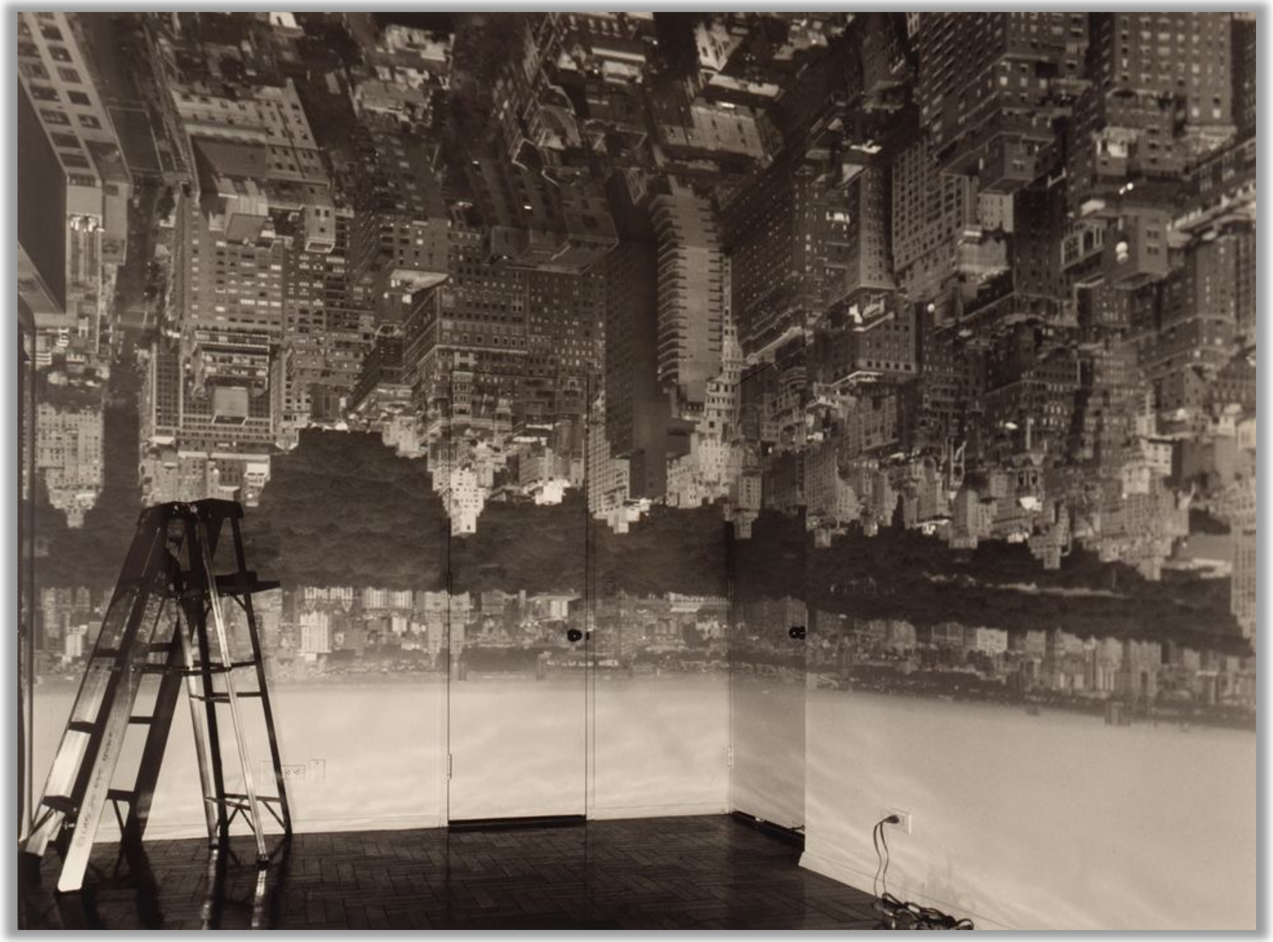
Camera Models



CSC420
David Lindell
University of Toronto
cs.toronto.edu/~lindell/teaching/420
Slide credit: Babak Taati ←Ahmed Ashraf ←Sanja Fidler

Textbook

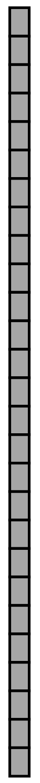
- If you are interested, this book has it all:

A. Zisserman and R. Hartley

Multiview Geometry

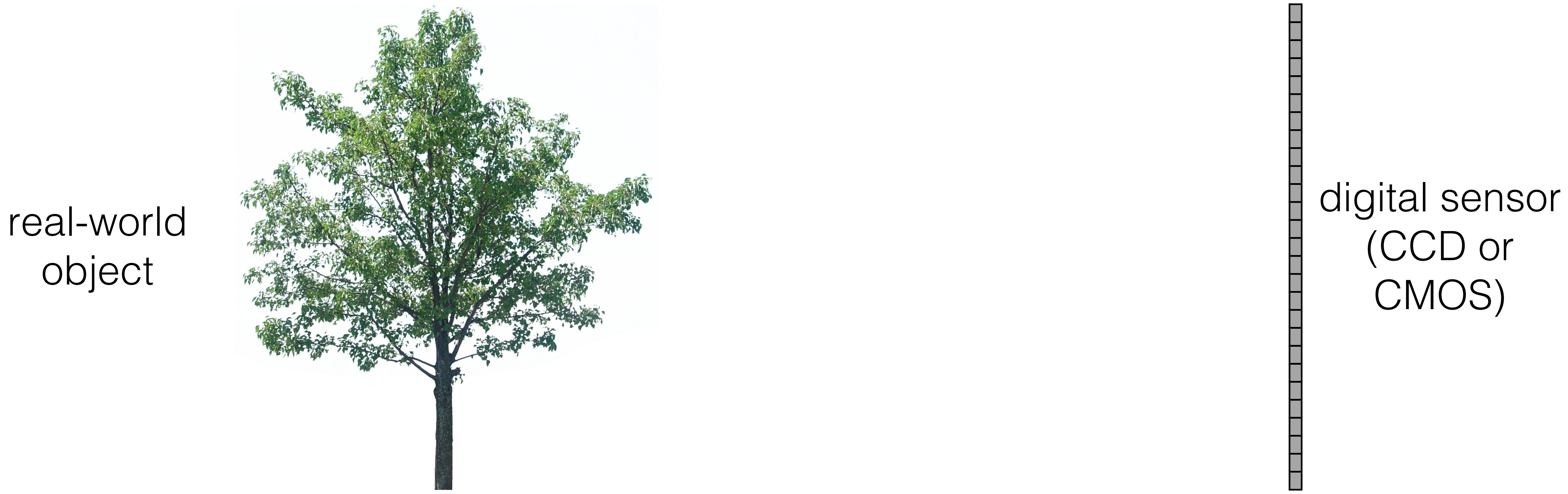
Cambridge University Press, 2003

Let's say we have a sensor…



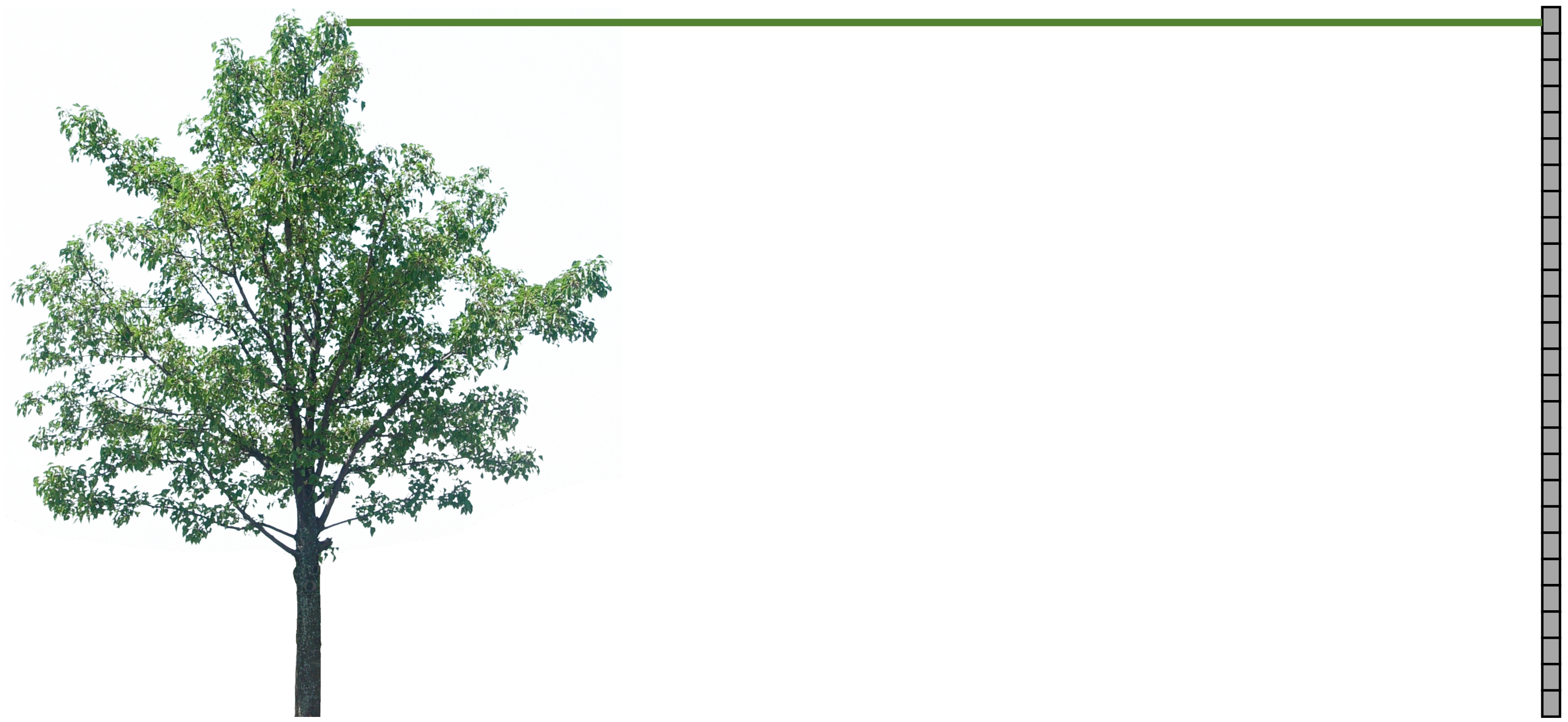
digital sensor (CCD or CMOS)

… and an object we like to photograph

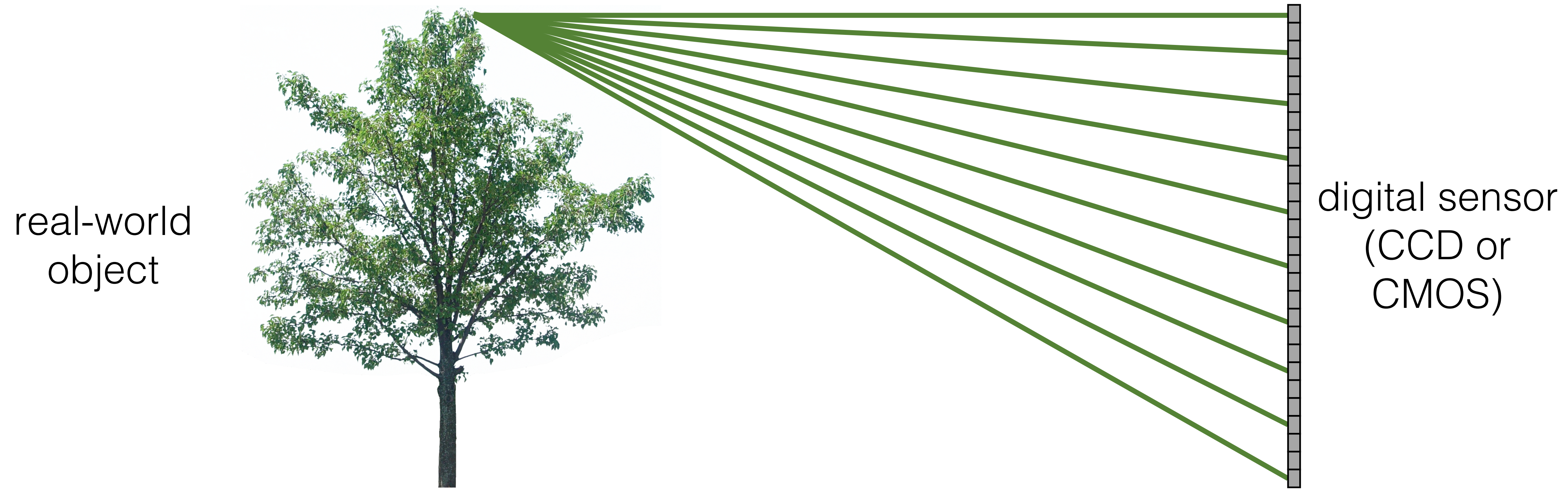


What would an image taken like this look like?

Bare-sensor imaging

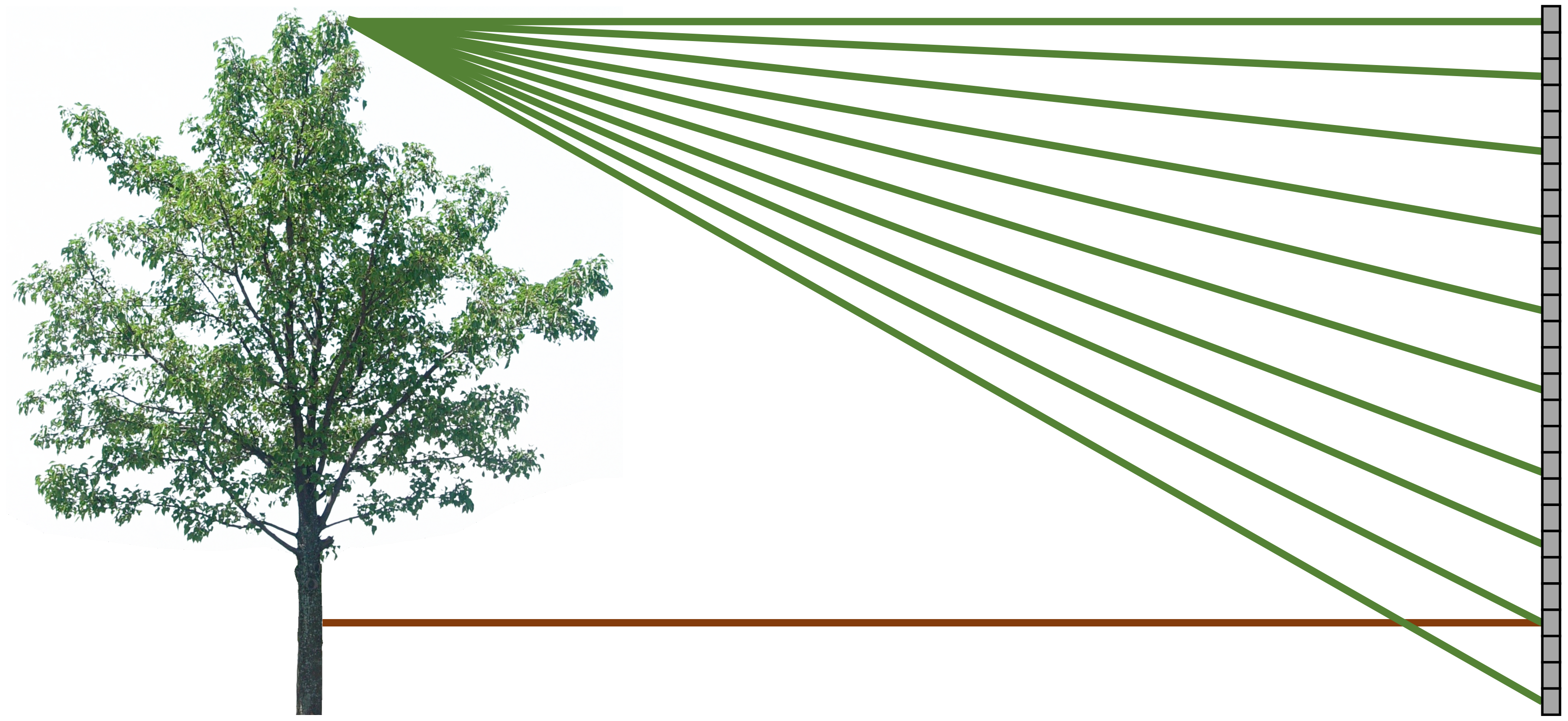


Bare-sensor imaging



Bare-sensor imaging

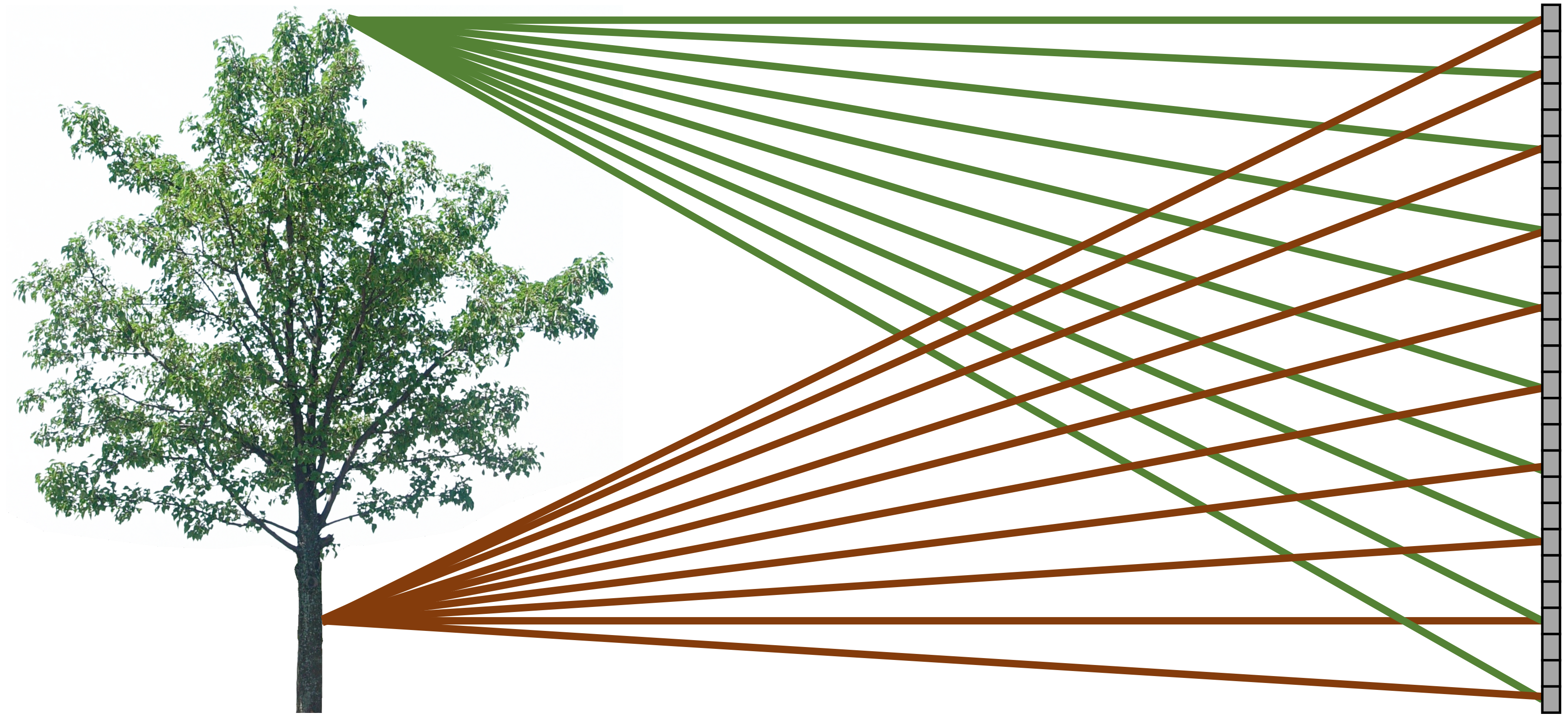
real-world object



digital sensor (CCD or CMOS)

Bare-sensor imaging

real-world object



digital sensor (CCD or CMOS)

All scene points contribute to all sensor pixels

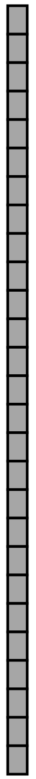
What does the image on the sensor look like?

Bare-sensor imaging

All scene points contribute to all sensor pixels

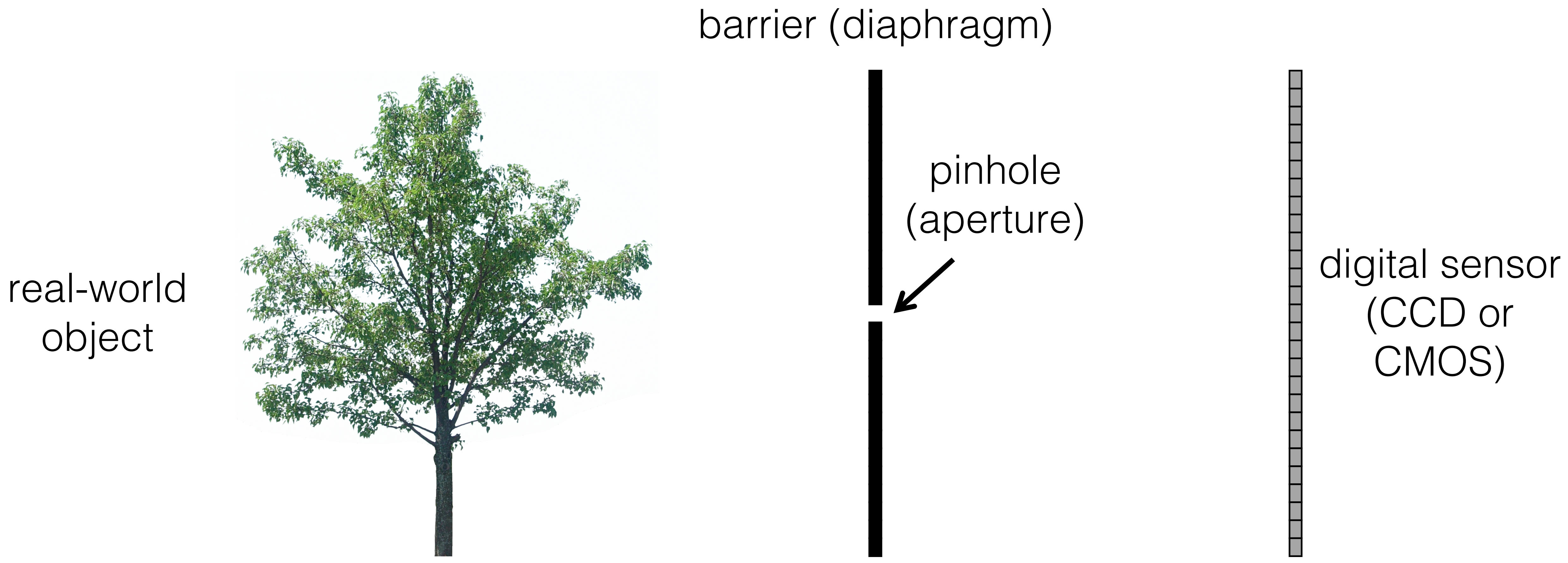
What can we do to make our image look better?

real-world object



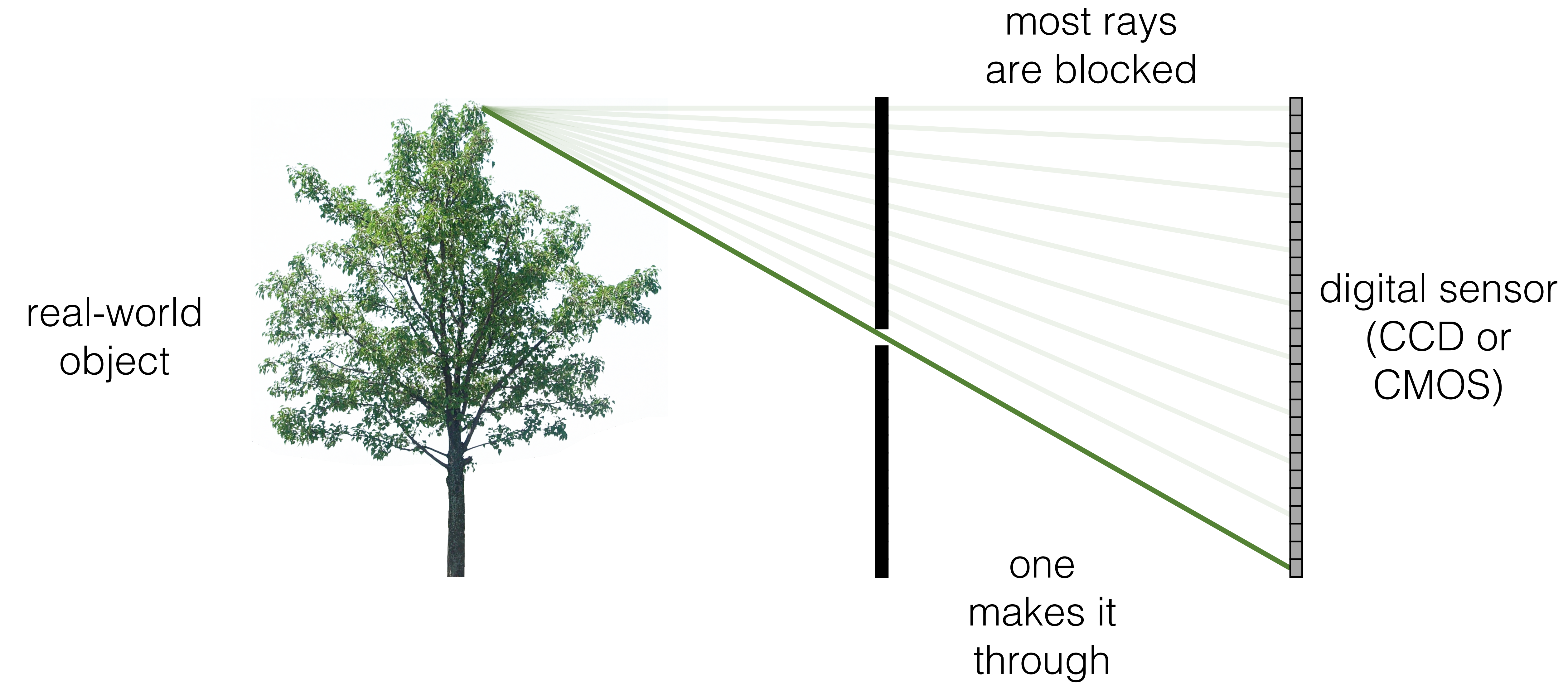
digital sensor (CCD or CMOS)

