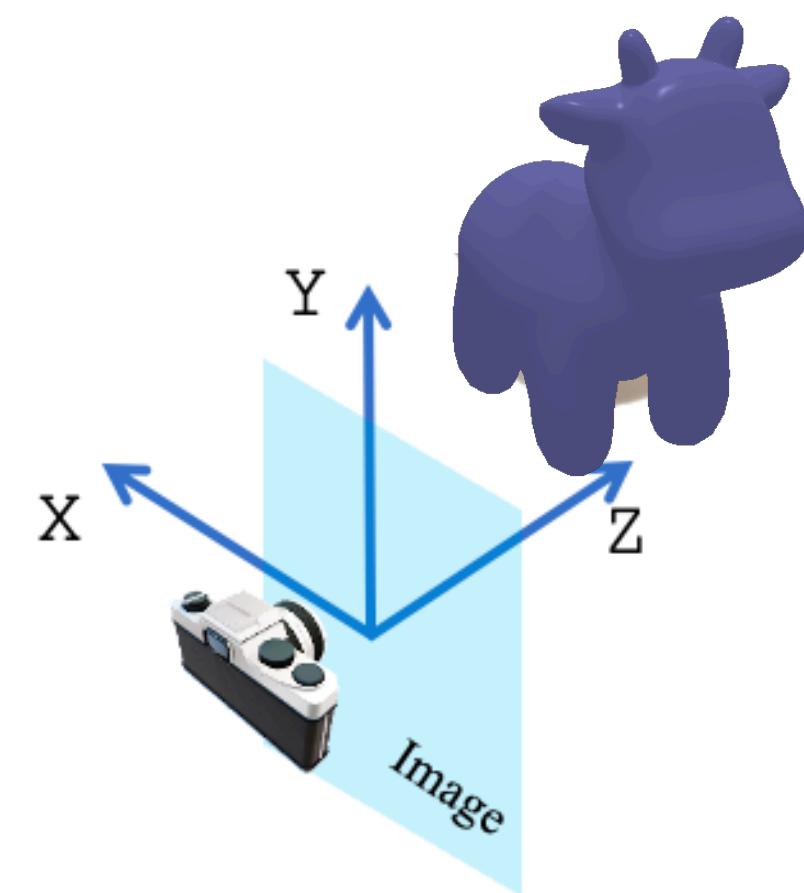
$$\underline{K_{cam \rightarrow img}}$$



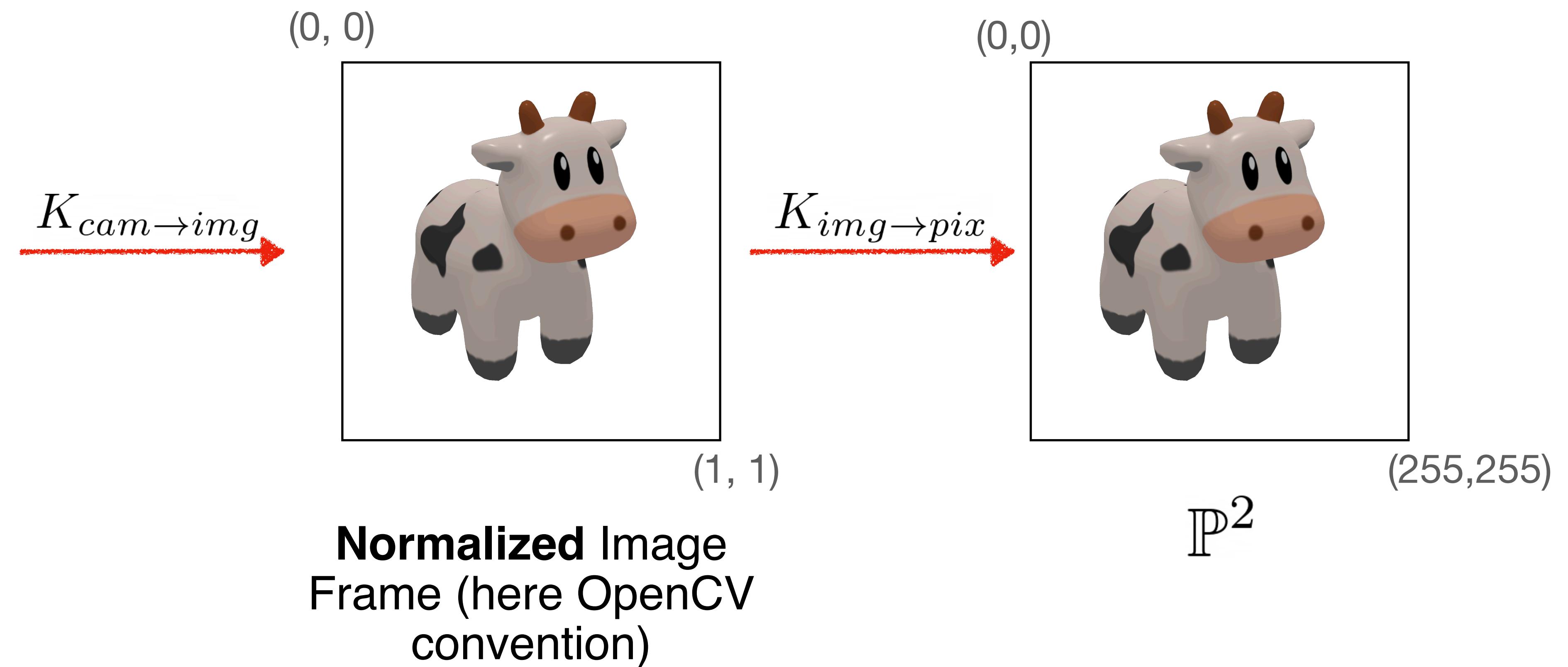
$\mathbb{P}^2$

# In practice: Decoupling Projection from Image Size

$$K \equiv K_{img \rightarrow pix} \ K_{cam \rightarrow img}$$



$\mathbb{P}^3$   
3D in Camera  
Frame

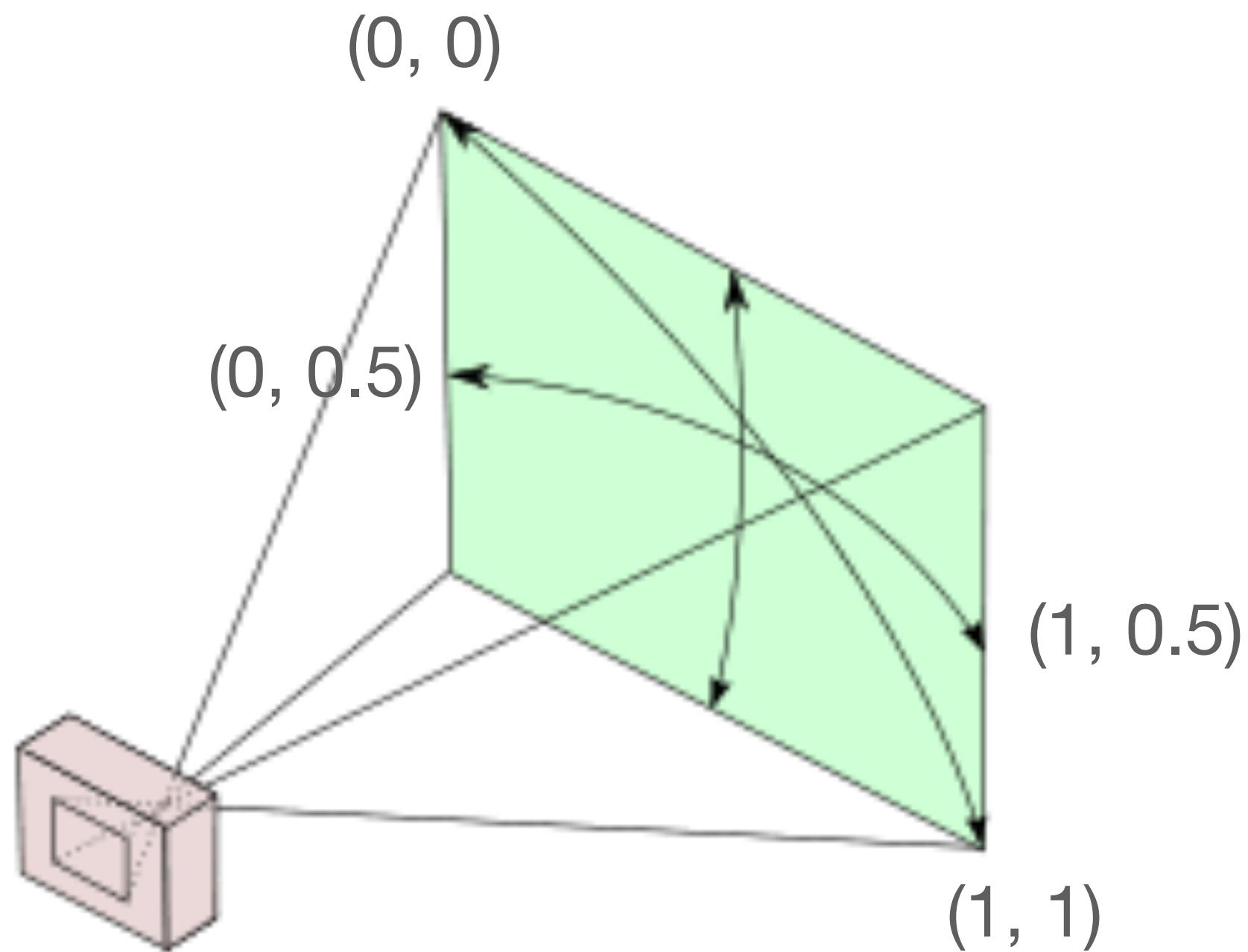


# Exercise: Focal Length as Function of FOV

$$K_{cam \rightarrow img}$$

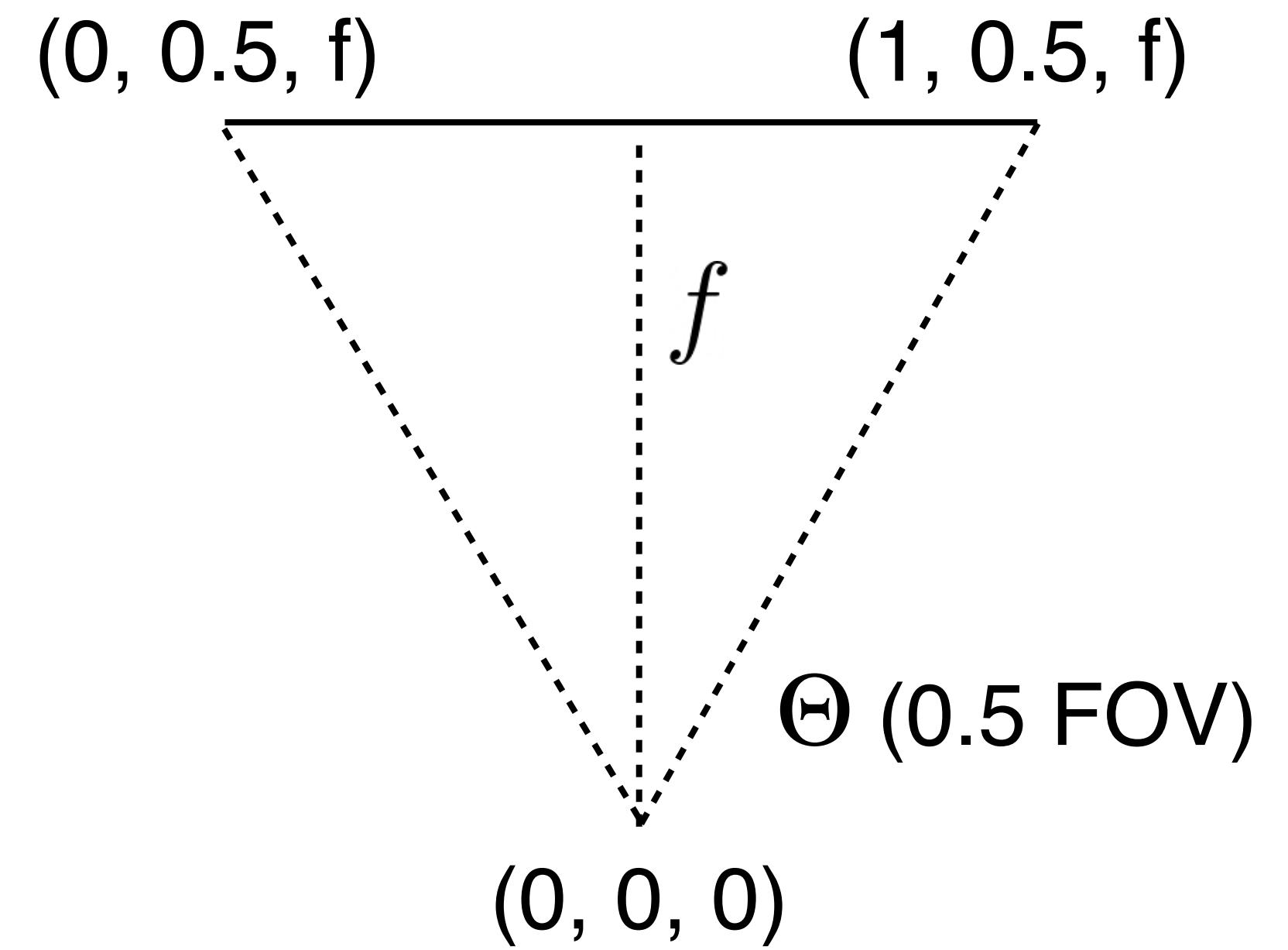
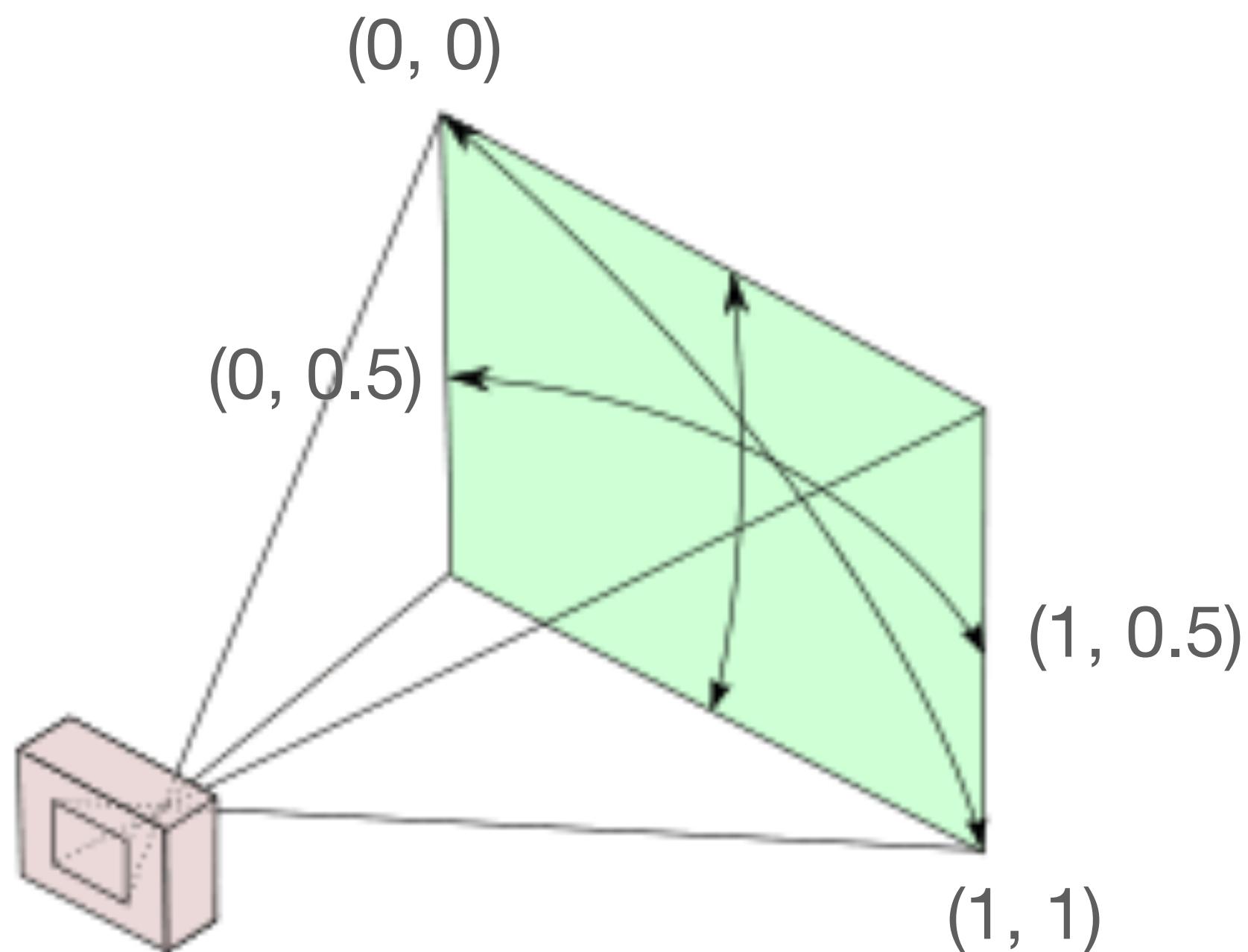
# Exercise: Focal Length as Function of FOV

$$K_{cam \rightarrow img}$$



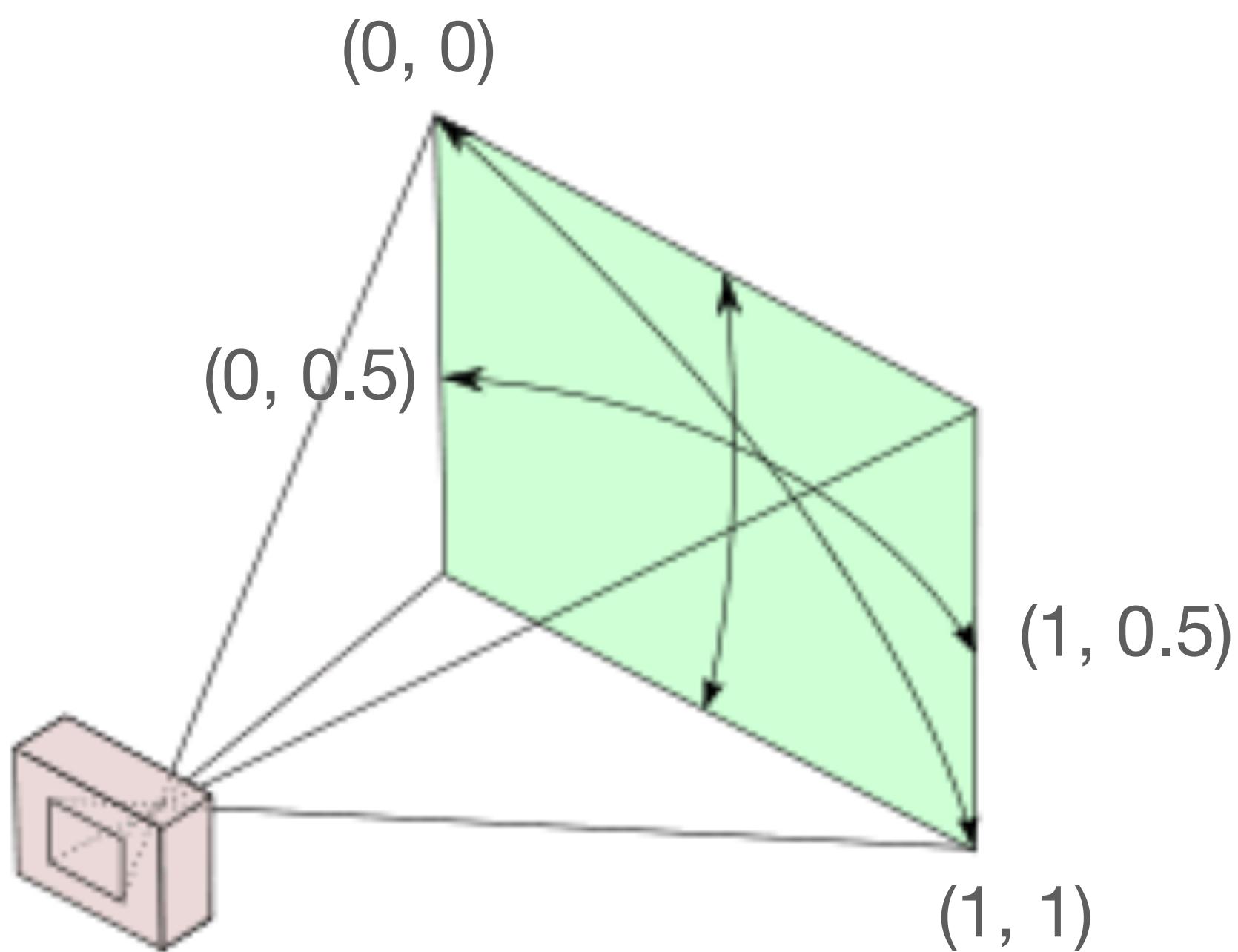
# Exercise: Focal Length as Function of FOV

$$K_{cam \rightarrow img}$$

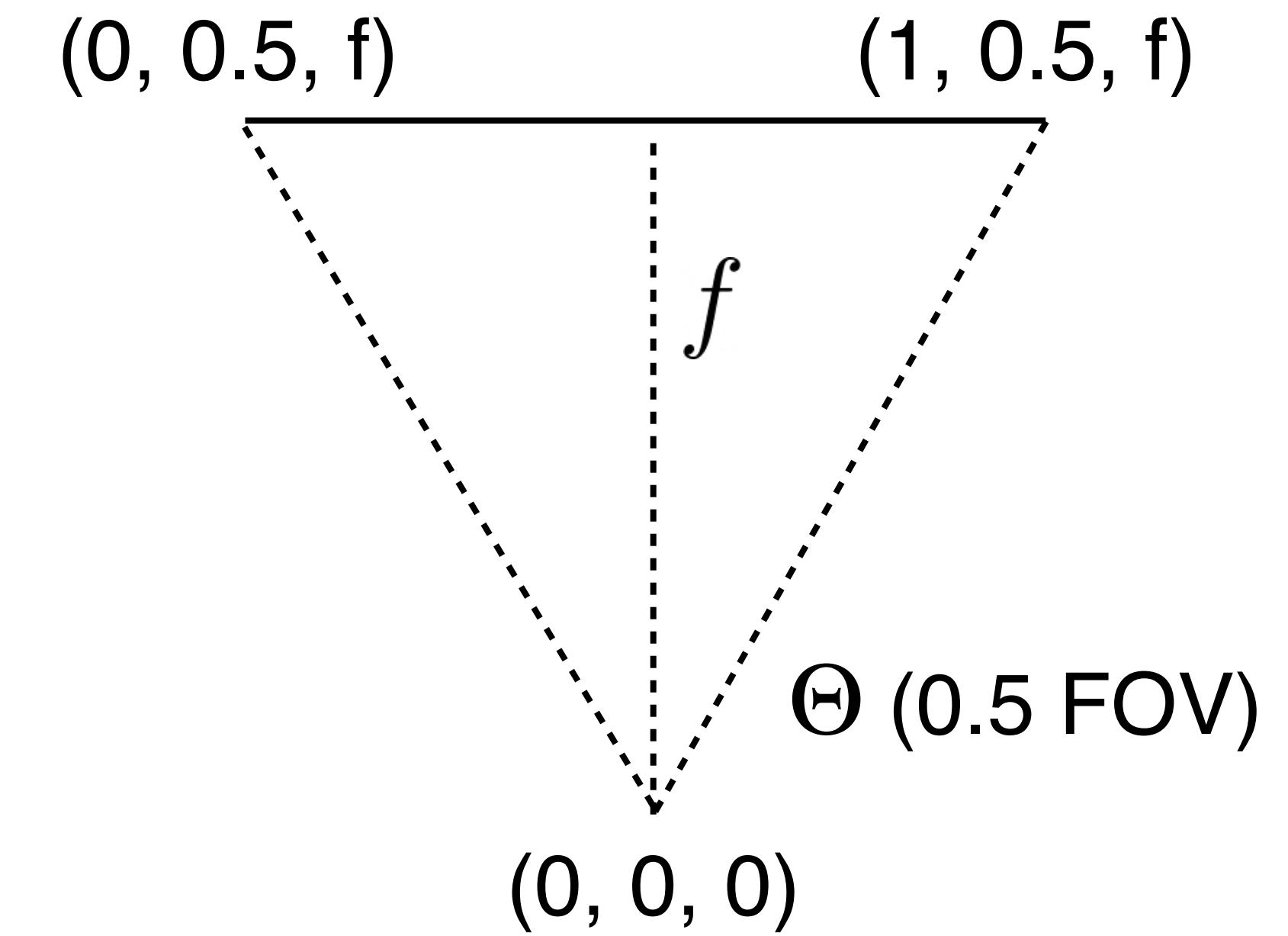


# Exercise: Focal Length as Function of FOV

$K_{cam \rightarrow img}$



$$\begin{bmatrix} \frac{1}{2 \tan(\Theta)} & 0 & 0.5 & 0 \\ 0 & \frac{1}{2 \tan(\Theta)} & 0.5 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$



# Exercise: Cropping an Image

$K_{cam \rightarrow img}$

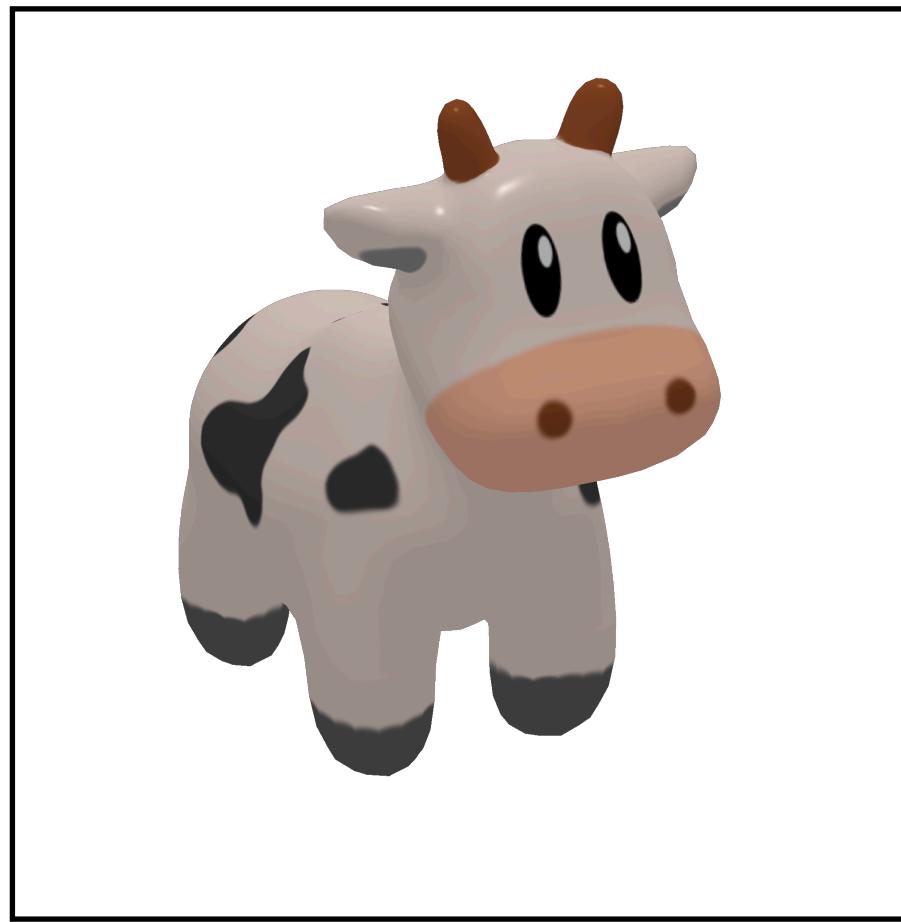


$K'_{cam \rightarrow img}$



# Exercise: Cropping an Image

$K_{cam \rightarrow img}$

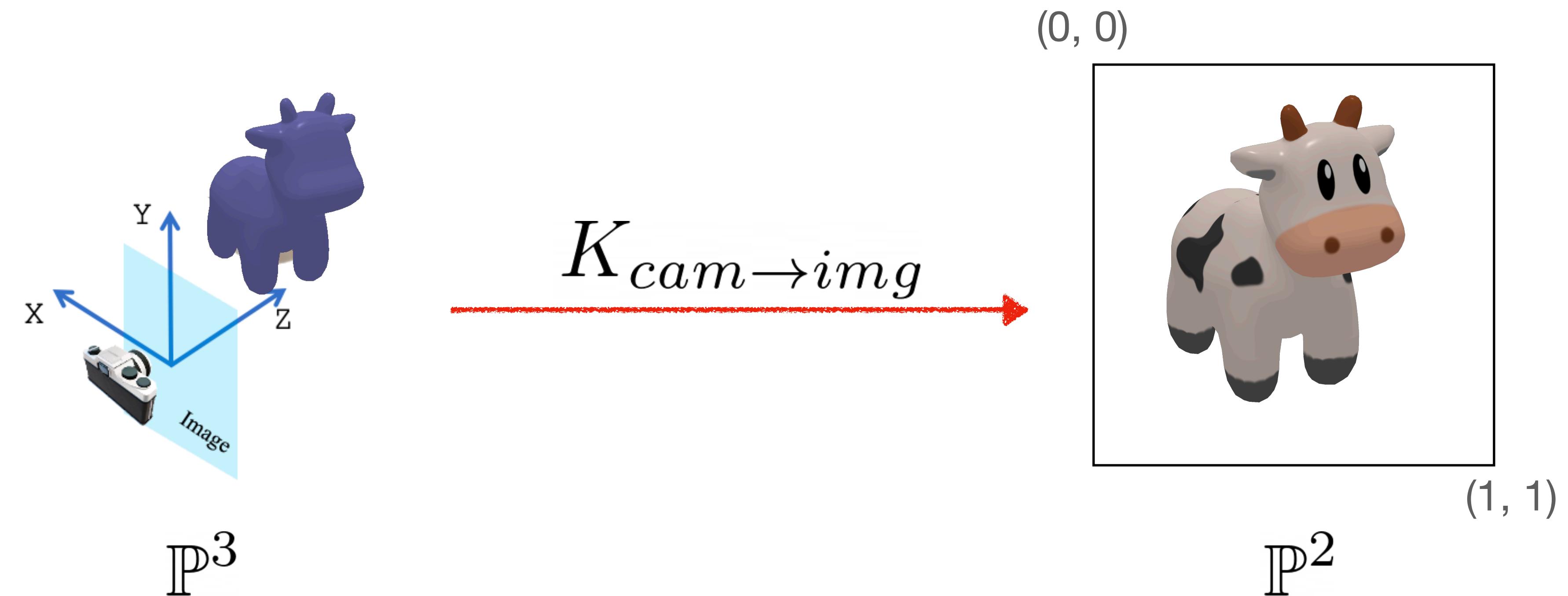


$K'_{cam \rightarrow img}$



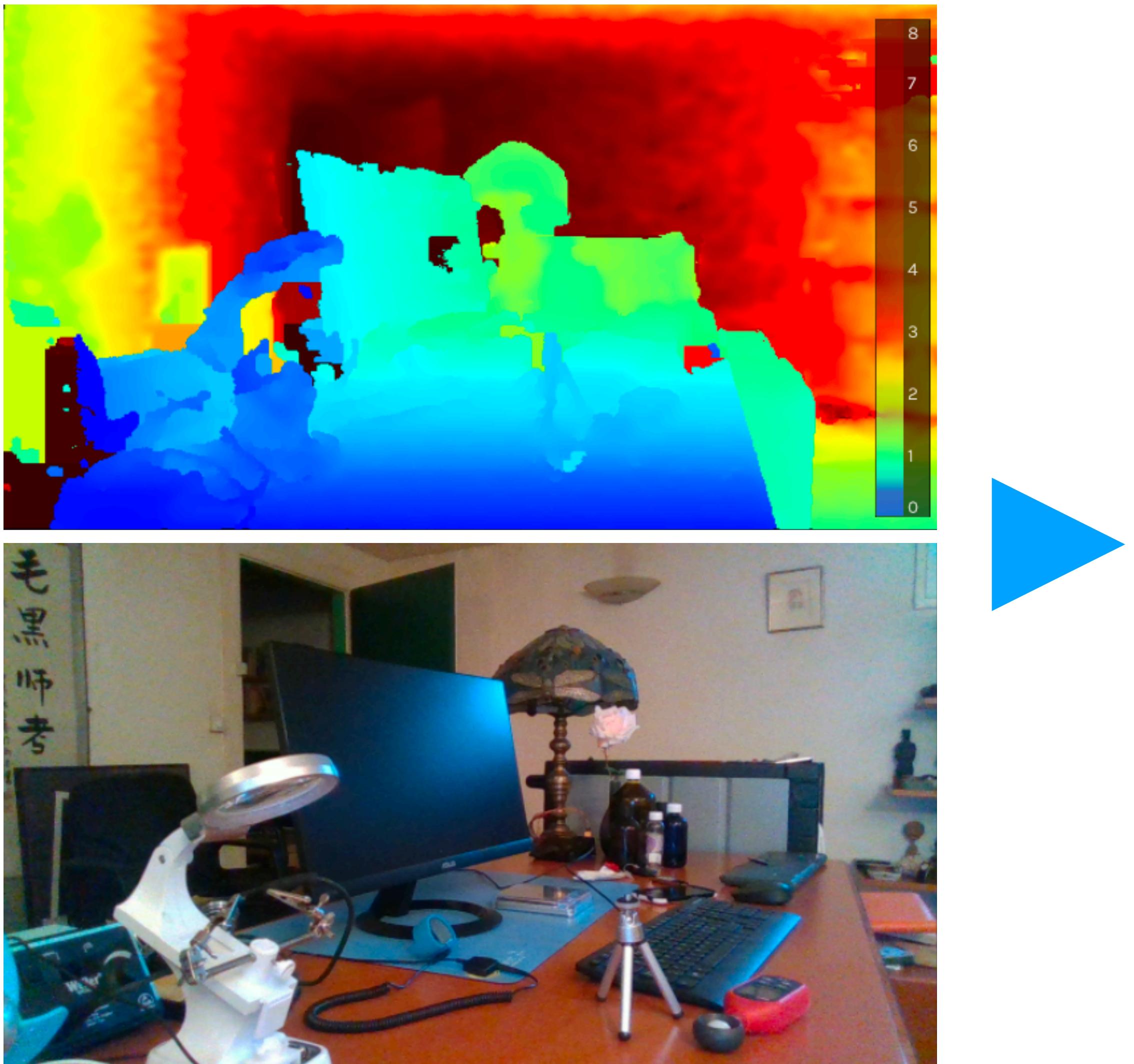
$$K'_{cam \rightarrow img} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} K_{cam \rightarrow img}$$

# In practice: Decoupling Projection from Image Size

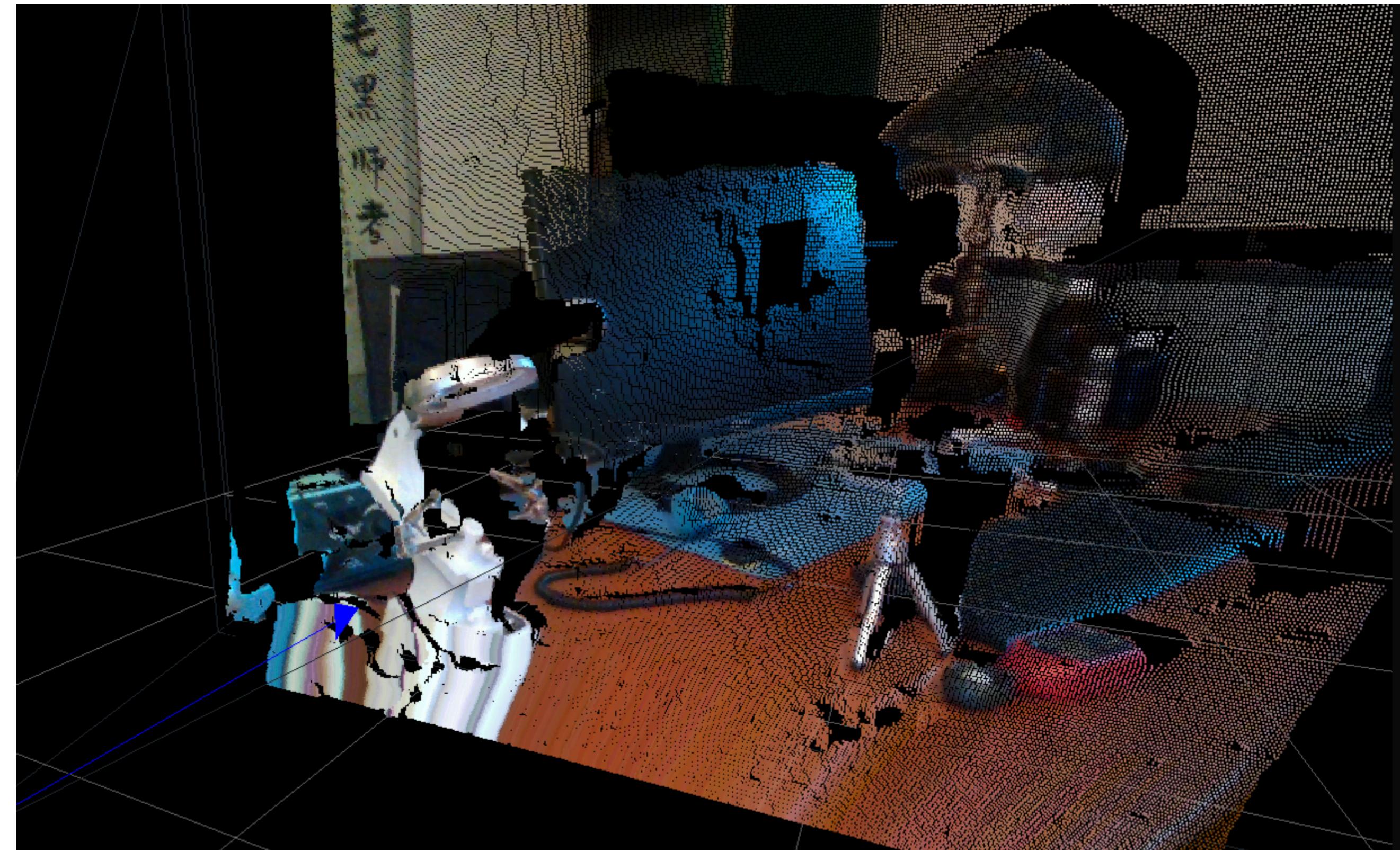
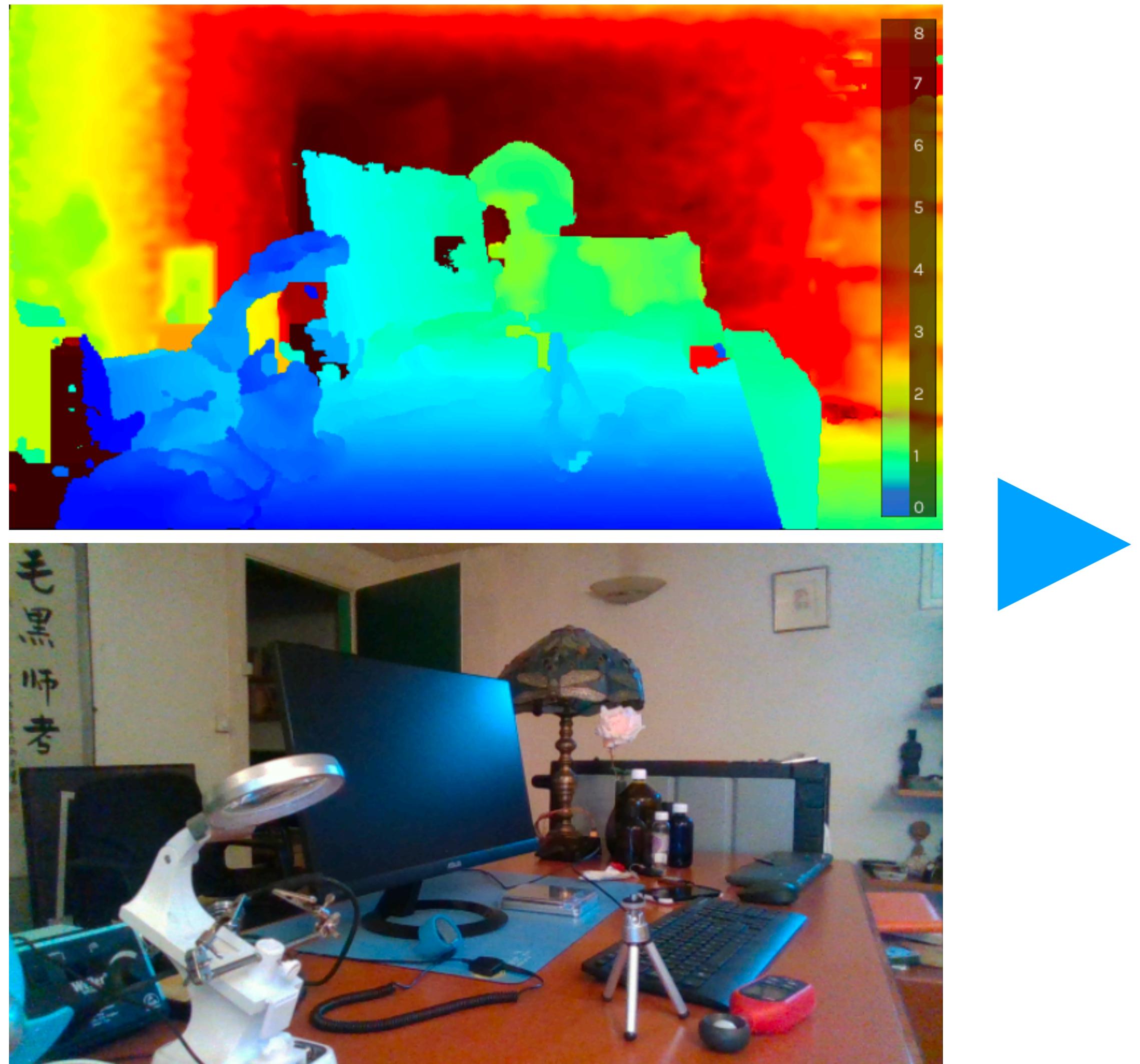


Helps (conceptually and in implementation) to reason in normalized image coordinates

# “Unproject”: Go from pixel and depth to 3D point



# “Unproject”: Go from pixel and depth to 3D point



“Unproject”: Go from pixel and depth to 3D point

$$\begin{pmatrix} f \cdot X + Z \cdot p_x \\ f \cdot Y + Z \cdot p_y \\ Z \end{pmatrix} = \mathbf{K} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

“Unproject”: Go from pixel and depth to 3D point

$$\begin{pmatrix} f \cdot X + Z \cdot p_x \\ f \cdot Y + Z \cdot p_y \\ Z \end{pmatrix} = \mathbf{K} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \mathbf{K}^{-1} \begin{pmatrix} f \cdot X + Z \cdot p_x \\ f \cdot Y + Z \cdot p_y \\ Z \end{pmatrix}$$

“Unproject”: Go from pixel and depth to 3D point

$$\begin{pmatrix} f \cdot X + Z \cdot p_x \\ f \cdot Y + Z \cdot p_y \\ Z \end{pmatrix} = \mathbf{K} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \mathbf{K}^{-1} \begin{pmatrix} f \cdot X + Z \cdot p_x \\ f \cdot Y + Z \cdot p_y \\ Z \end{pmatrix}$$

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = Z \cdot \mathbf{K}^{-1} \begin{pmatrix} \frac{f \cdot X}{Z} + p_x \\ \frac{f \cdot Y}{Z} + p_y \\ 1 \end{pmatrix}$$

“Unproject”: Go from pixel and depth to 3D point

$$\begin{pmatrix} f \cdot X + Z \cdot p_x \\ f \cdot Y + Z \cdot p_y \\ Z \end{pmatrix} = \mathbf{K} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \mathbf{K}^{-1} \begin{pmatrix} f \cdot X + Z \cdot p_x \\ f \cdot Y + Z \cdot p_y \\ Z \end{pmatrix}$$

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = Z \cdot \mathbf{K}^{-1} \begin{pmatrix} \frac{f \cdot X}{Z} + p_x \\ \frac{f \cdot Y}{Z} + p_y \\ 1 \end{pmatrix}$$

$$\mathbf{x} = Z \cdot \mathbf{K}^{-1} \begin{pmatrix} x_{pix} \\ y_{pix} \\ 1 \end{pmatrix}$$

“Unproject”: Go from pixel and depth to 3D point

$$\mathbf{X} = Z \cdot \mathbf{K}^{-1} \begin{pmatrix} x_{pix} \\ y_{pix} \\ 1 \end{pmatrix}$$

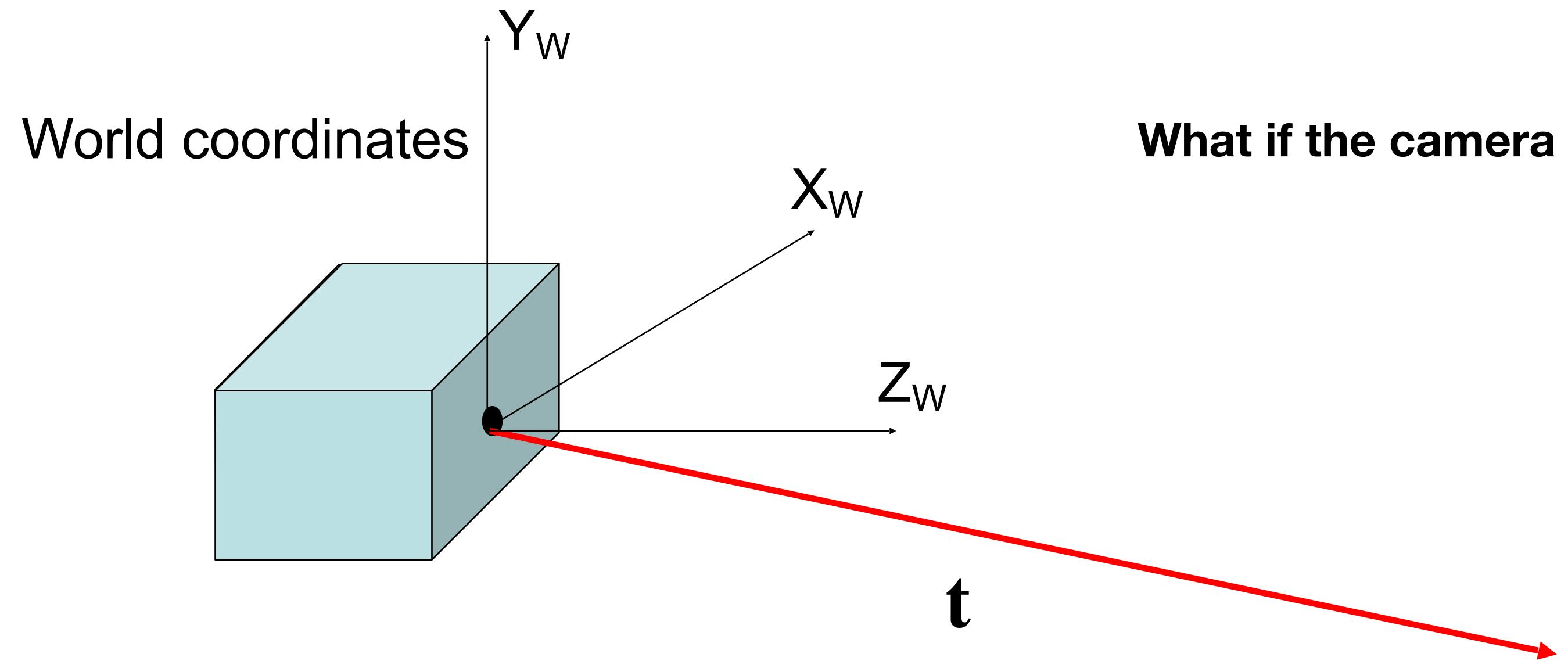
“Unproject”: Go from pixel and depth to 3D point

$$\mathbf{X} = Z \cdot \mathbf{K}^{-1} \begin{pmatrix} x_{pix} \\ y_{pix} \\ 1 \end{pmatrix}$$

$$\text{unproject}(\mathbf{x}, Z) = Z \cdot \mathbf{K}^{-1} \begin{pmatrix} x_{pix} \\ y_{pix} \\ 1 \end{pmatrix}$$

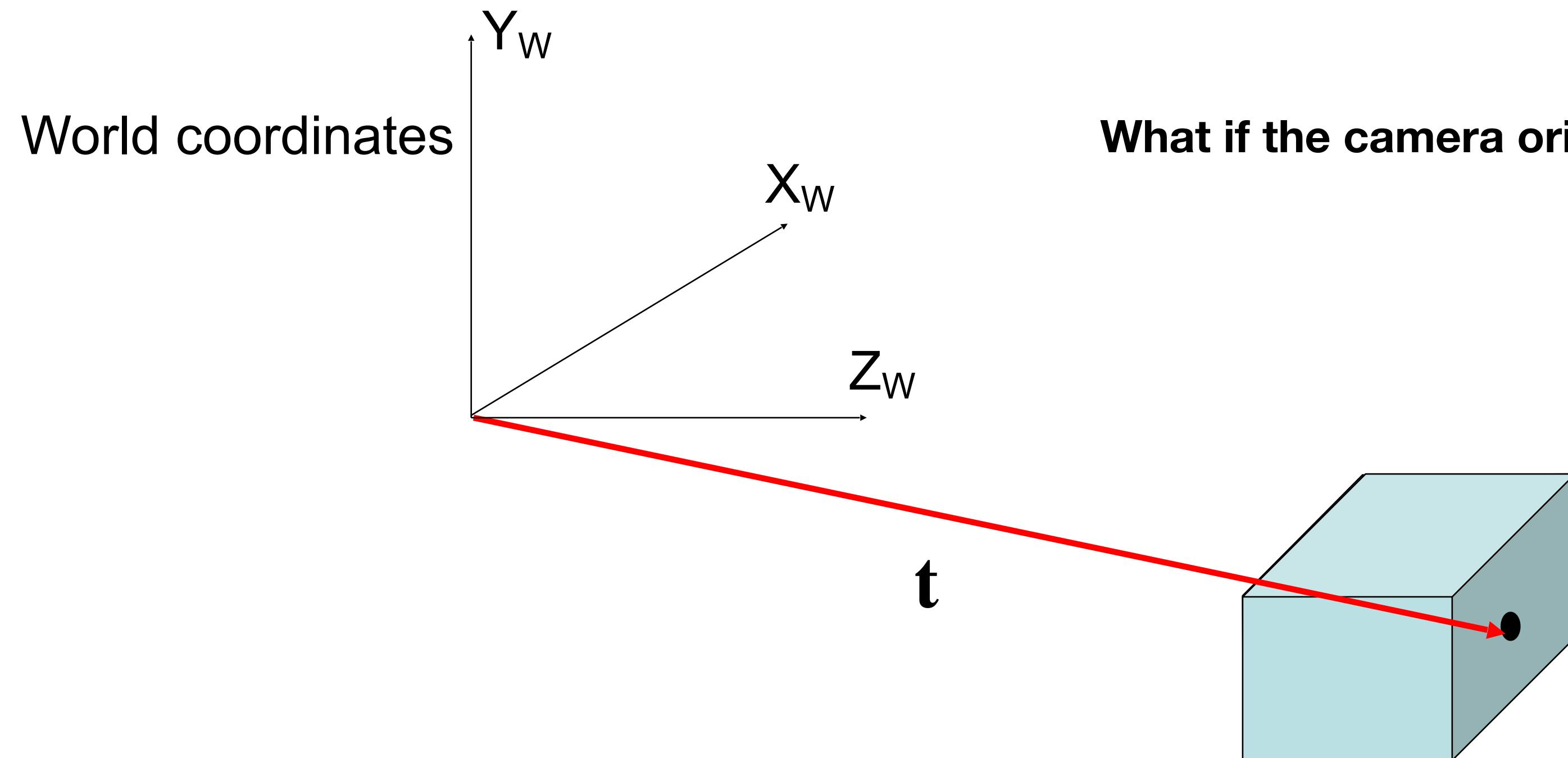
# Questions?

# Camera parameters



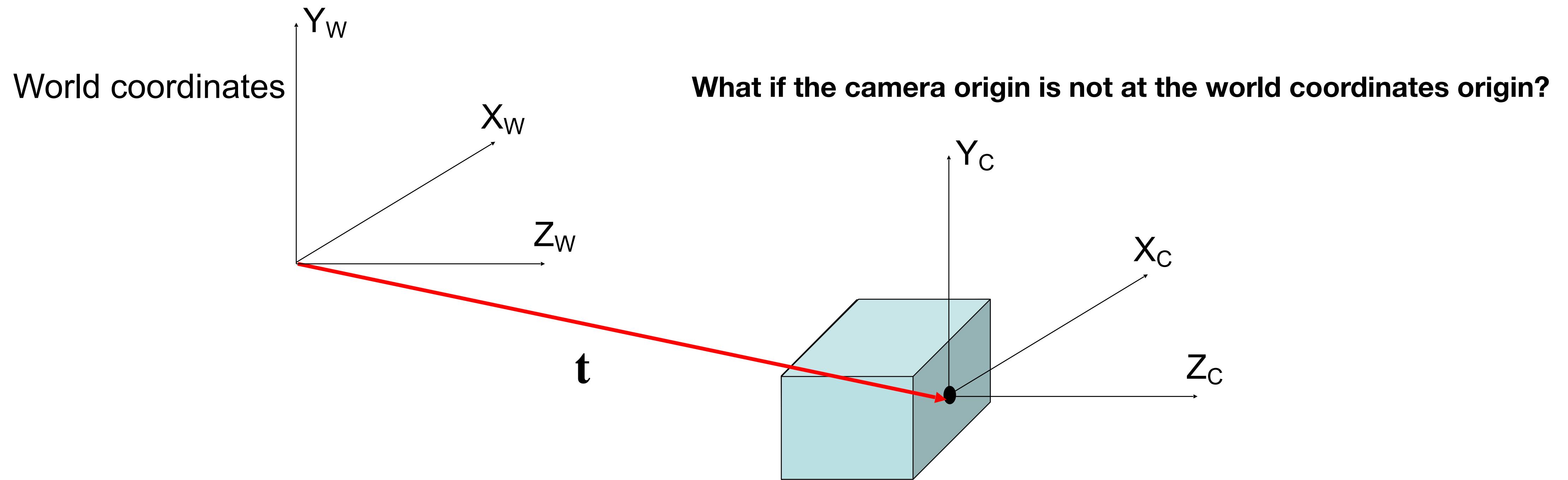
**What if the camera origin is not at the world coordinates origin?**

# Camera parameters

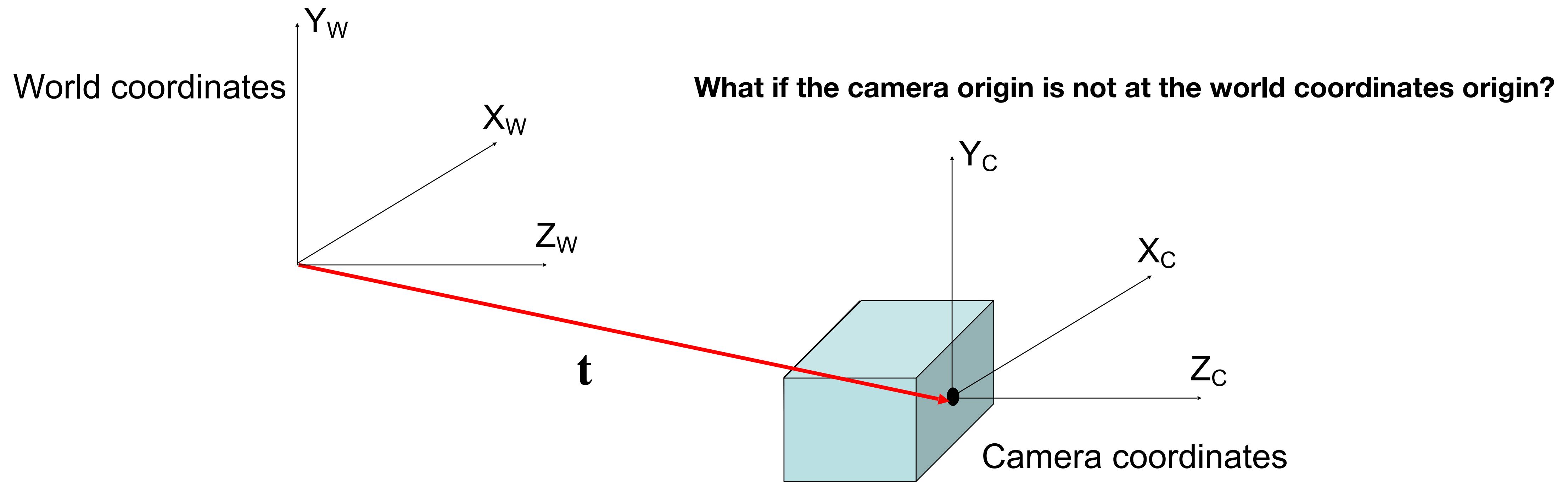


**What if the camera origin is not at the world coordinates origin?**

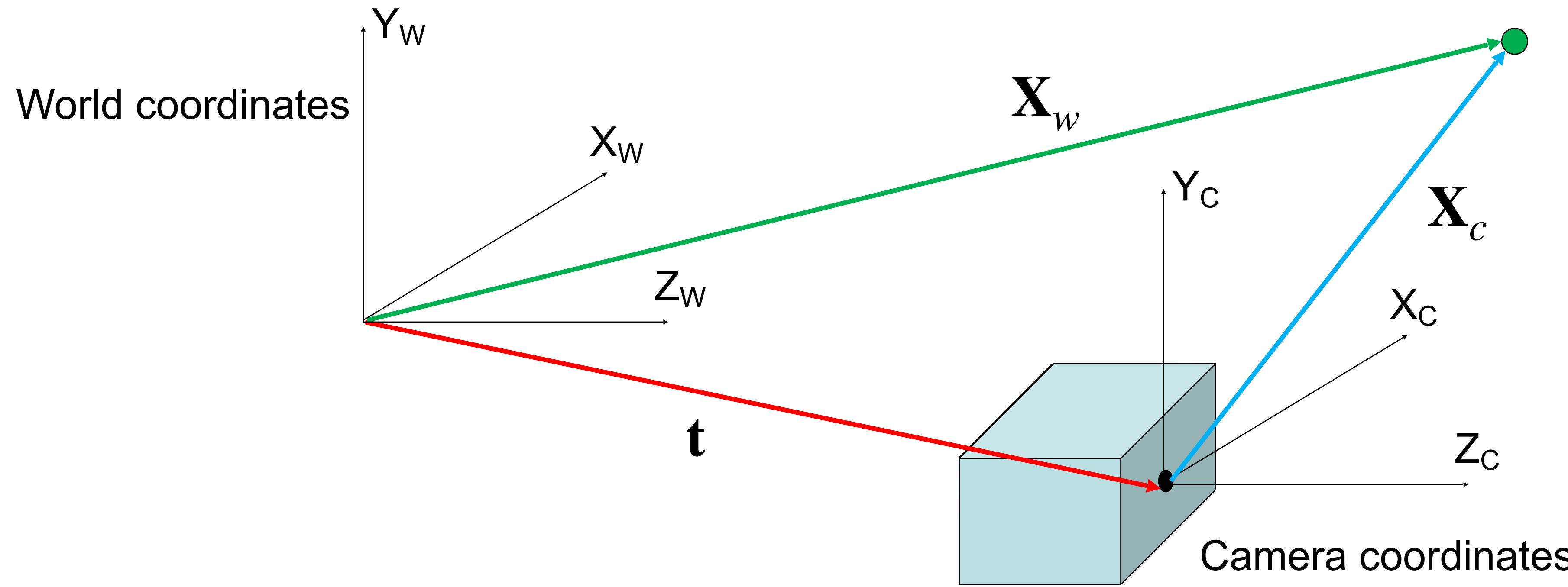
# Camera parameters



# Camera parameters



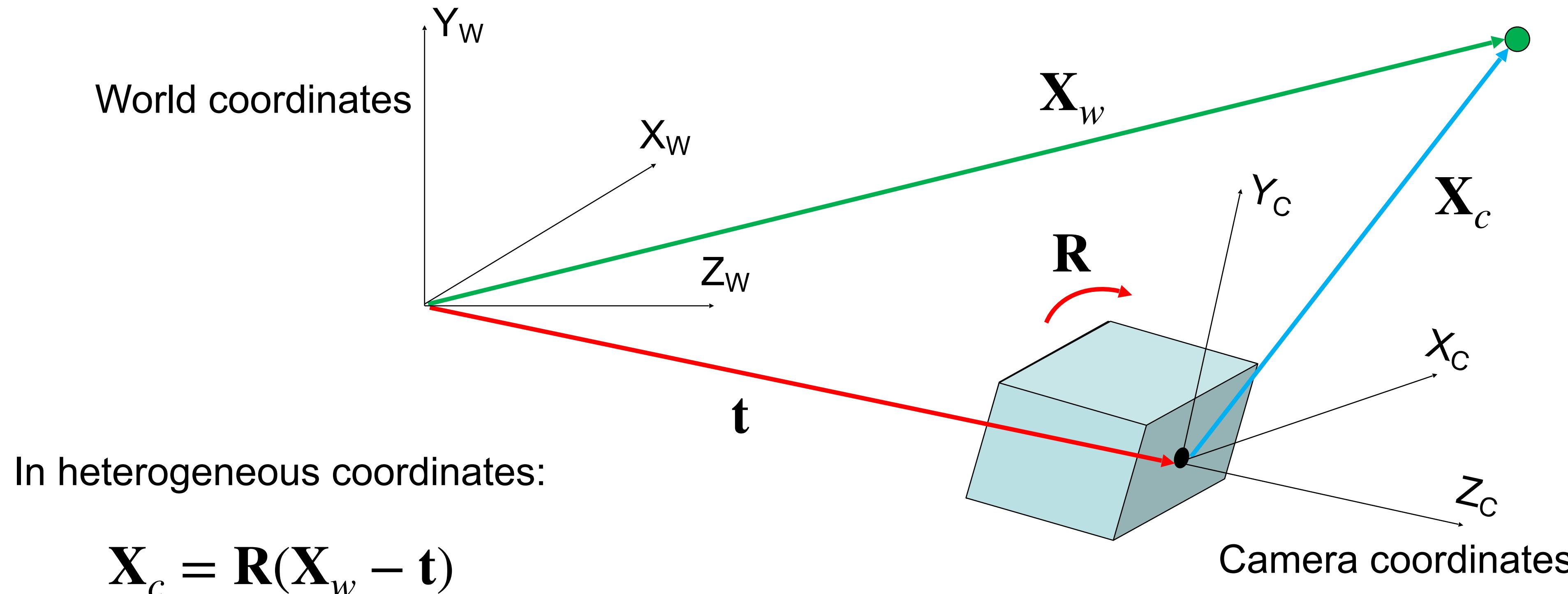
# Camera parameters



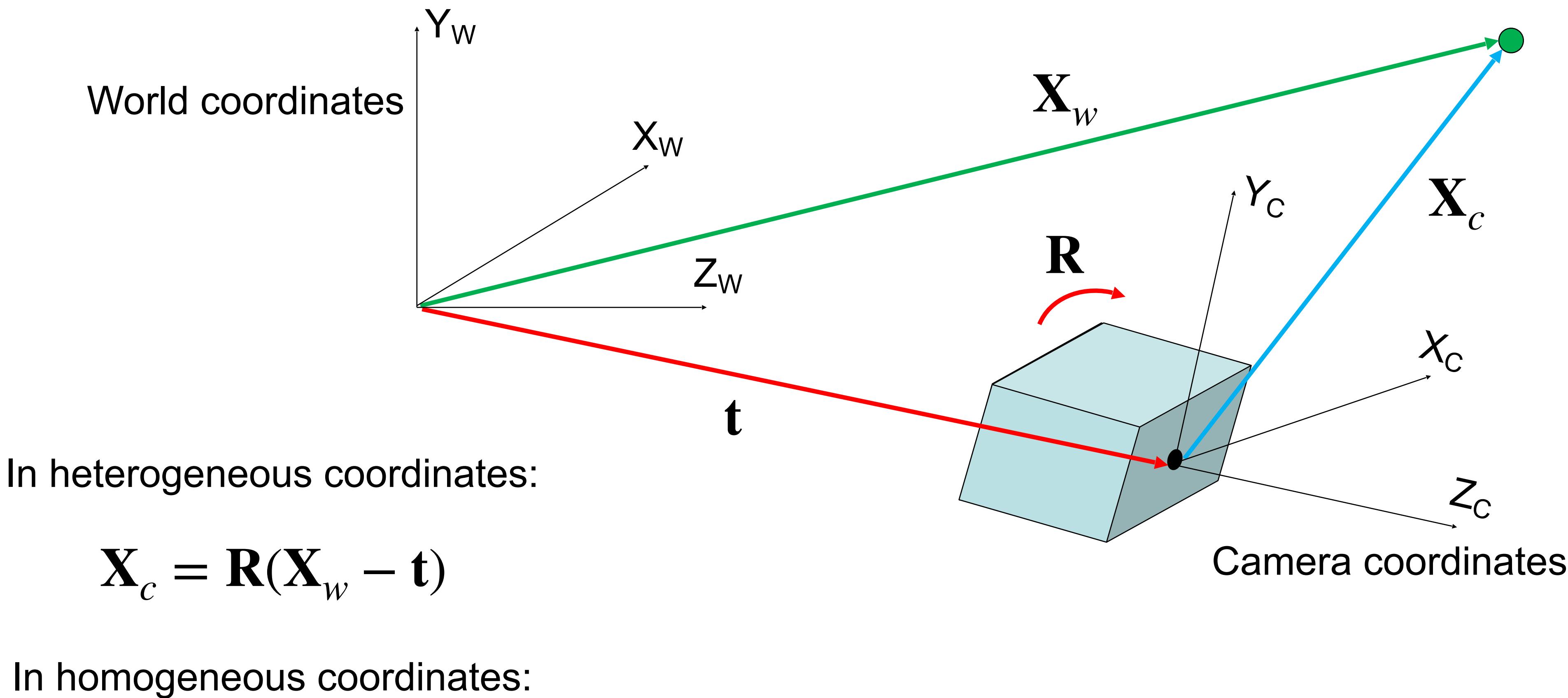
In heterogeneous coordinates:

$$\mathbf{X}_c = \mathbf{X}_w - \mathbf{t}$$

# Camera parameters

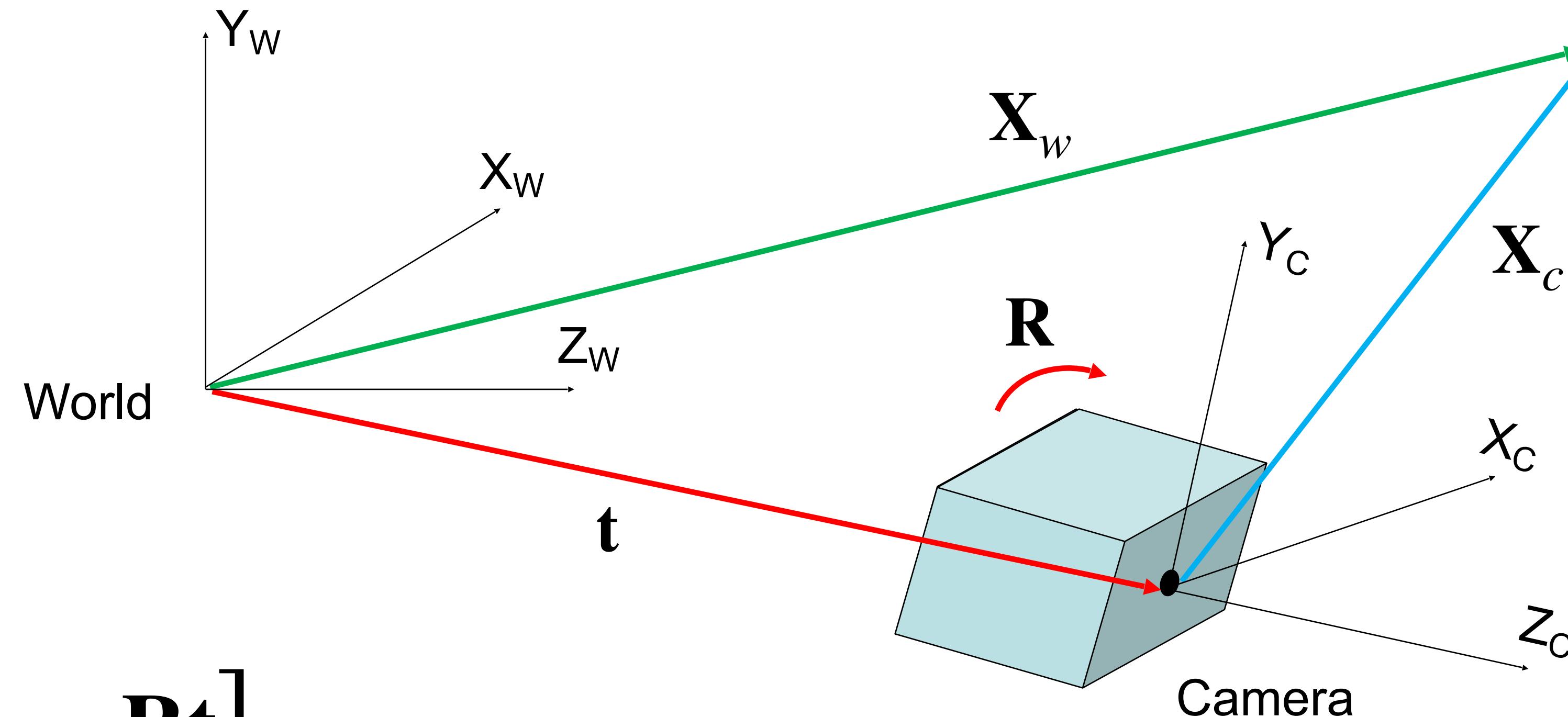


# Camera parameters



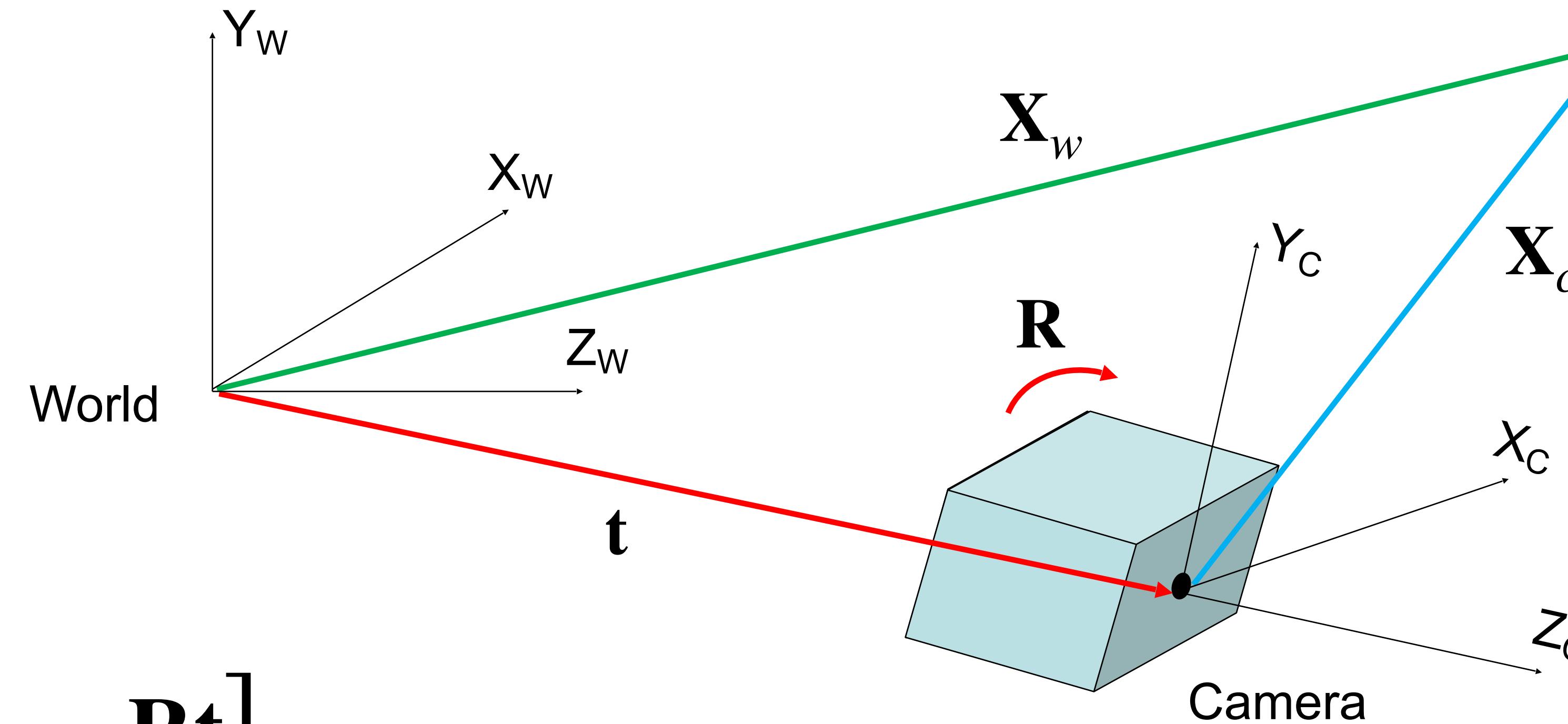
$$\tilde{\mathbf{X}}_c = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \tilde{\mathbf{X}}_w$$

# Cam2World vs. World2Cam extrinsic parameters



$$\tilde{\mathbf{X}}_c = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \tilde{\mathbf{X}}_w = \mathbf{C}^{W2C} \tilde{\mathbf{X}}_w$$

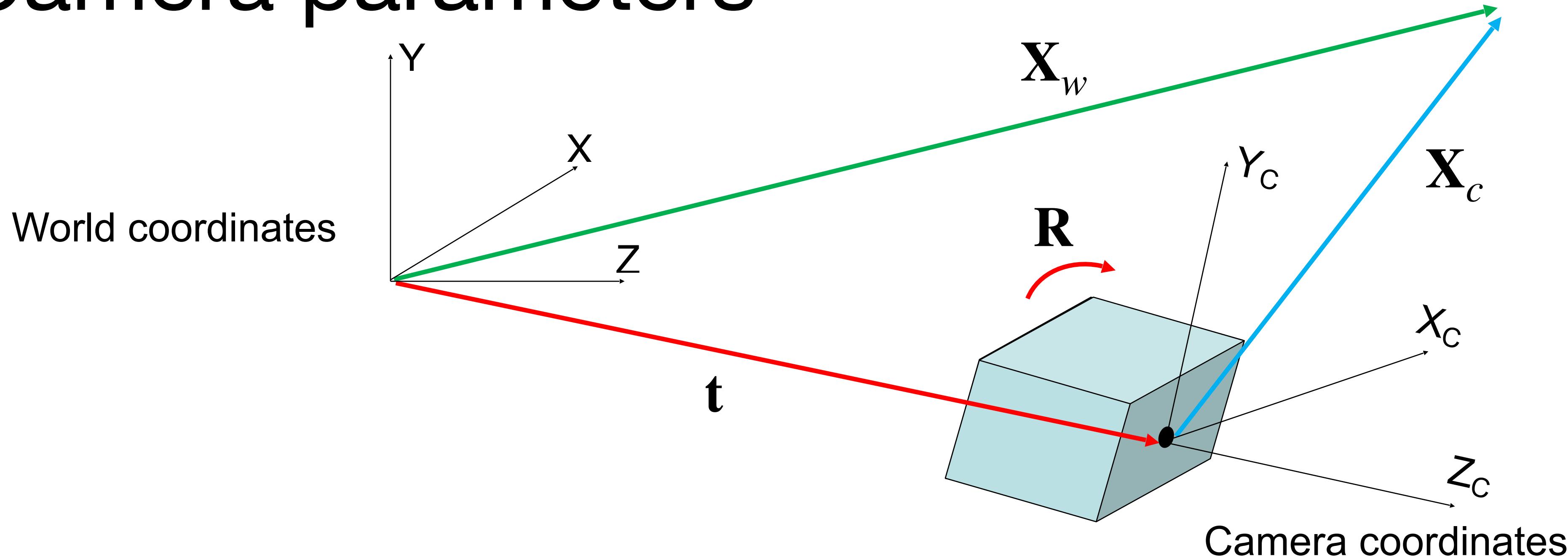
# Cam2World vs. World2Cam extrinsic parameters



$$\tilde{\mathbf{X}}_c = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \tilde{\mathbf{X}}_w = \mathbf{C}^{W2C} \tilde{\mathbf{X}}_w$$

$$\tilde{\mathbf{X}}_w = \begin{bmatrix} \mathbf{R}^T & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \tilde{\mathbf{X}}_c = (\mathbf{C}^{W2C})^{-1} \tilde{\mathbf{X}}_c = \mathbf{C}^{C2W} \tilde{\mathbf{X}}_c$$

# Camera parameters



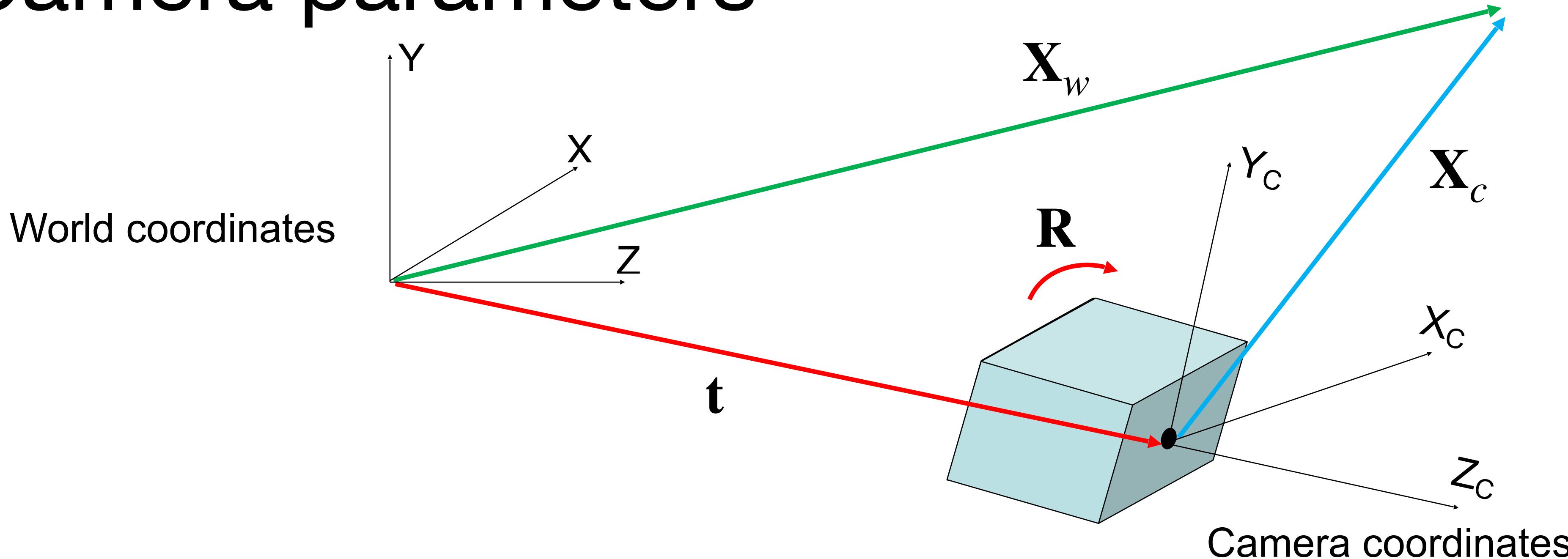
World coordinates to camera coordinates

$$\tilde{\mathbf{X}}_c = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \tilde{\mathbf{X}}_w$$

Camera coordinates to image coordinates

$$\tilde{\mathbf{x}} = \begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tilde{\mathbf{X}}_c$$

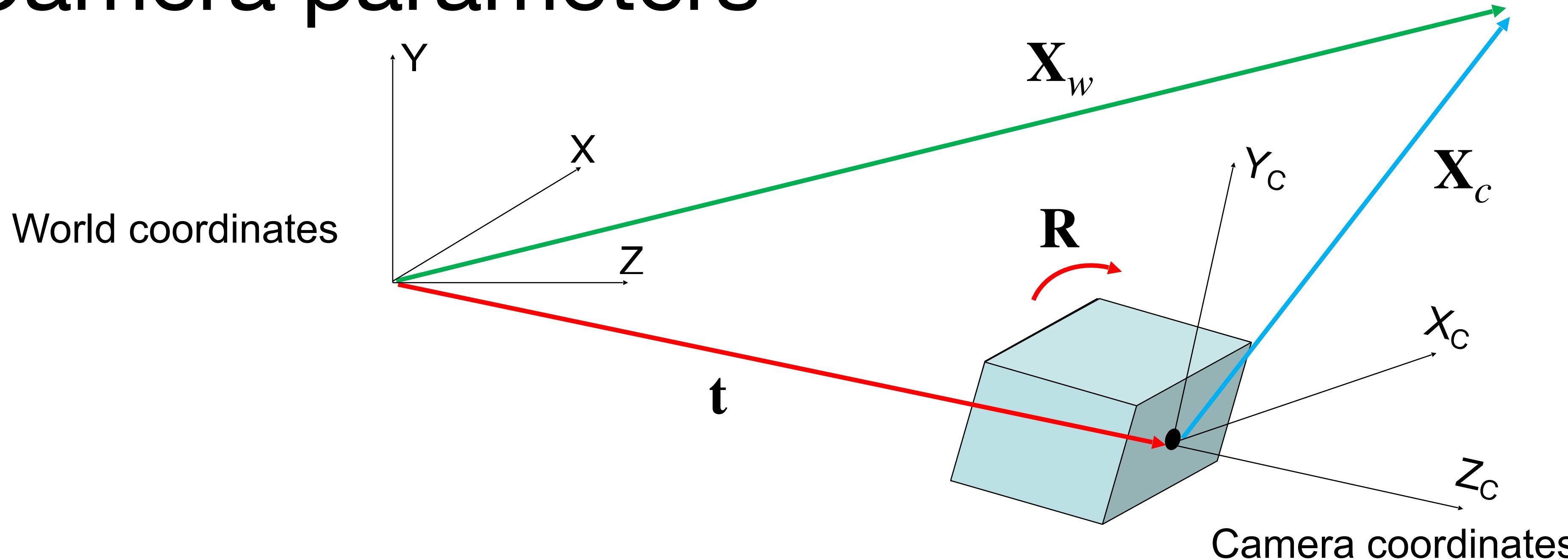
# Camera parameters



**World coordinates to image coordinates**

$$\tilde{\mathbf{x}} = \begin{bmatrix} ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{bmatrix} \tilde{\mathbf{X}}_w$$

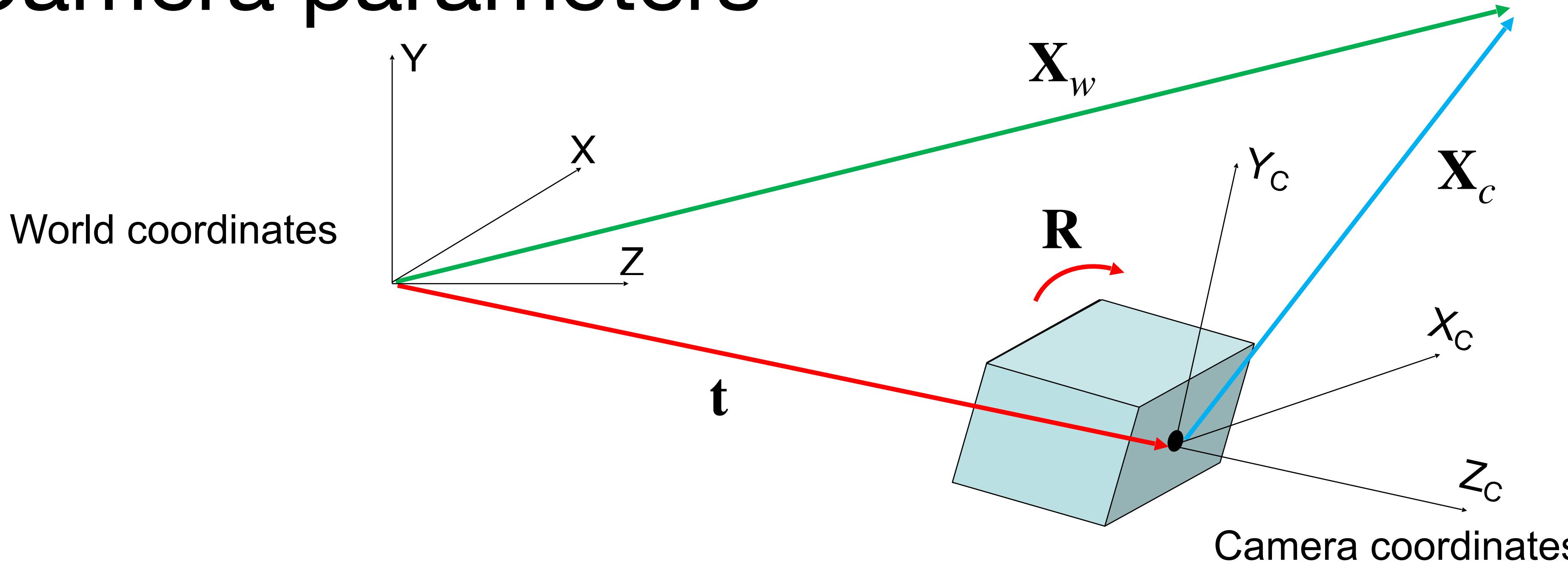
# Camera parameters



World coordinates to image coordinates

$$\tilde{\mathbf{x}} = \begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R} & -\mathbf{R}\mathbf{t} \\ 0 & 1 \end{bmatrix} \tilde{\mathbf{X}}_w$$

# Camera parameters



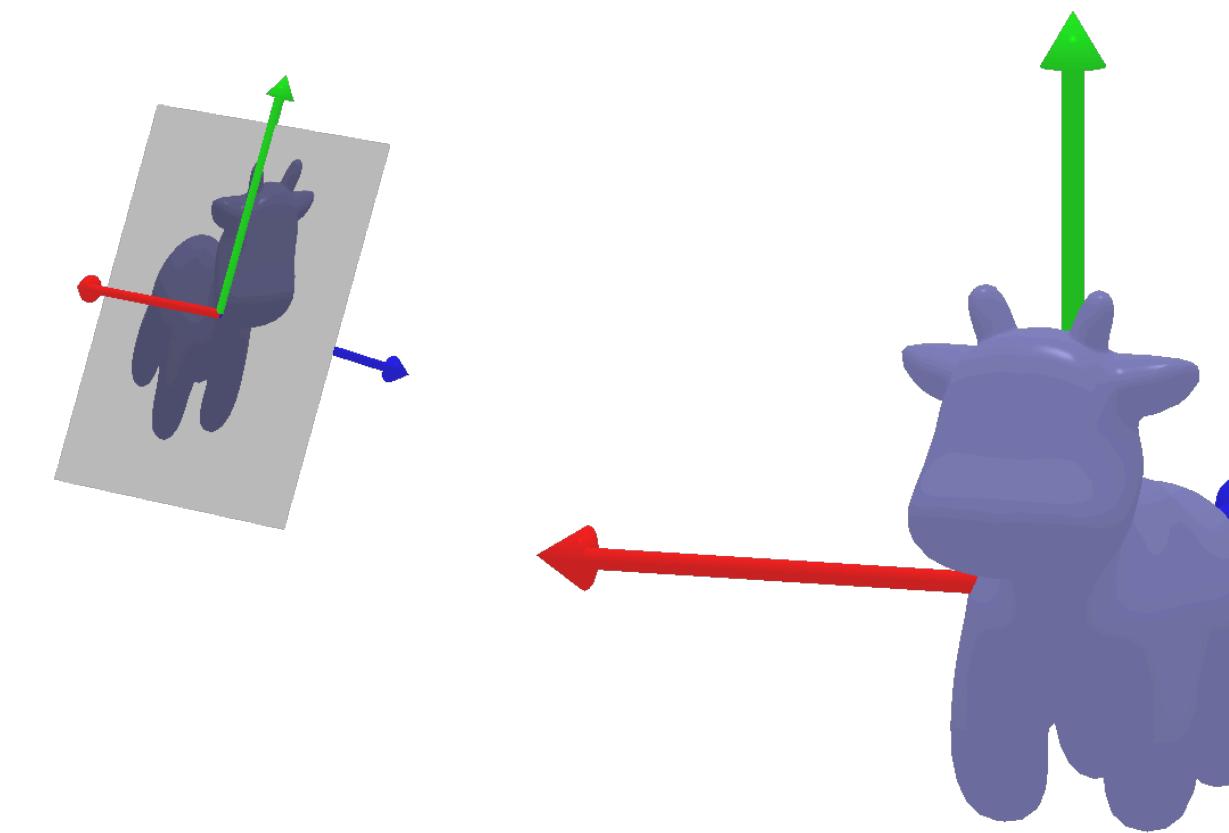
**World coordinates to image coordinates**

$$\tilde{\mathbf{x}} = \begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R} & -\mathbf{R}\mathbf{t} \\ 0 & 1 \end{bmatrix} \tilde{\mathbf{X}}_w = \mathbf{P}\tilde{\mathbf{X}}_w$$

**Intrinsic parameters**      **Extrinsic parameters**

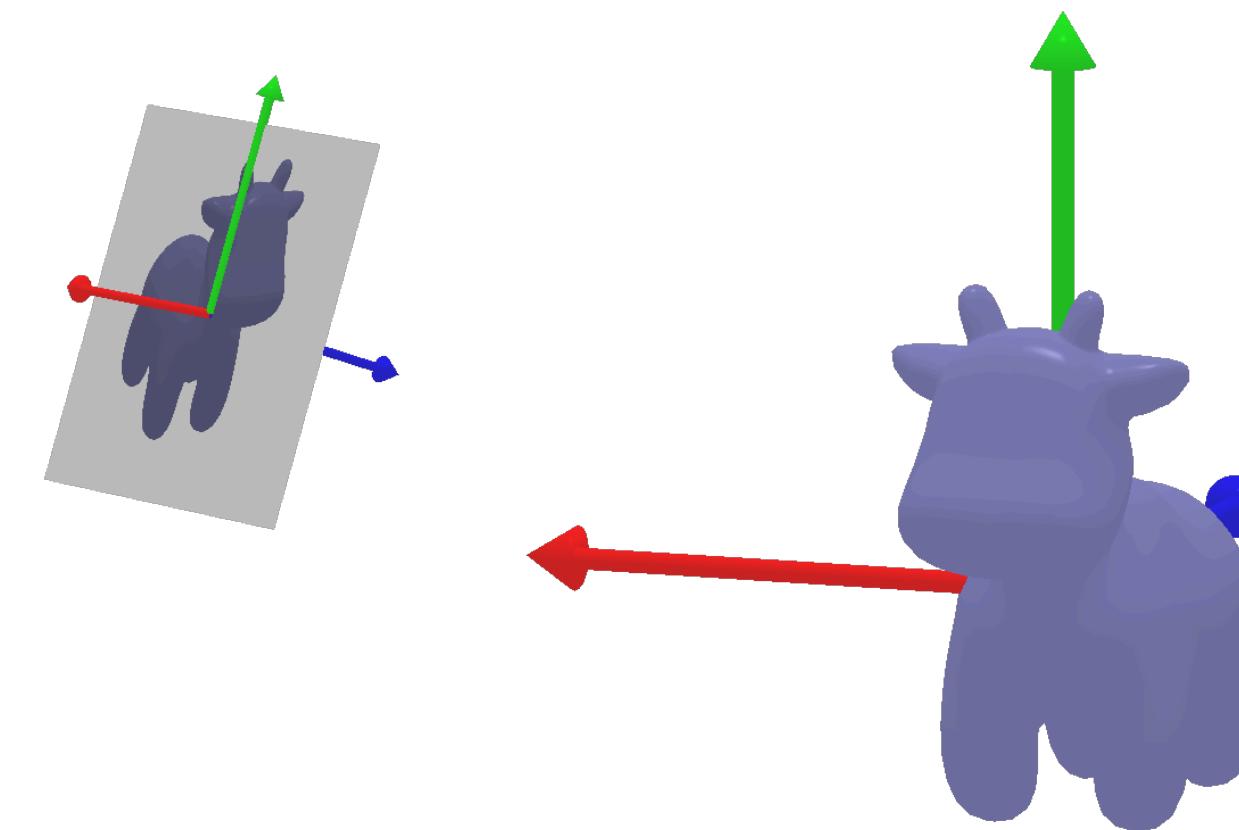
# World2Camera Transformations: Exercise

$$X_c = \begin{bmatrix} R & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} X_w; \quad \mathbf{t} = -R\tilde{\mathbf{C}}$$



# World2Camera Transformations: Exercise

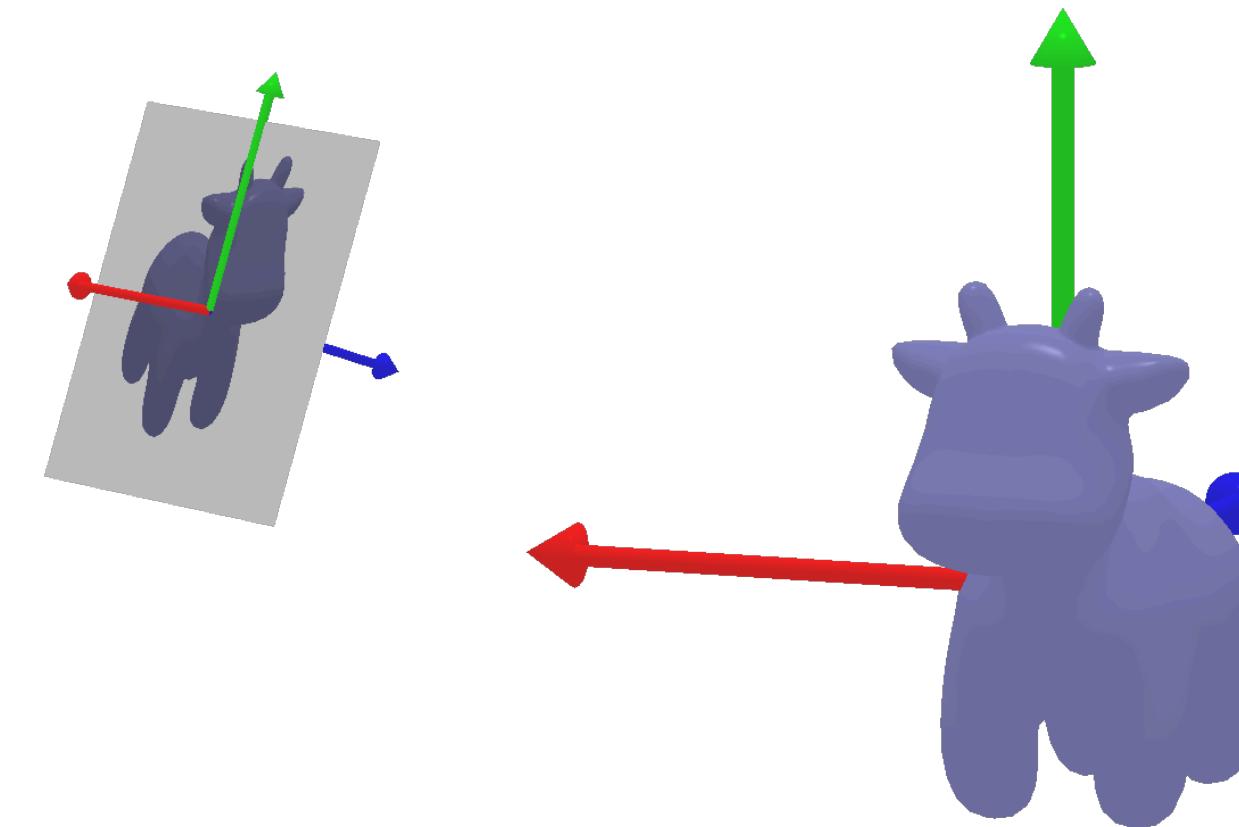
$$X_c = \begin{bmatrix} R & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} X_w; \quad \mathbf{t} = -R\tilde{\mathbf{C}}$$



What are  $R$ ,  $\mathbf{t}$  for an upright (world and camera y-directions align) origin-facing camera 2m away from origin located at  $(0,0,-2)$ ?