Let’s add something to this scene

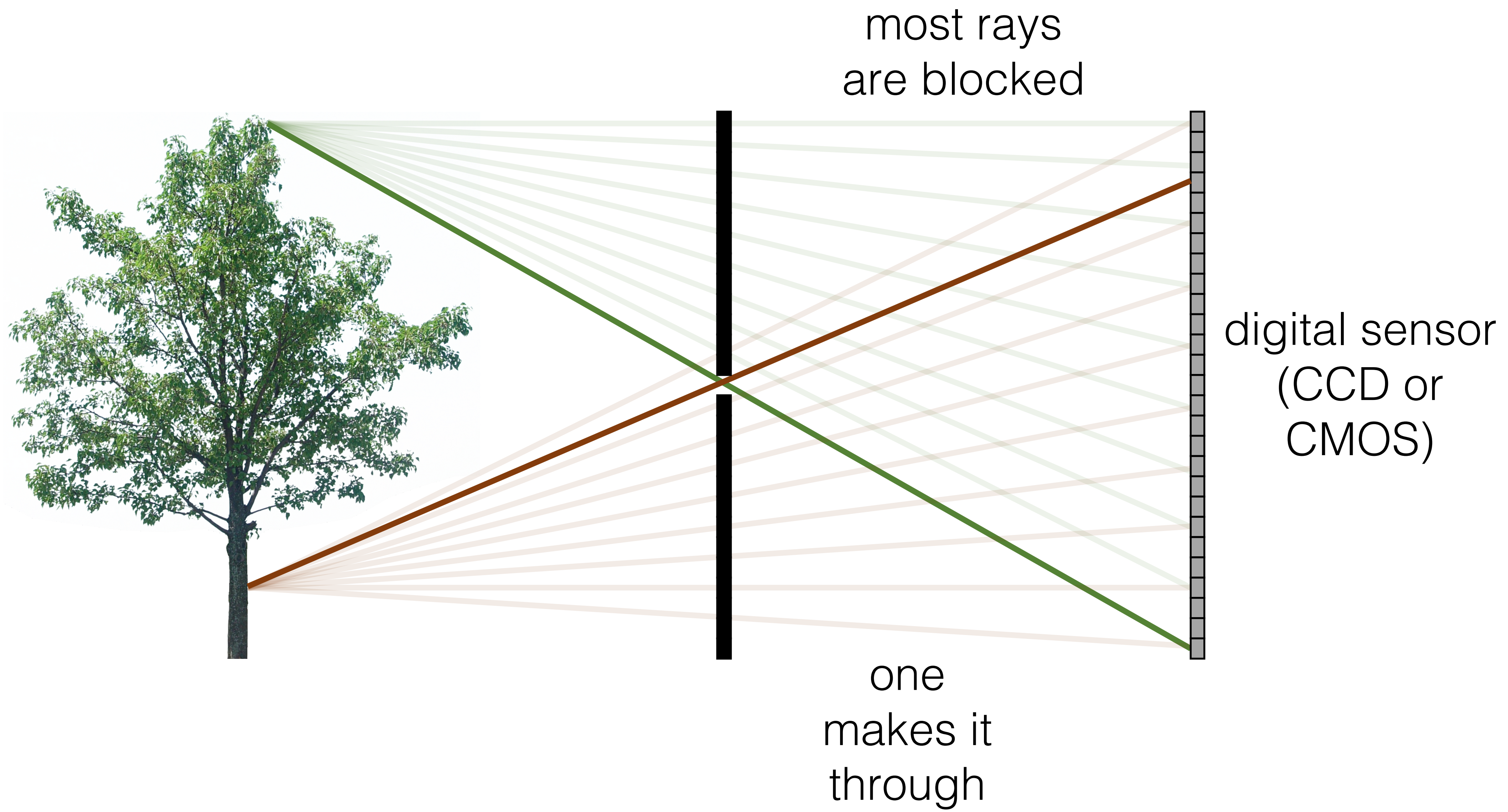


What would an image taken like this look like?

Pinhole imaging

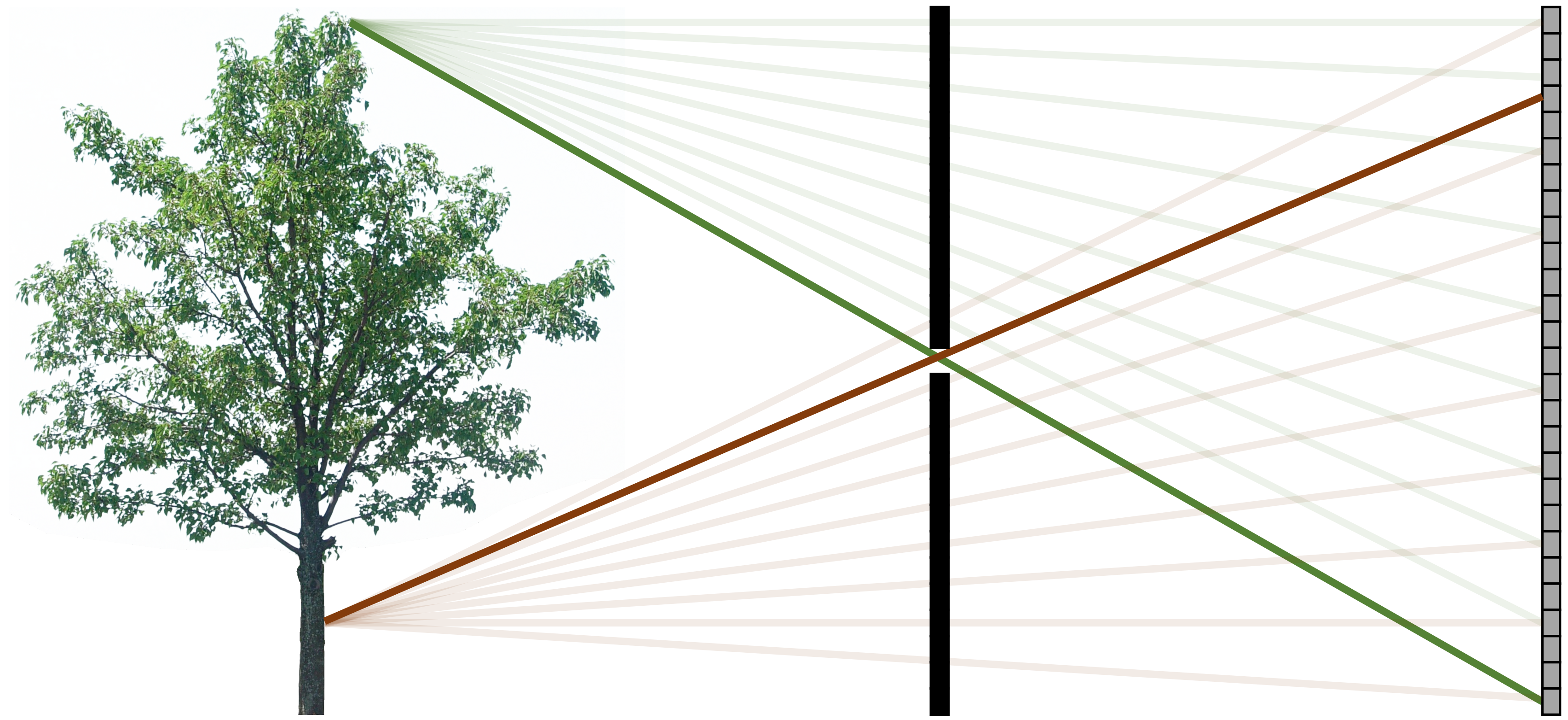


Pinhole imaging



Pinhole imaging

real-world object

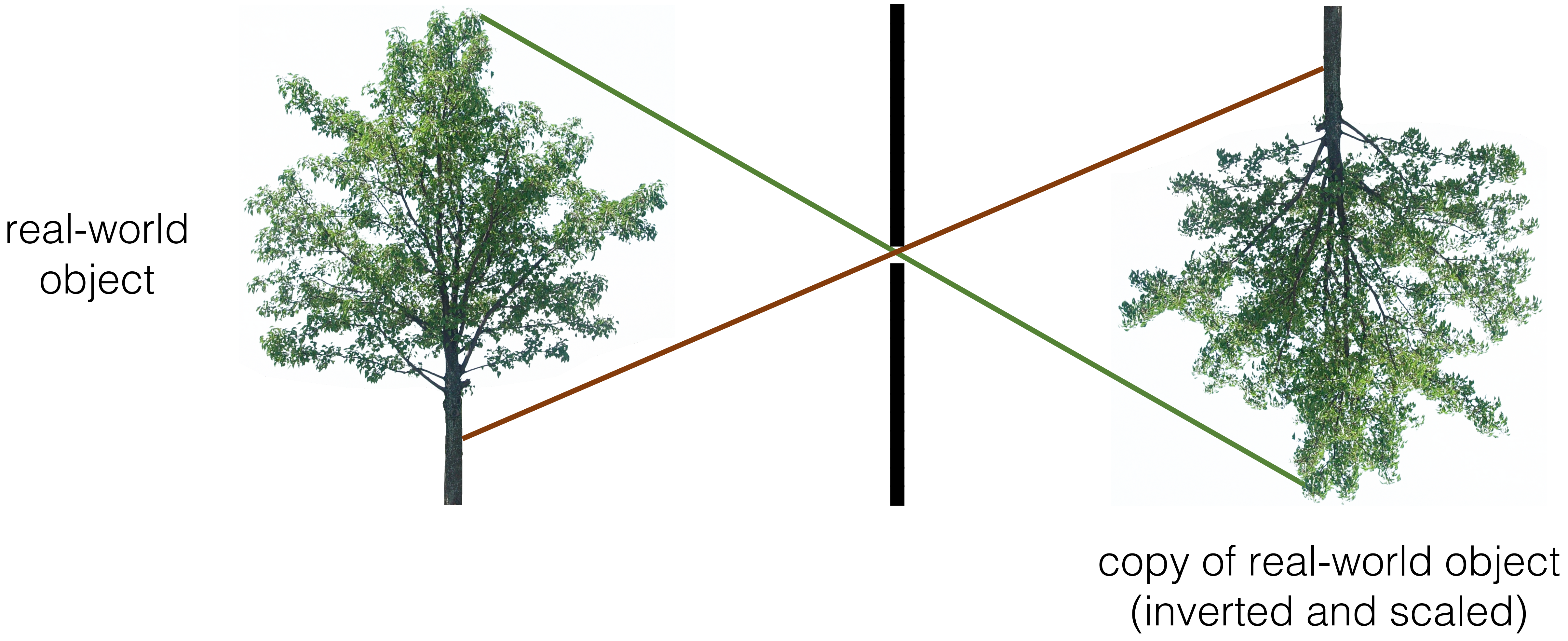


digital sensor (CCD or CMOS)

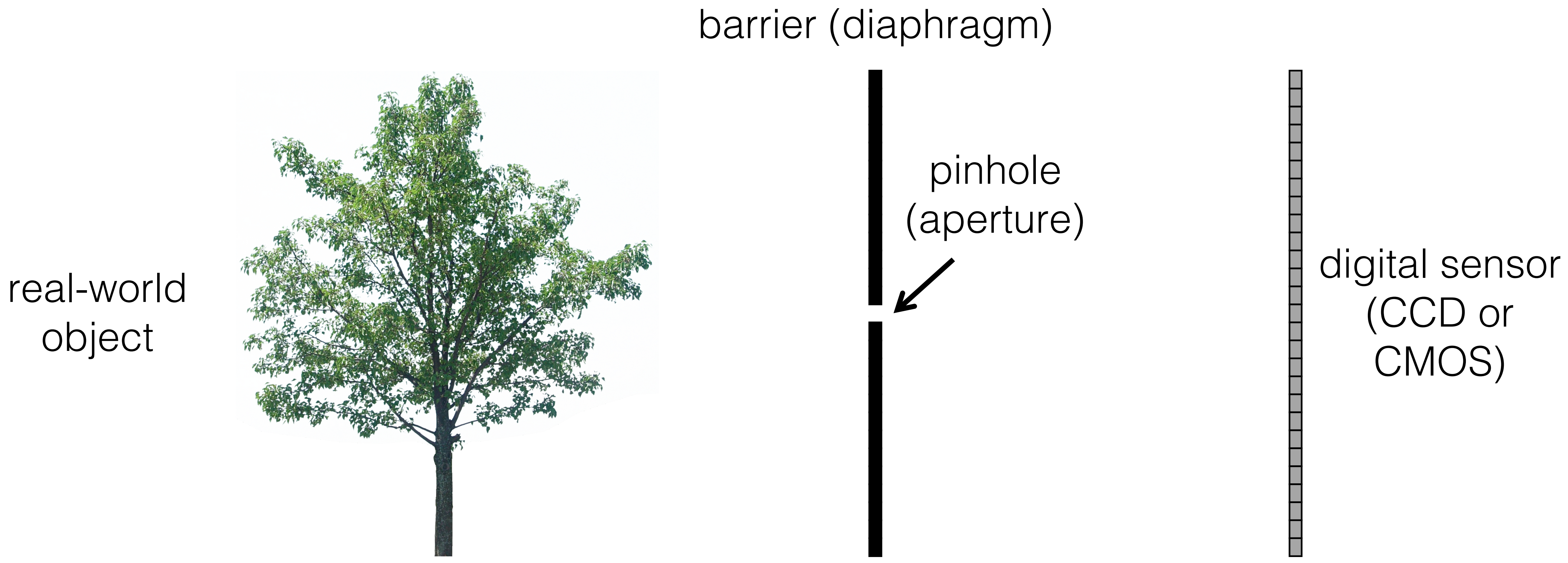
Each scene point contributes to only one sensor pixel

What does the image on the sensor look like?

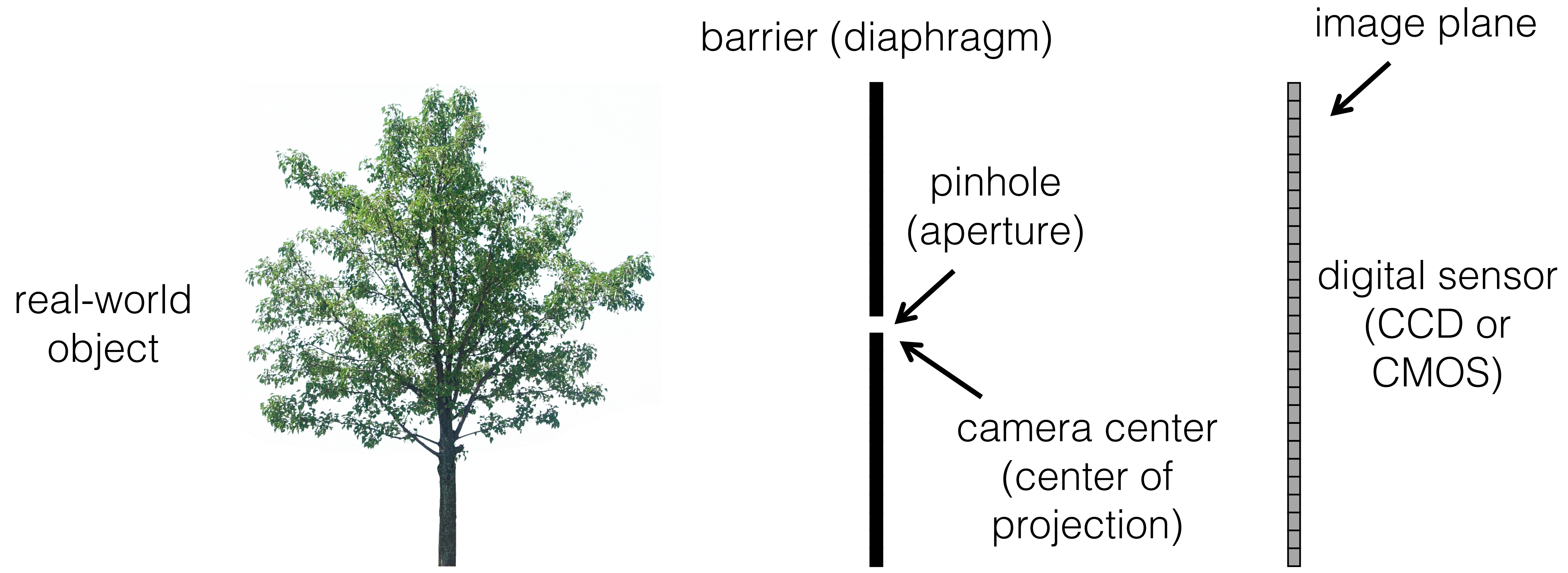
Pinhole imaging



Pinhole camera terms

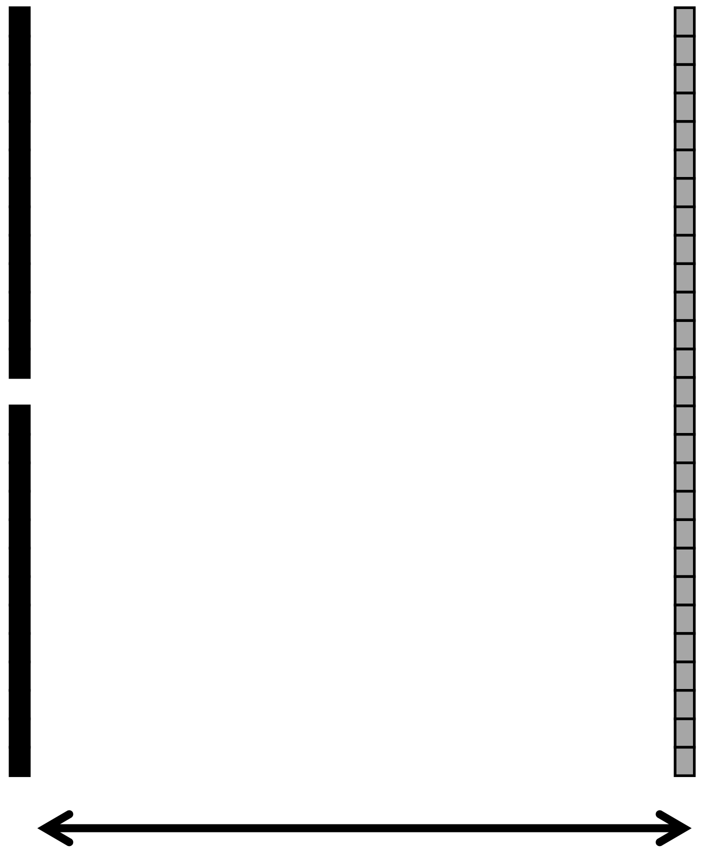


Pinhole camera terms



Focal length

real-world
object

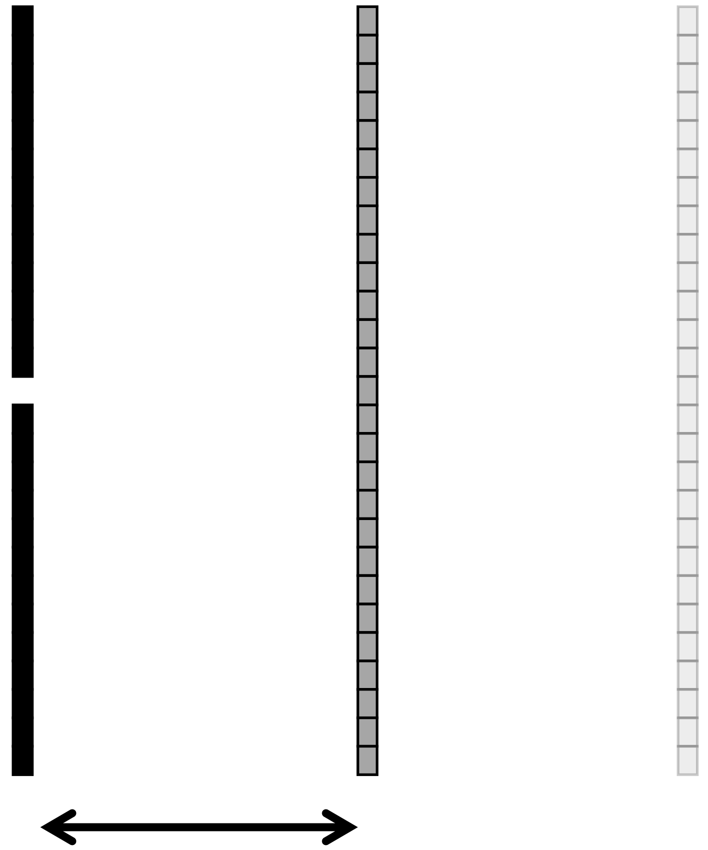


focal length f

Focal length

What happens as we change the focal length?