# World2Camera Transformations: Exercise

$$X_c = \begin{bmatrix} R & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} X_w; \quad \mathbf{t} = -R\tilde{\mathbf{C}}$$

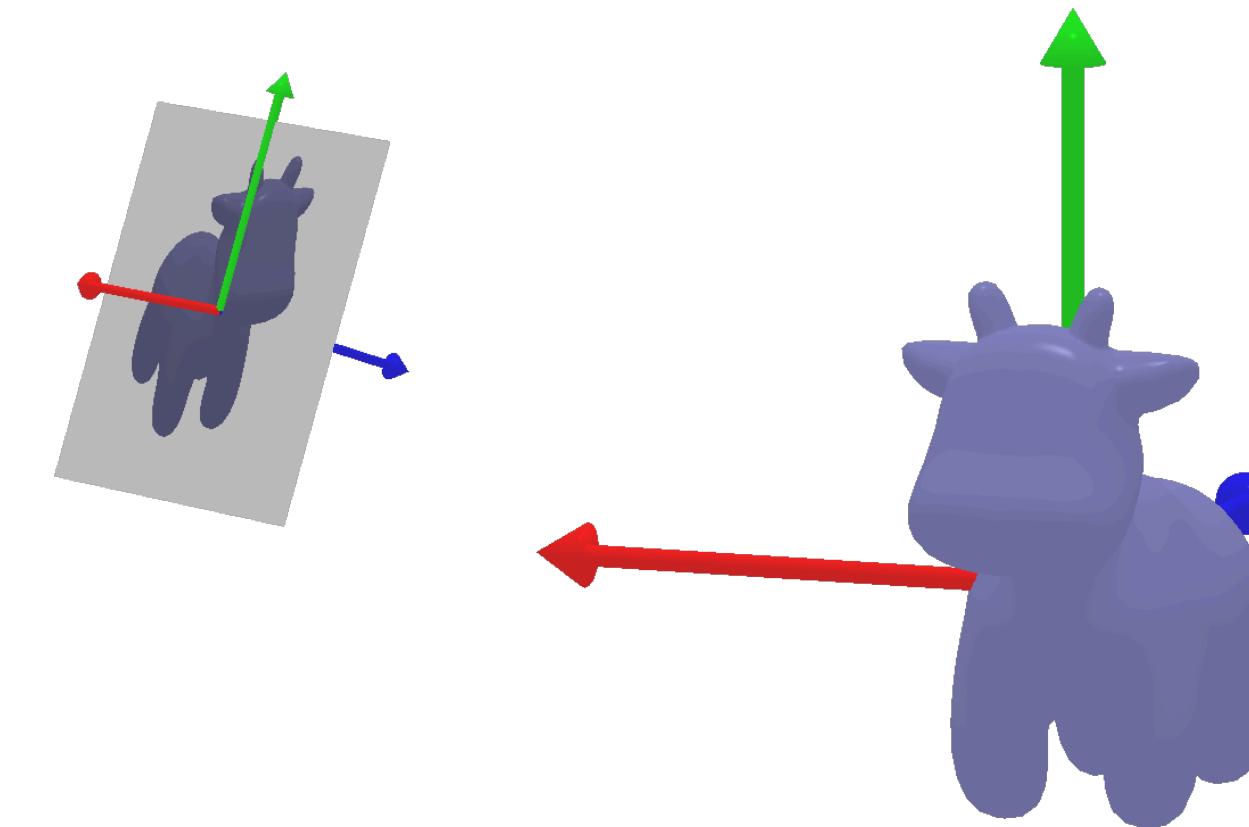


What are  $R$ ,  $\mathbf{t}$  for an upright (world and camera y-directions align) origin-facing camera 2m away from origin located at  $(0,0,-2)$ ?

$$R = \mathbf{I}; \quad \mathbf{t} = \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix}$$

# World2Camera Transformations: Exercise

$$X_c = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} X_w; \quad t = -R\tilde{C}$$



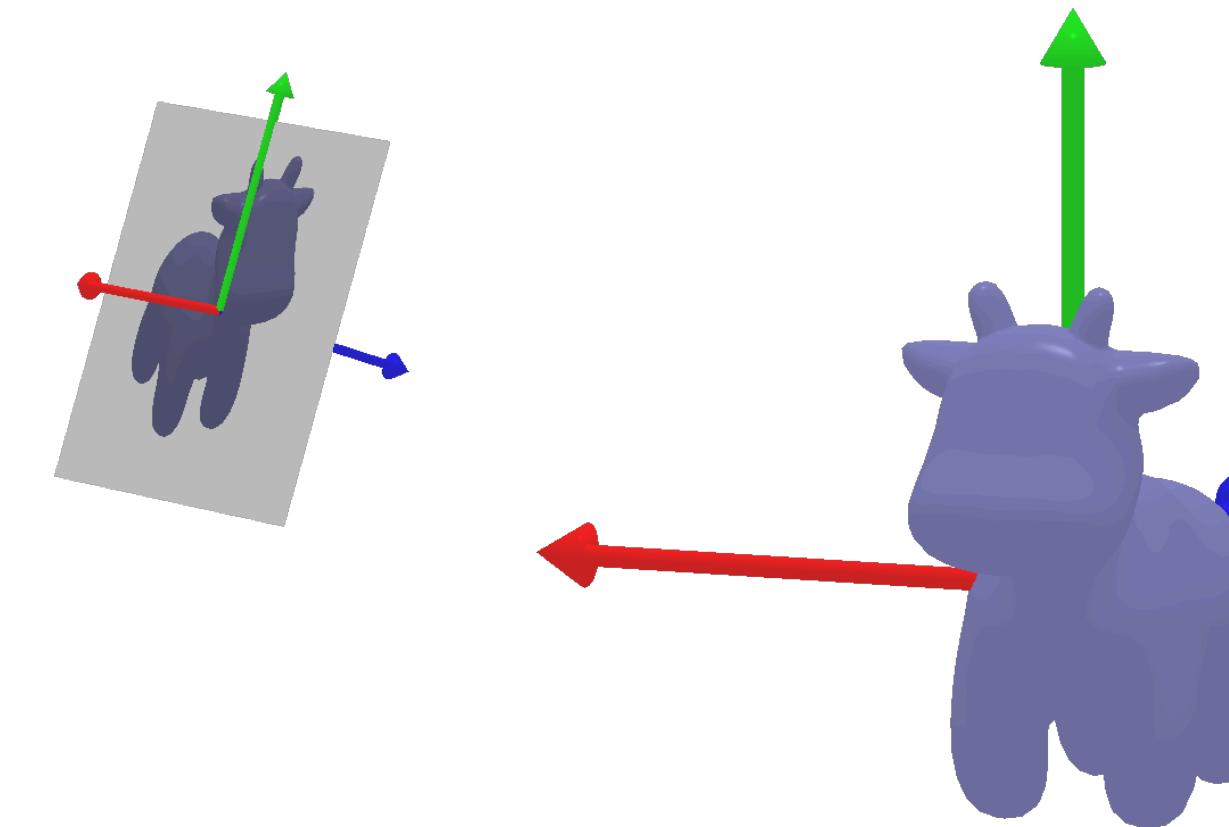
What are  $R$ ,  $t$  for an upright (world and camera y-directions align) origin-facing camera 2m away from origin located at  $(0,0,-2)$ ?

$$R = I; \quad t = \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix}$$

What are  $R$ ,  $t$  for an upright (world and camera y-directions align) origin-facing camera 2m away from origin located at  $(-2,0,0)$ ?

# World2Camera Transformations: Exercise

$$X_c = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} X_w; \quad t = -R\tilde{C}$$



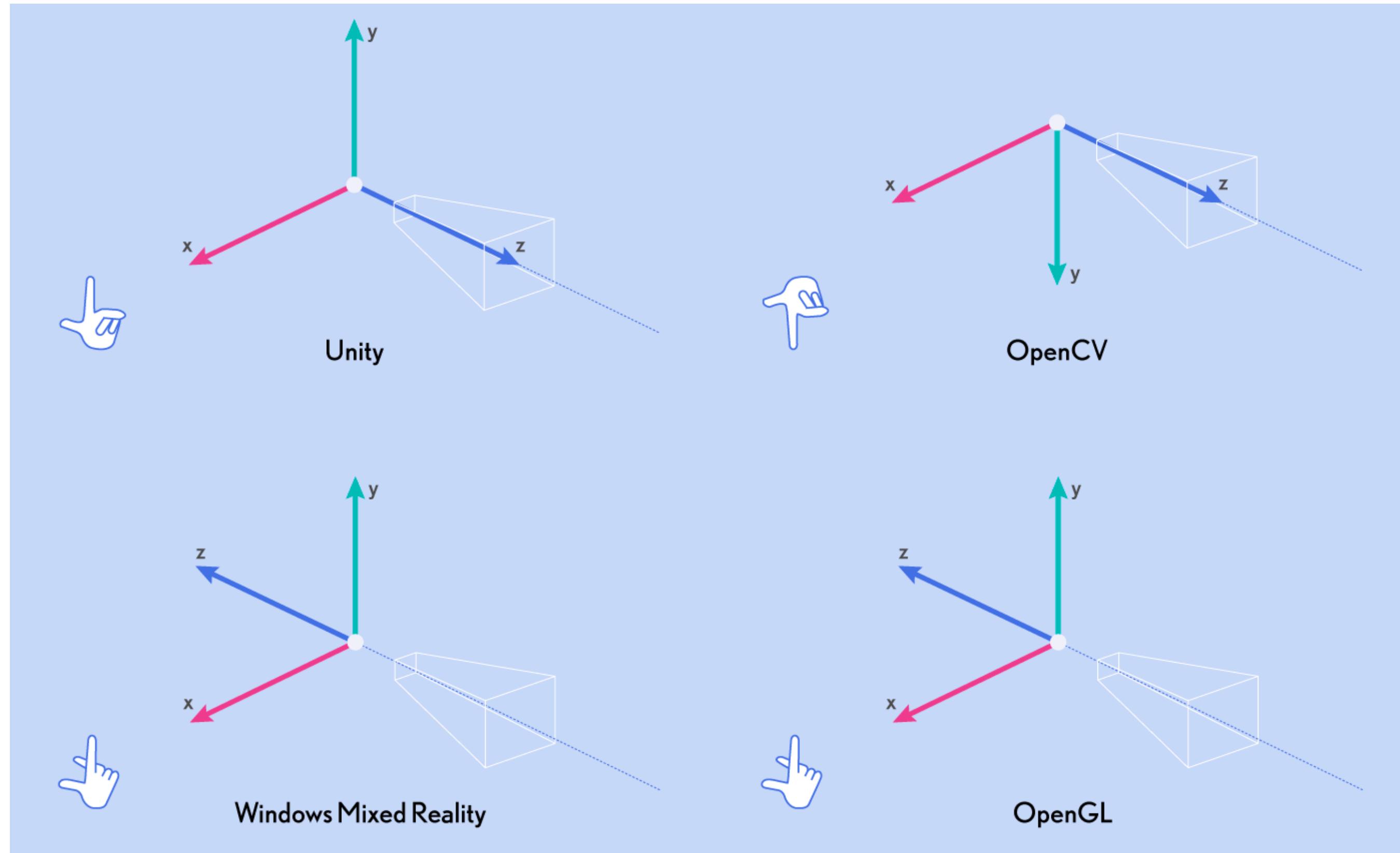
What are  $R$ ,  $t$  for an upright (world and camera y-directions align) origin-facing camera 2m away from origin located at  $(0,0,-2)$ ?

$$R = I; \quad t = \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix}$$

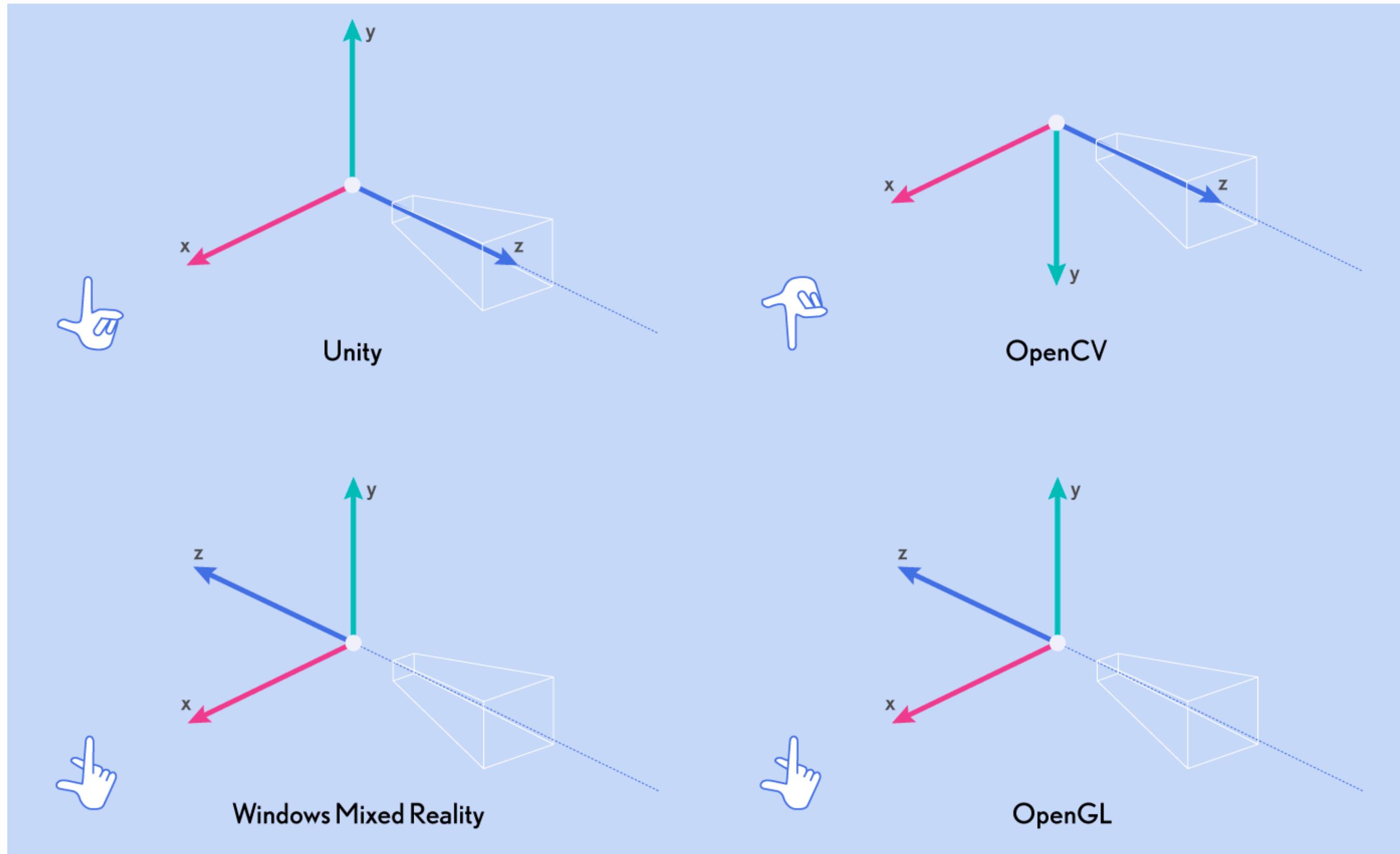
What are  $R$ ,  $t$  for an upright (world and camera y-directions align) origin-facing camera 2m away from origin located at  $(-2,0,0)$ ?

$$R = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}; \quad t = \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix}$$

# Beware of Conventions

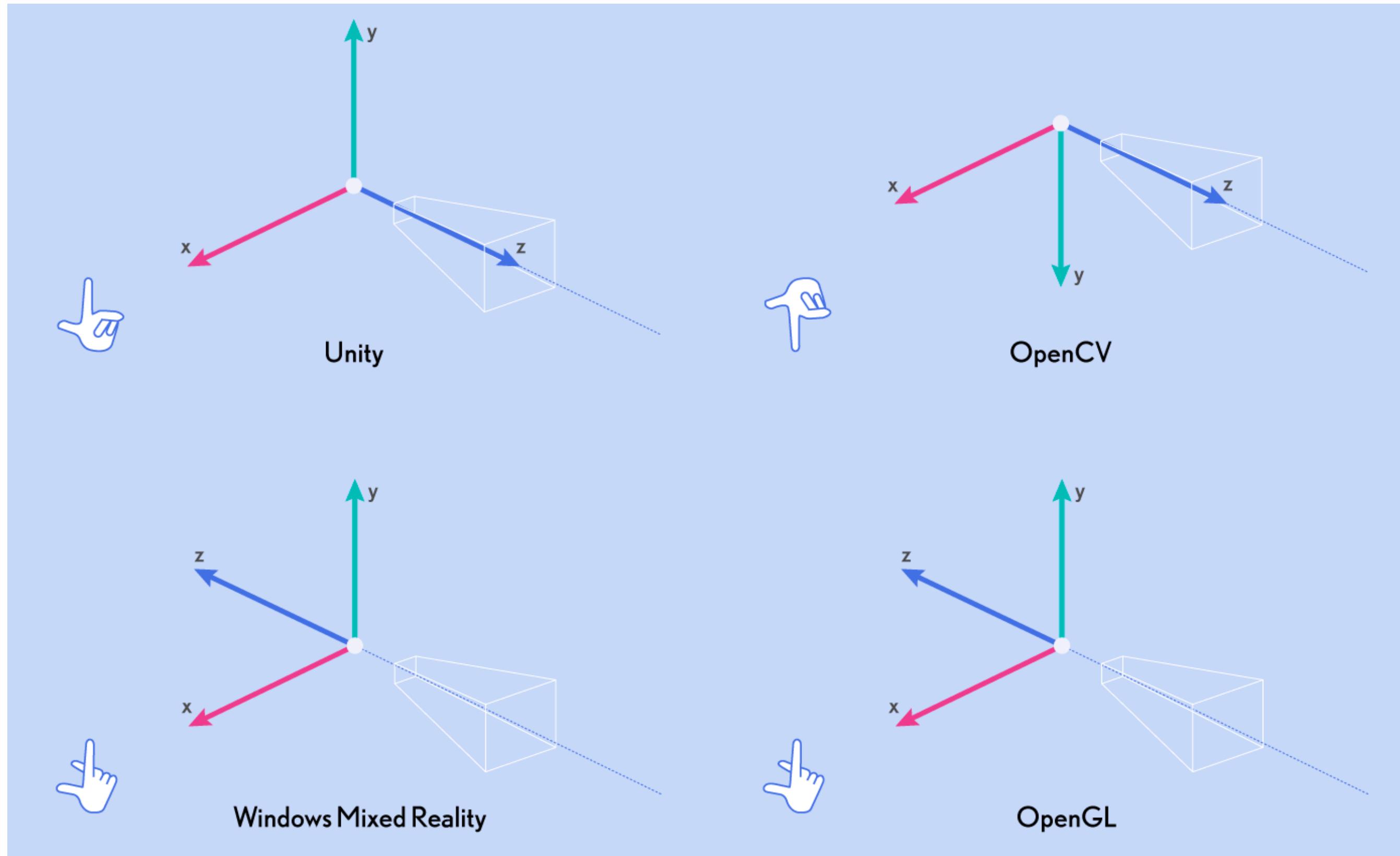


# Beware of Conventions



X=left or right? Y=up or down? etc..

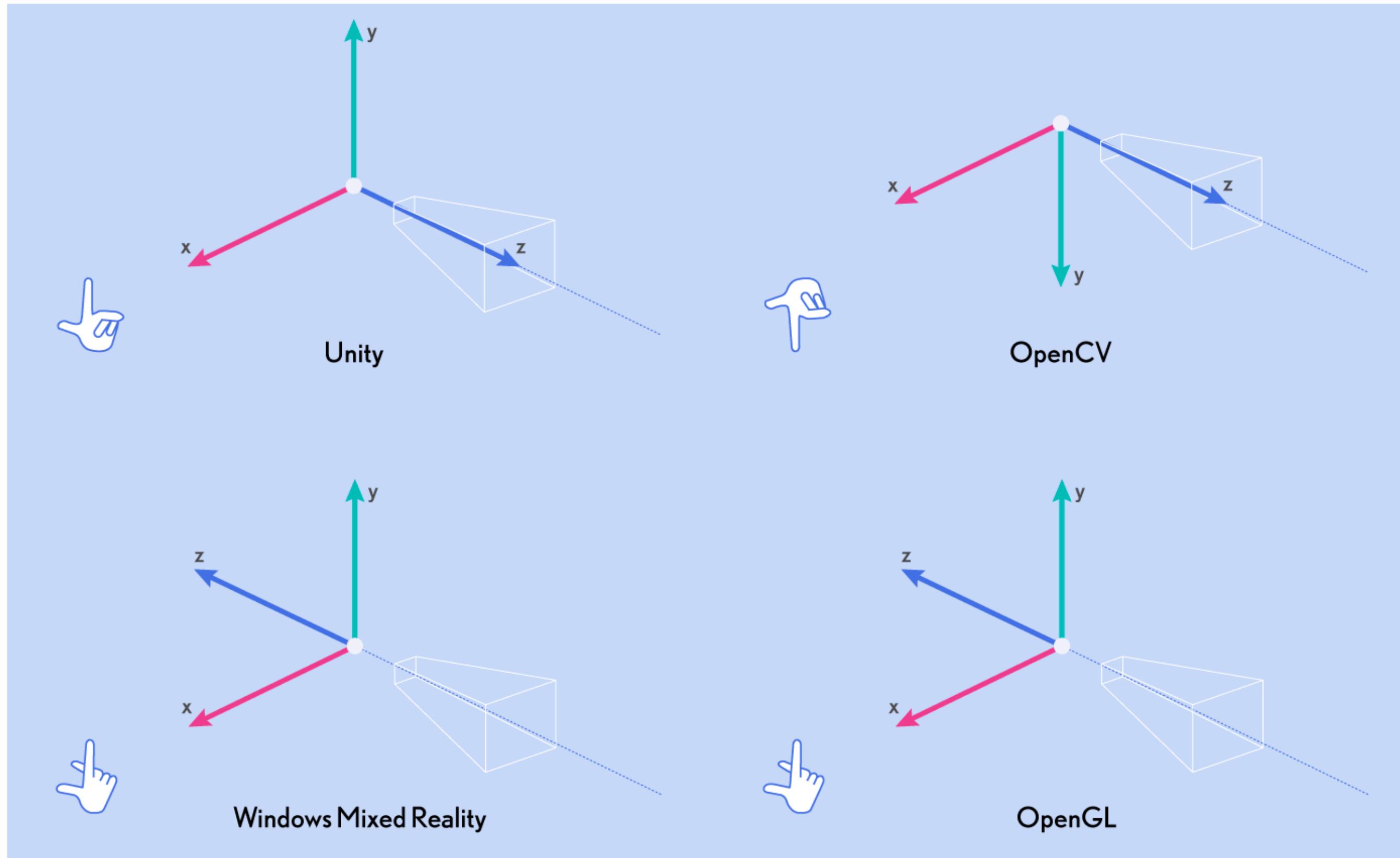
# Beware of Conventions



X=left or right? Y=up or down? etc..

If you externally obtain transformation matrices (e.g. using someone else's code), make sure of convention compatibility (one of the *most common* sources of bugs!)

# Beware of Conventions



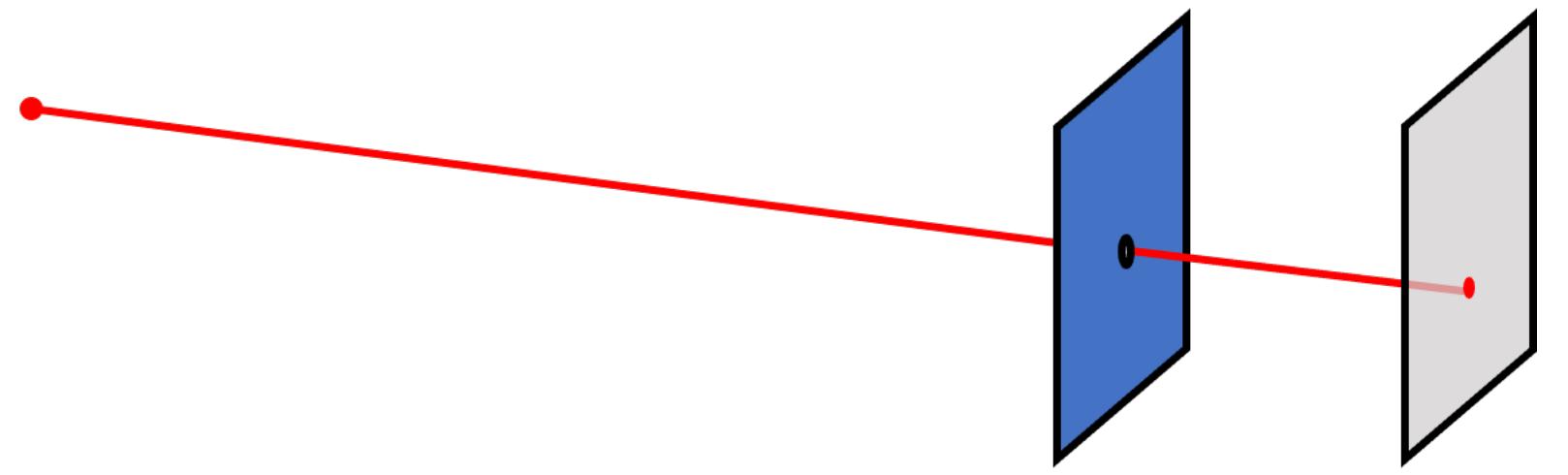
X=left or right? Y=up or down? etc..

If you externally obtain transformation matrices (e.g. using someone else's code), make sure of convention compatibility (one of the *most common* sources of bugs!)

read <https://pytorch3d.org/docs/cameras>

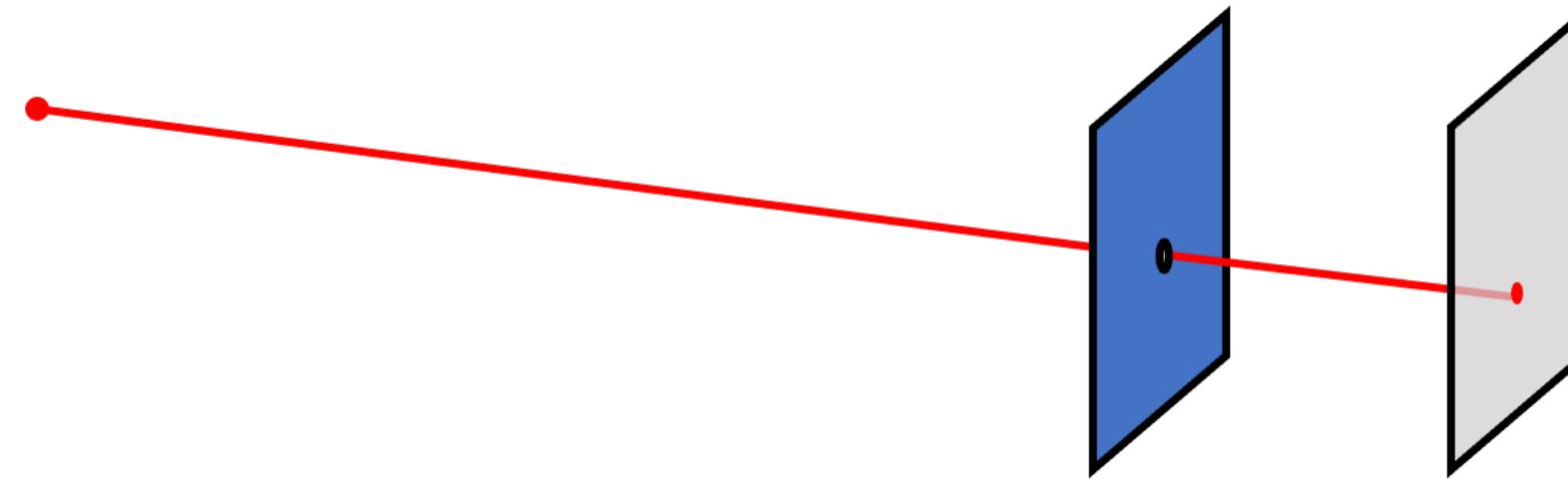
# Summary

# Summary

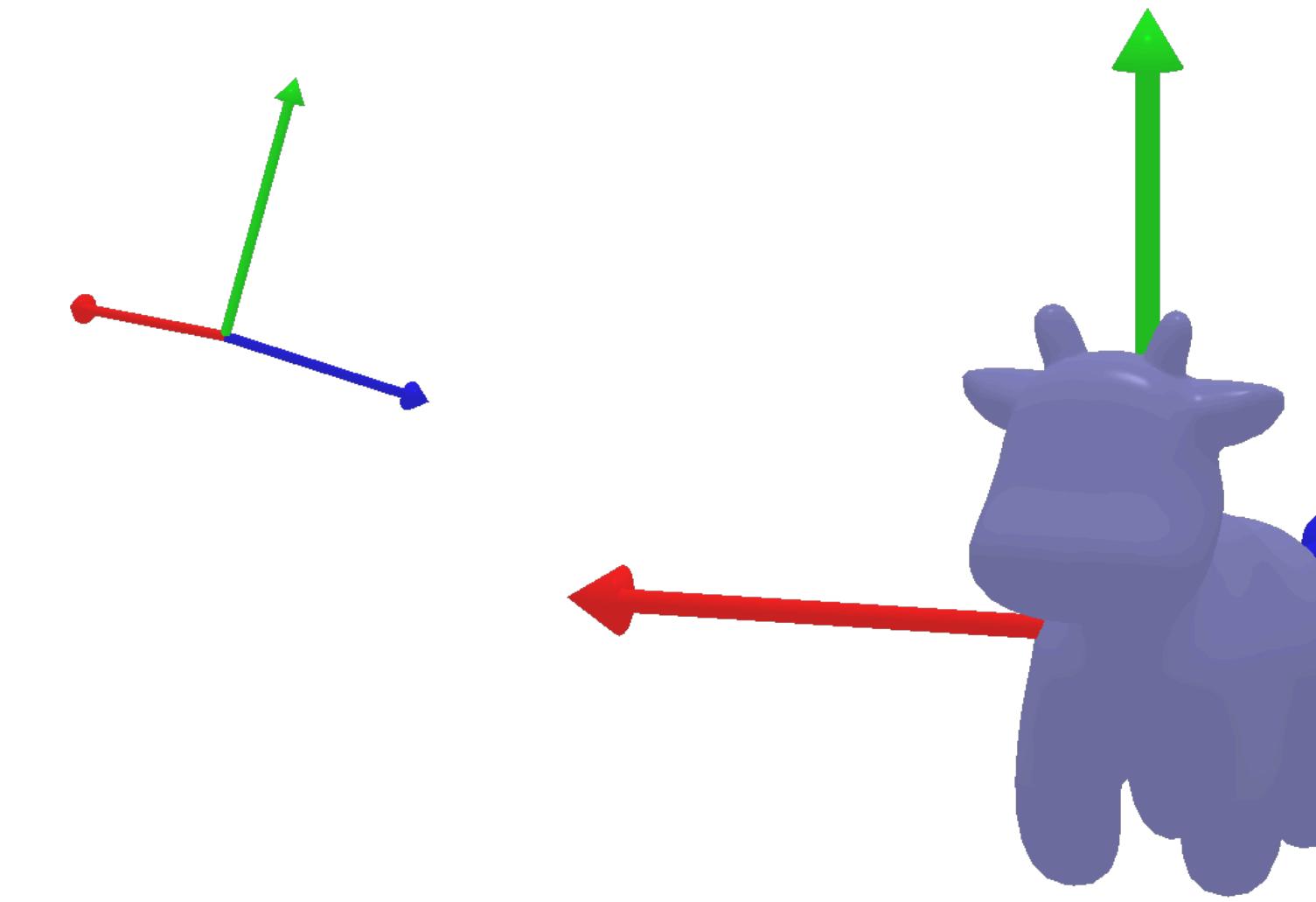


***Projection*** – associating rays to points in a plane

# Summary



***Projection*** — associating rays to points in a plane



***Camera Transformation*** — from world to camera coordinates

# Questions?

# Vision systems

One camera



Two cameras



N cameras



# Let's consider two eyes

One camera



Two cameras



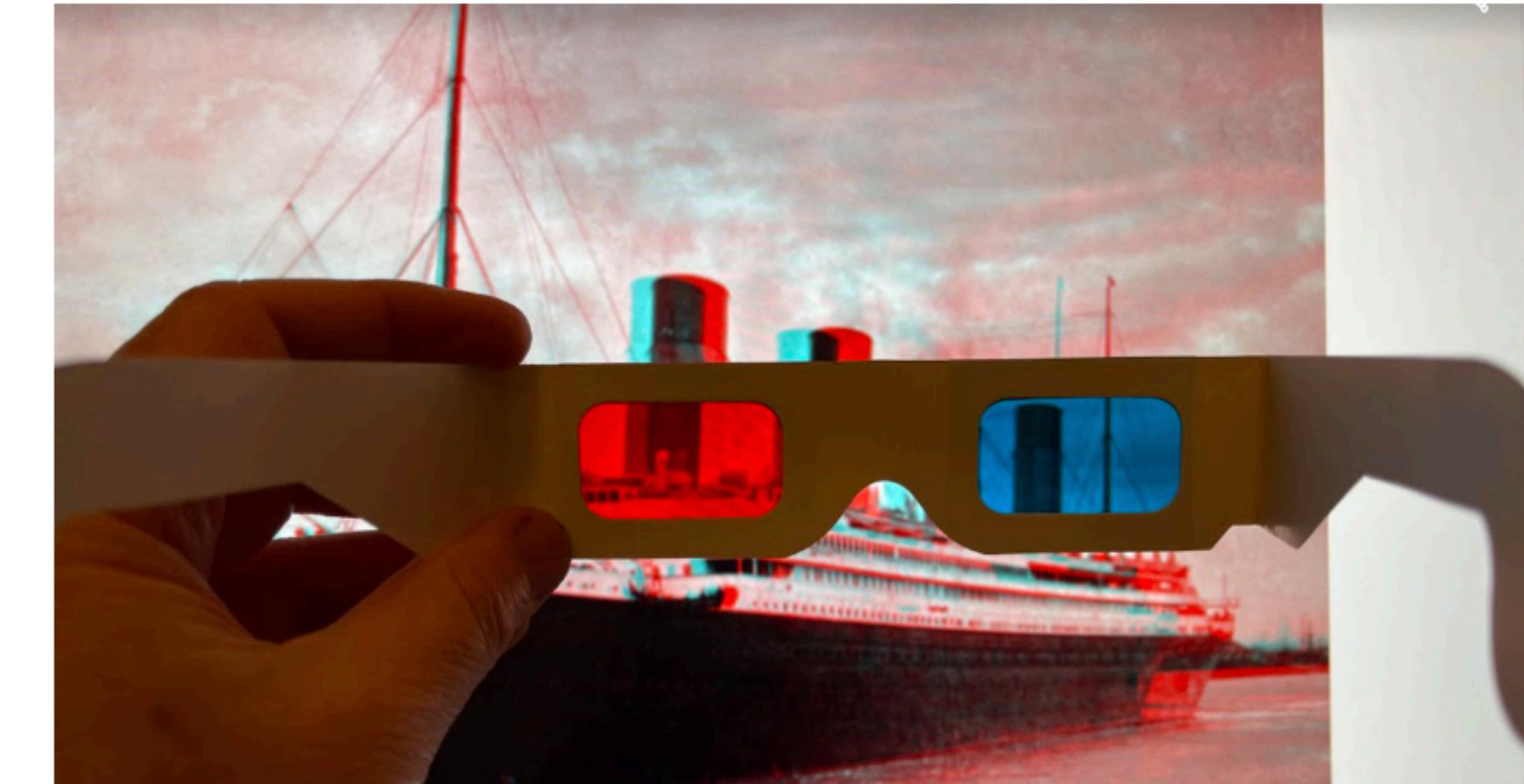
N cameras



# Stereo images of the Titanic



(a)



(b)

Figure 1.1: (a) Stereo anaglyph of the ocean liner, the Titanic [McManus2022]. The red image shows the right eye's view, and cyan the left eye's view. When viewed through stereo red/cyan stereo glasses, as in (b), the cyan contrast appears in the left eye image and the red variations appear to the right eye, creating a the perception of 3d.