real-world object

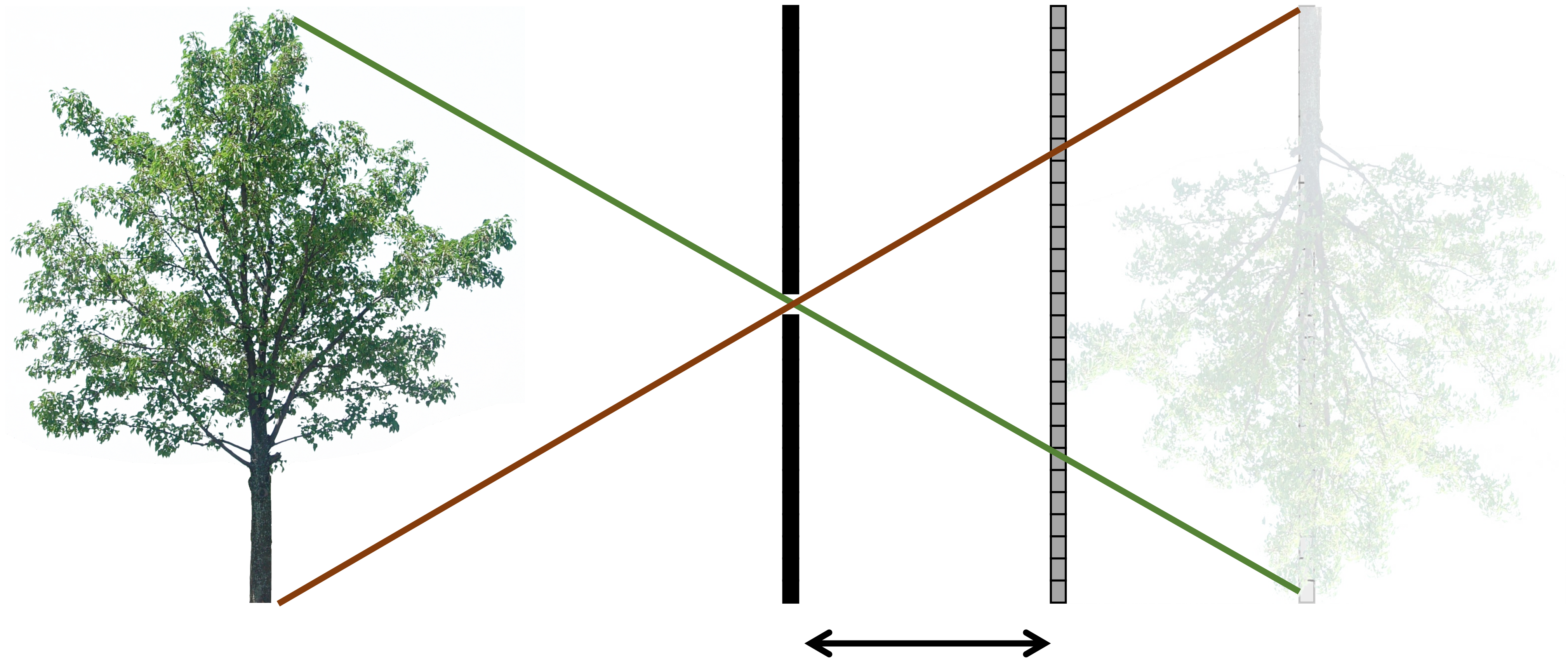


focal length 0.5 f

Focal length

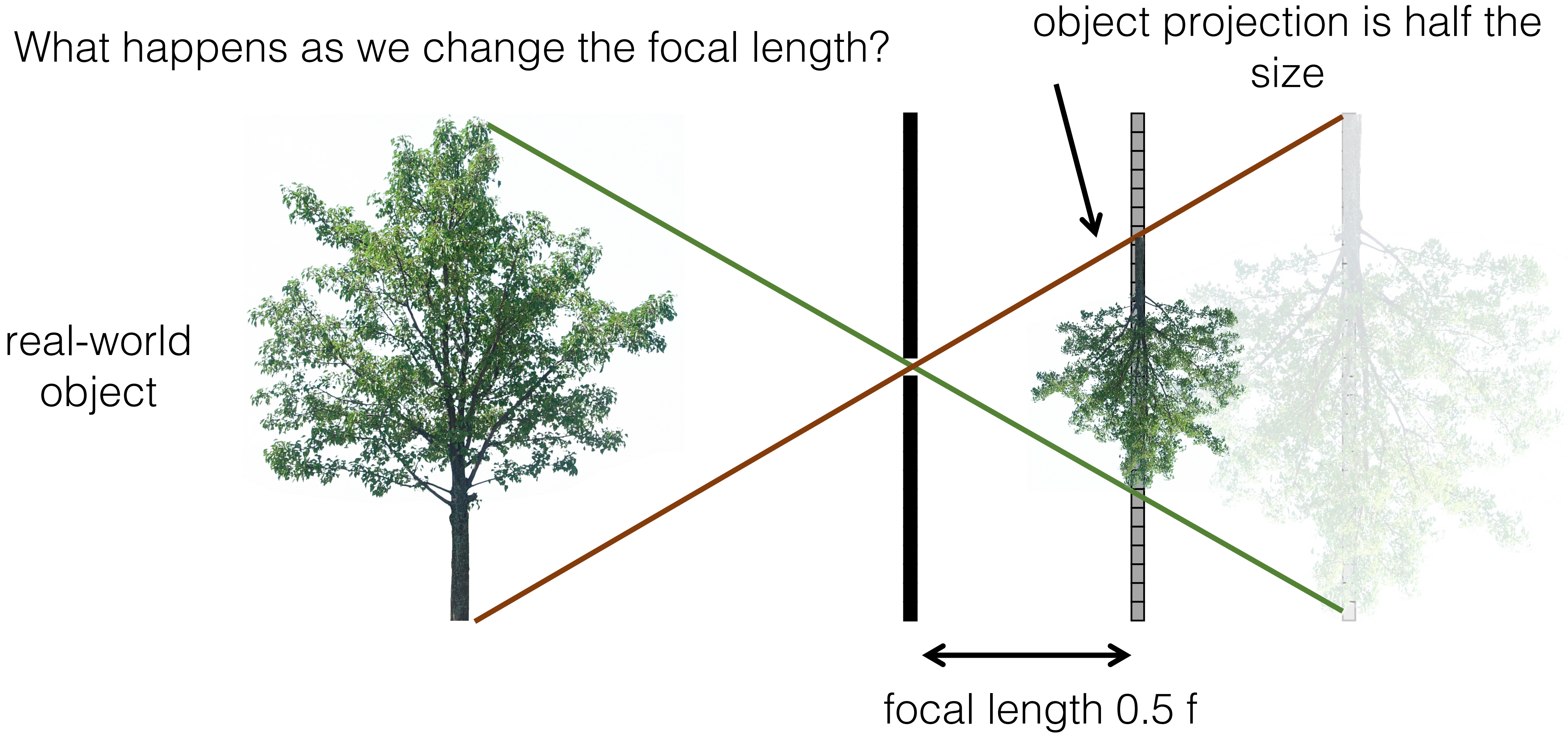
What happens as we change the focal length?

real-world object

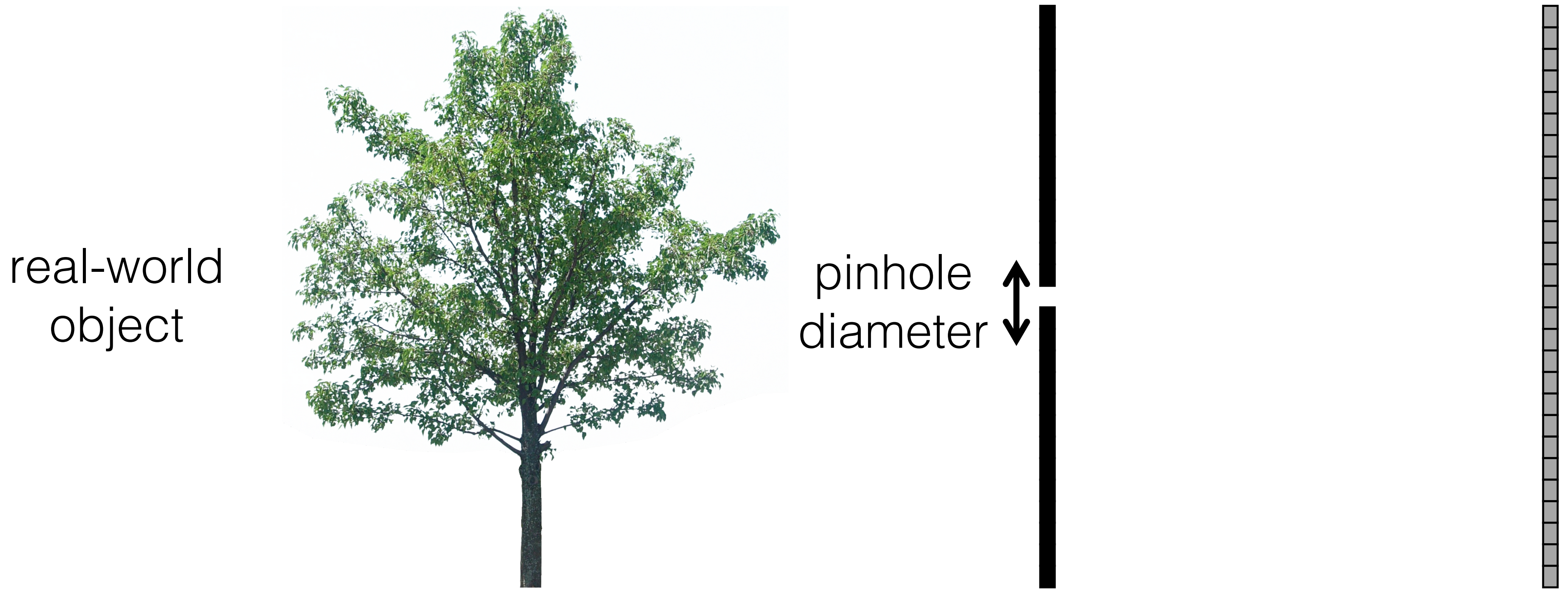


Focal length

What happens as we change the focal length?



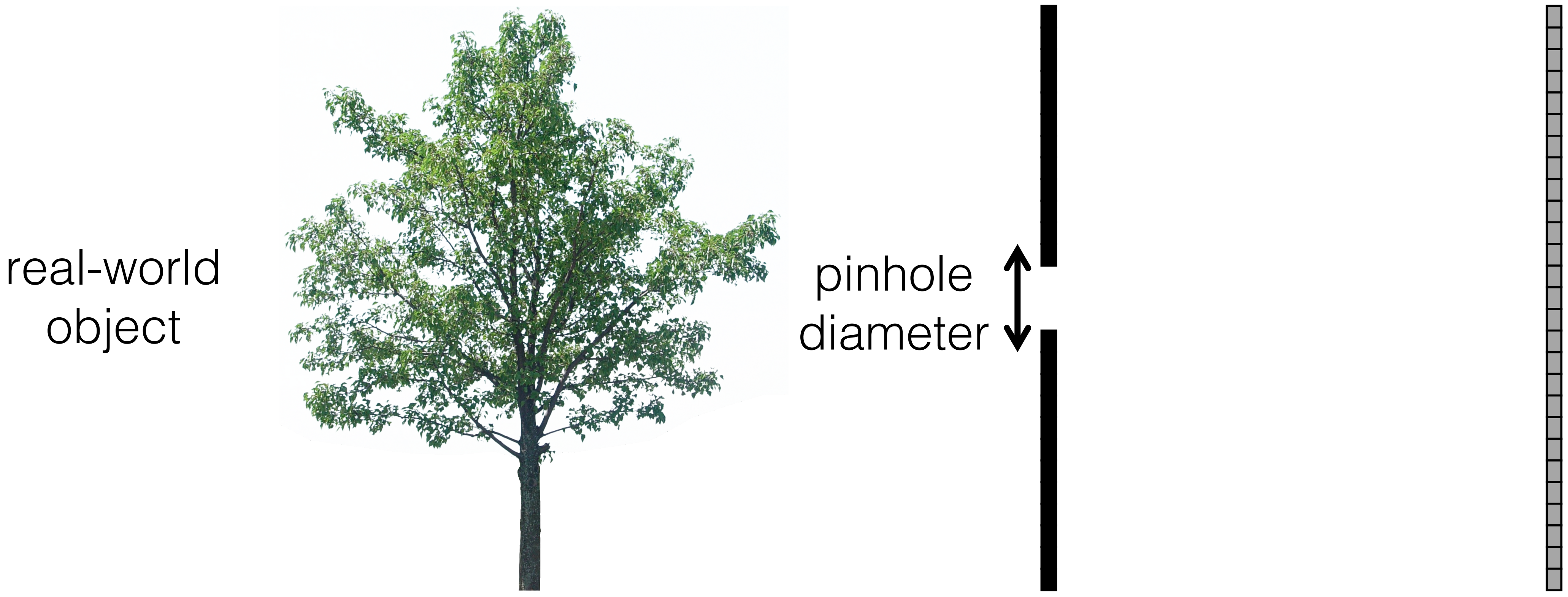
Pinhole size



Ideal pinhole has infinitesimally small size
- In practice that is impossible.

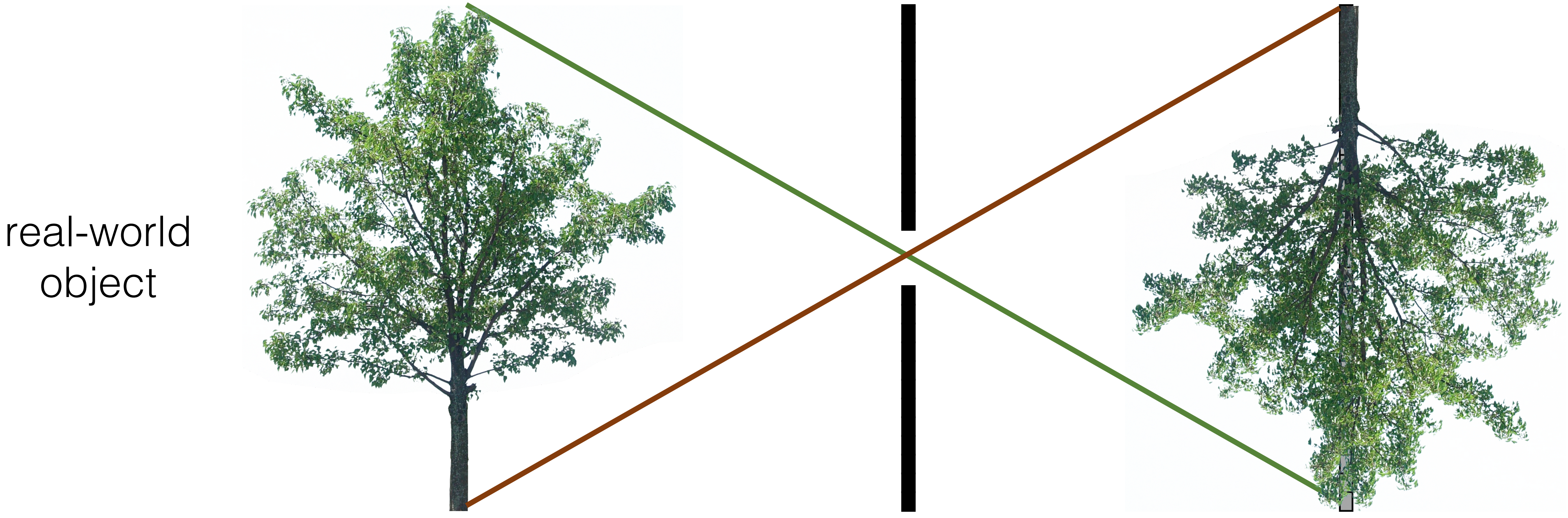
Pinhole size

What happens as we change the pinhole diameter?



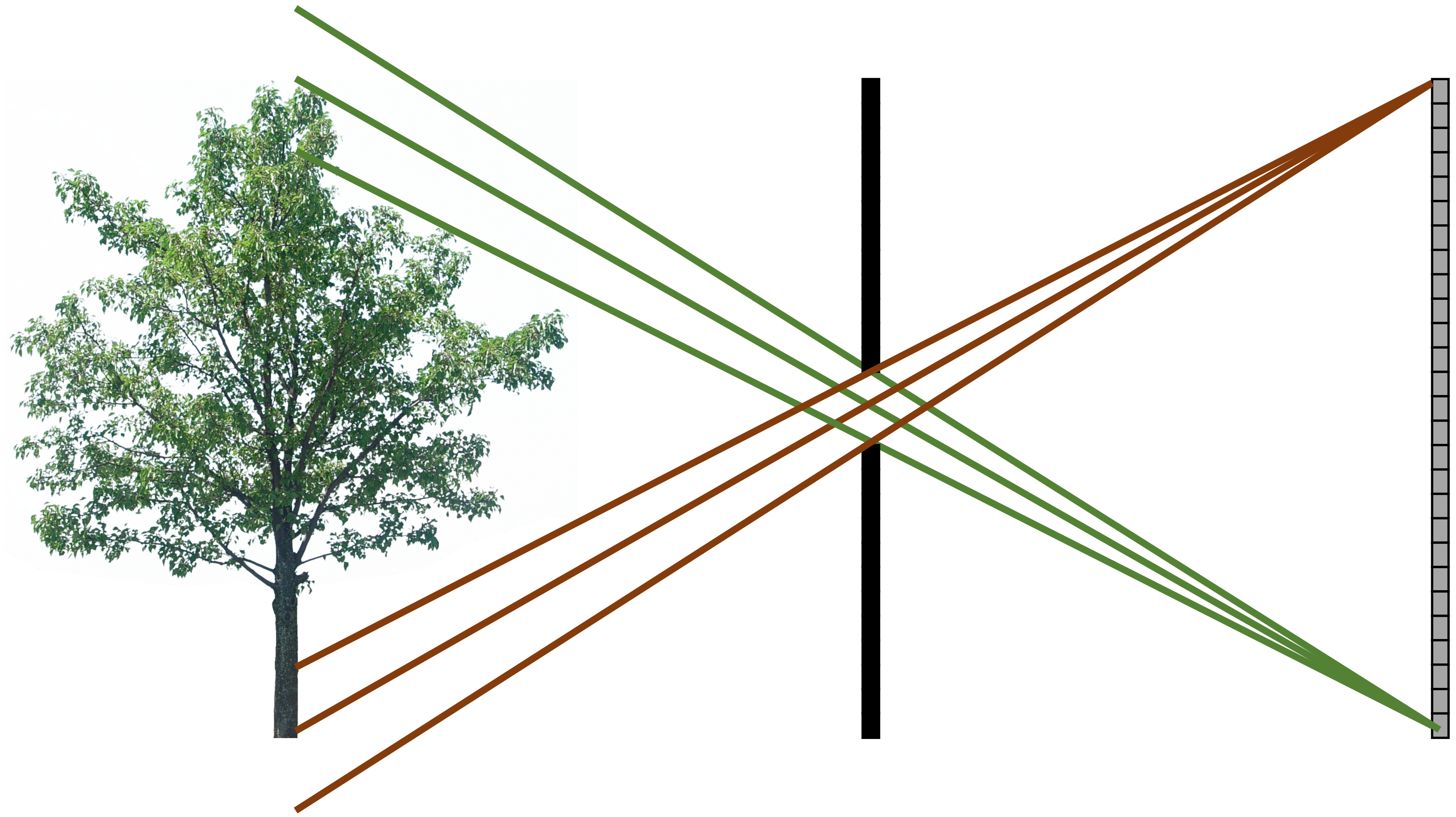
Pinhole size

What happens as we change the pinhole diameter?



Pinhole size

What happens as we change the pinhole diameter?

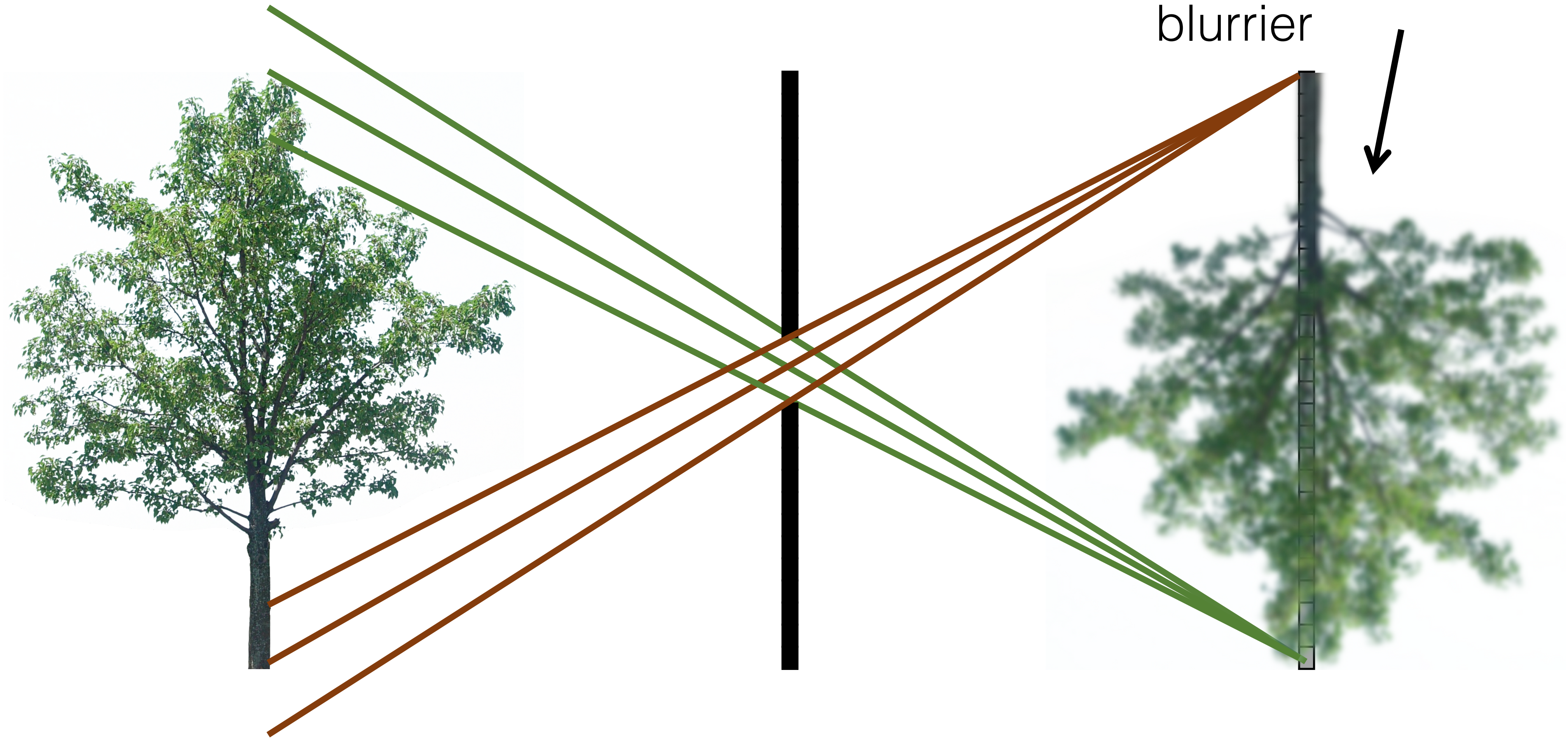


real-world object

Pinhole size

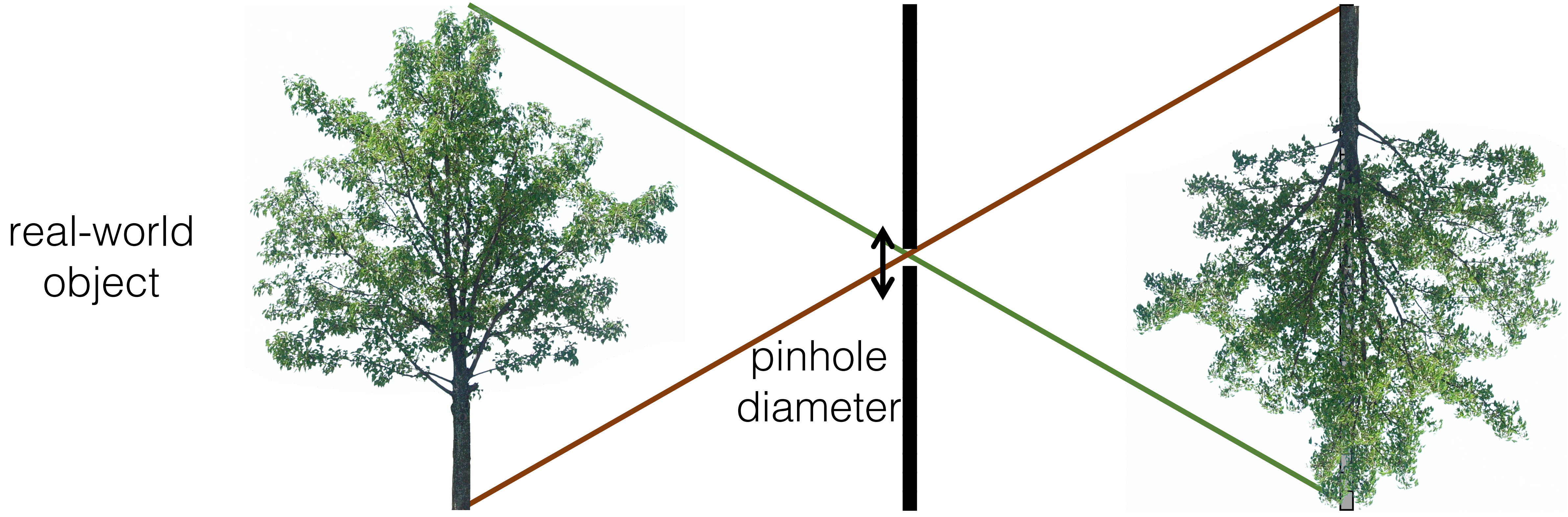
What happens as we change the pinhole diameter?

object projection becomes blurrier



Pinhole size

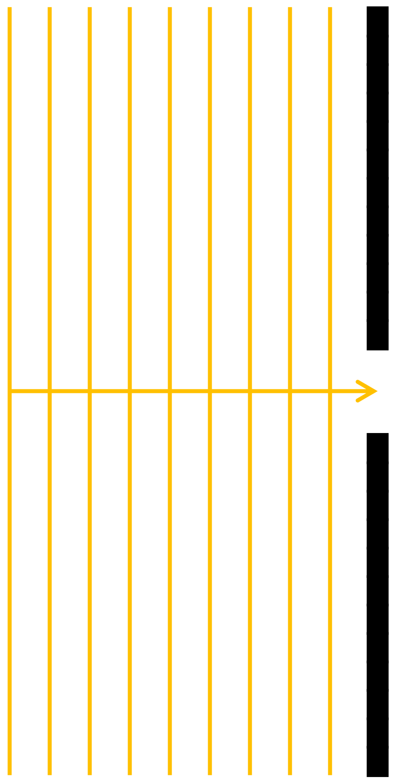
What happens as we change the pinhole diameter?



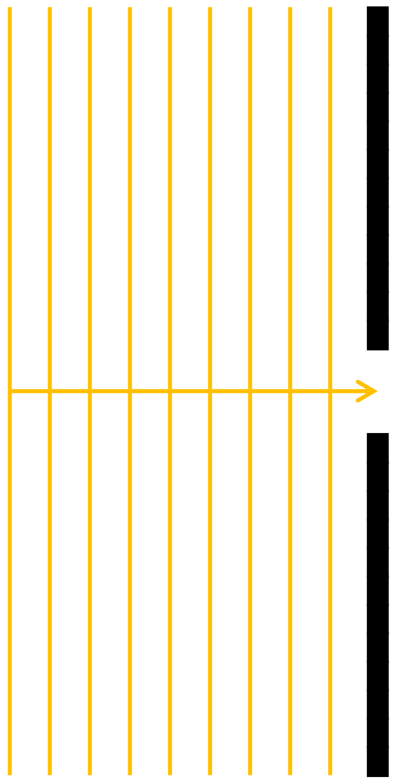
Will the image keep getting sharper the smaller we make the pinhole?