# Stereoscope



View of [Boston](#), c. 1860; an early stereoscopic card for viewing a scene from nature

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](#)

Public Domain

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

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.

More details

CC BY-SA 4.0

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

Created: 1 July 2014

# Depth without objects

Random dot stereograms (Bela Julesz)



# Depth without objects

Random dot stereograms (Bela Julesz)

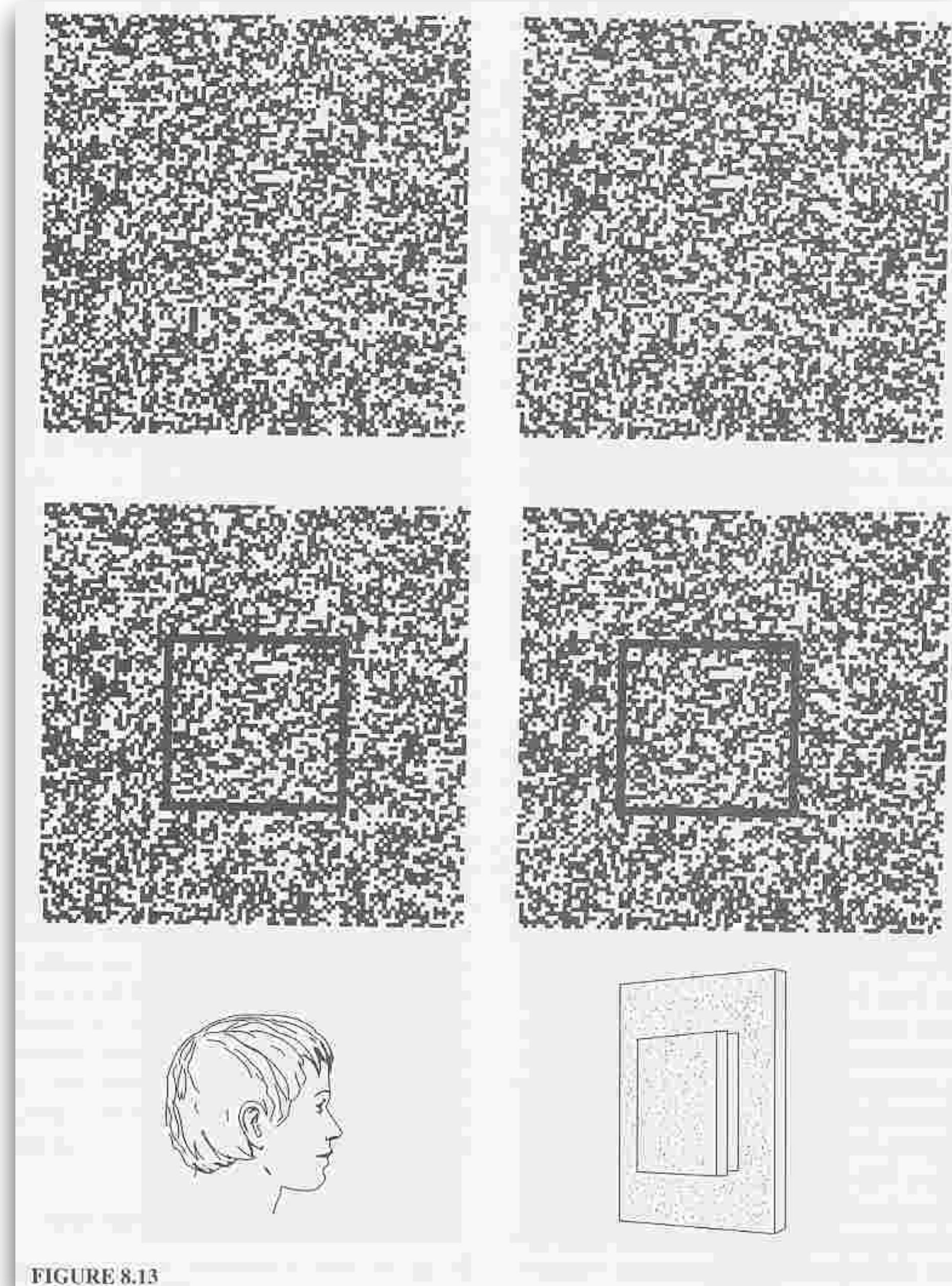


FIGURE 8.13

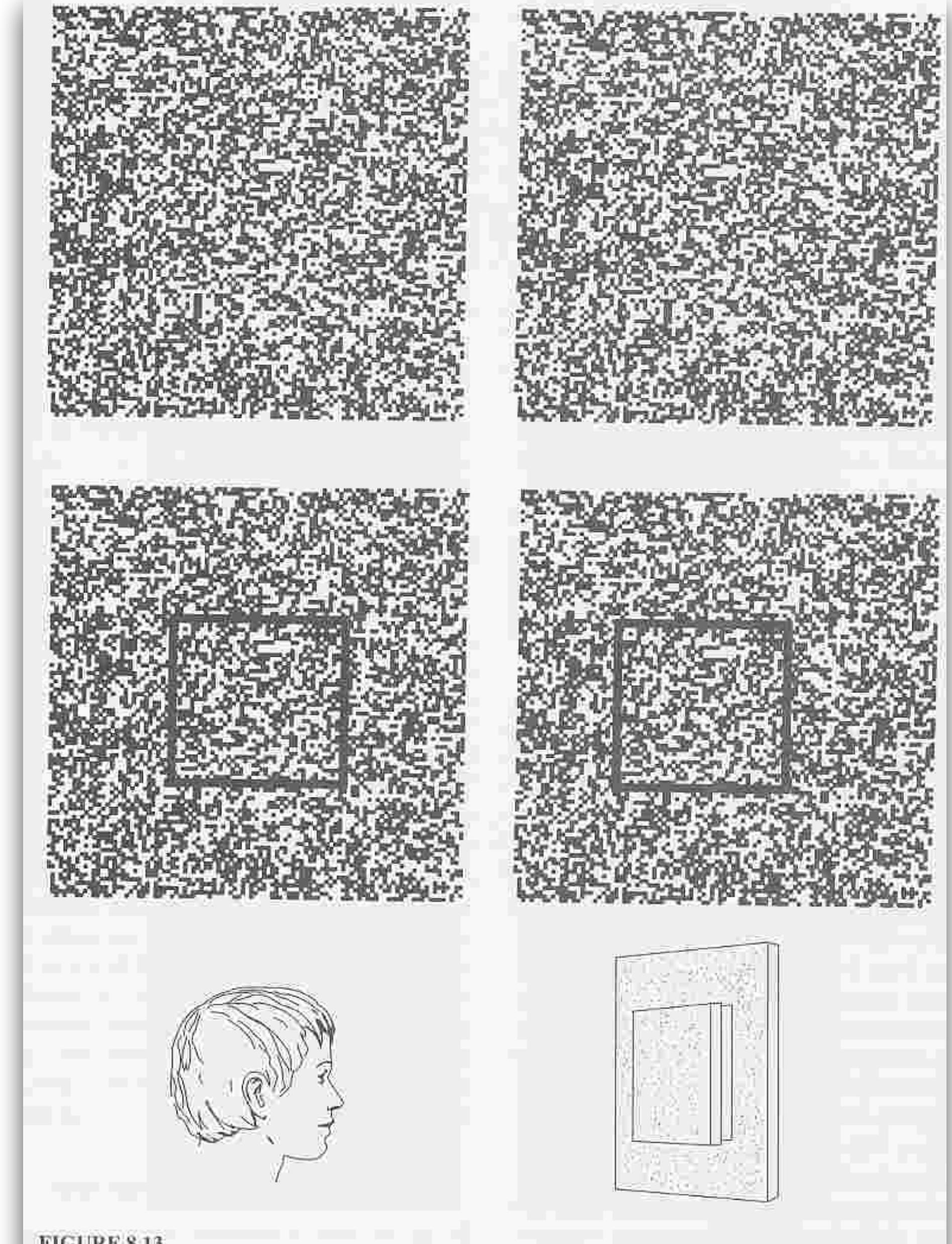
# Depth without objects

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

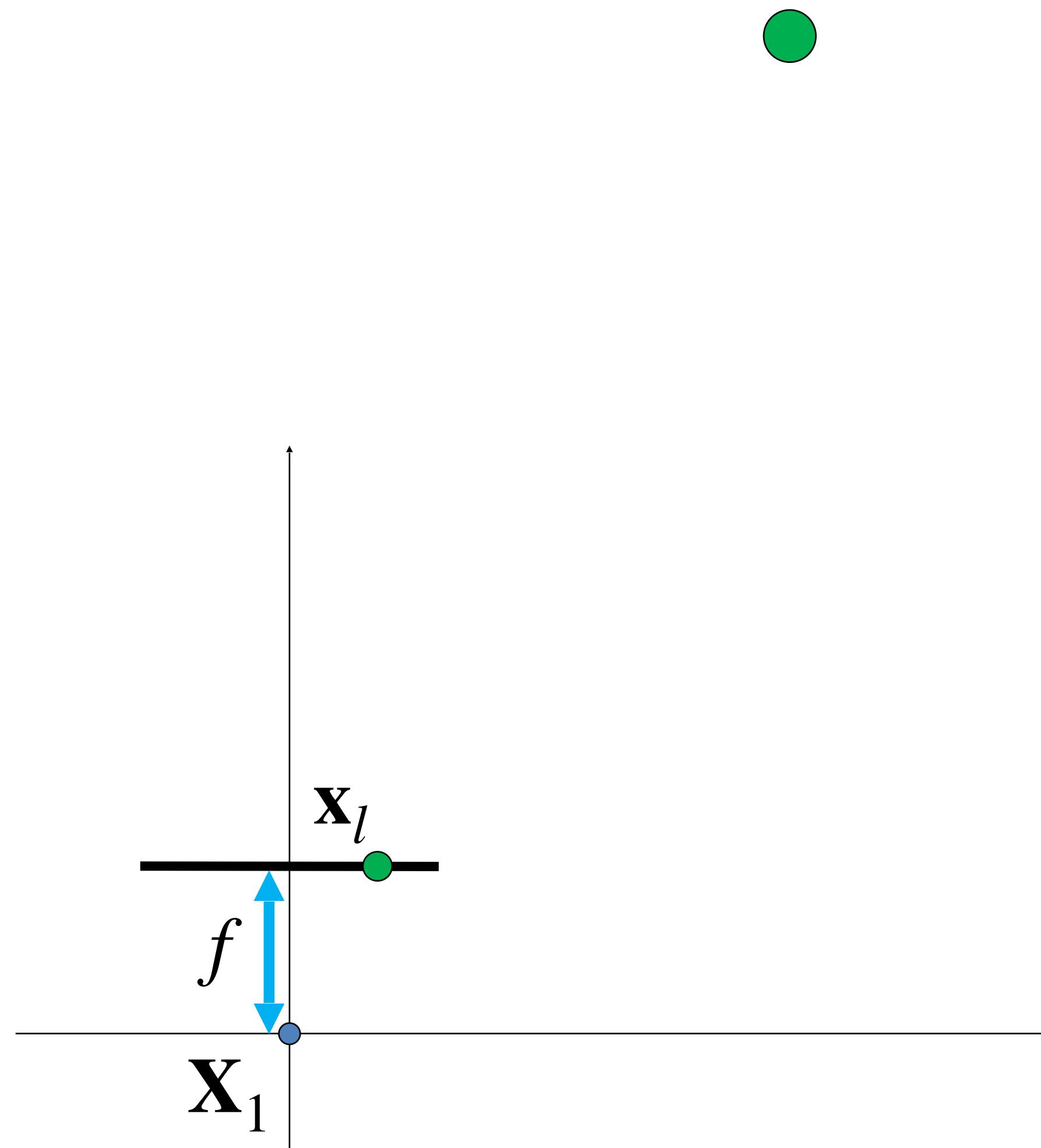
1	0	1	0	1	0	0	1	0	1
1	0	0	1	0	1	0	1	0	0
0	0	1	1	0	1	1	0	1	0
0	1	0	4	A	S	S	X	0	1
1	1	1	8	A	B	A	Y	0	1
0	0	1	4	A	S	A	Y	1	0
1	1	1	8	B	A	S	X	0	1
1	0	0	1	1	0	1	1	0	1
1	1	0	0	1	1	0	1	1	1
0	1	0	0	0	1	1	1	1	0