Diffraction limit

A consequence of the wave nature of light



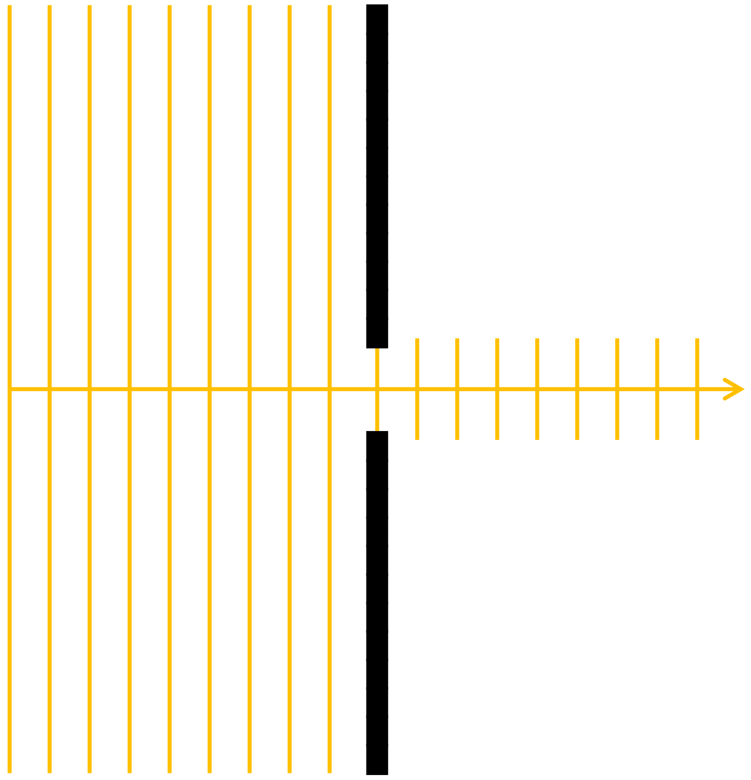
What do geometric optics predict will happen?



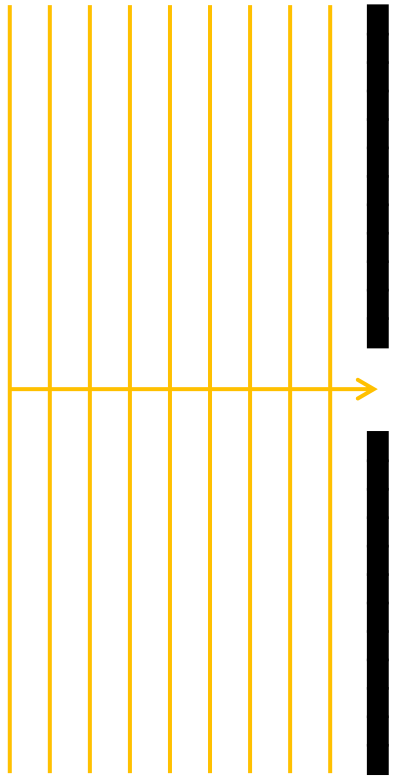
What do wave optics predict will happen?

Diffraction limit

A consequence of the wave nature of light



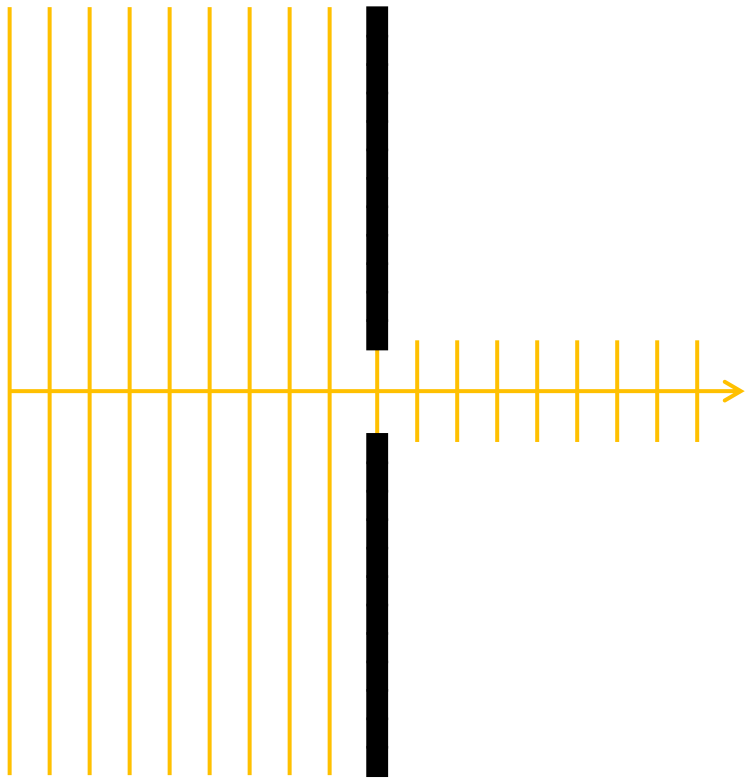
What do geometric optics predict will happen?



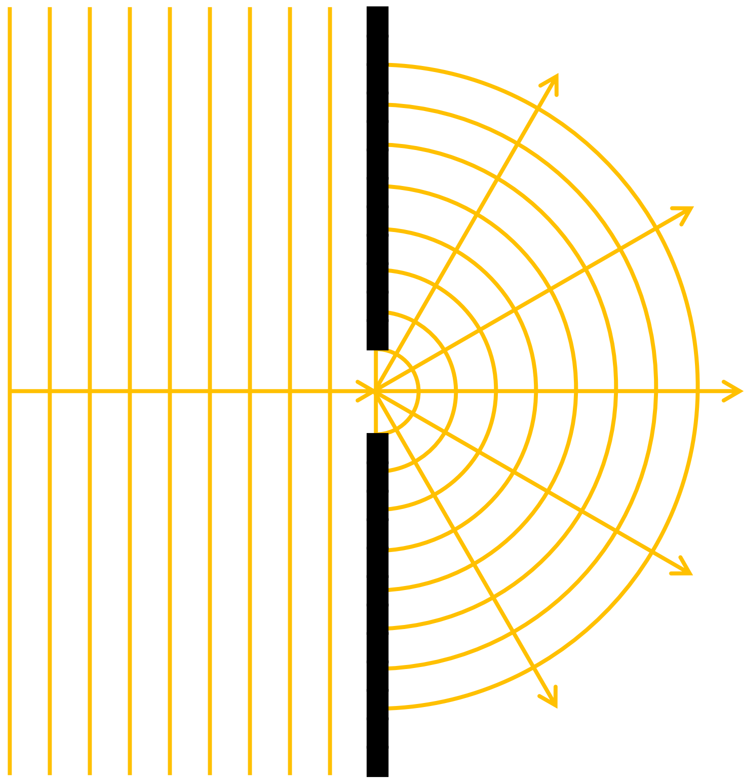
What do wave optics predict will happen?

Diffraction limit

A consequence of the wave nature of light



What do geometric optics predict will happen?

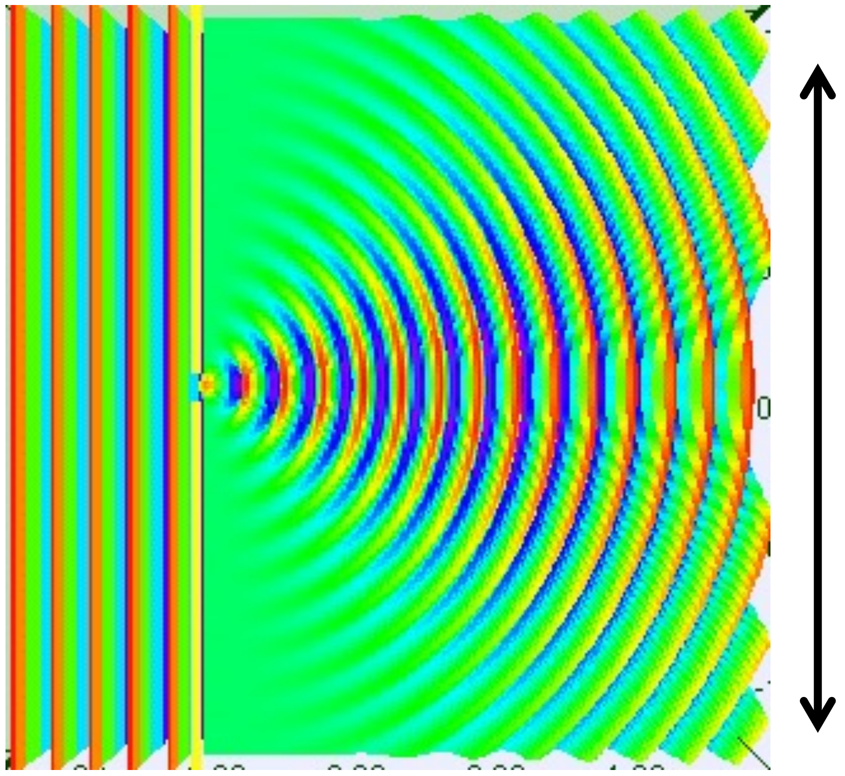


What do wave optics predict will happen?

Diffraction limit

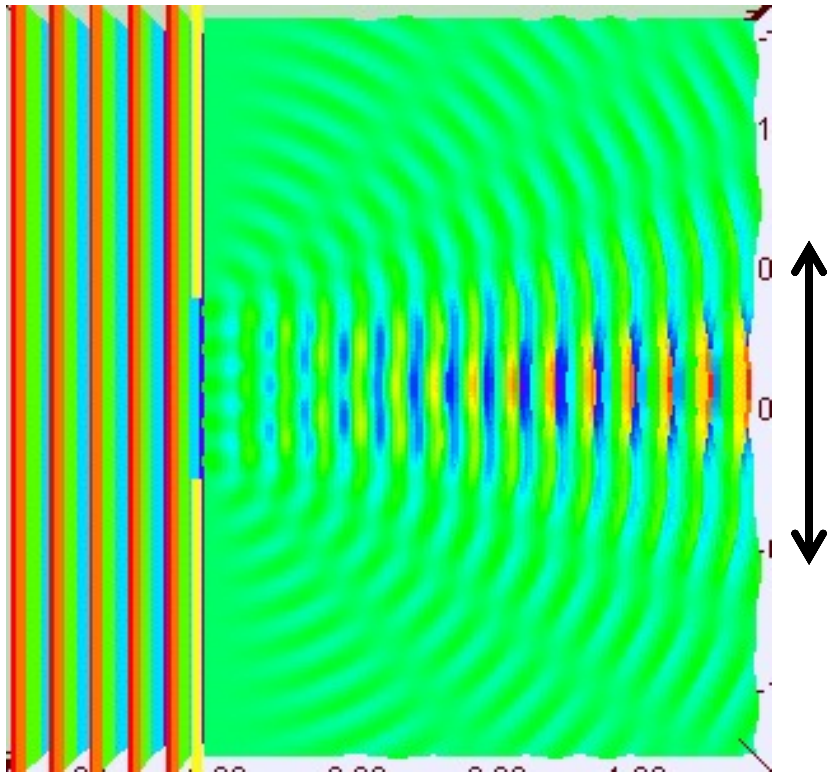
Diffraction pattern = Fourier transform of the pinhole.

- Smaller pinhole means bigger Fourier spectrum.
- Smaller pinhole means more diffraction.



wide diffraction pattern

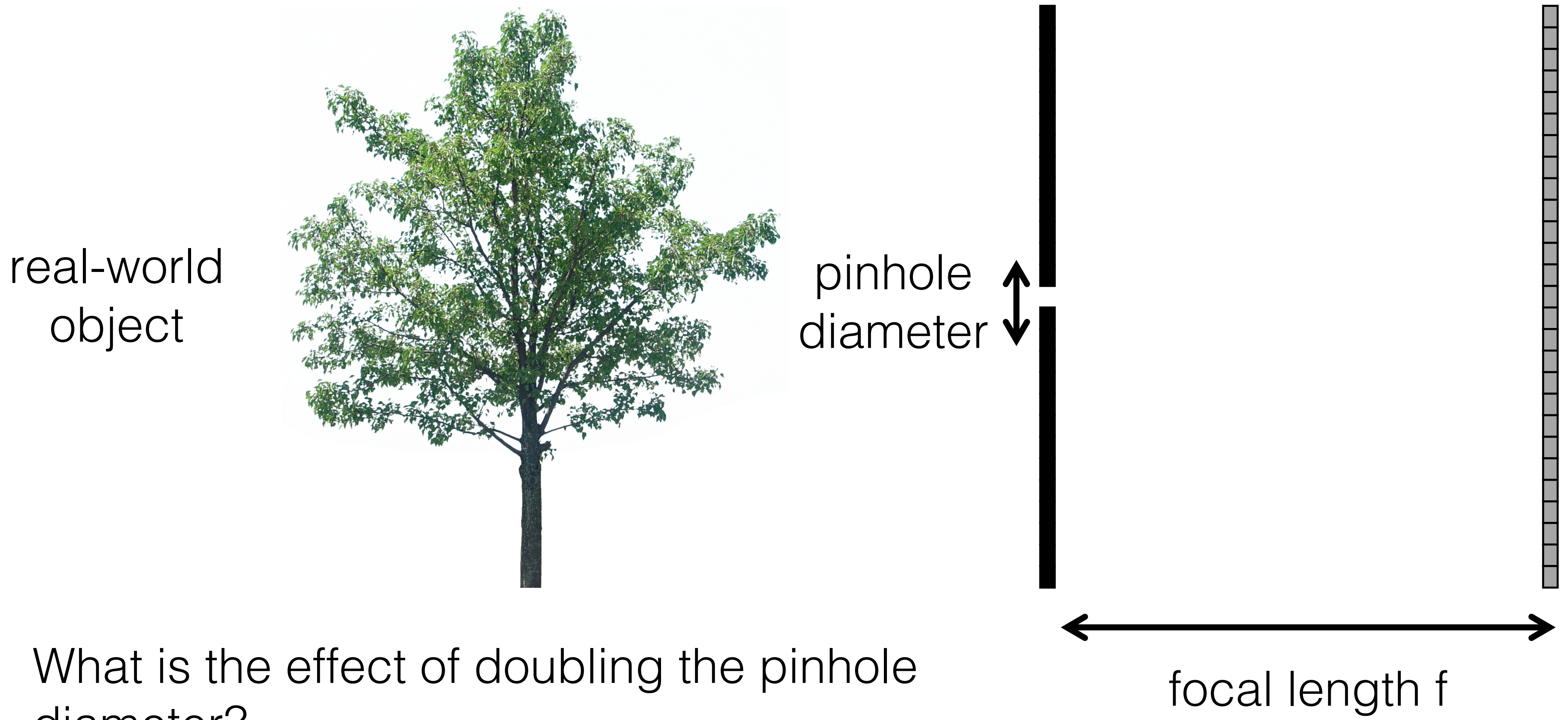
small pinhole



narrow diffraction pattern

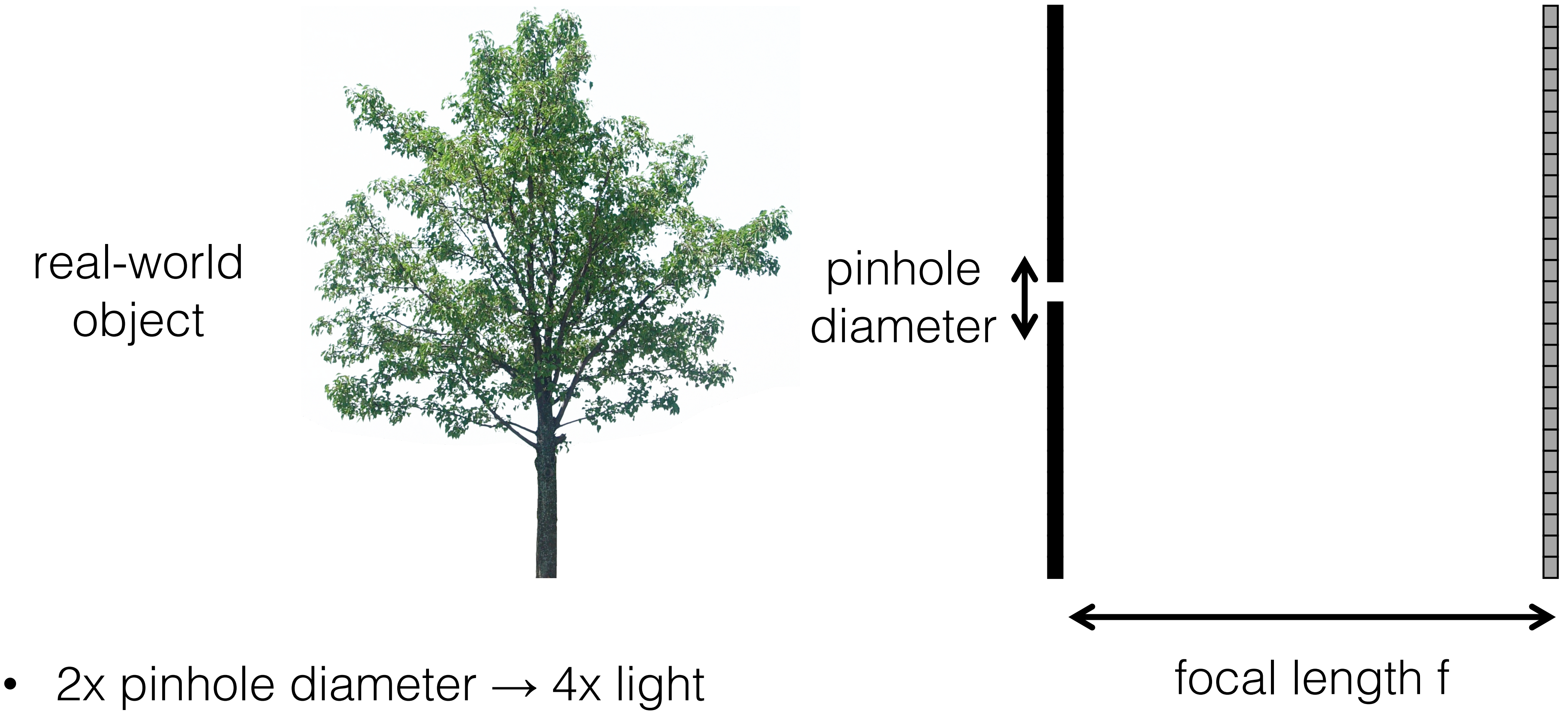
large pinhole

What about light efficiency?



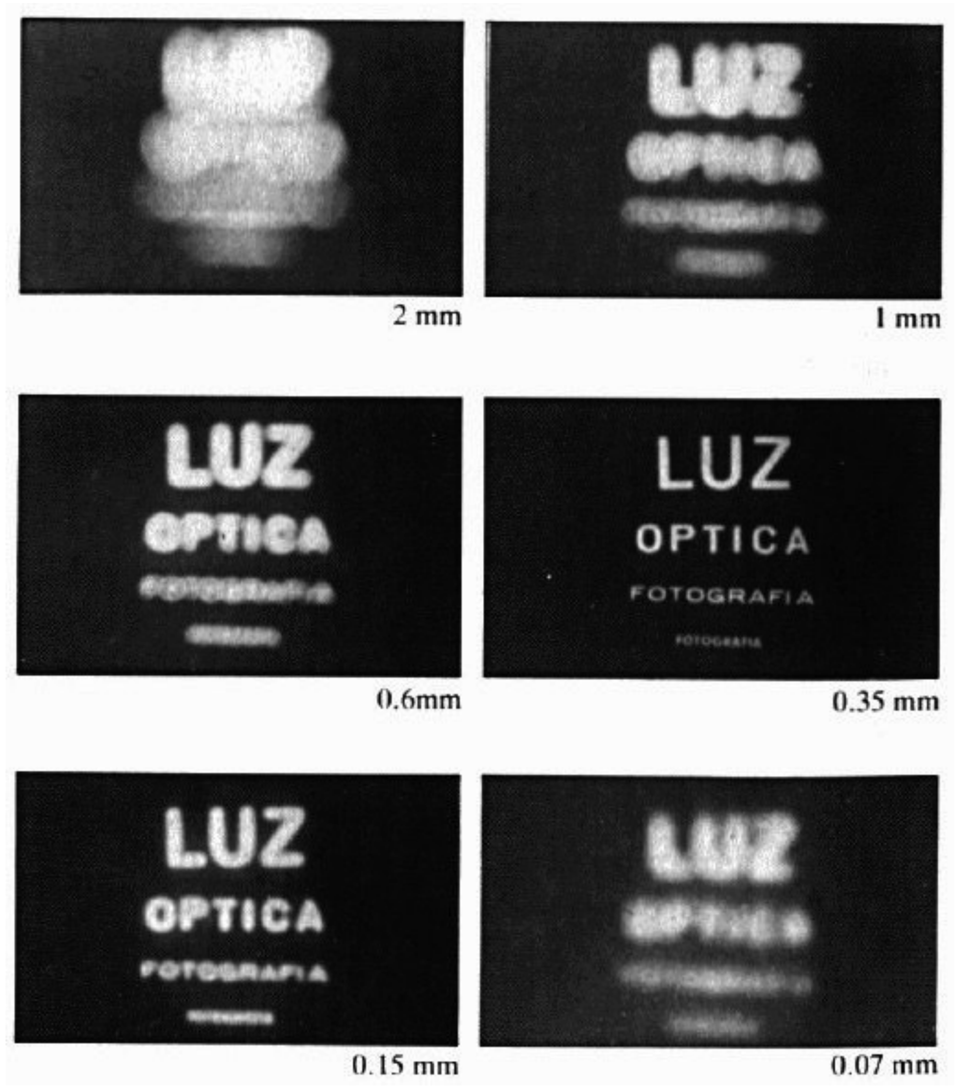
- What is the effect of doubling the pinhole diameter?
- What is the effect of doubling the focal length?

What about light efficiency?