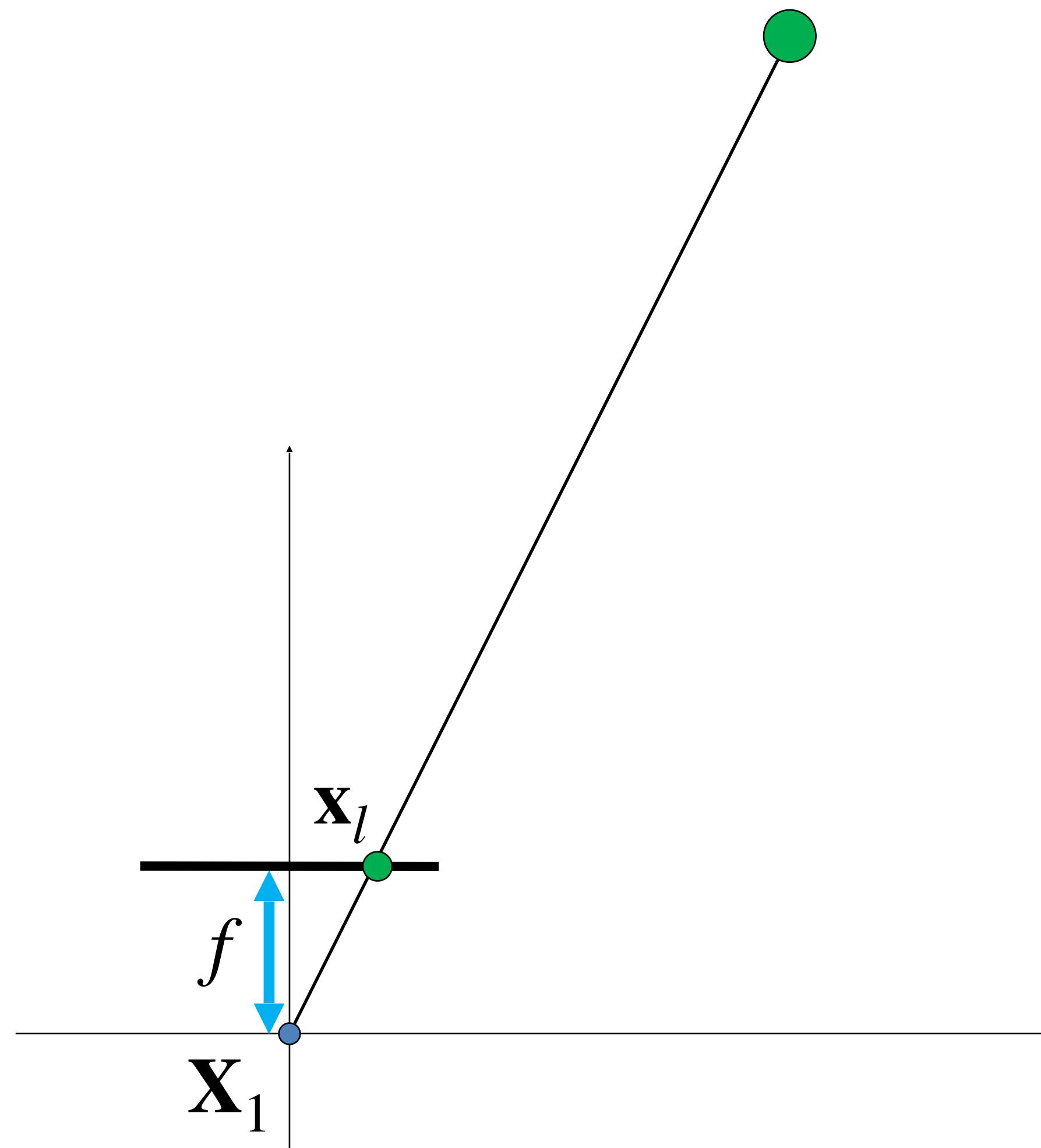
Julesz, 1971

FIGURE 8.13

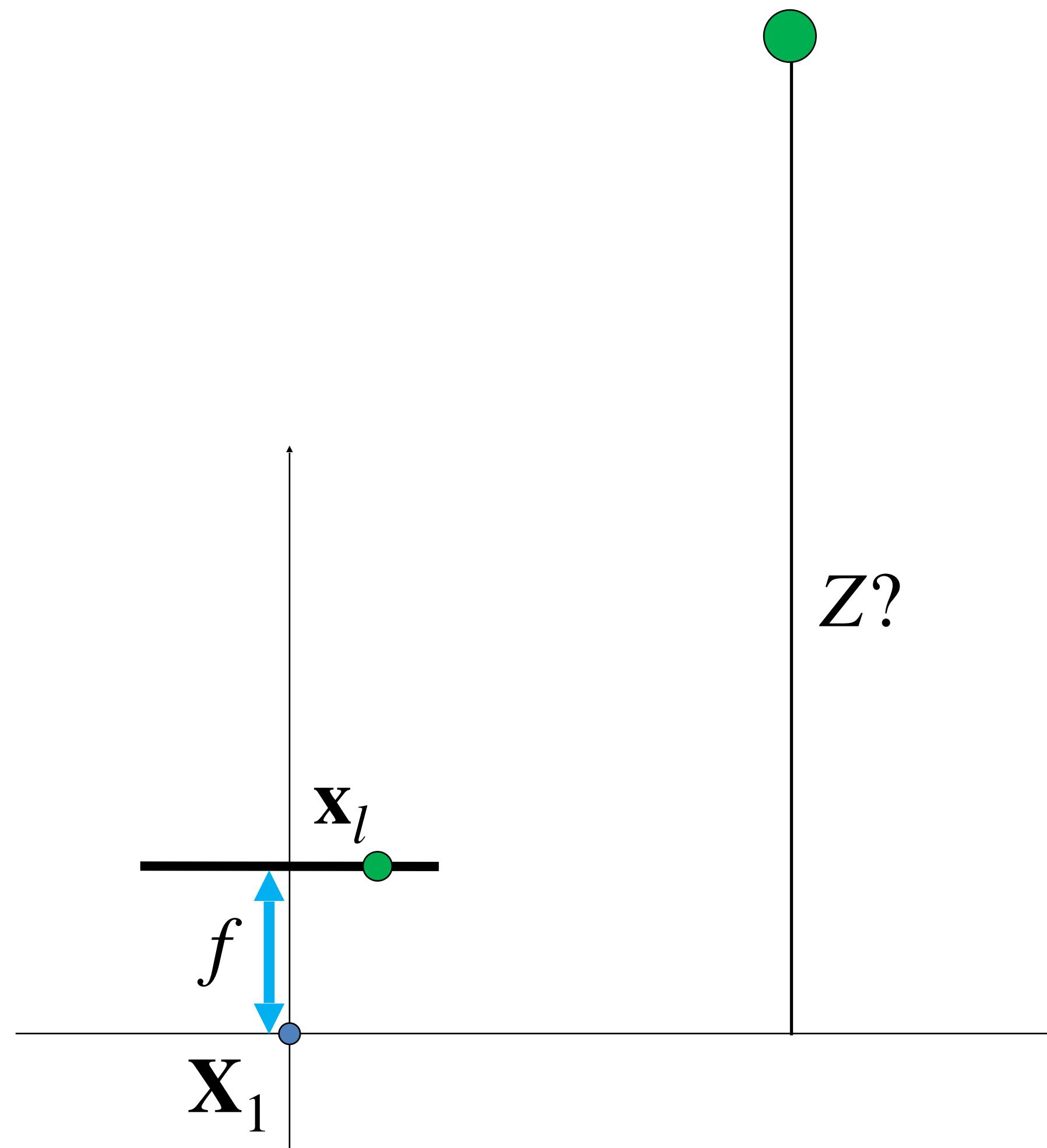
# Geometry for a simple stereo system



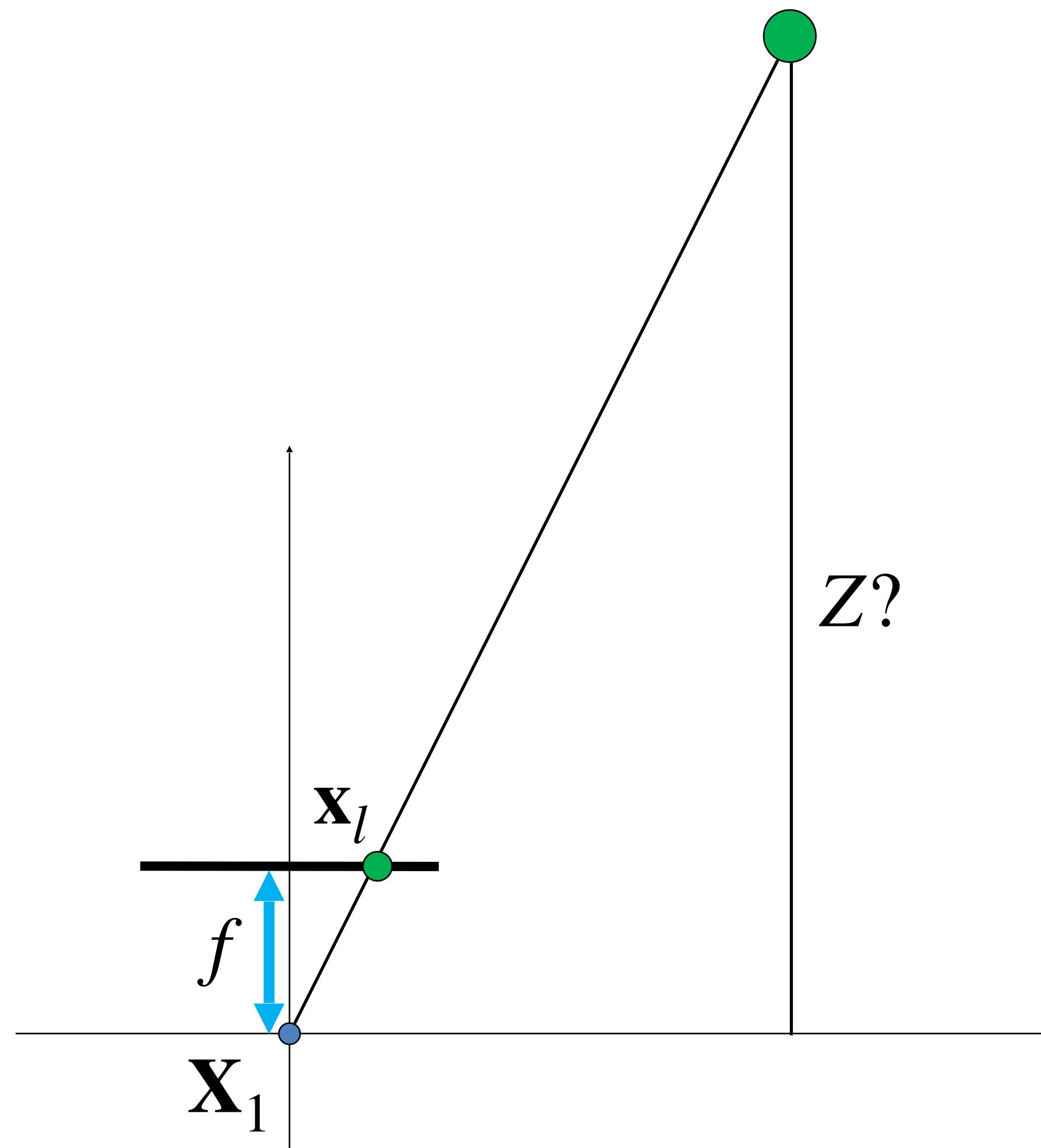
# Geometry for a simple stereo system



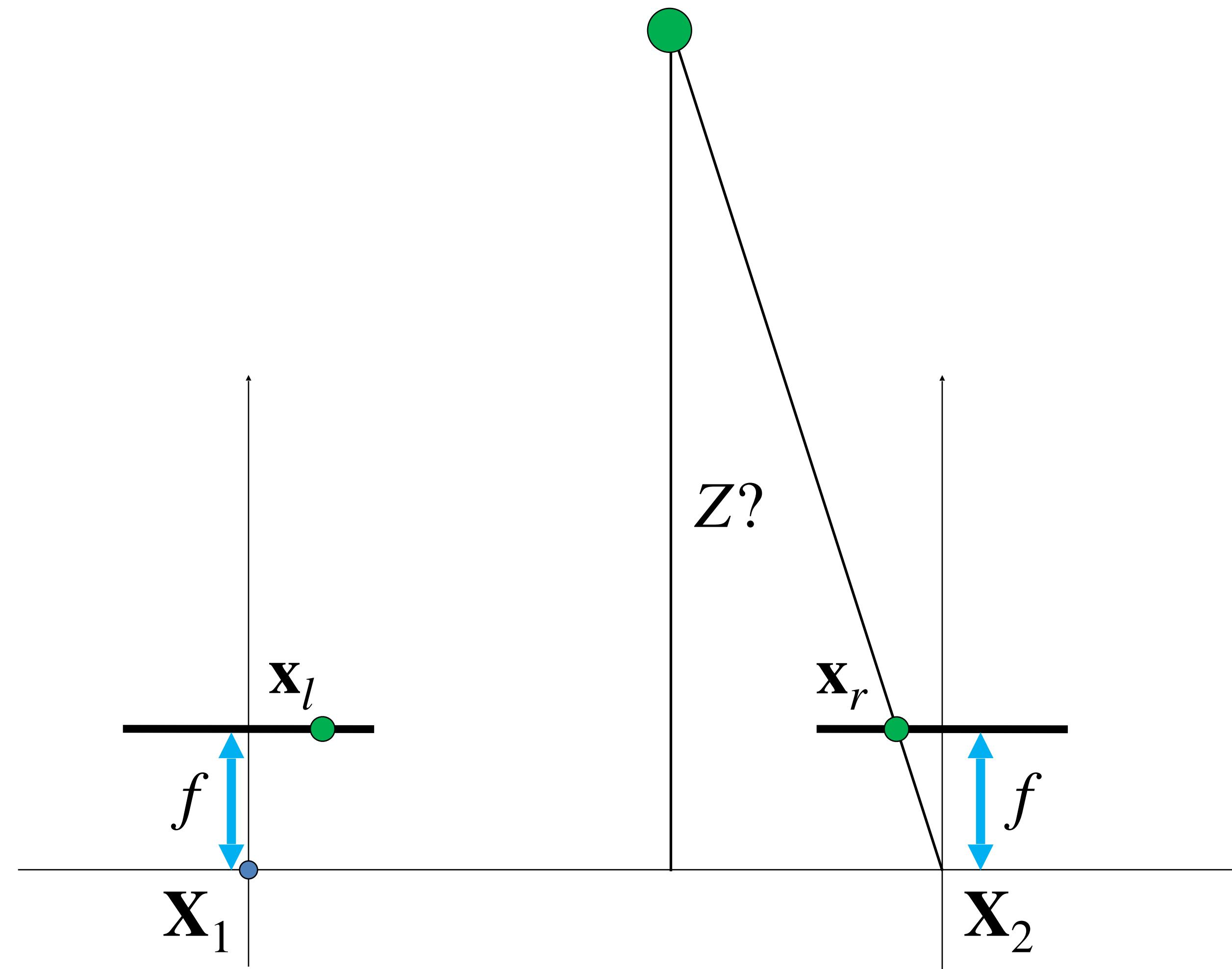
# Geometry for a simple stereo system



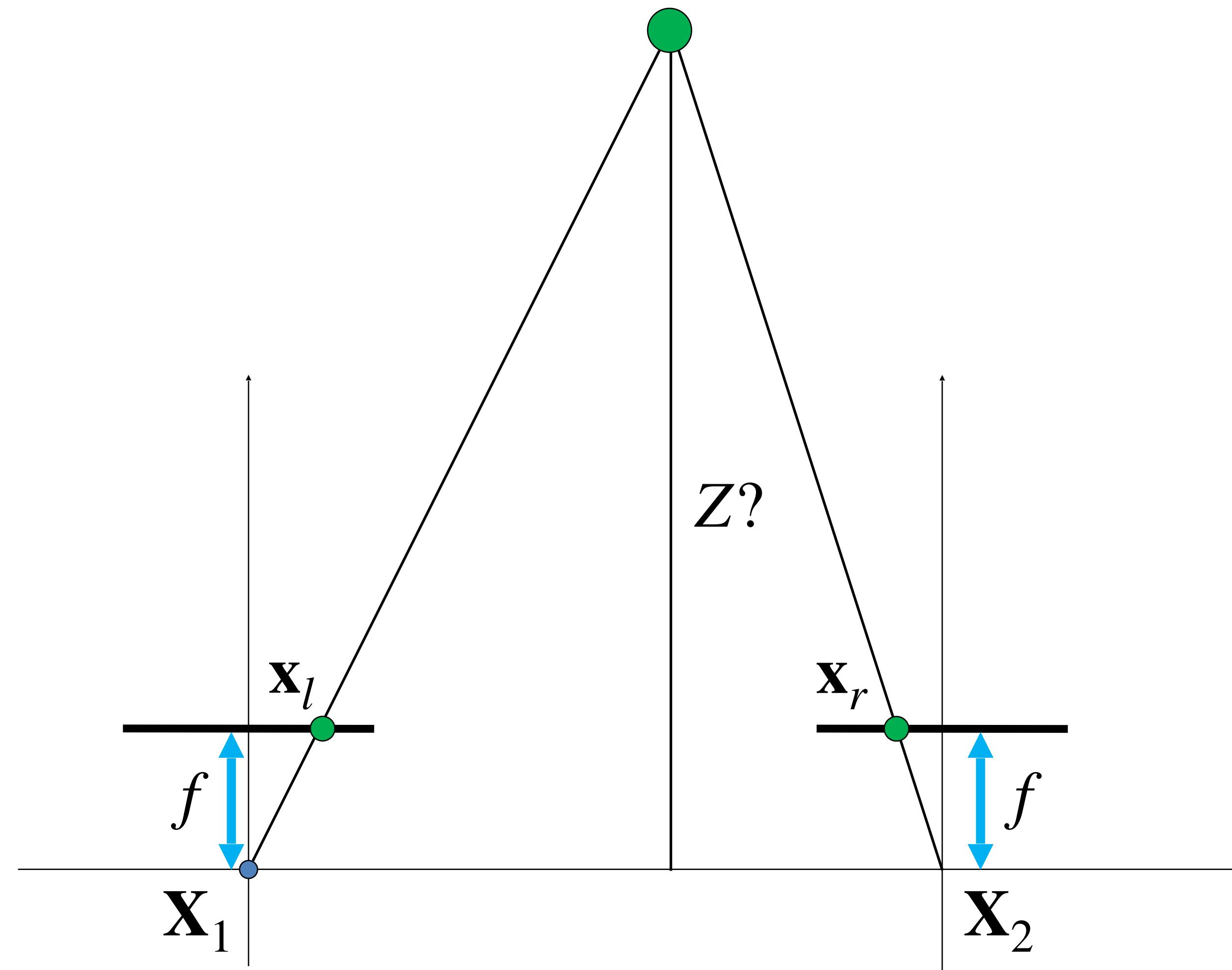
# Geometry for a simple stereo system



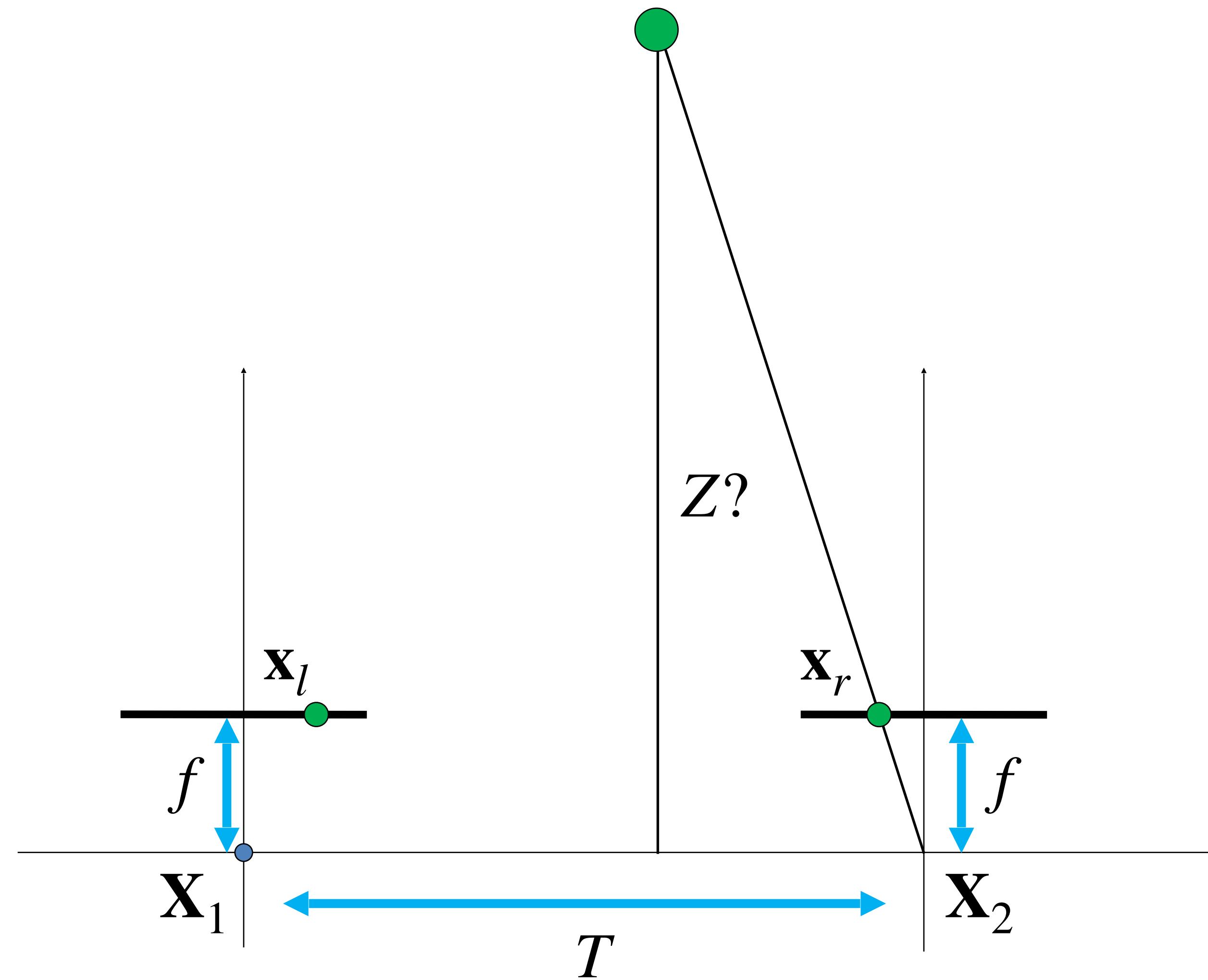
# Geometry for a simple stereo system



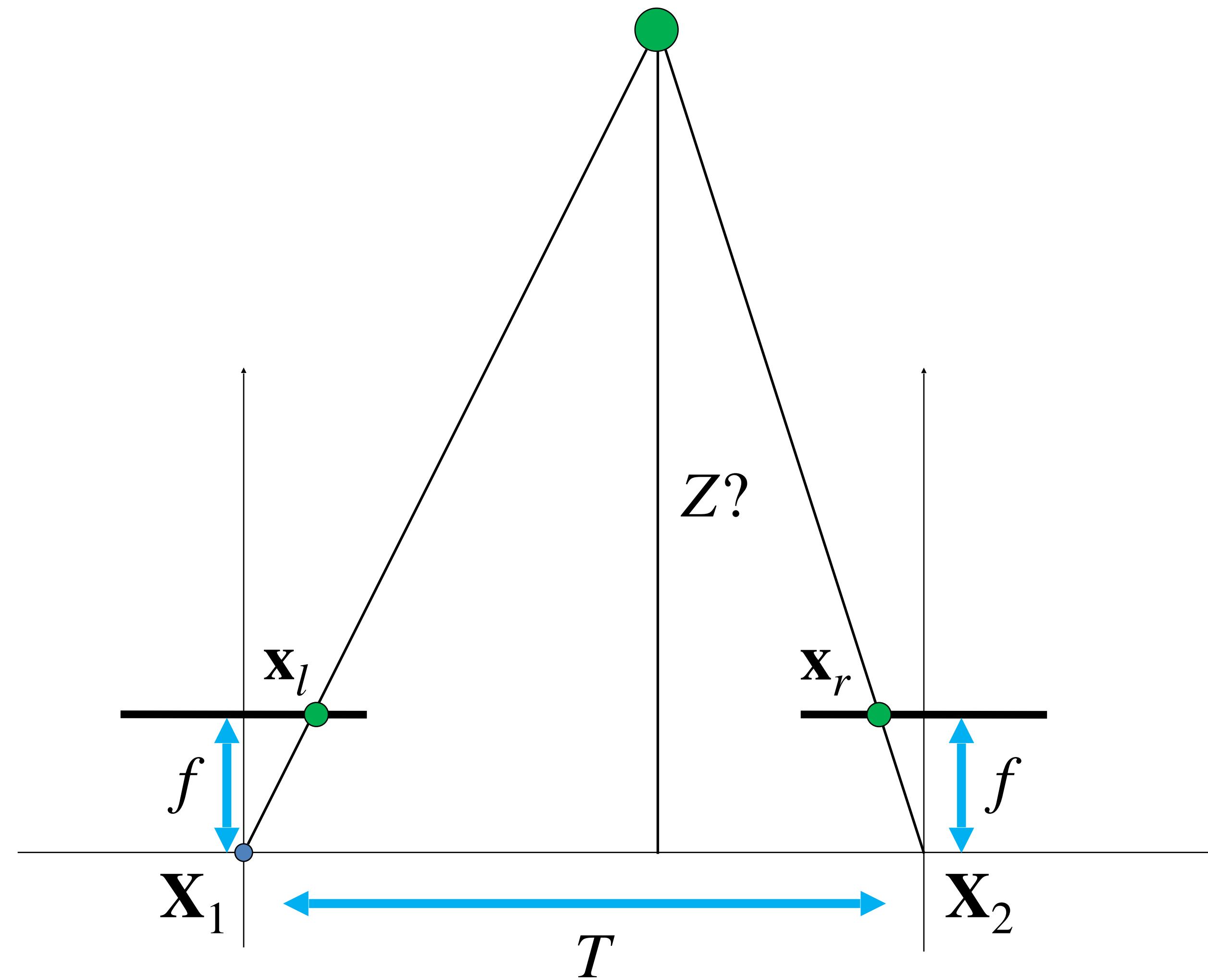
# Geometry for a simple stereo system



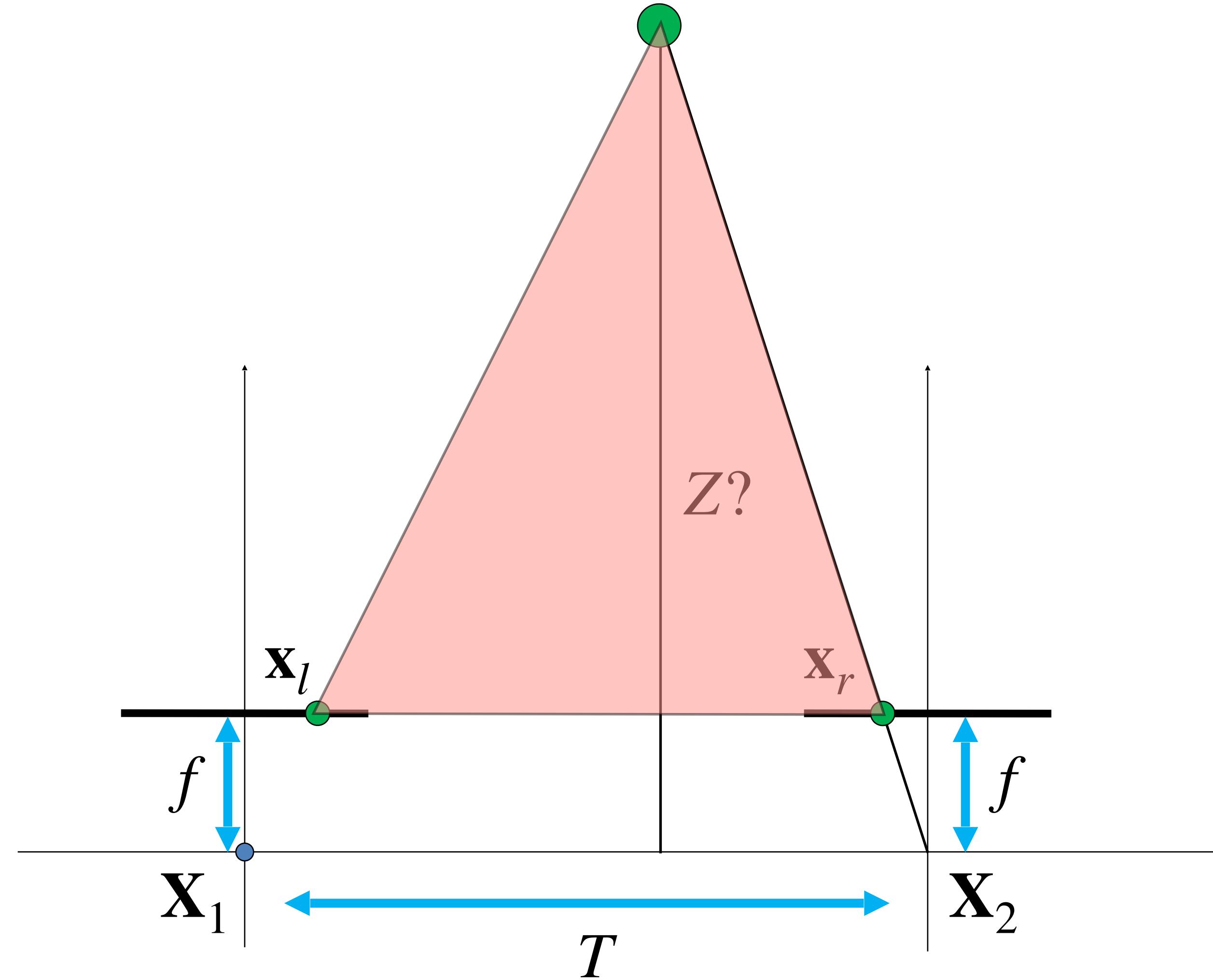
# Geometry for a simple stereo system



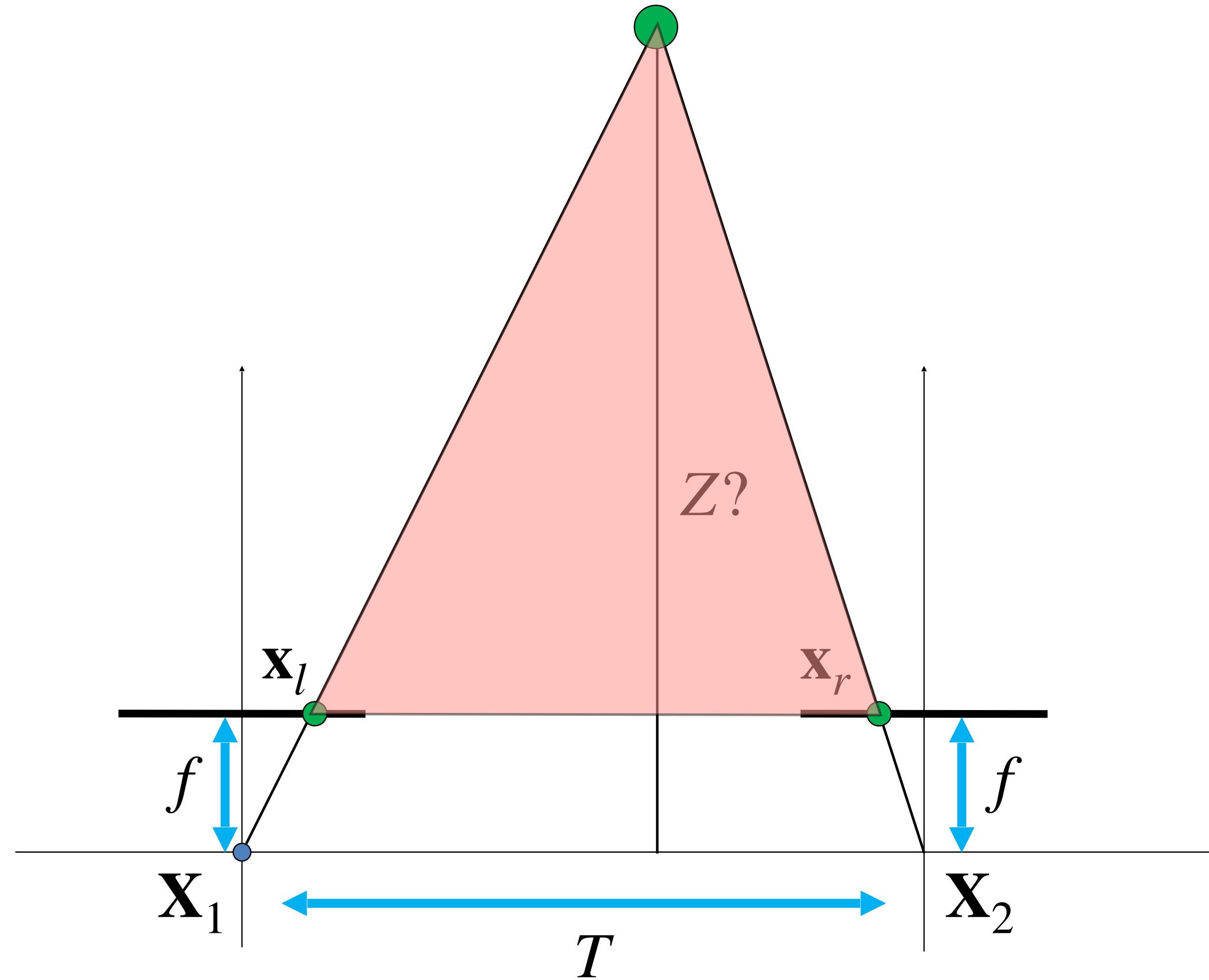
# Geometry for a simple stereo system



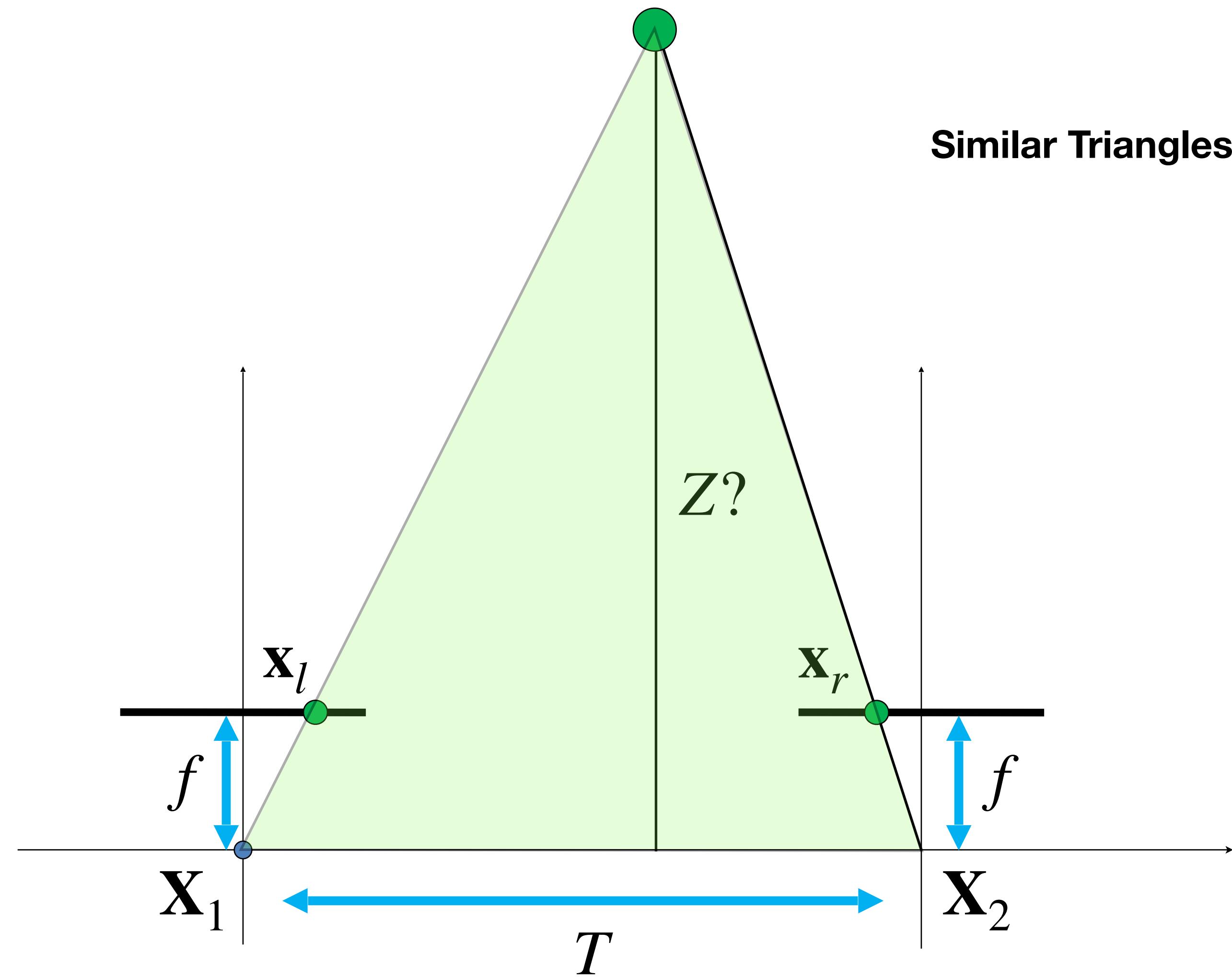
# Geometry for a simple stereo system



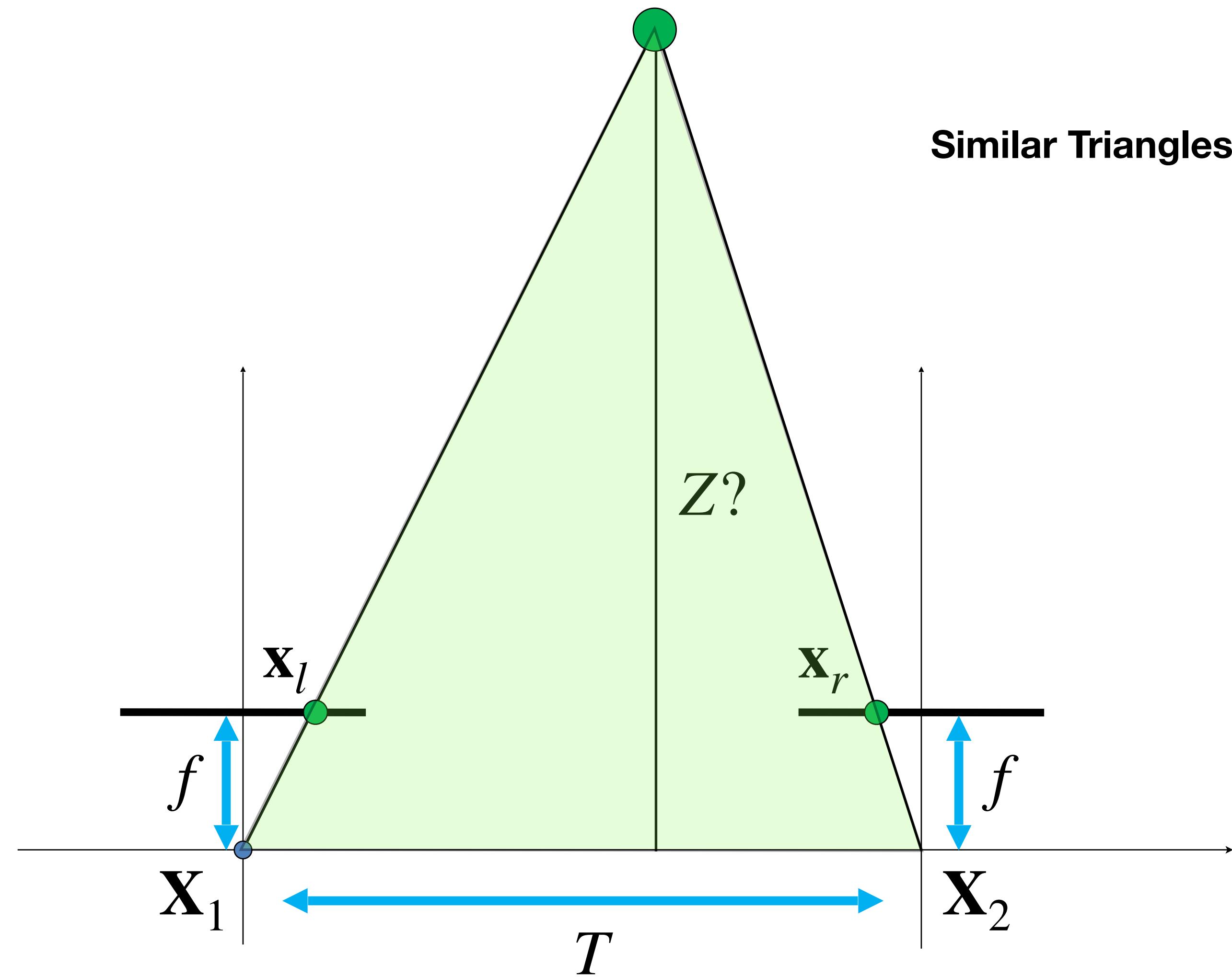
# Geometry for a simple stereo system



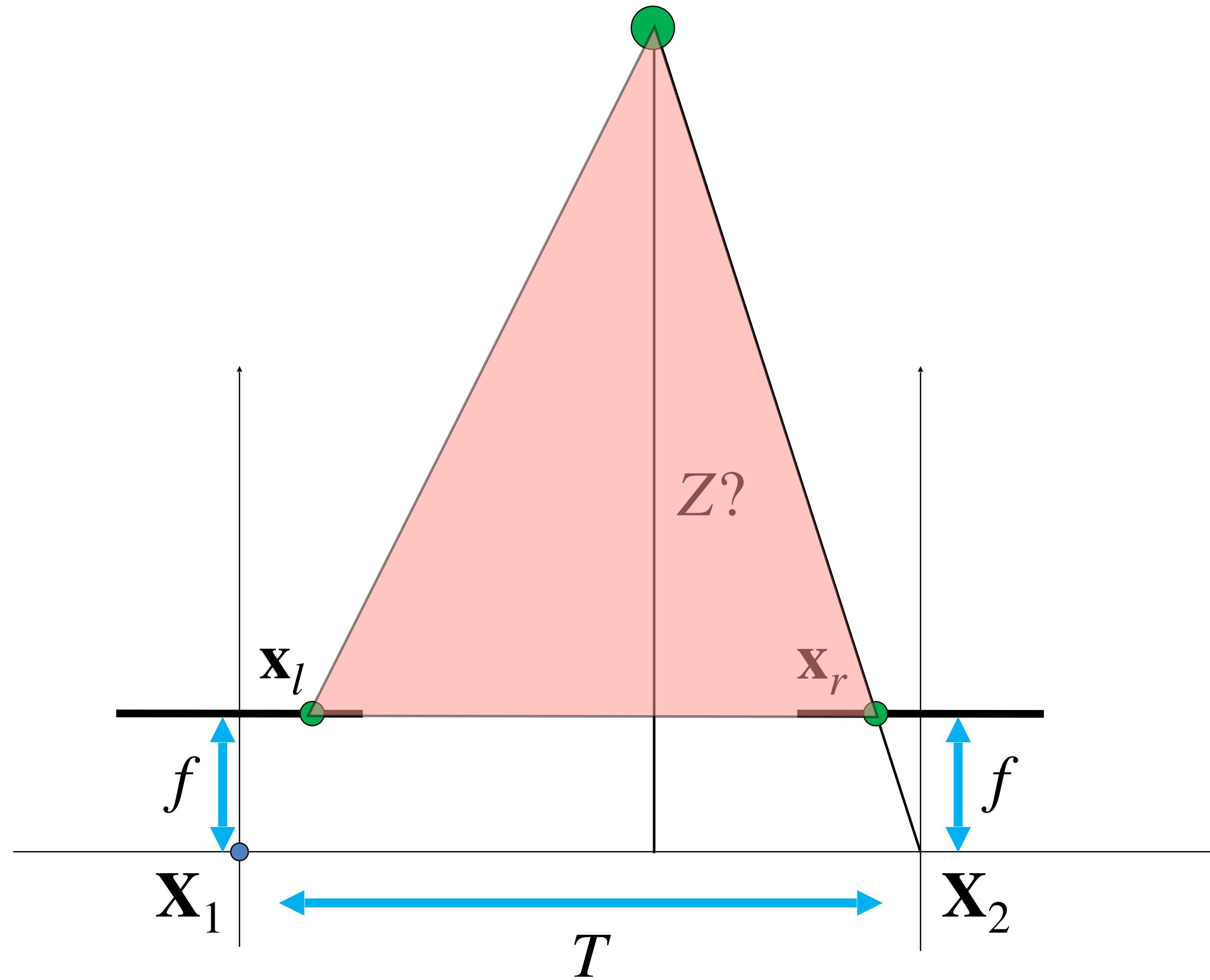
# Geometry for a simple stereo system



# Geometry for a simple stereo system



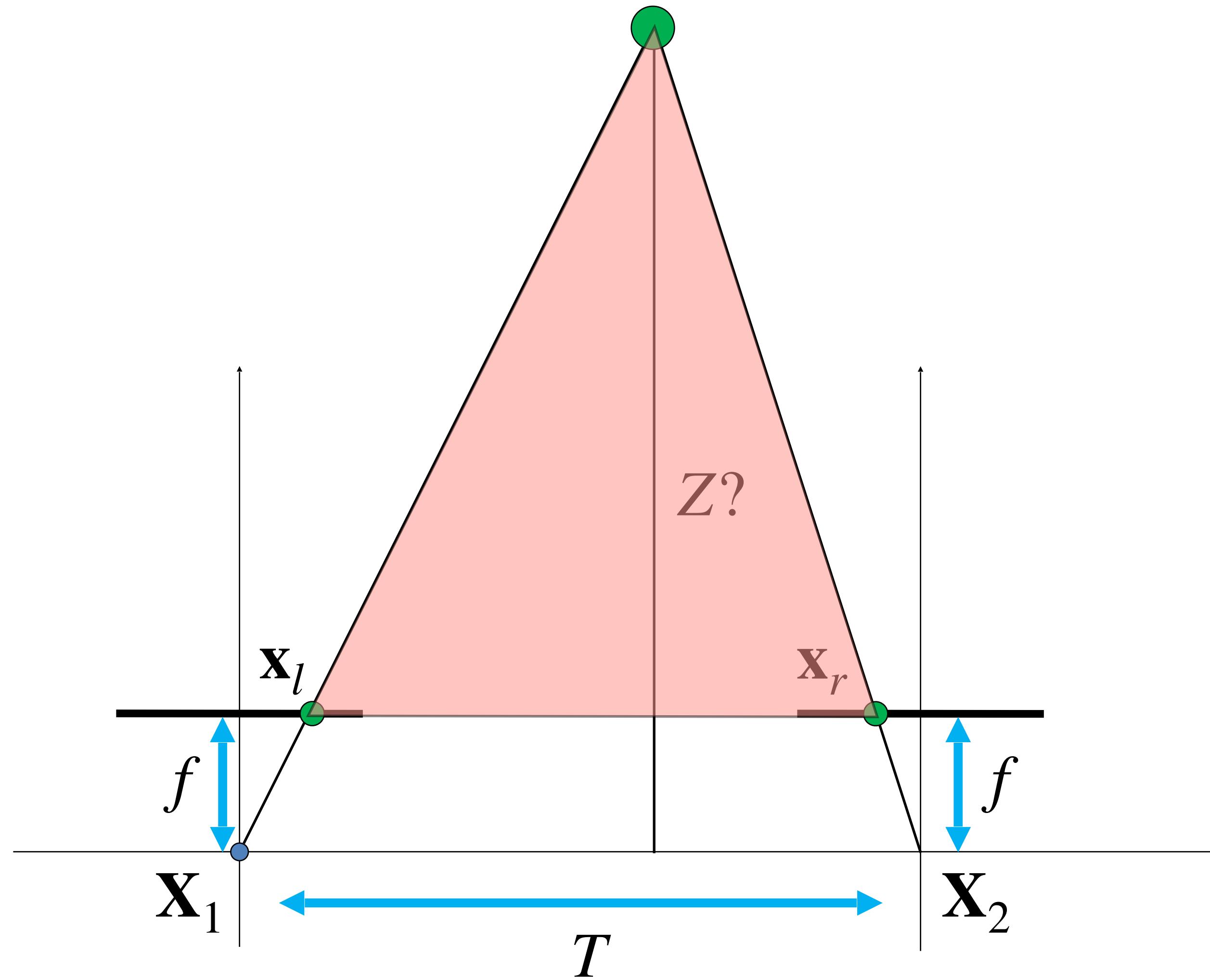
# Geometry for a simple stereo system



Similar Triangles:

$$\frac{T + \mathbf{x}_r - \mathbf{x}_l}{Z - f} =$$

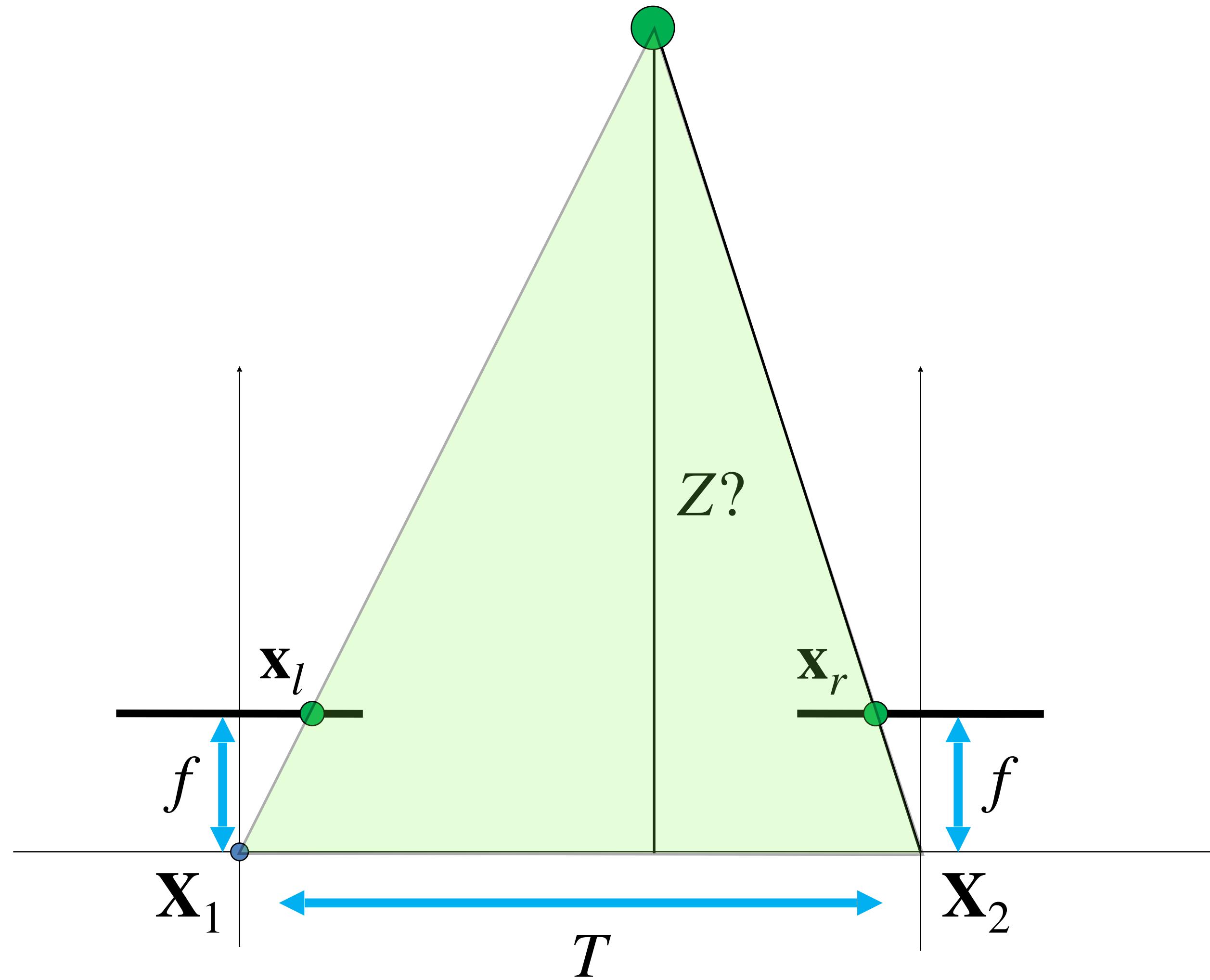
# Geometry for a simple stereo system



Similar Triangles:

$$\frac{T + \mathbf{x}_r - \mathbf{x}_l}{Z - f} =$$

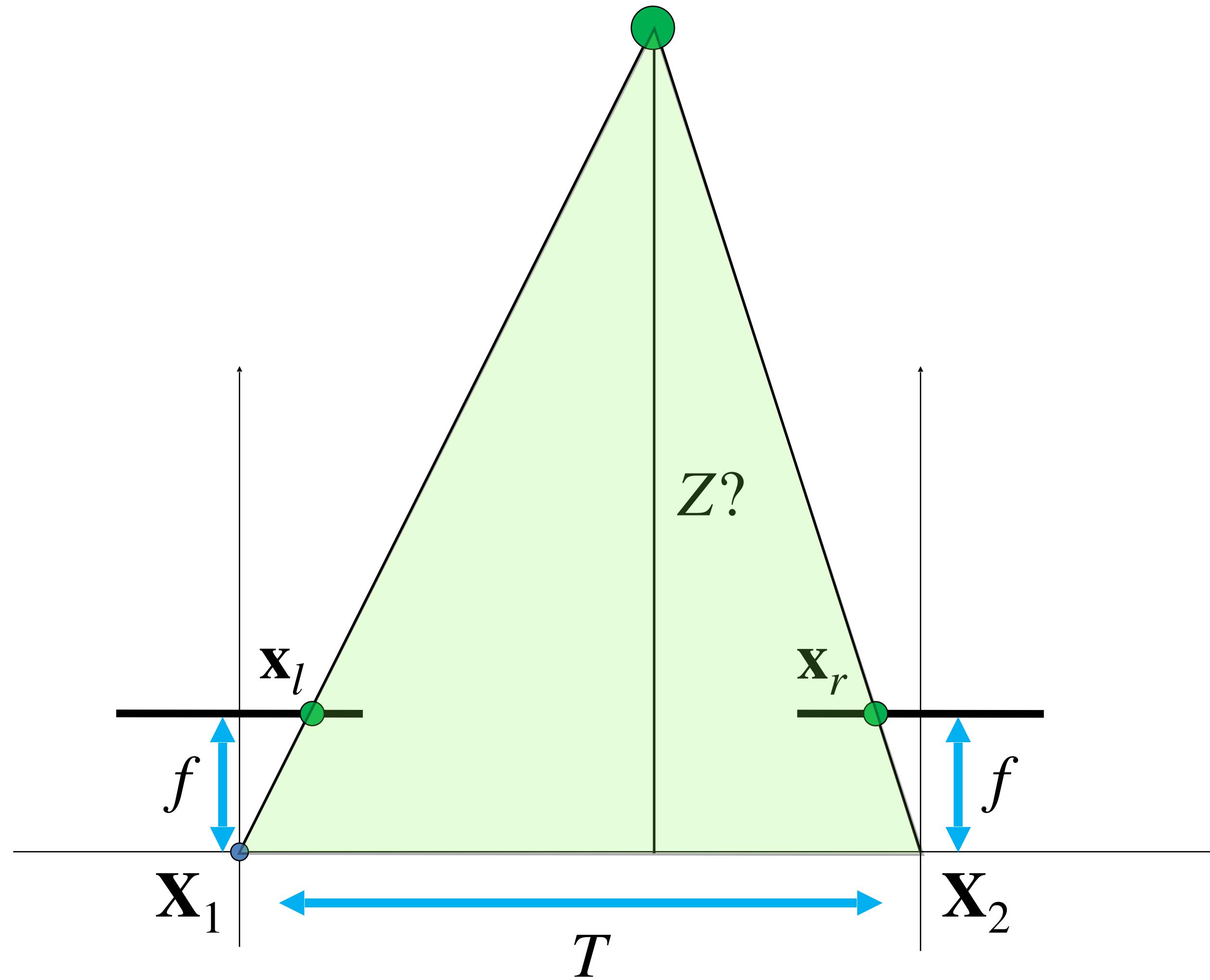
# Geometry for a simple stereo system



Similar Triangles:

$$\frac{T + \mathbf{x}_r - \mathbf{x}_l}{Z - f} = \frac{T}{Z}$$

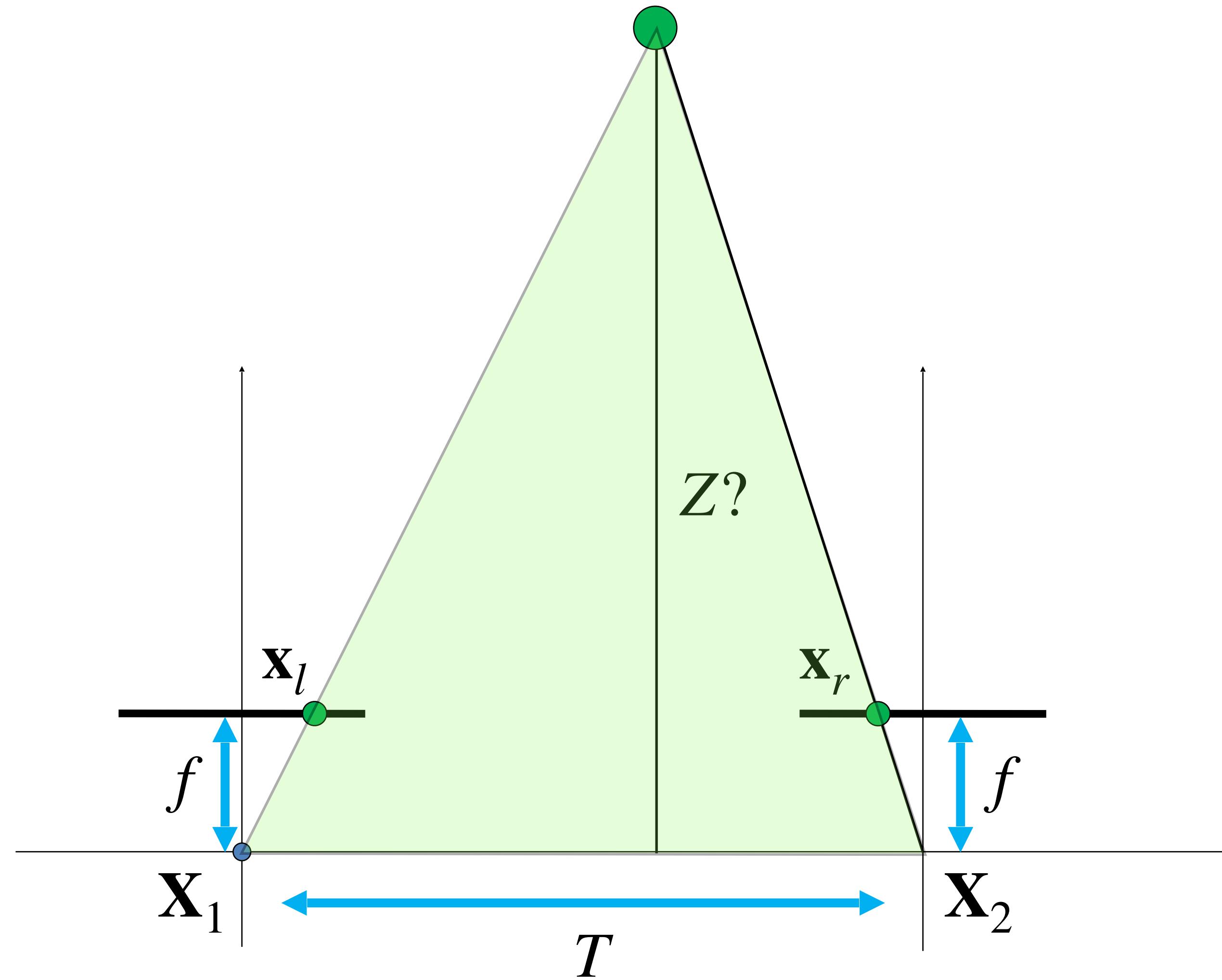
# Geometry for a simple stereo system



Similar Triangles:

$$\frac{T + \mathbf{x}_r - \mathbf{x}_l}{Z - f} = \frac{T}{Z}$$

# Geometry for a simple stereo system



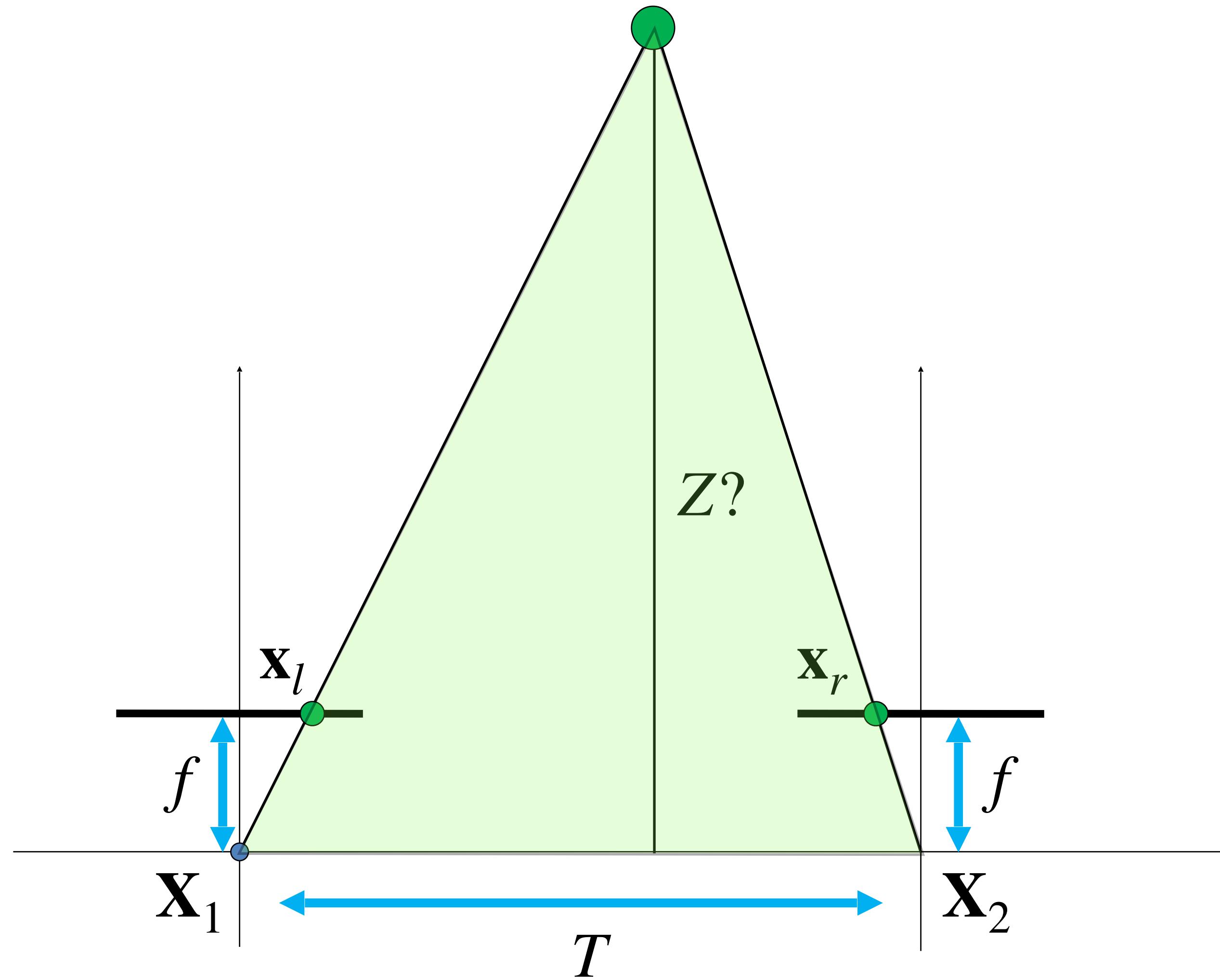
**Similar Triangles:**

$$\frac{T + \mathbf{x}_r - \mathbf{x}_l}{Z - f} = \frac{T}{Z}$$

**Solve for  $Z$ :**

$$Z = f \frac{T}{\mathbf{x}_l - \mathbf{x}_r}$$

# Geometry for a simple stereo system



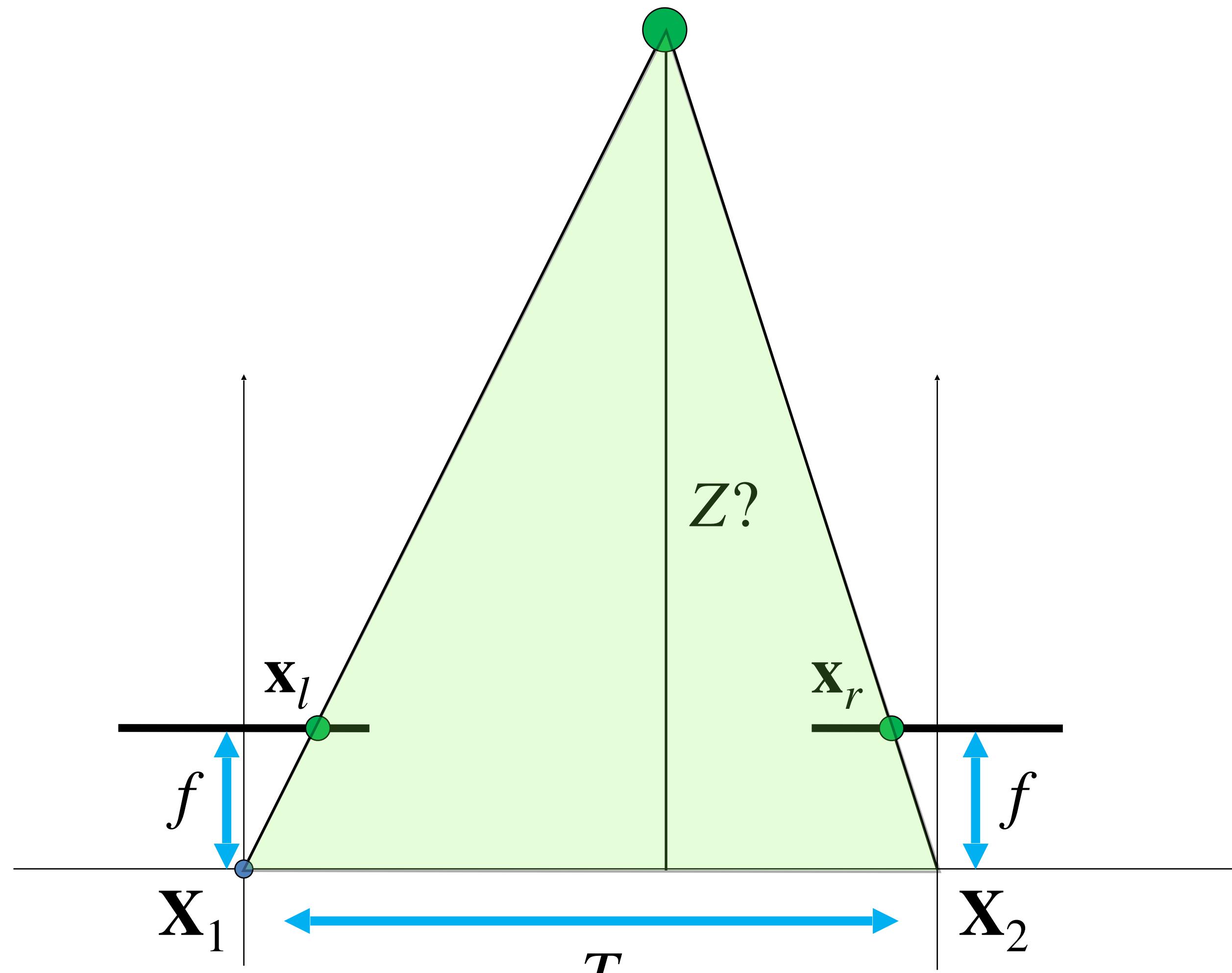
**Similar Triangles:**

$$\frac{T + \mathbf{x}_r - \mathbf{x}_l}{Z - f} = \frac{T}{Z}$$

**Solve for Z:**

$$Z = f \frac{T}{\mathbf{x}_l - \mathbf{x}_r}$$

# Geometry for a simple stereo system



**Similar Triangles:**

$$\frac{T + \mathbf{x}_r - \mathbf{x}_l}{Z - f} = \frac{T}{Z}$$

**Solve for  $Z$ :**

$$Z = f \frac{T}{\mathbf{x}_l - \mathbf{x}_r}$$

Disparity

# Measuring disparity



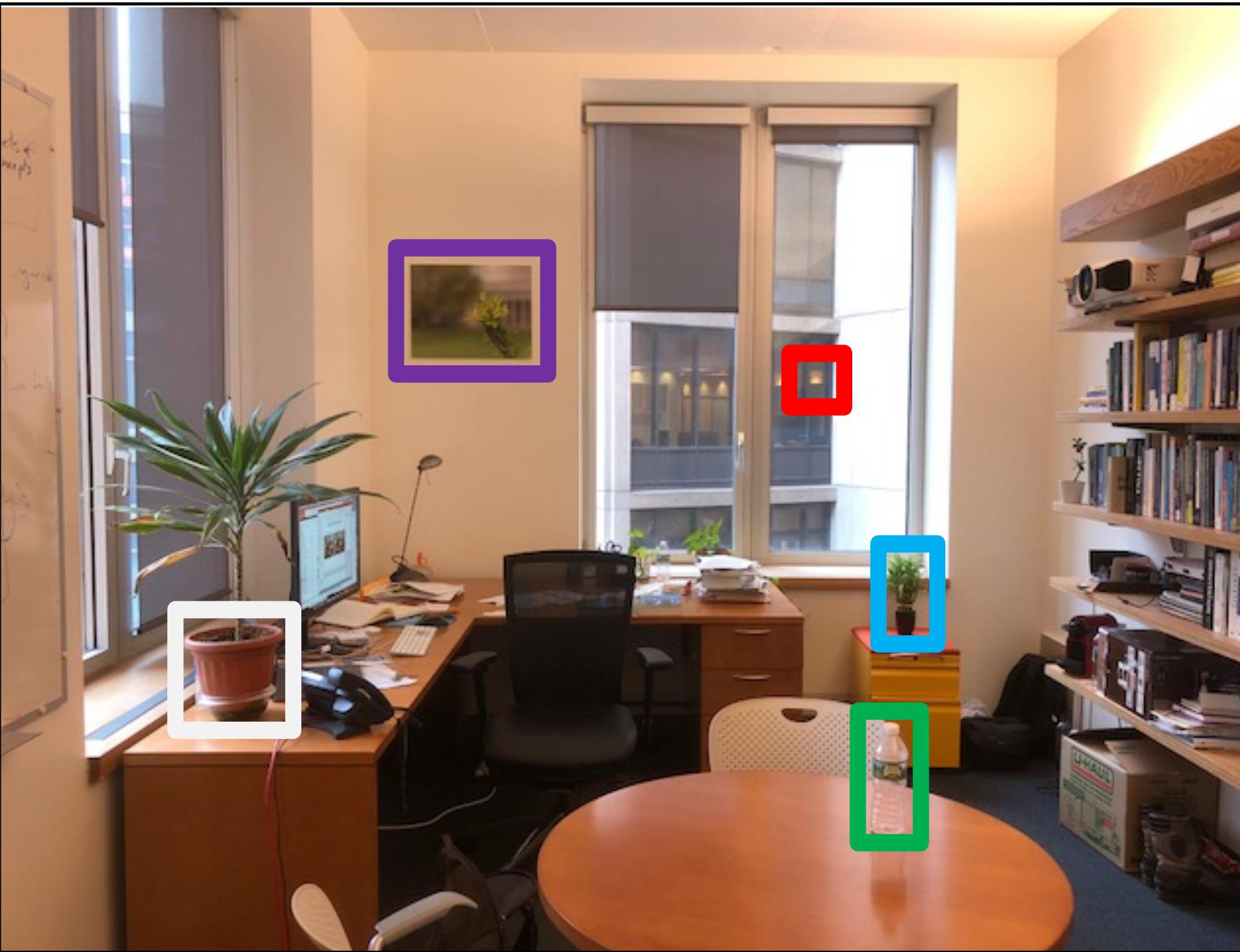
Left image



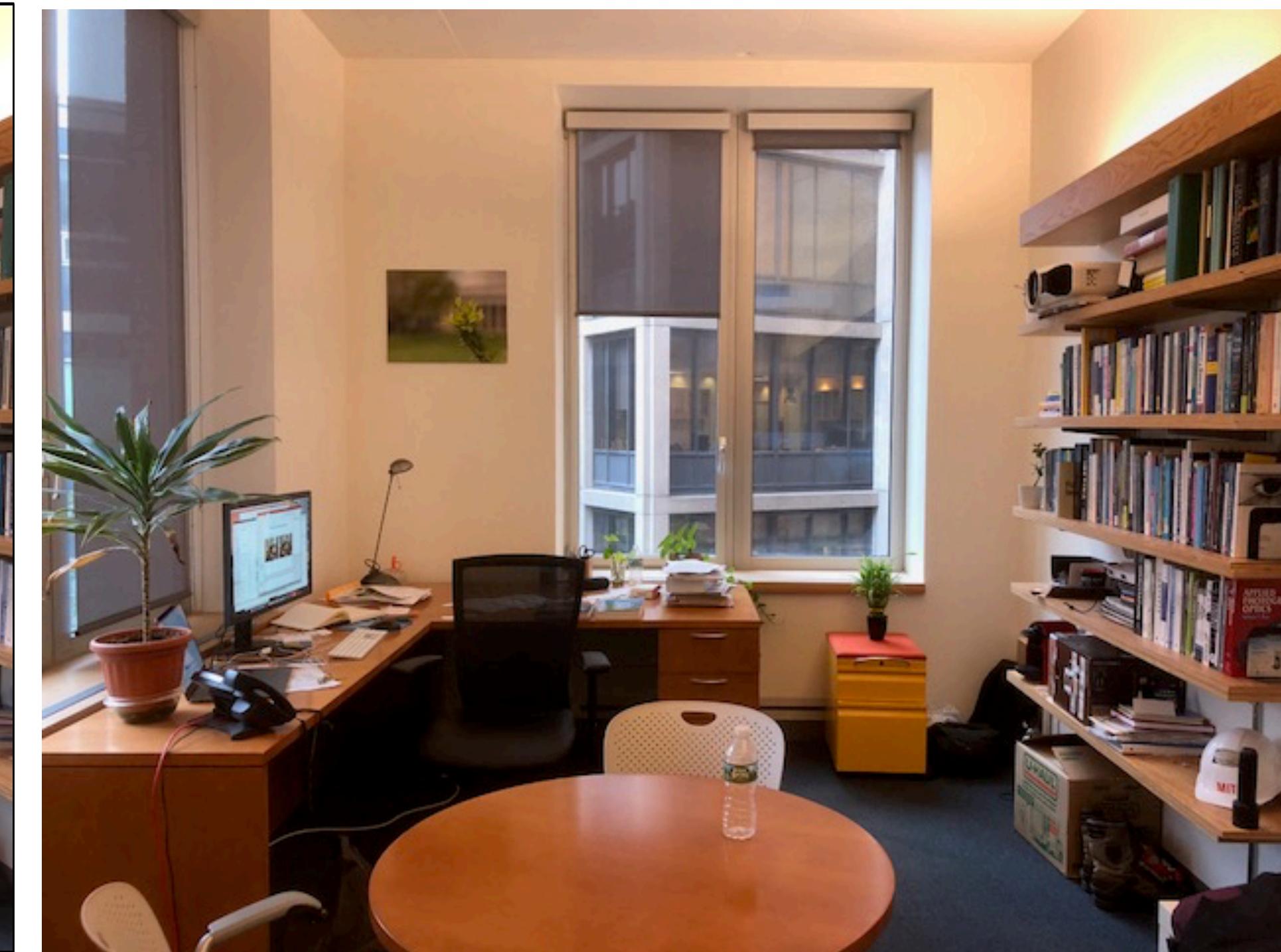
Right image

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

# Measuring disparity

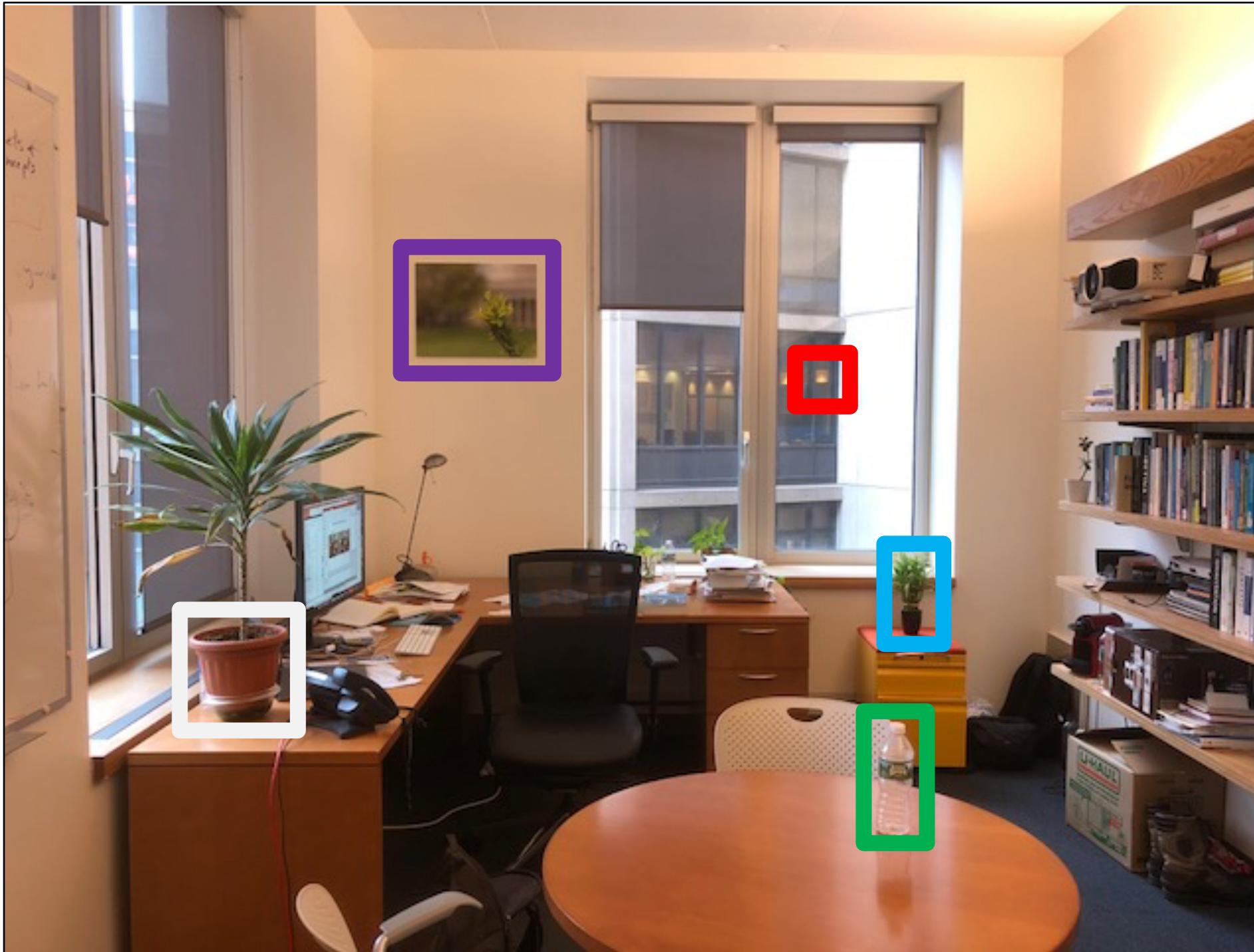


Left image

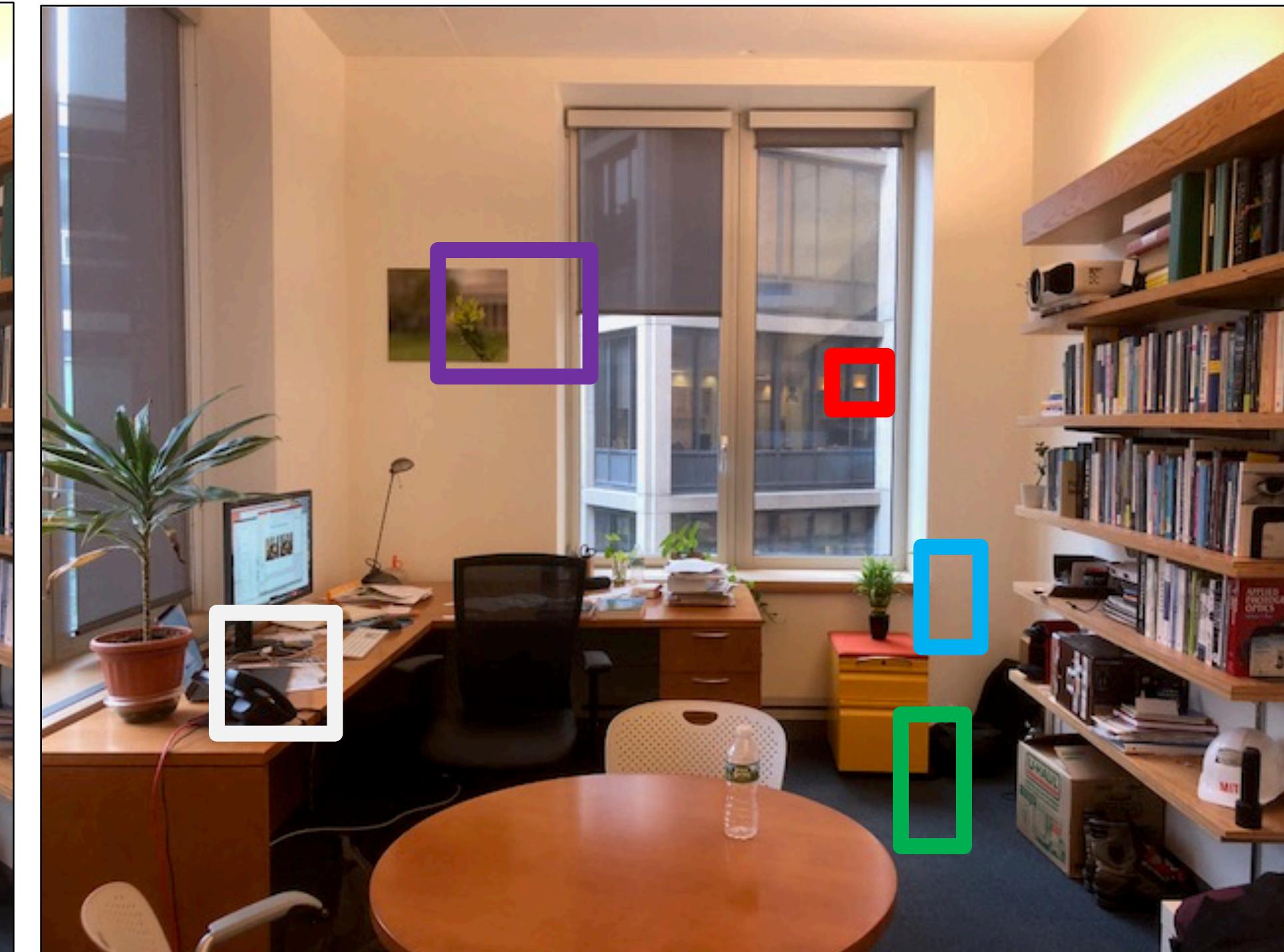


Right image

# Measuring disparity

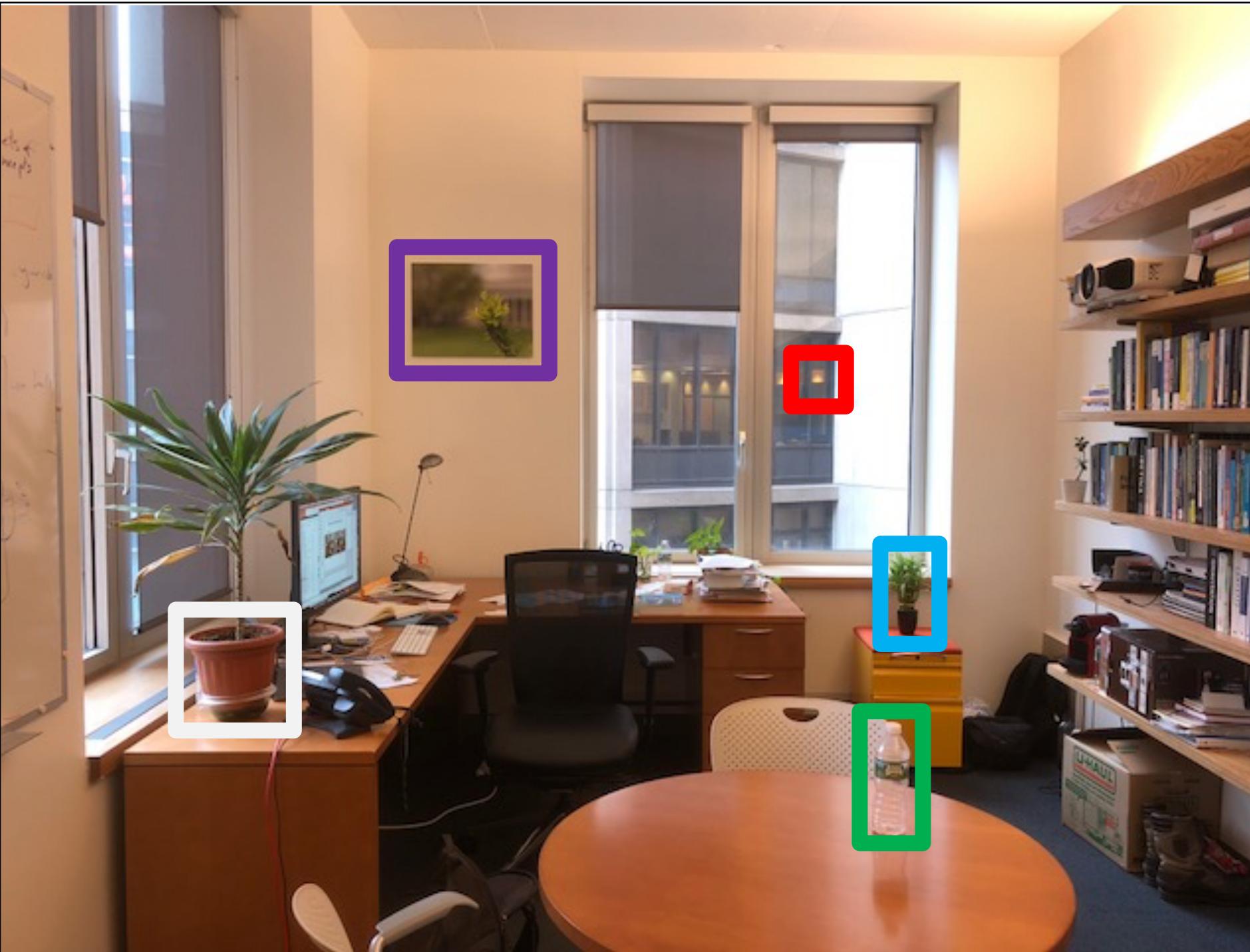


Left image

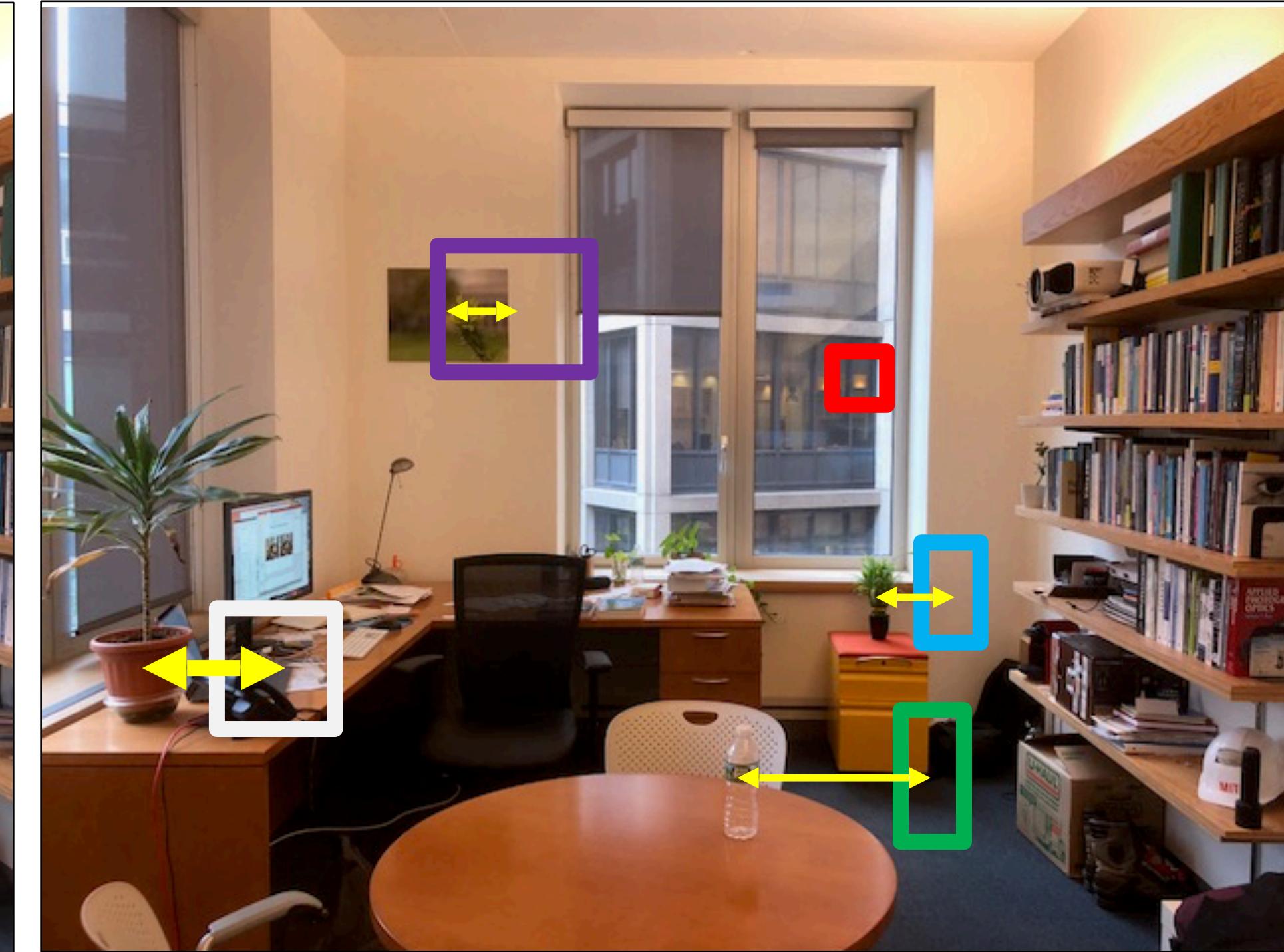


Right image

# Measuring disparity



Left image



Right image

# Disparity map

$$I(x, y)$$



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



# Disparity map

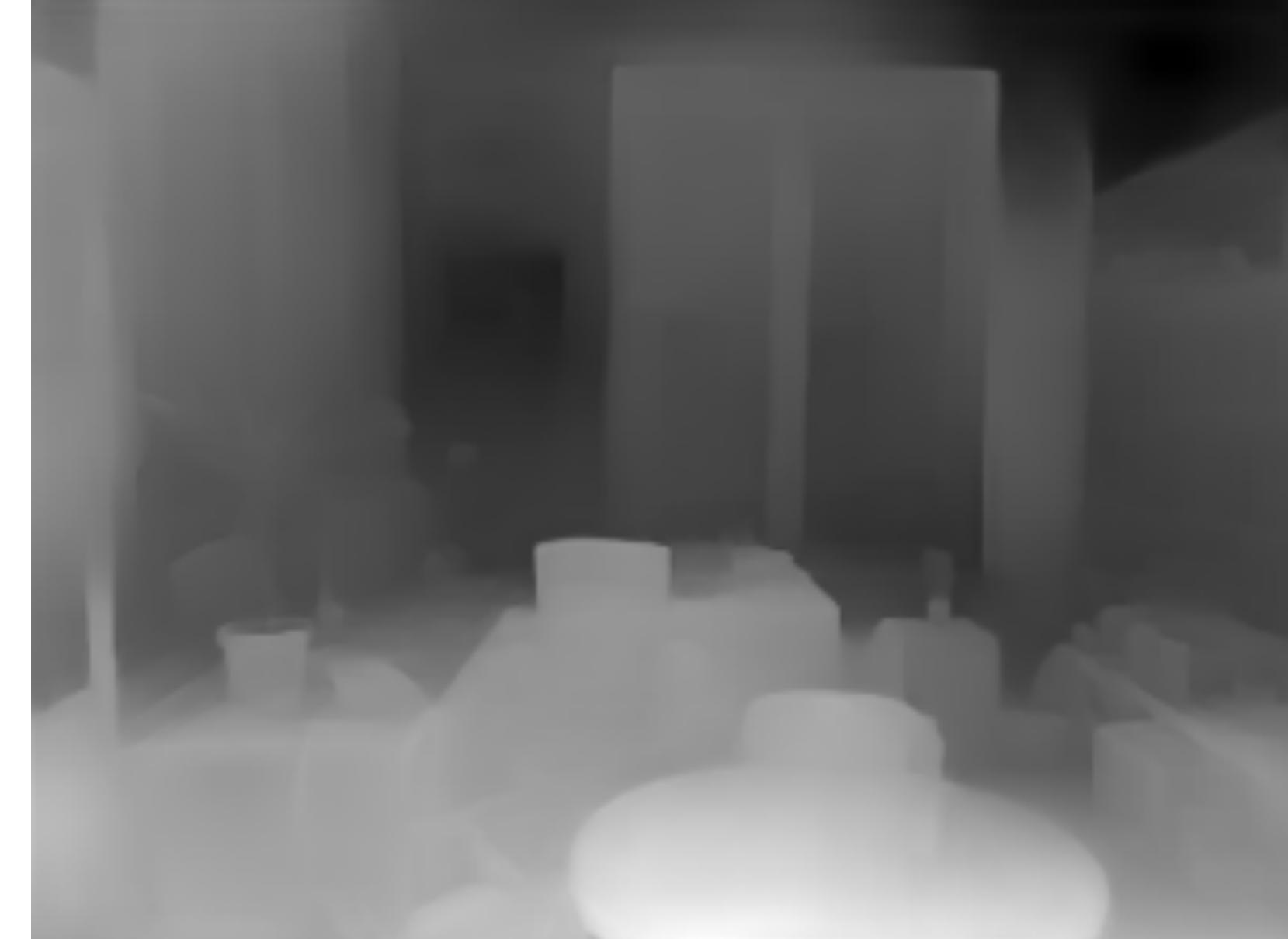
$I(x, y)$



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



$D(x, y)$



# Disparity map

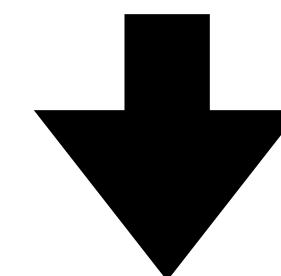
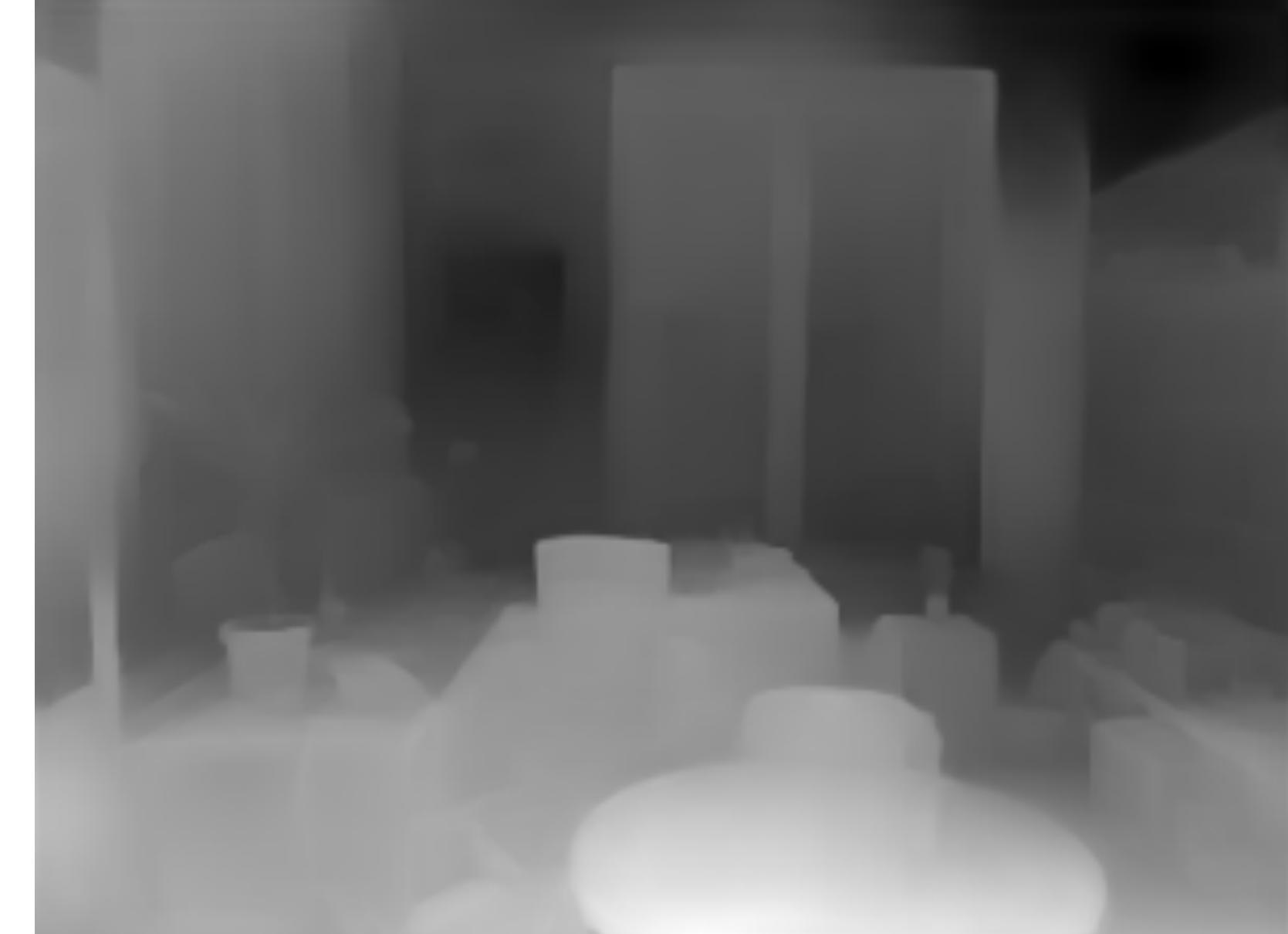
$I(x, y)$



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



$D(x, y)$



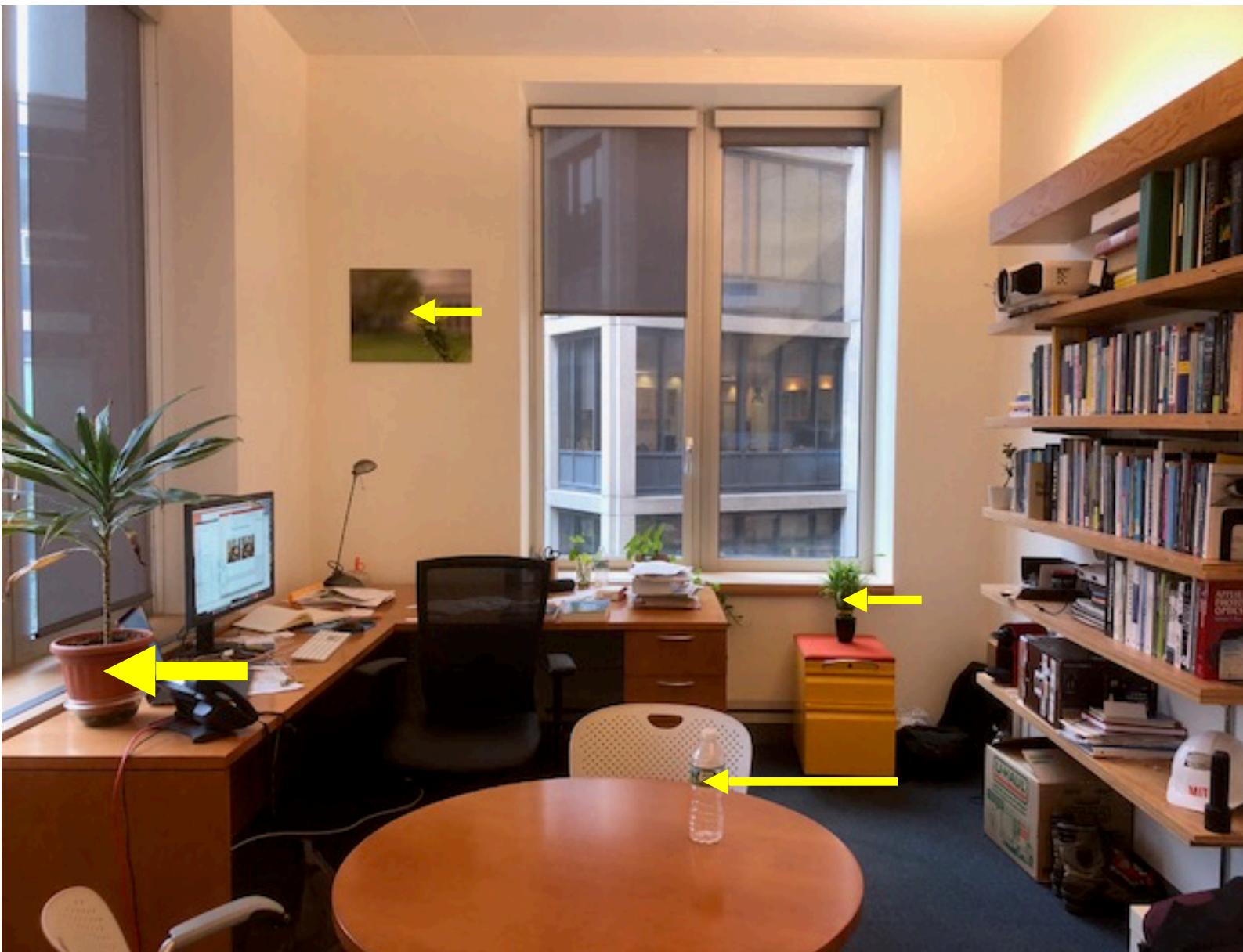
$$Z = f \frac{T}{D(x, y)}$$

# Disparity map

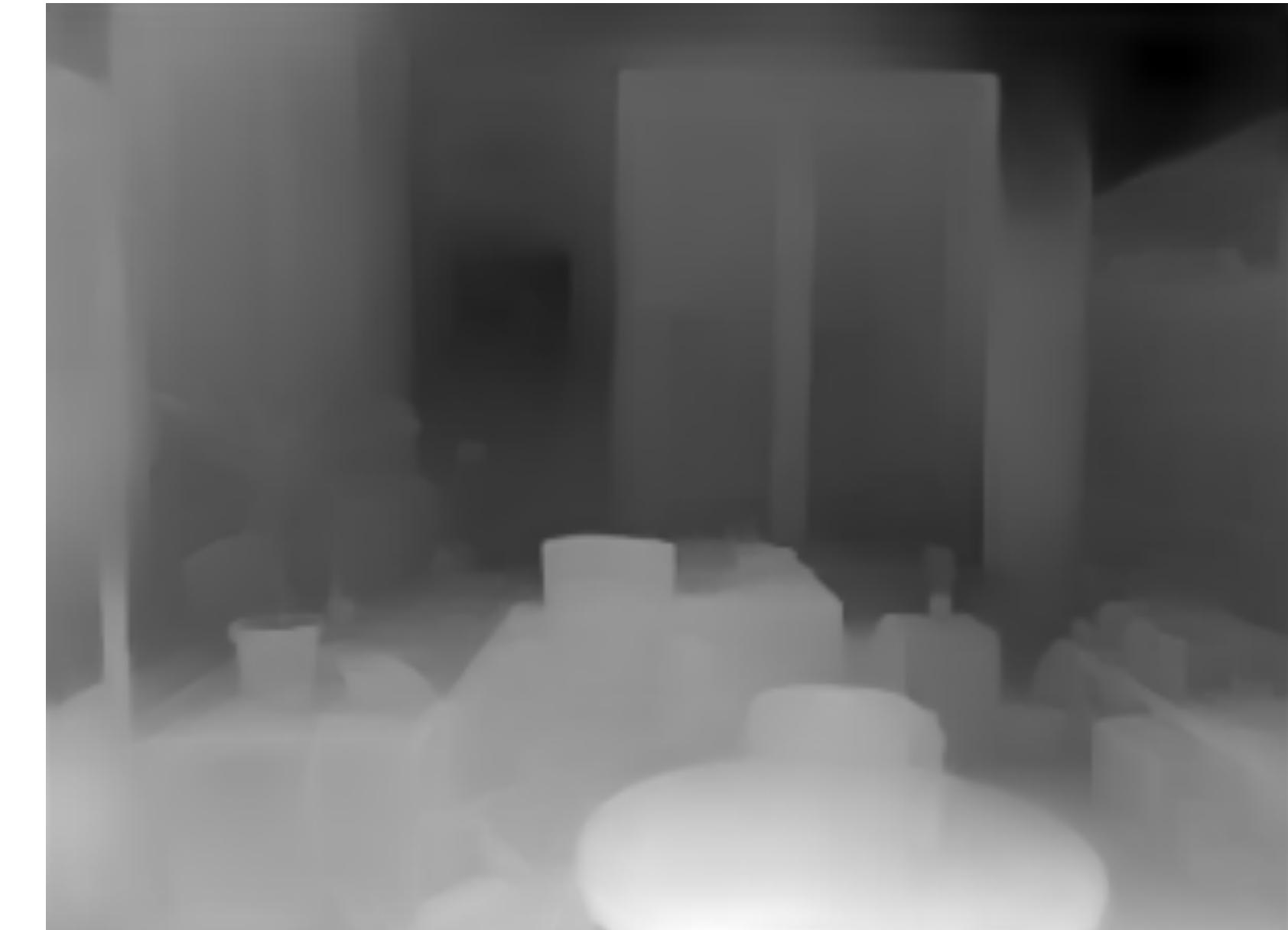
$I(x, y)$



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



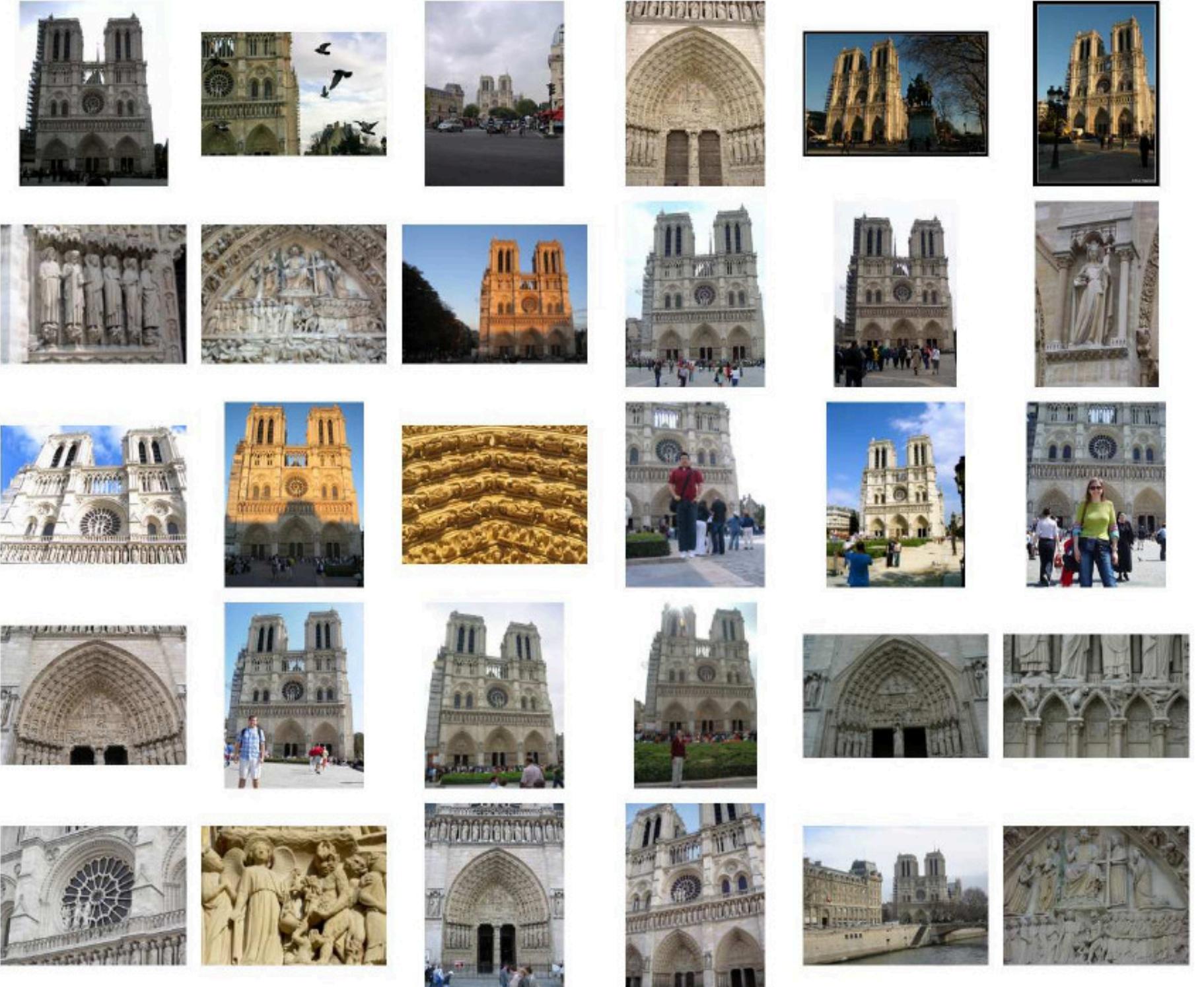
$D(x, y)$



Depth map

$$z = f \frac{T}{D(x, y)}$$

# Next time: Multi-View Geometry



Why?

We want to understand 3D world only from 2D observations (images). For that, we need to have a mathematical understanding of how they are connected.

What you'll learn.

Mathematical model of cameras. Reconstruct camera poses, approximate geometry, and camera parameters from 2D images of a scene.