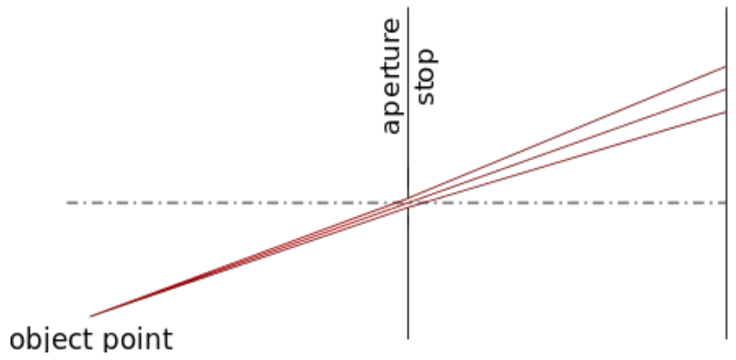
- 2x pinhole diameter → 4x light
- 2x focal length → ¼x light

Shrinking the Aperture

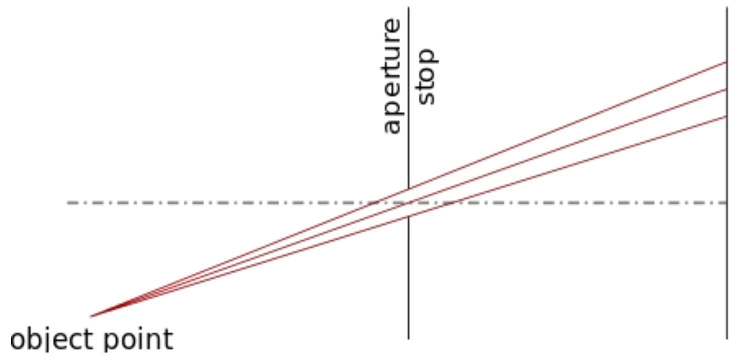


[Source: N. Snavely]

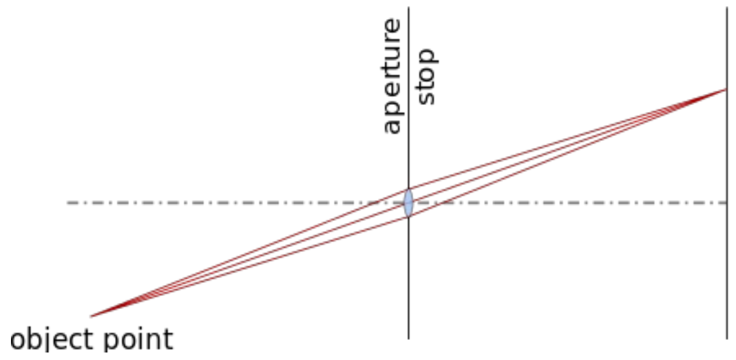
Adding a Lens



Small Pinhole



Big pinhole

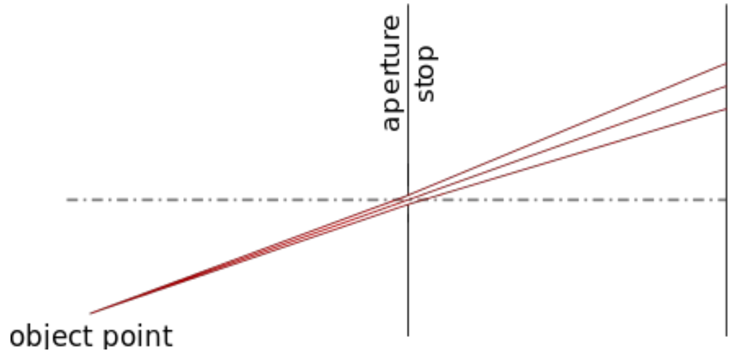


Lens

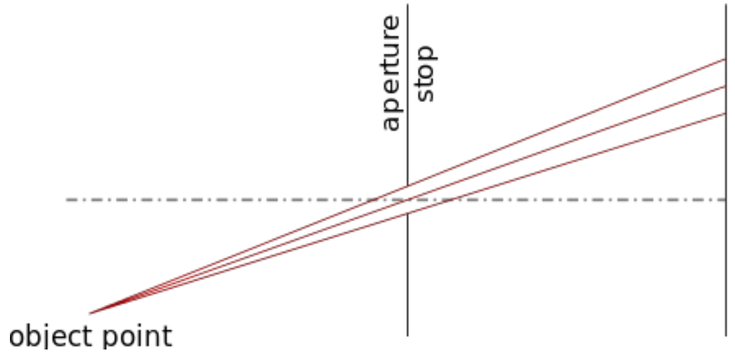
- A lens focuses light onto the film

[Source: N. Snavely]

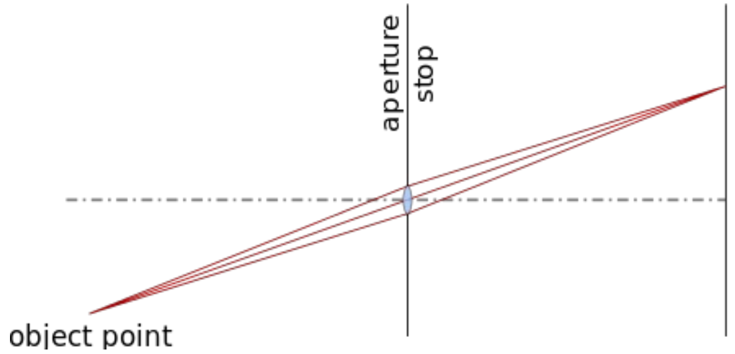
Adding a Lens



Small Pinhole



Big pinhole

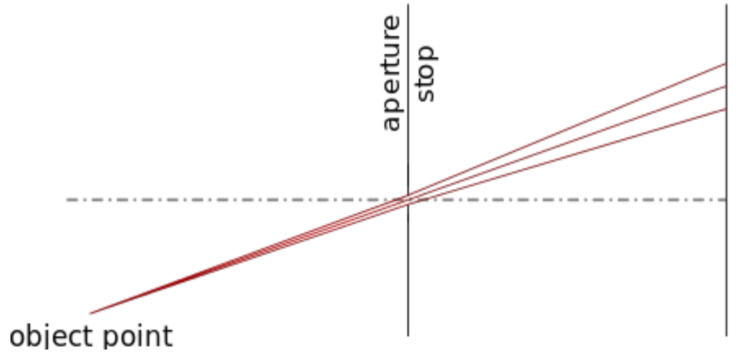


Lens

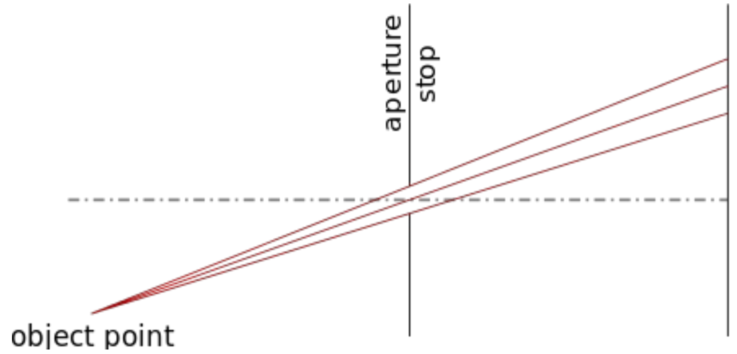
- A lens focuses light onto the film
- There is a specific distance at which objects are in focus

[Source: N. Snavely]

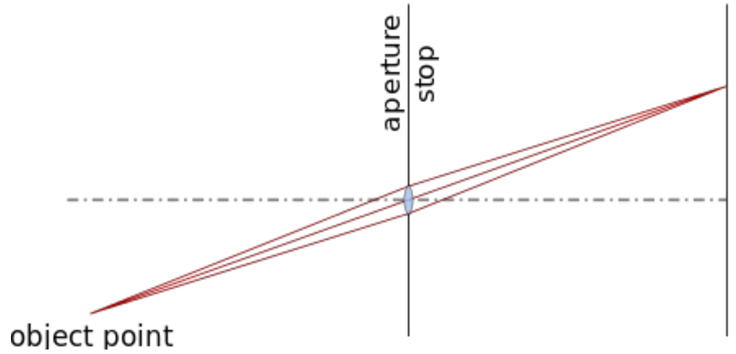
Adding a Lens



Small Pinhole



Big pinhole

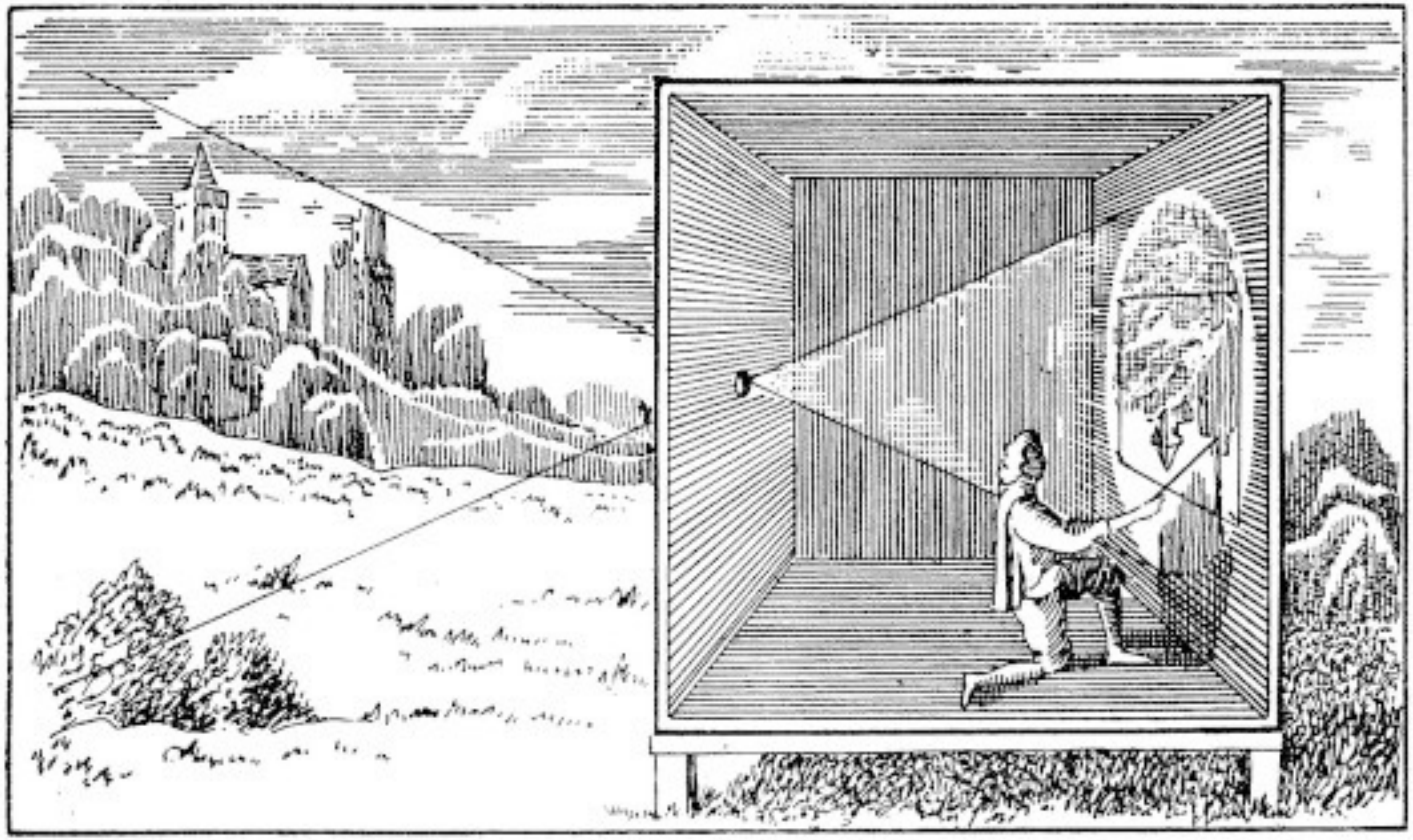


Lens

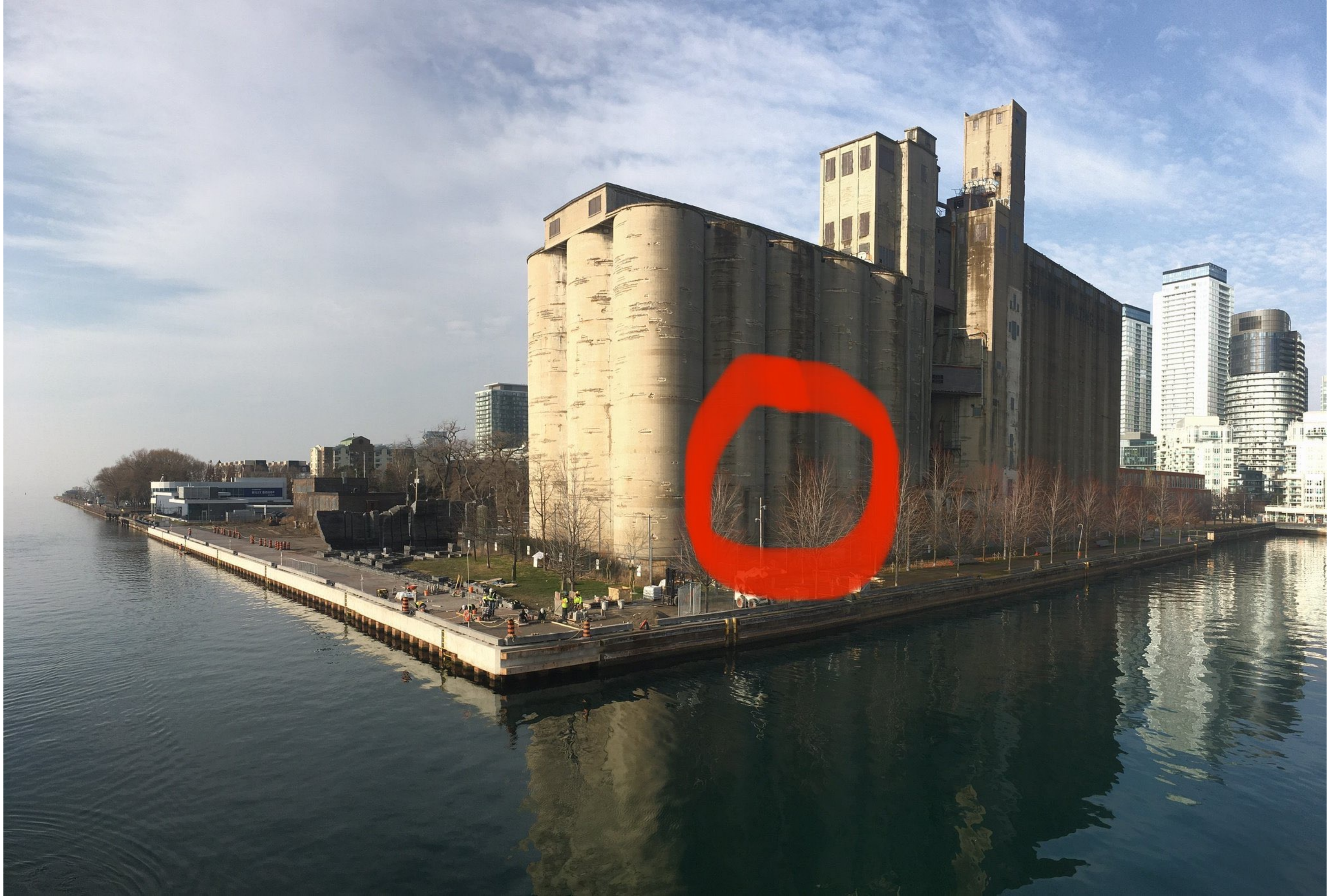
- A lens focuses light onto the film
- There is a specific distance at which objects are in focus
- Changing the shape of the lens changes this distance

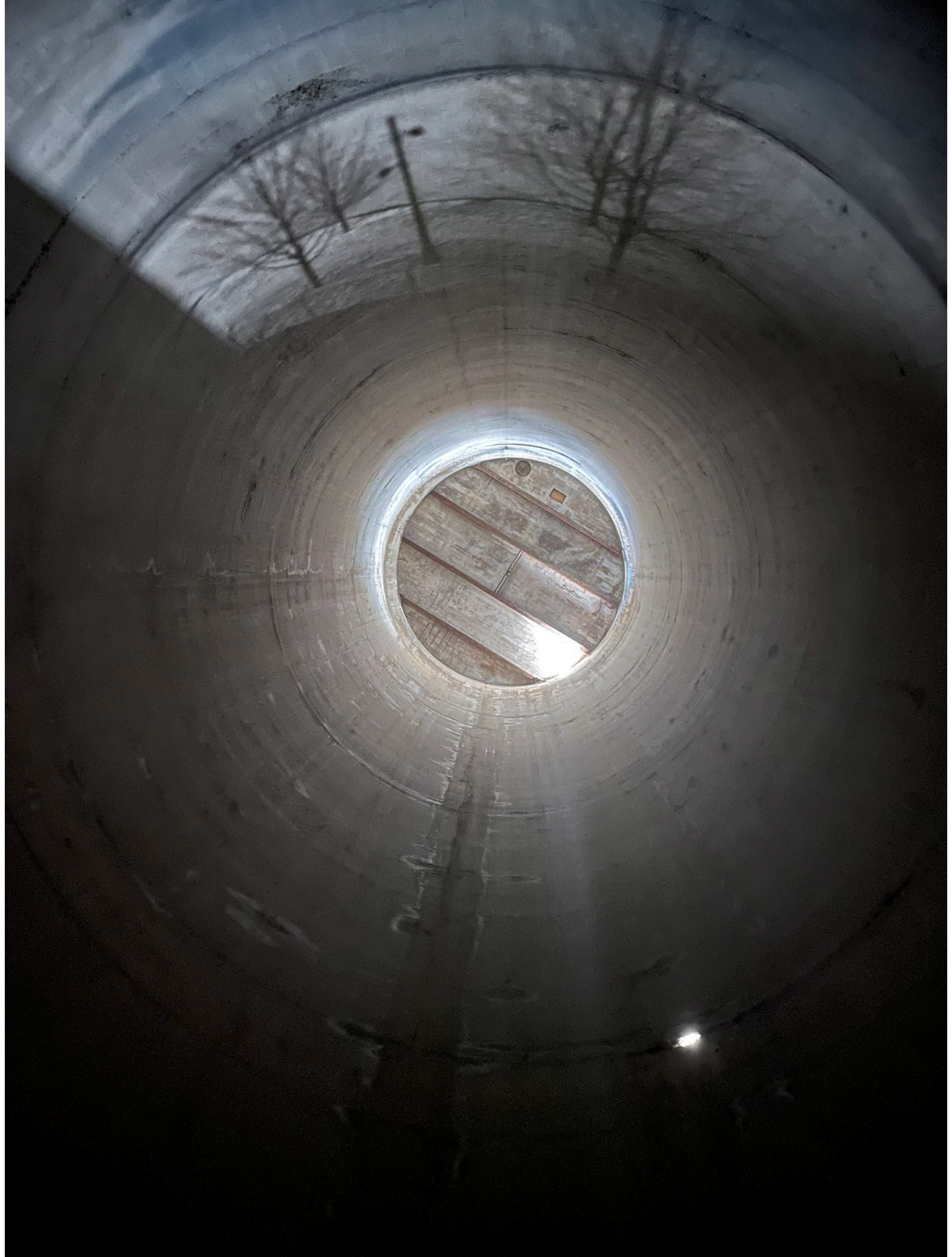
[Source: N. Snavely]

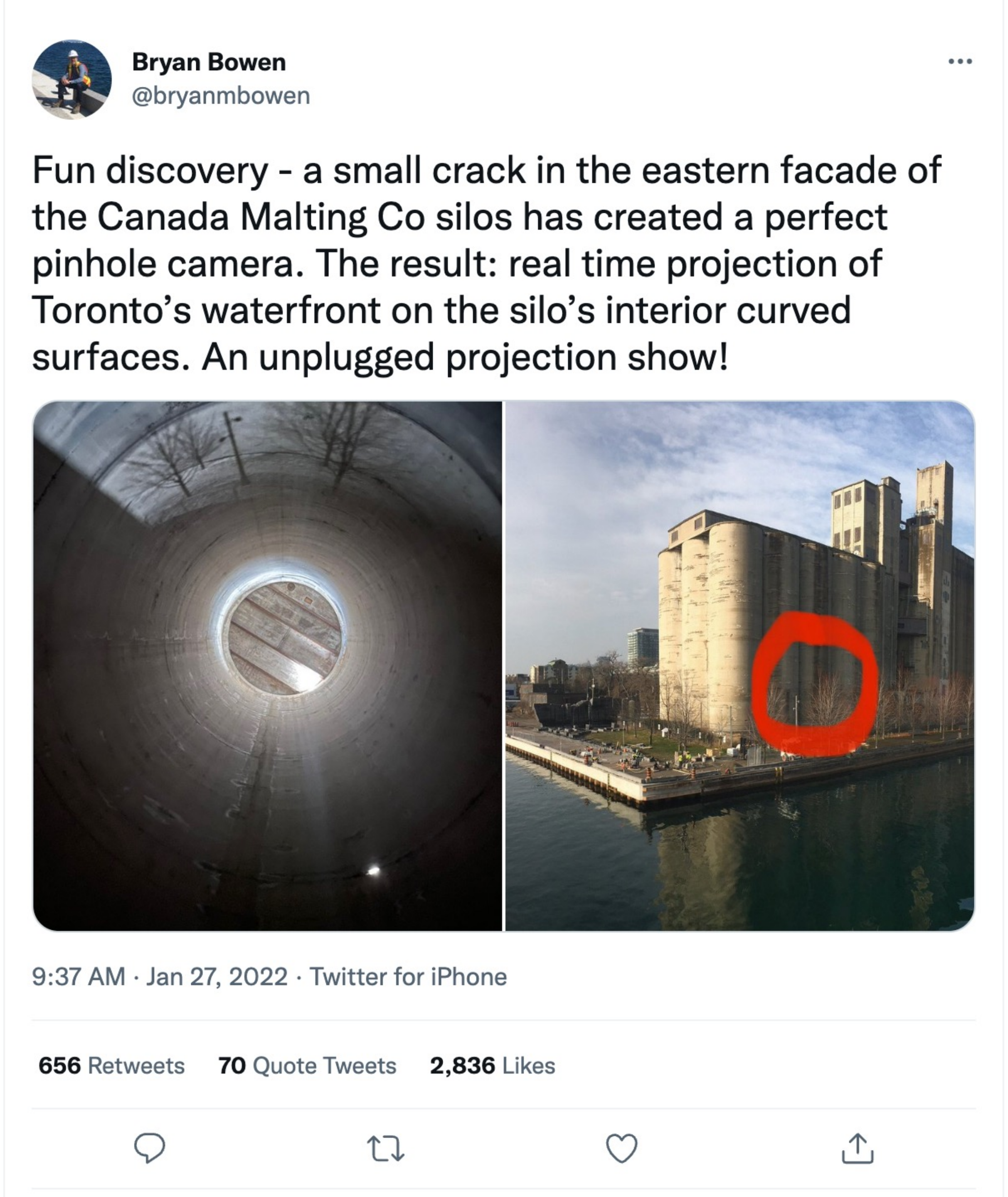
Pinhole Camera / Camera Obscura



Mo-Ti (Chinese Philosopher) 470-390 BC


BILLY BISHOP





Bryan Bowen
@bryanmbowen

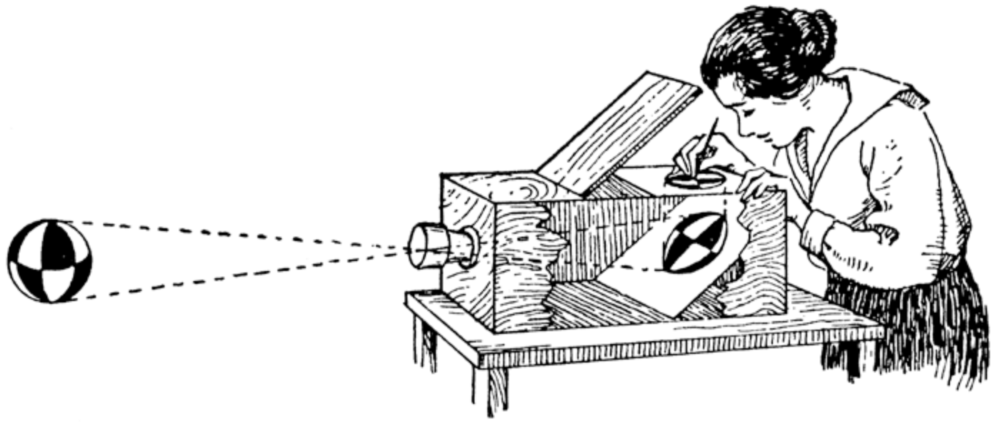
Fun discovery - a small crack in the eastern facade of the Canada Malting Co silos has created a perfect pinhole camera. The result: real time projection of Toronto's waterfront on the silo's interior curved surfaces. An unplugged projection show!

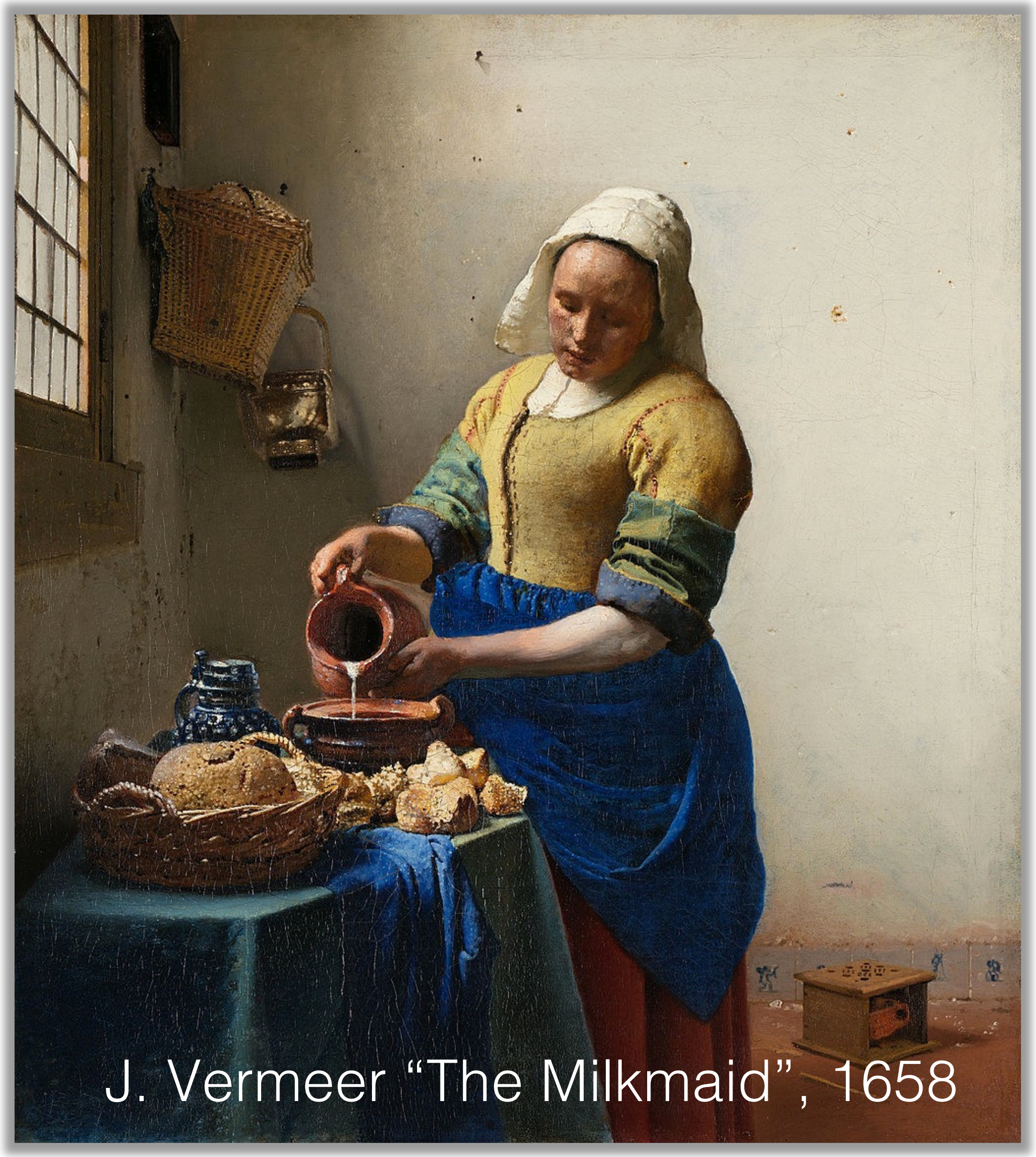
9:37 AM · Jan 27, 2022 · Twitter for iPhone

656 Retweets 70 Quote Tweets 2,836 Likes

Abelardo Morell

Pinhole Camera / Camera Obscura





J. Vermeer “The Milkmaid”, 1658

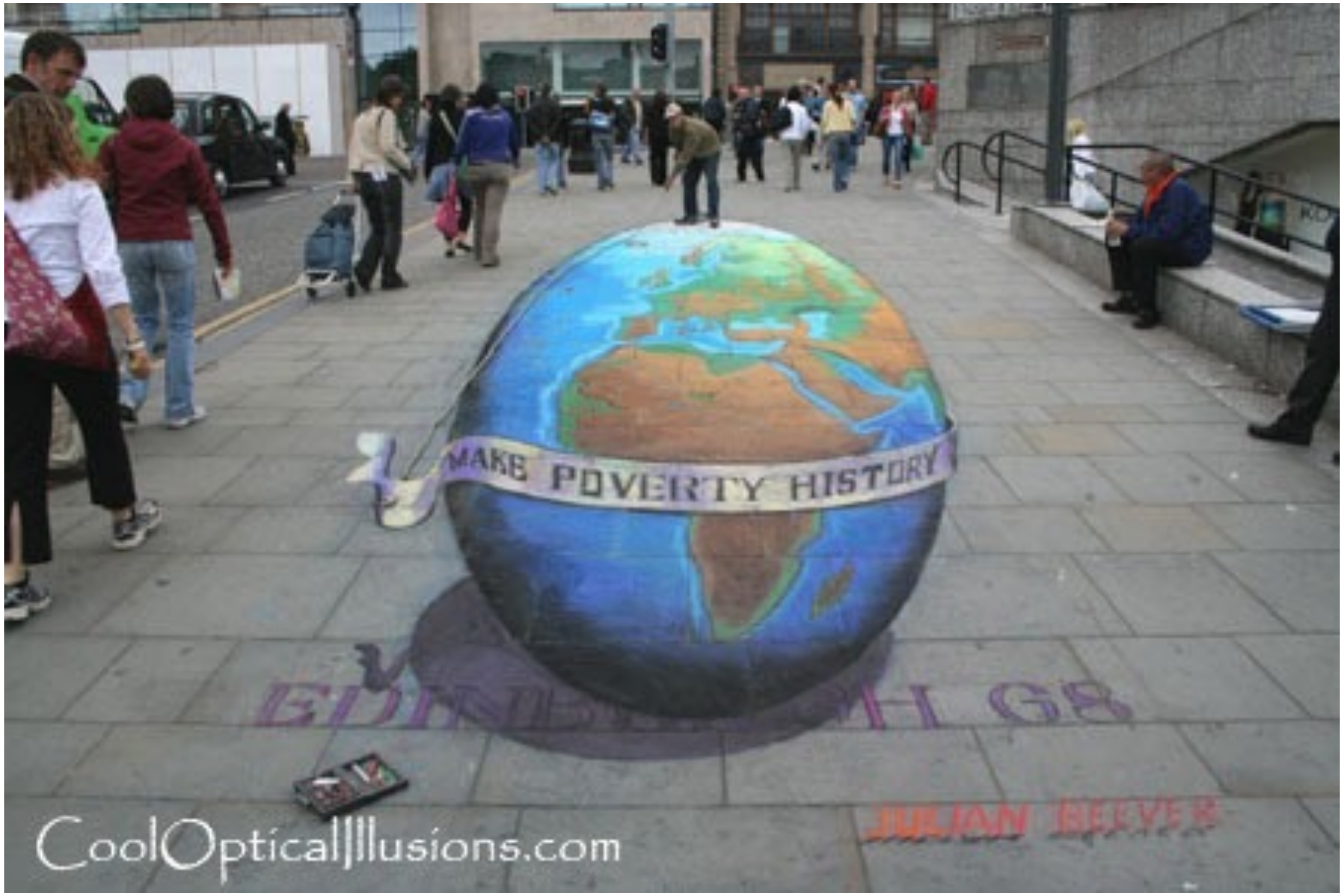
同サイズの《牛乳を注ぐ女》が投影される額縁の前に立ち

Imaging

- Images are 2D projections of real world scene
- Images capture two kinds of information:
  - Geometric: positions, points, lines, curves, etc.
  - Photometric: intensity, color
- Complex 3D-2D relationships
  - Camera models approximate these relationships

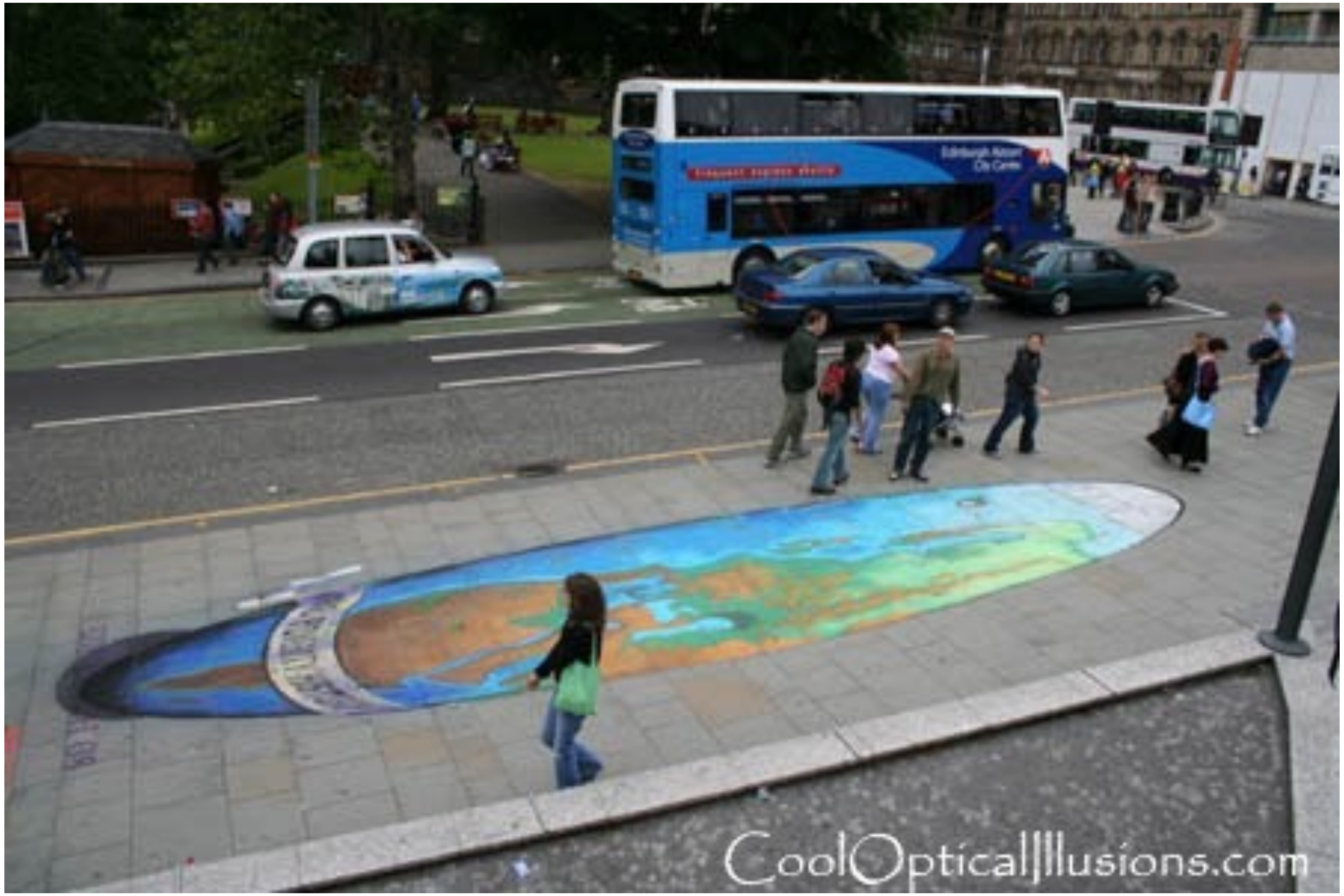
[Source: L.W. Kheng]

Projection



[Source: N. Snavely]

Projection



[Source: N. Snavely]

3D to 2D Projection

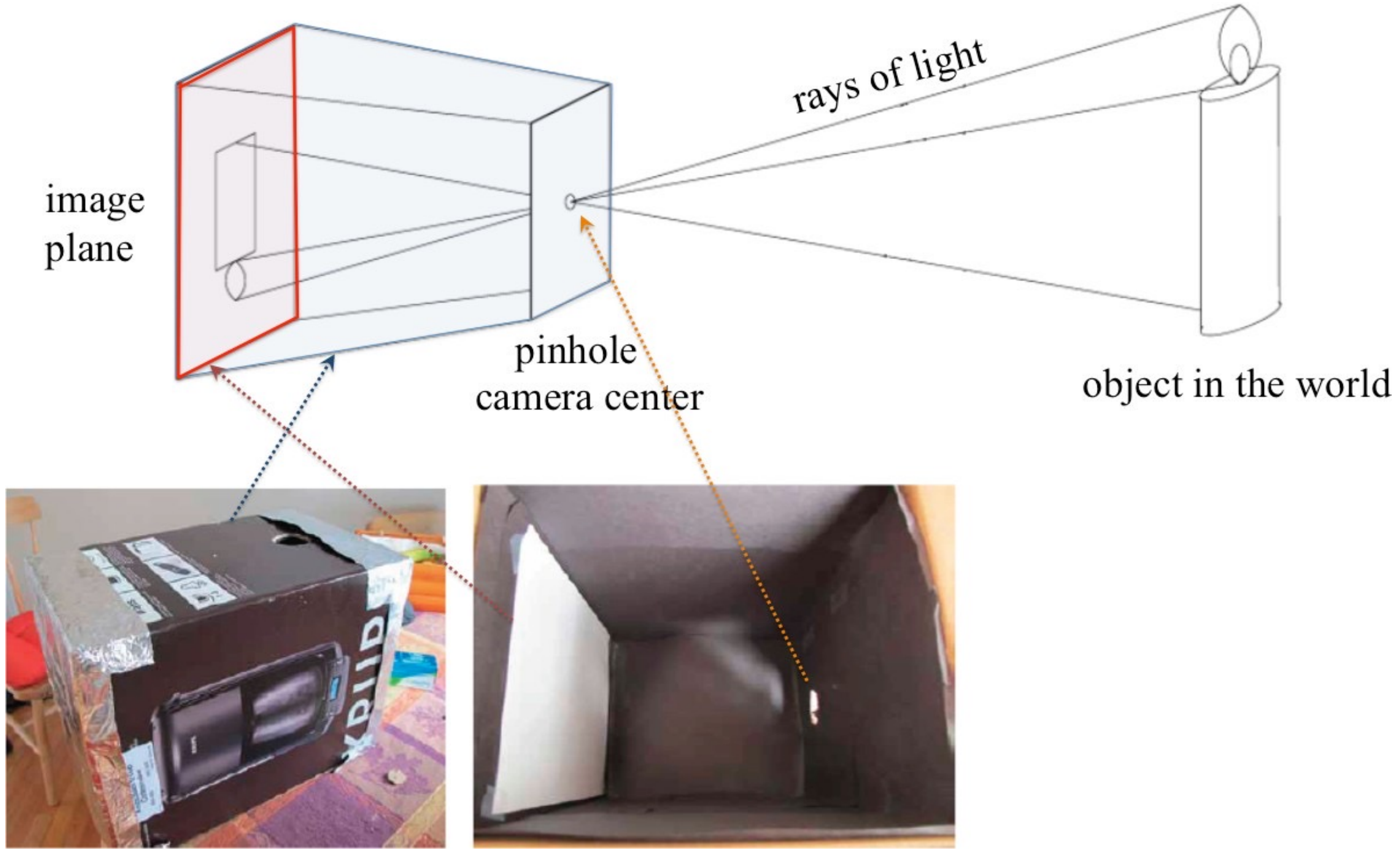
- How are 3D primitives projected onto the image plane?

3D to 2D Projection

- How are 3D primitives projected onto the image plane?
- We can do this using a linear 3D to 2D projection matrix

Modeling Projection

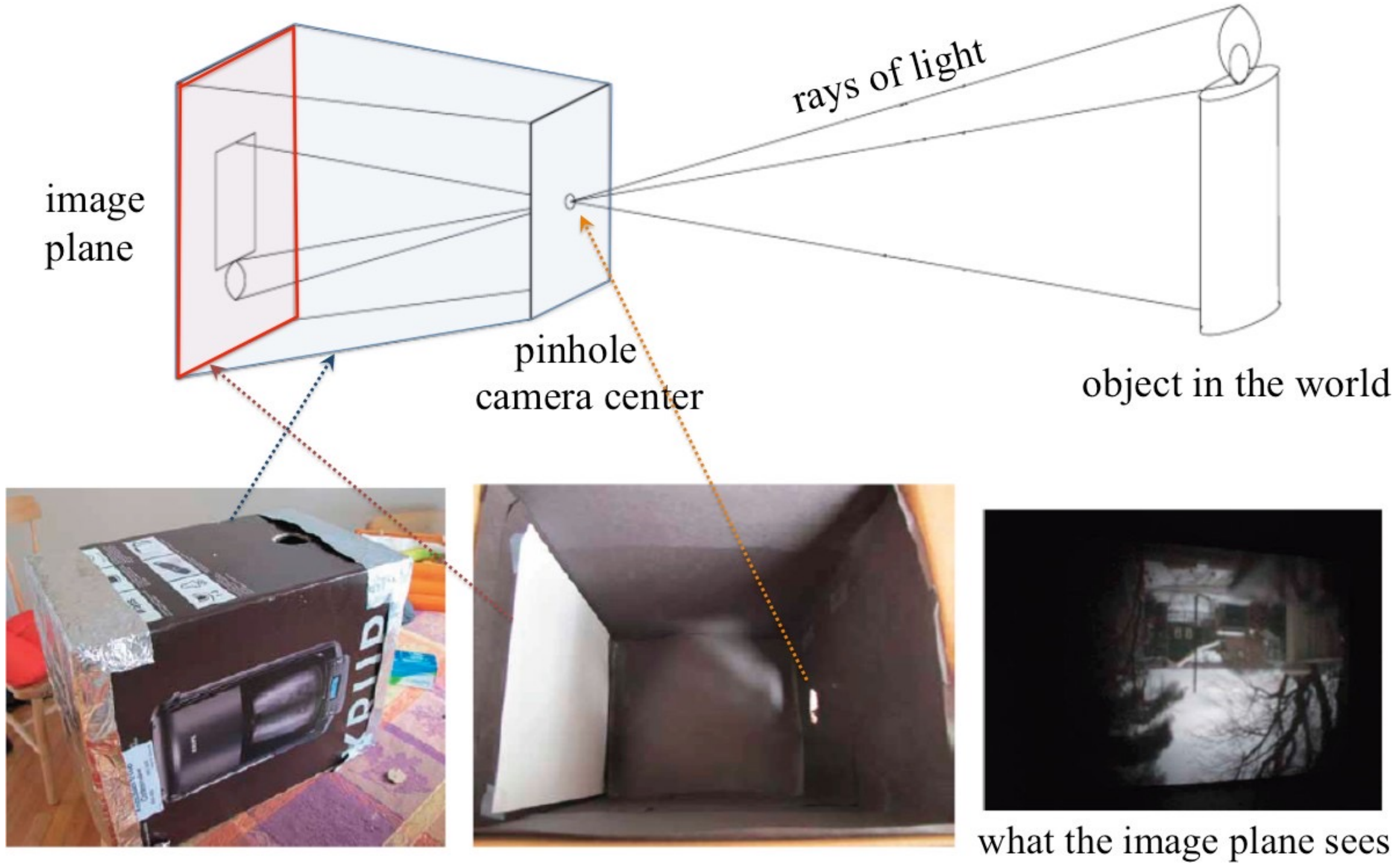
- We will use the pinhole model as an approximation



[Pics from: A. Torralba, Forsyth & Ponce]

Modeling Projection

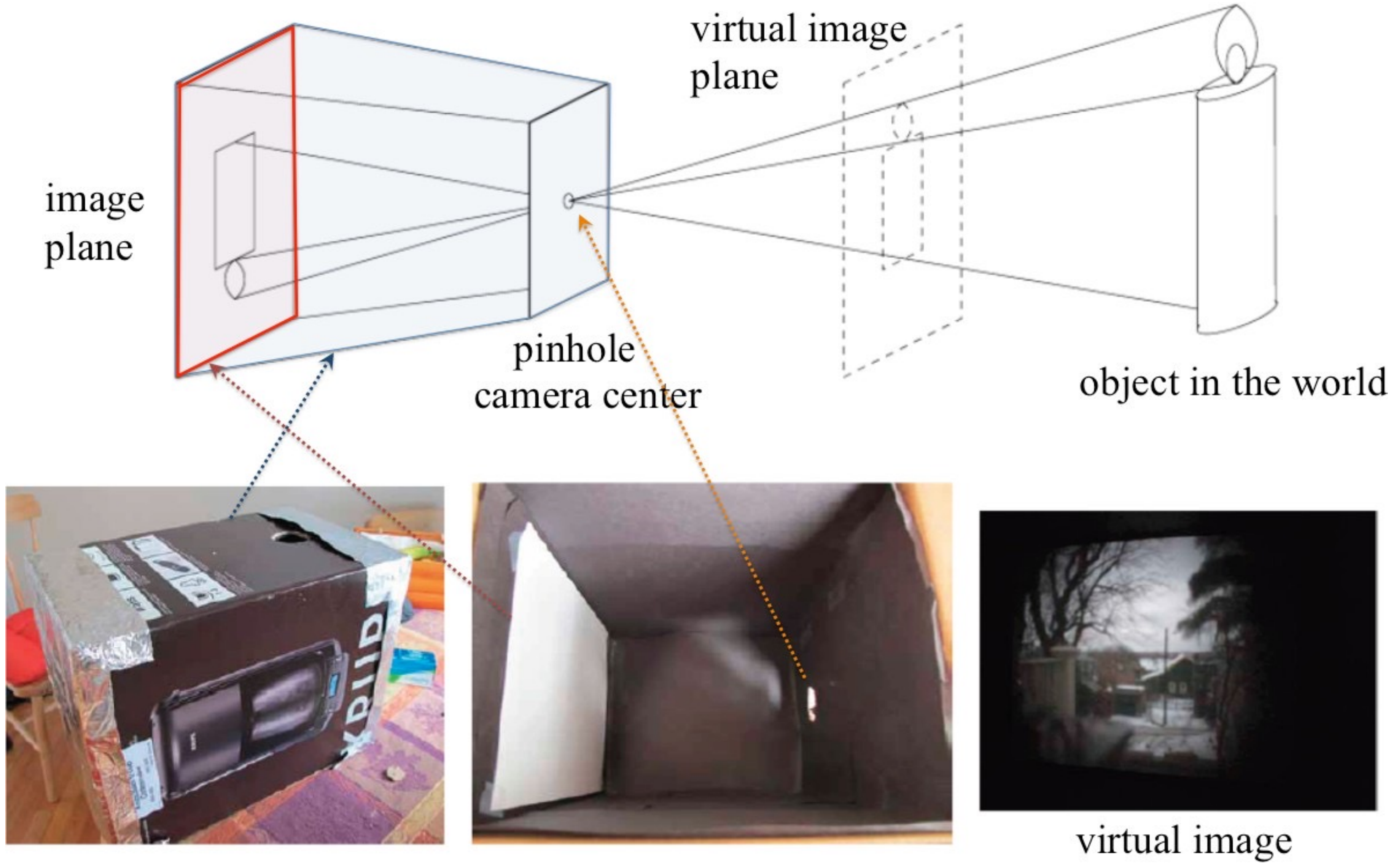
- We will use the pinhole model as an approximation



[Pics from: A. Torralba, Forsyth & Ponce]

Modeling Projection

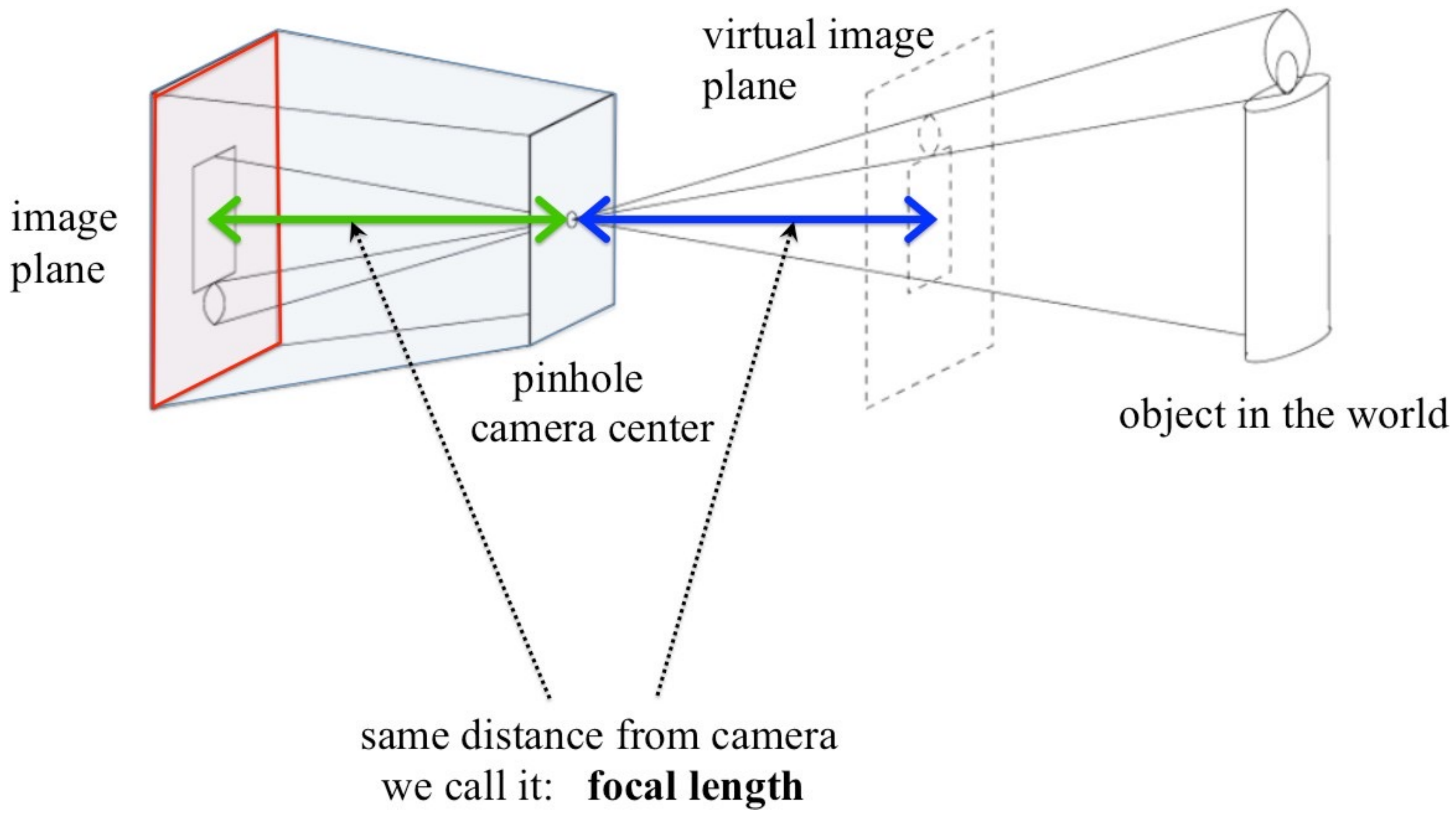
- We will use the pinhole model as an approximation



[Pics from: A. Torralba, Forsyth & Ponce]

Modeling Projection

- We will use the pinhole model as an approximation



[Pics from: A. Torralba, Forsyth & Ponce]

Focal Length

- Can be thought of as zoom
- Larger focal length narrows the field of view, more pixels per angle in the scene

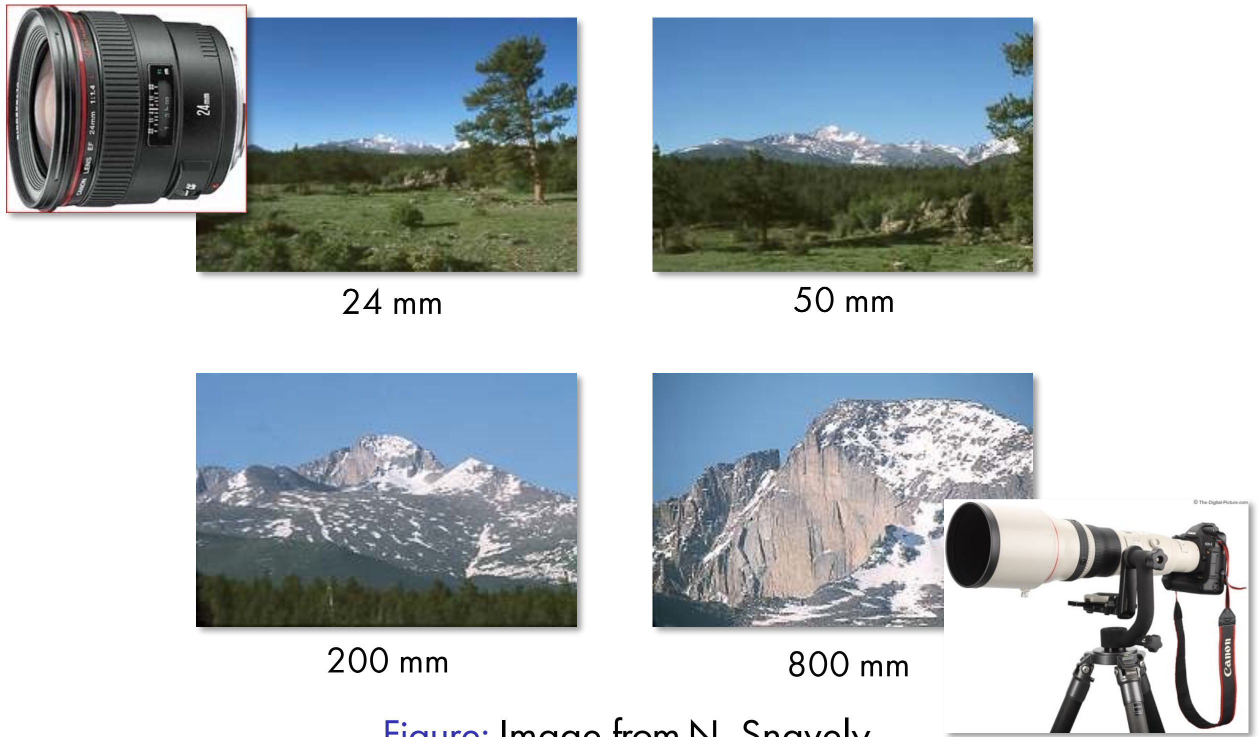
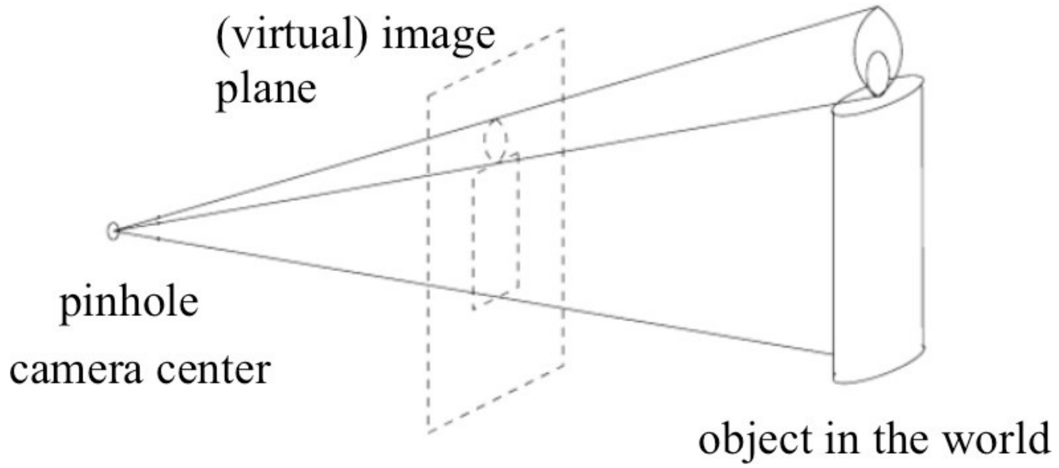


Figure: Image from N. Snavely

[Source: N. Snavely, slide credit: R. Urtasun]

Modeling Projection

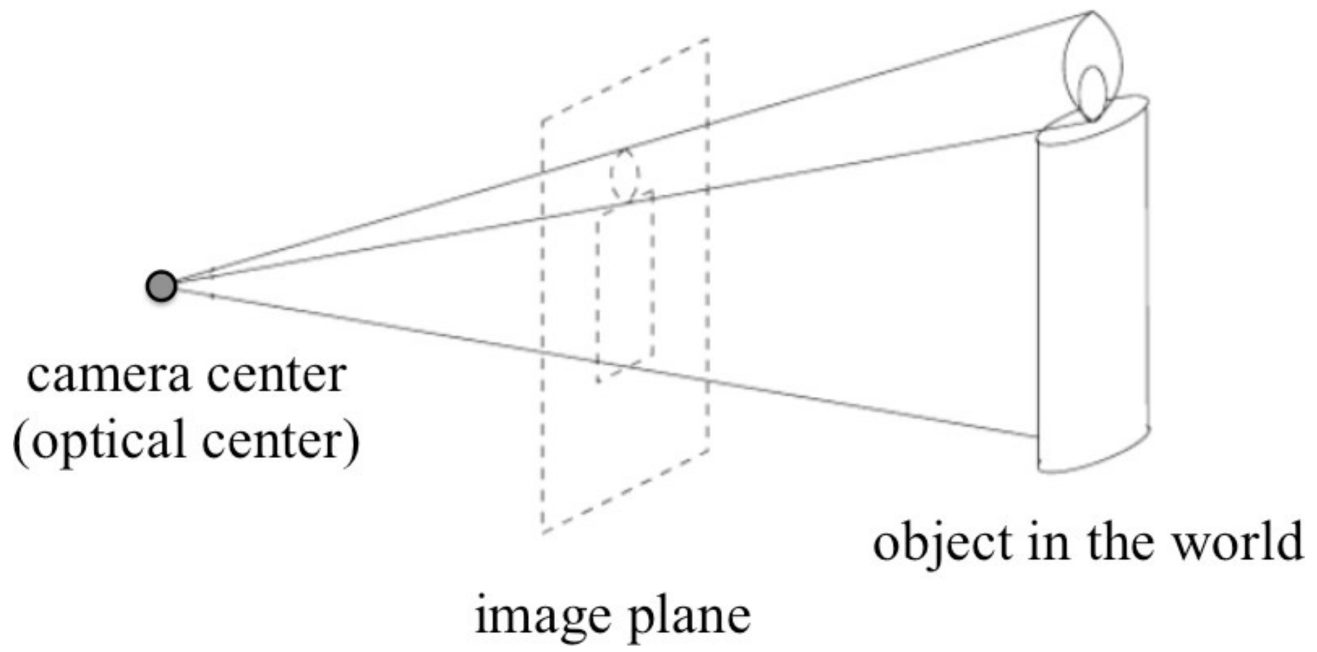
- We will use the pinhole model as an approximation



- Since it's easier to think in a non-upside down world, we will work with the virtual image plane, and just call it the image plane.
- How do points in 3D project to image plane? If I know a point in 3D, can I compute to which pixel it projects?

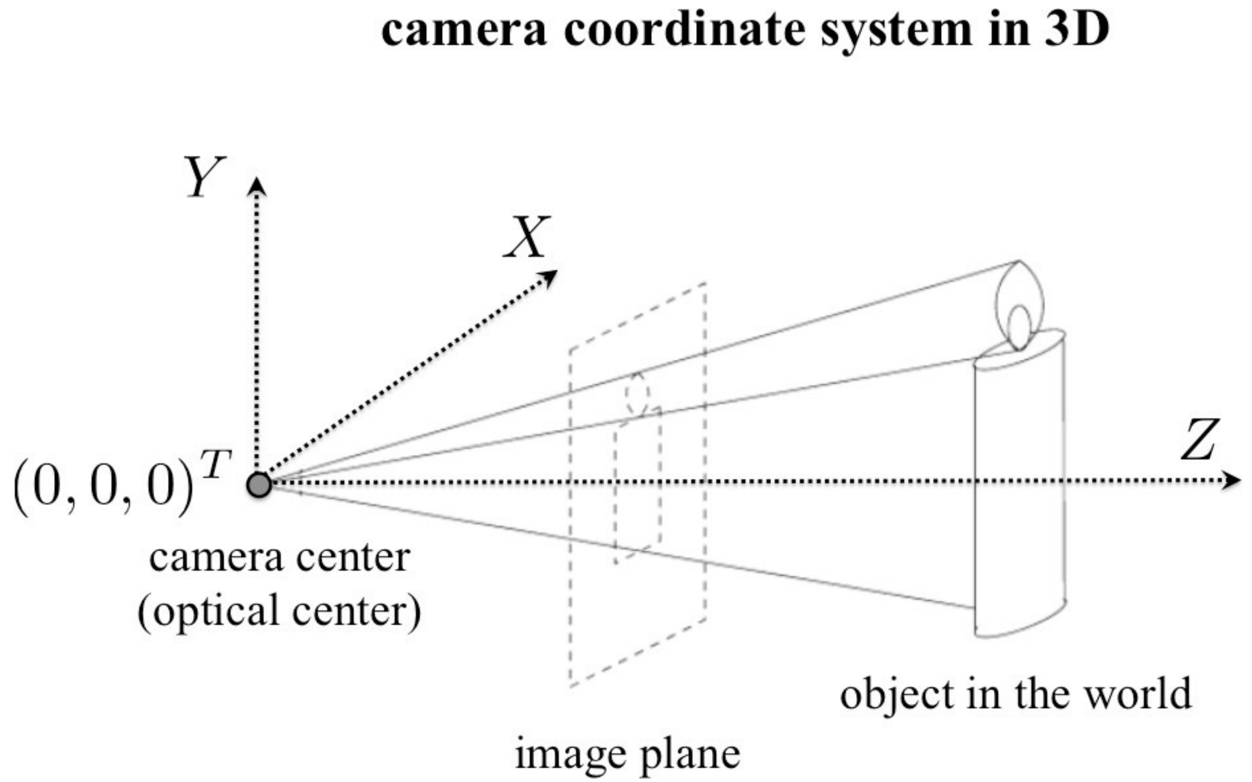
[Pics from: A. Torralba, Forsyth & Ponce]

Modeling Projection



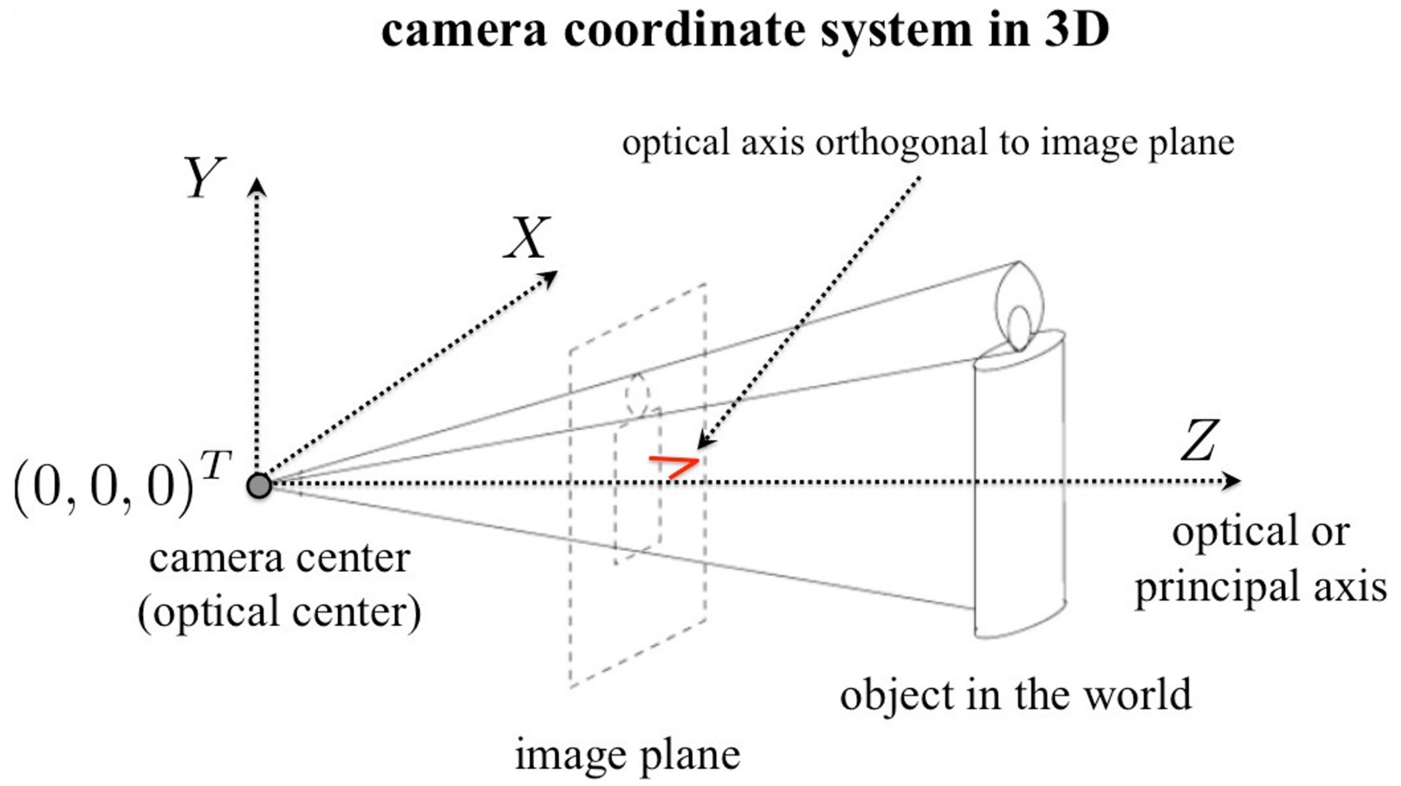
- First some notation which will help us derive the math
- To start with, we need a coordinate system

Modeling Projection



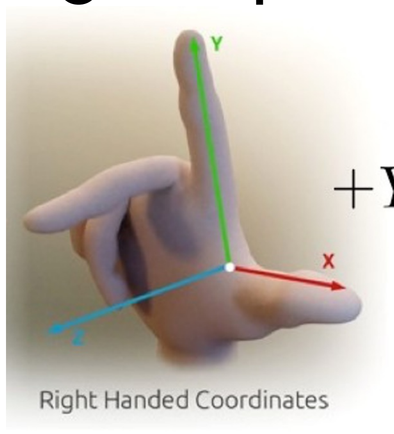
- We place a coordinate system relative to camera: optical center or camera center C is thus at origin (0, 0, 0).

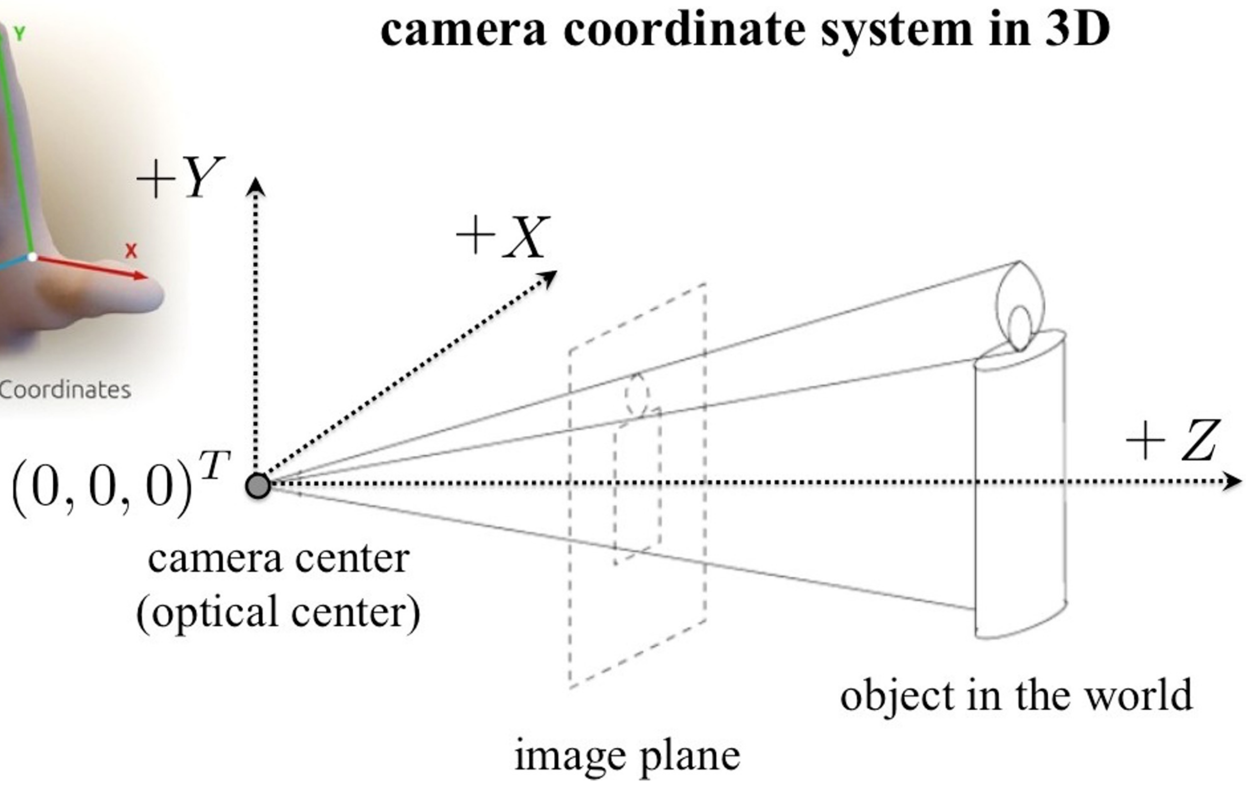
Modeling Projection



- The Z axis is called the optical or principal axis. It is orthogonal to the image plane. Axes X and Y are parallel to the image axes.

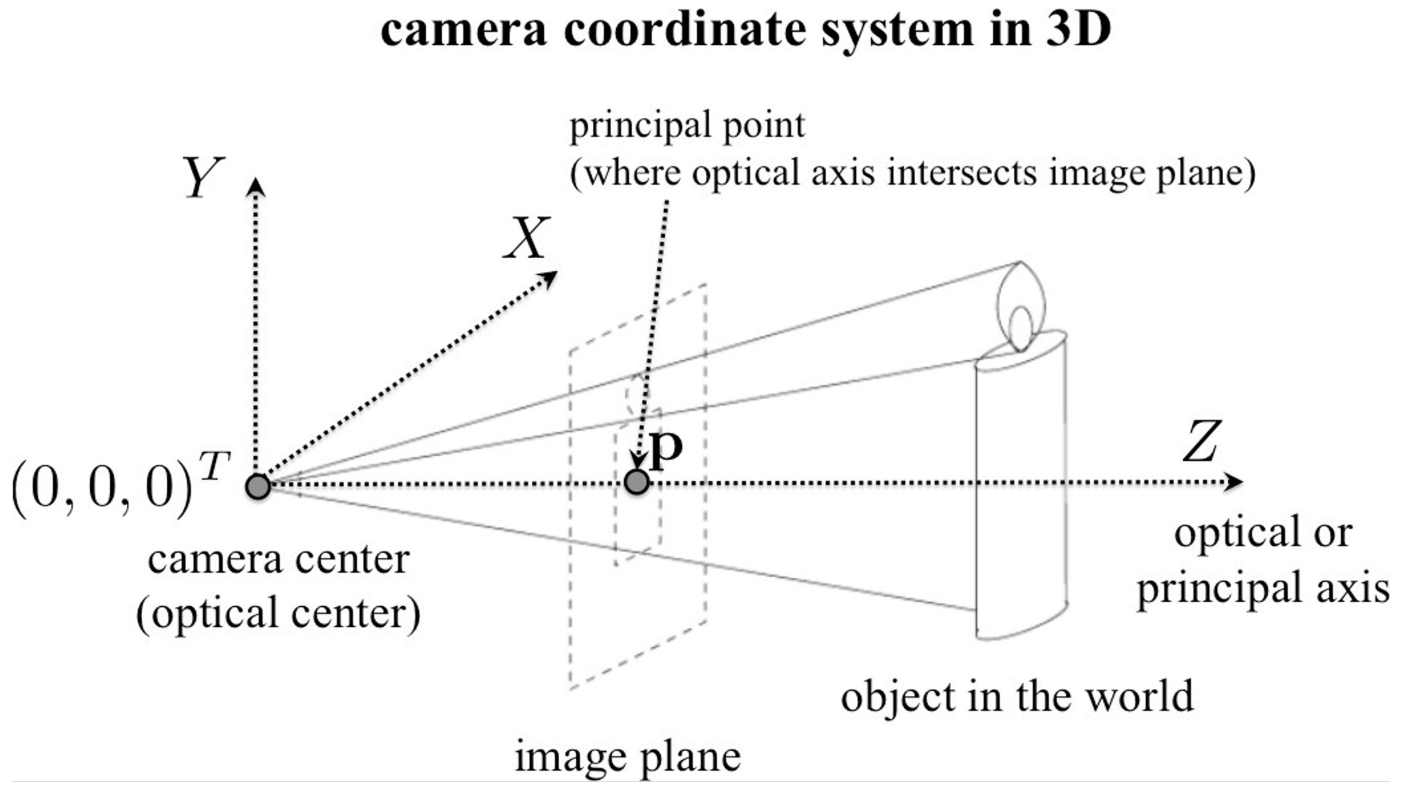
Modeling Projection





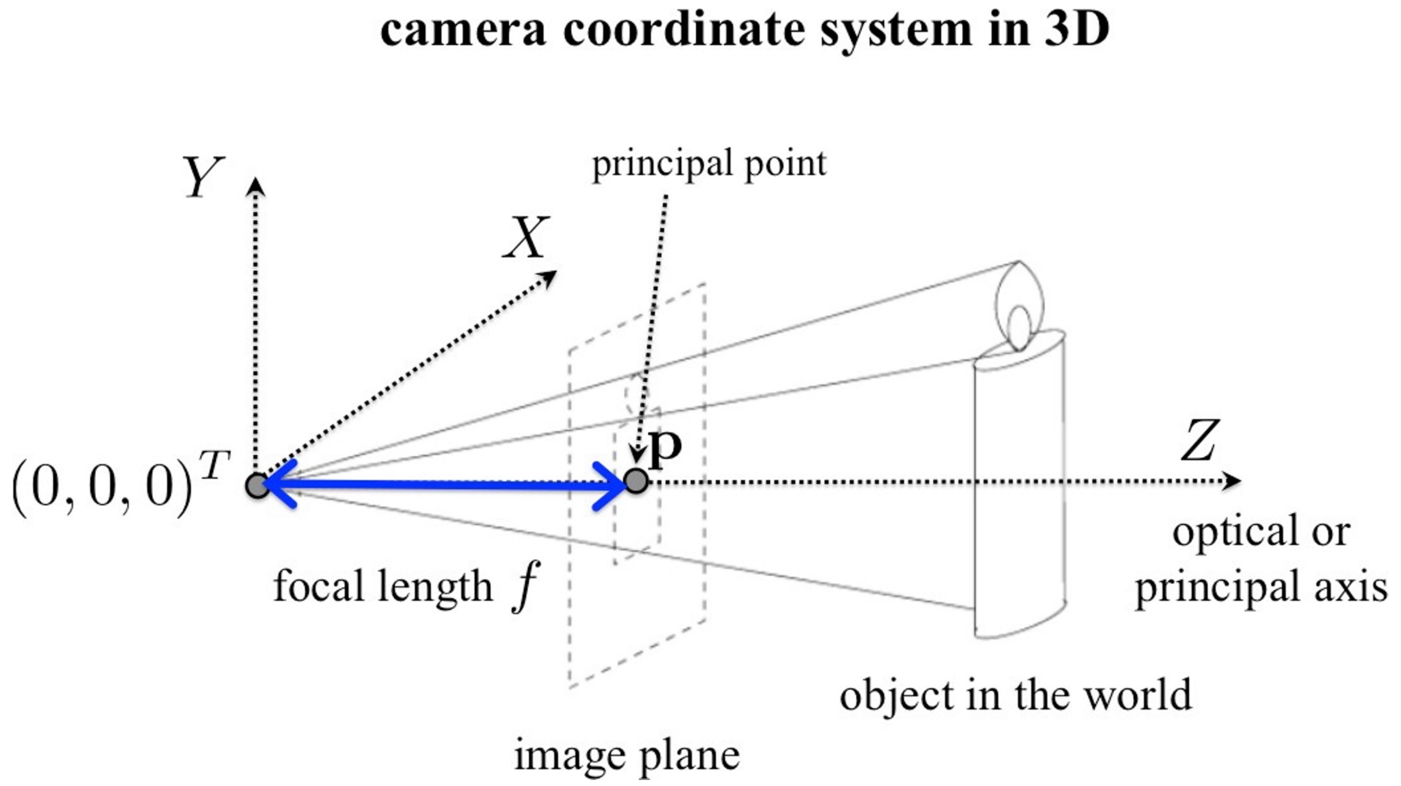
- We will use a right-handed coordinate system

Modeling Projection



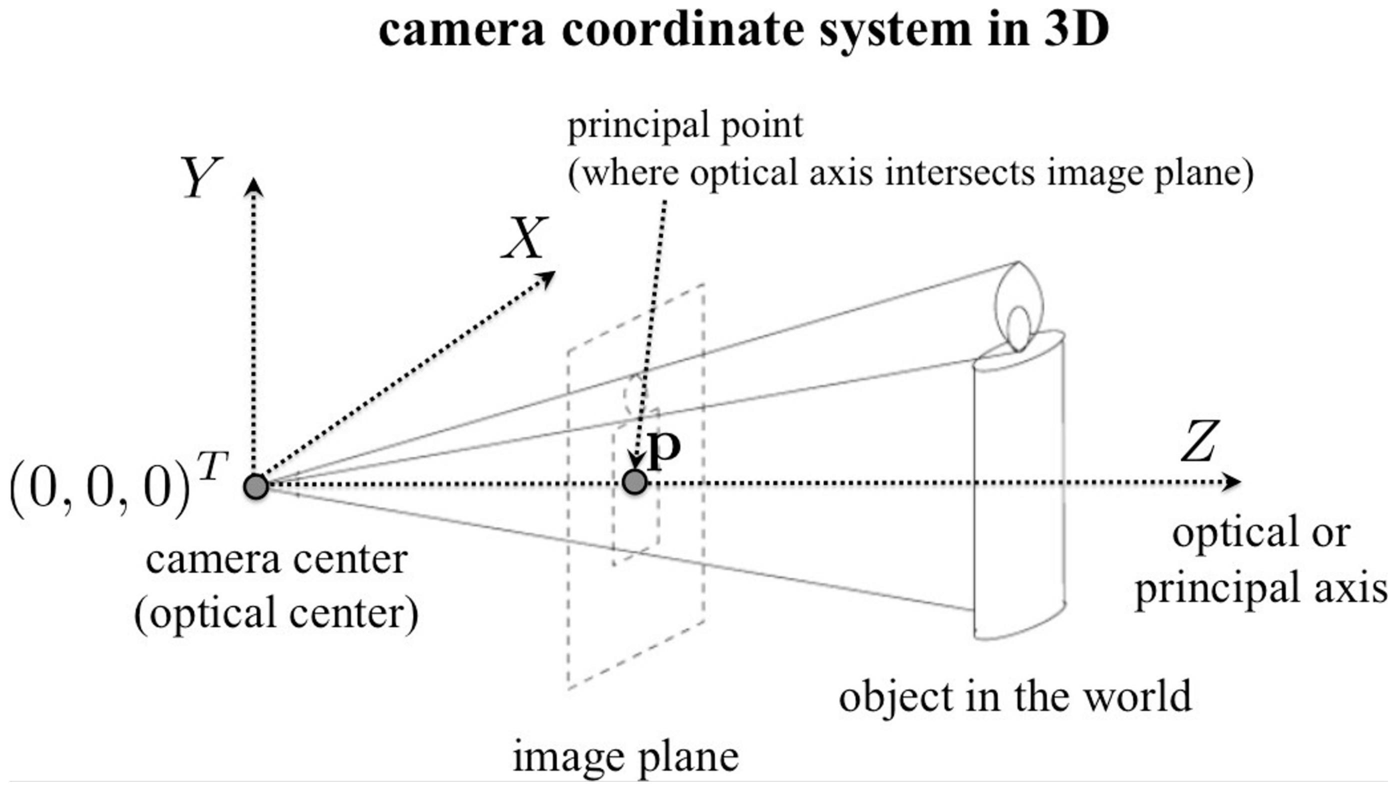
- The optical axis intersects the image plane in a point, p. We call this point a principal point. It’s worth to remember the principal point since it will appear again later in the math.

Modeling Projection



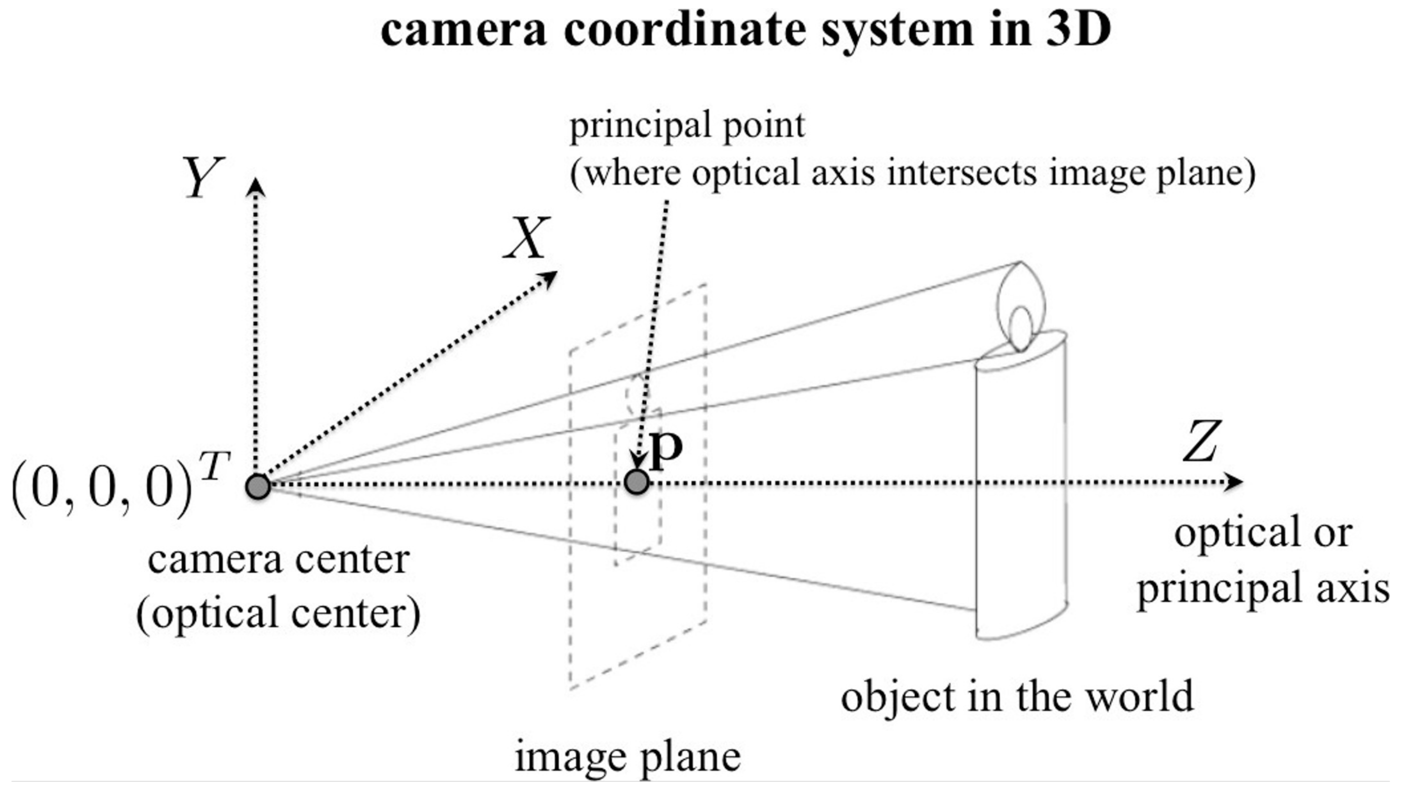
- The distance from the camera center to the principal point is called focal length, we will denote it with f. It's worth to remember the focal length since it will appear again later in the math.

Modeling Projection



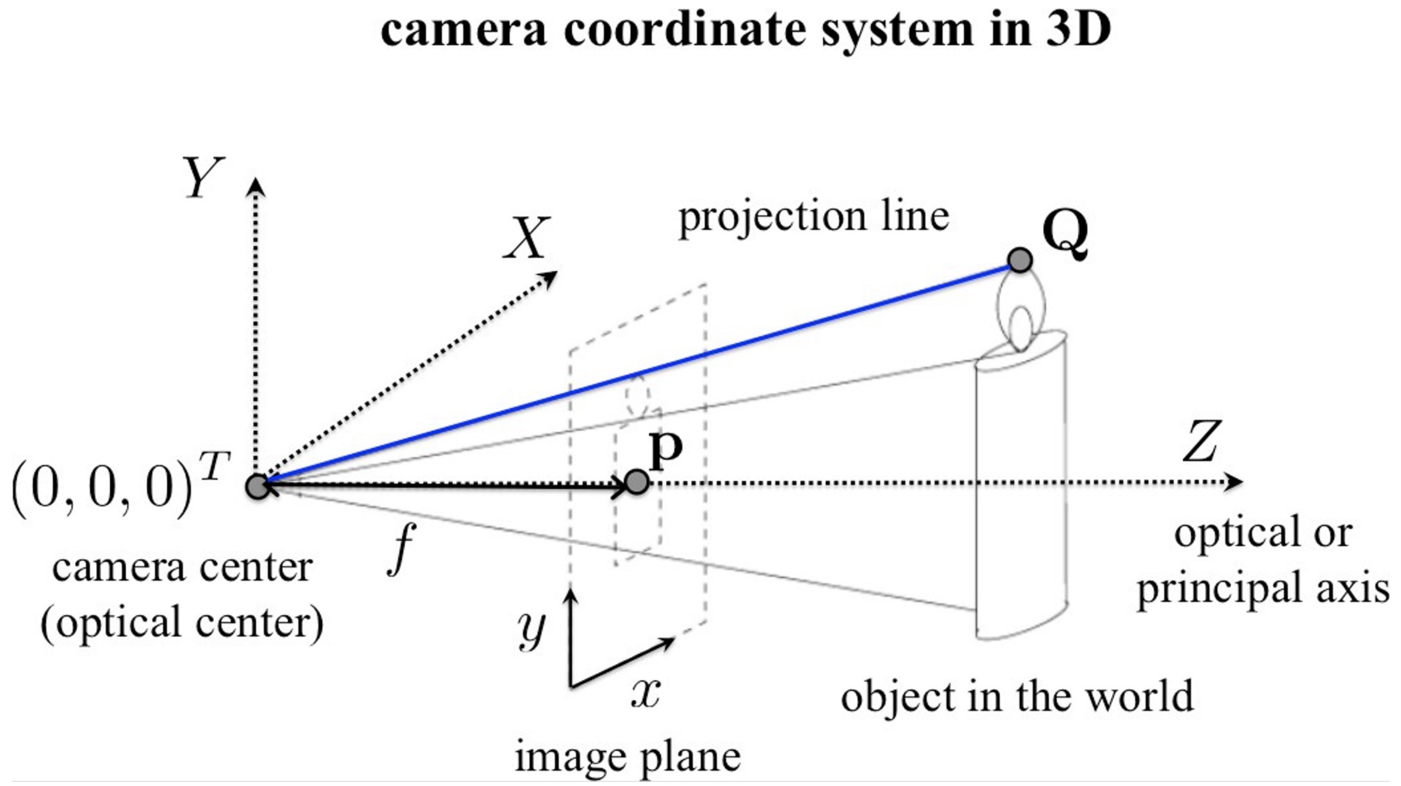
- We'll denote the image axes with x and y.
- The tricky part is how to get from the camera's coordinate system (3D) to the image coordinate system (2D).

Modeling Projection



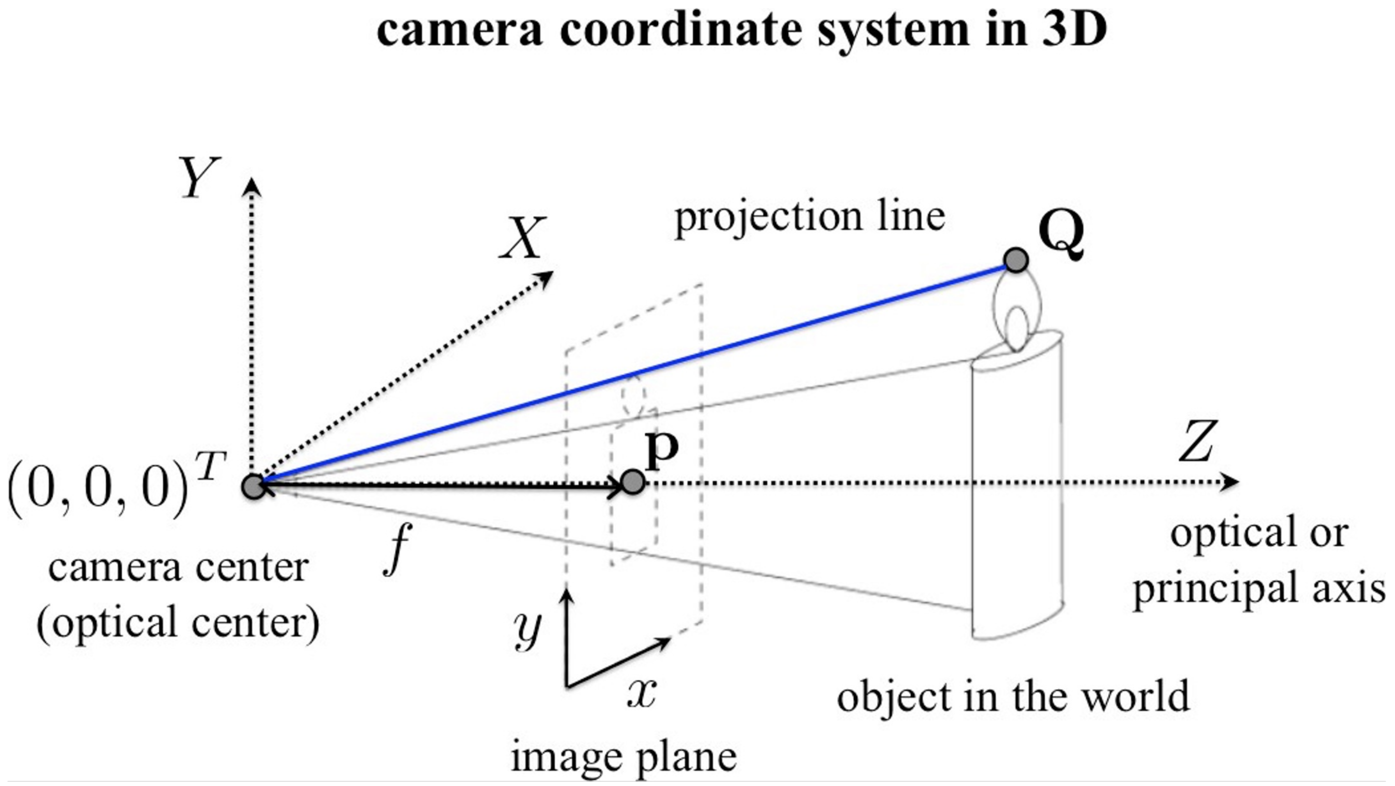
- Let's take some point Q in 3D.
- Q "lives" relative to the camera; its coordinates are assumed to be in camera's coordinate system.

Modeling Projection



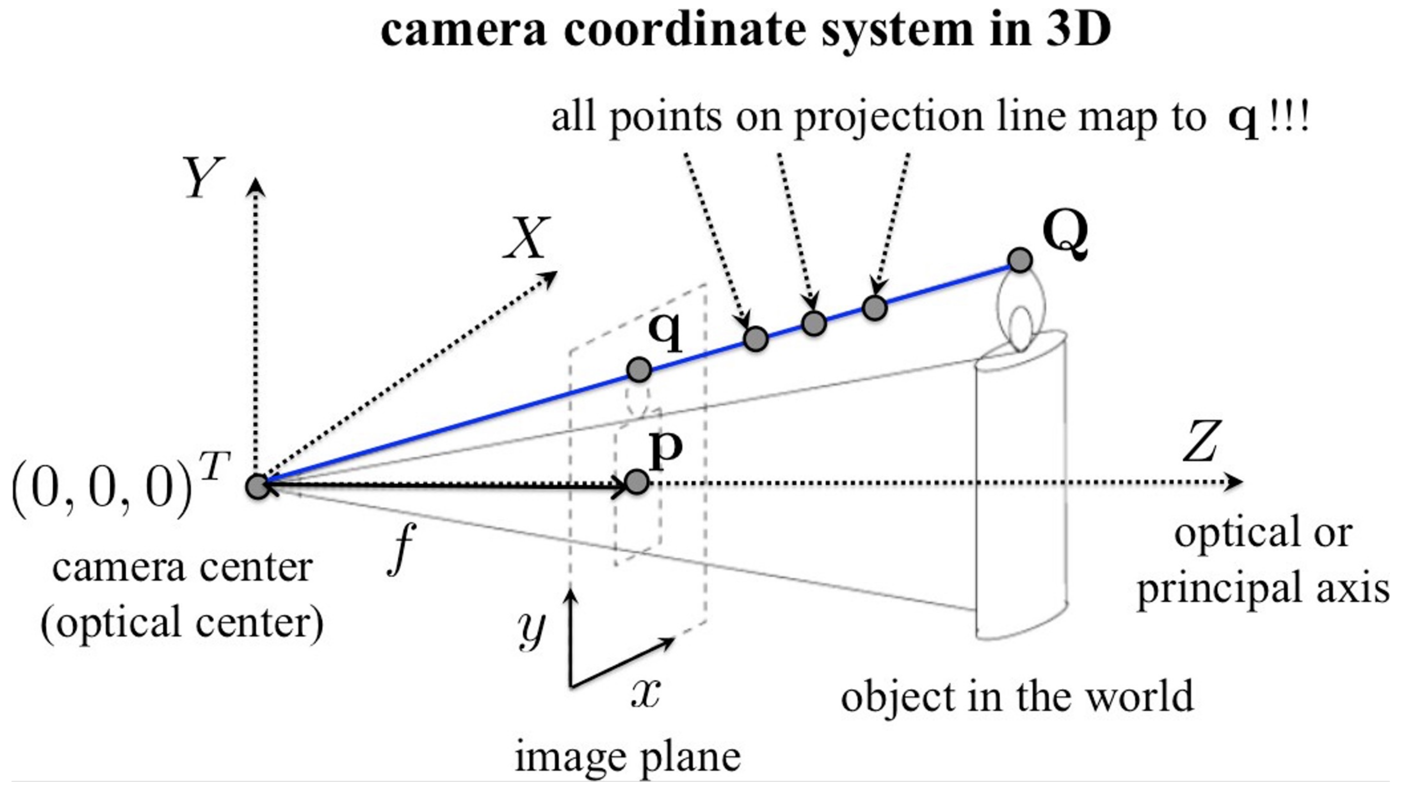
- We call the line from Q to camera center a projection line.

Modeling Projection



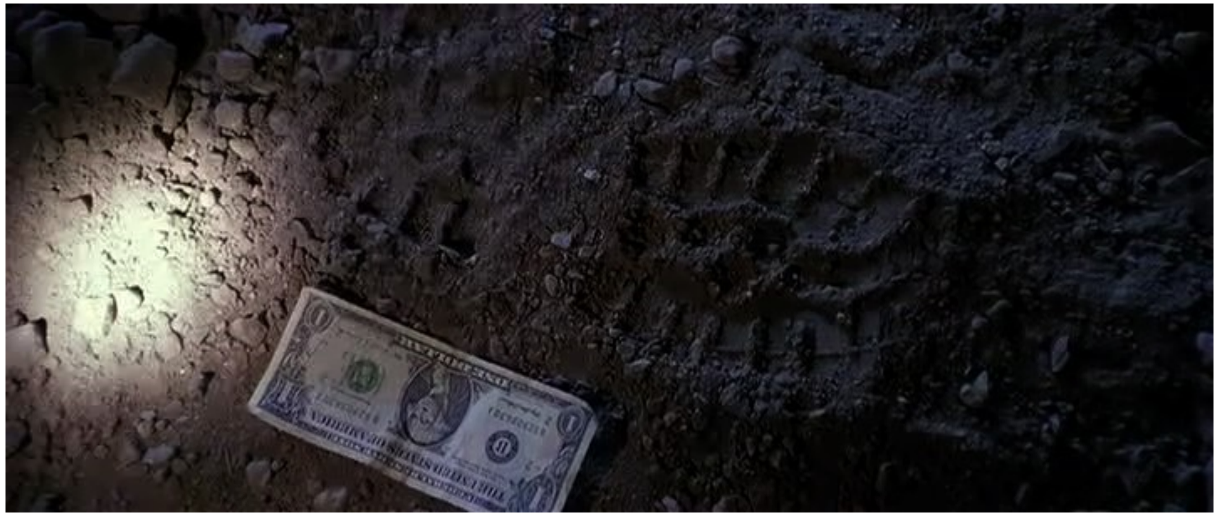
- The projection line intersects the image plane in a point q. This is the point we see in our image.

Modeling Projection



- First thing to notice is that all points from Q's projection line project to the same point q in the image!
- Ambiguity: It's impossible to know how far a 3D point is from the camera along the projection line by looking only at the image (point q).

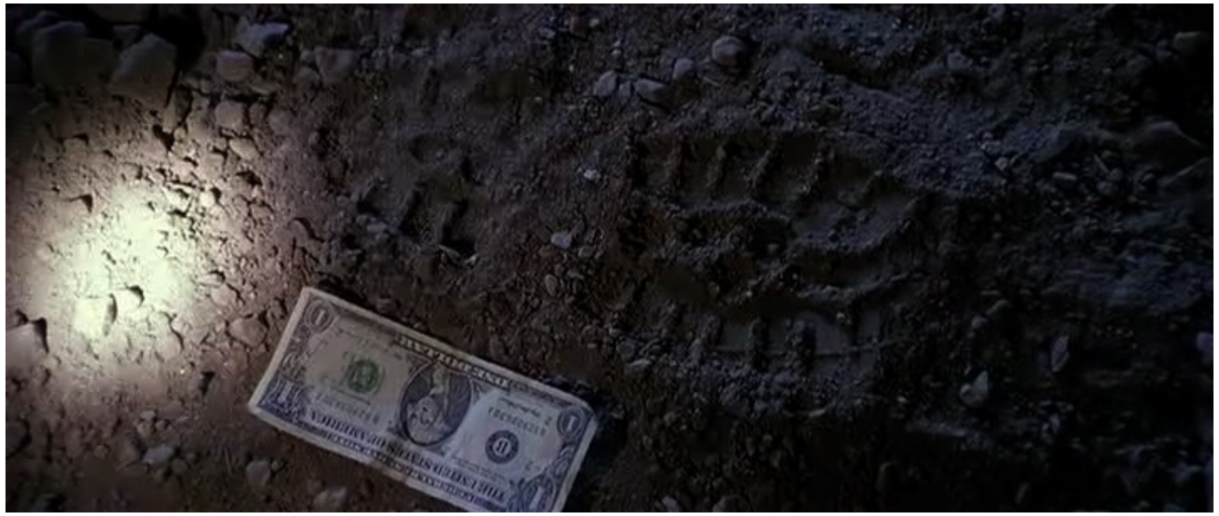
Modeling Projection



From the movie Bone Collector

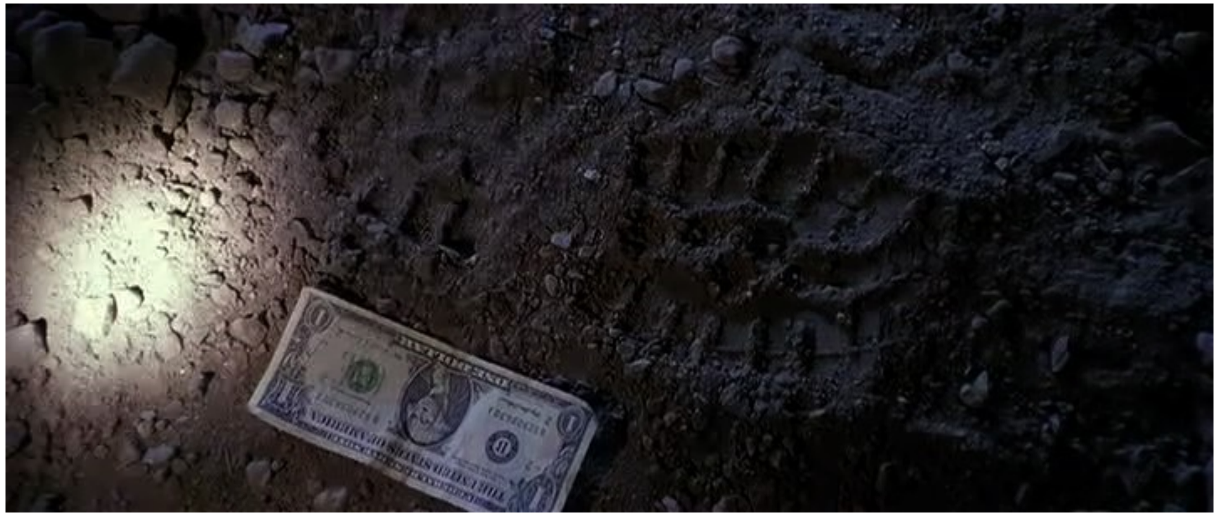
- Why did the detective put a dollar bill next to the footprint?

Modeling Projection


From the movie Bone Collector

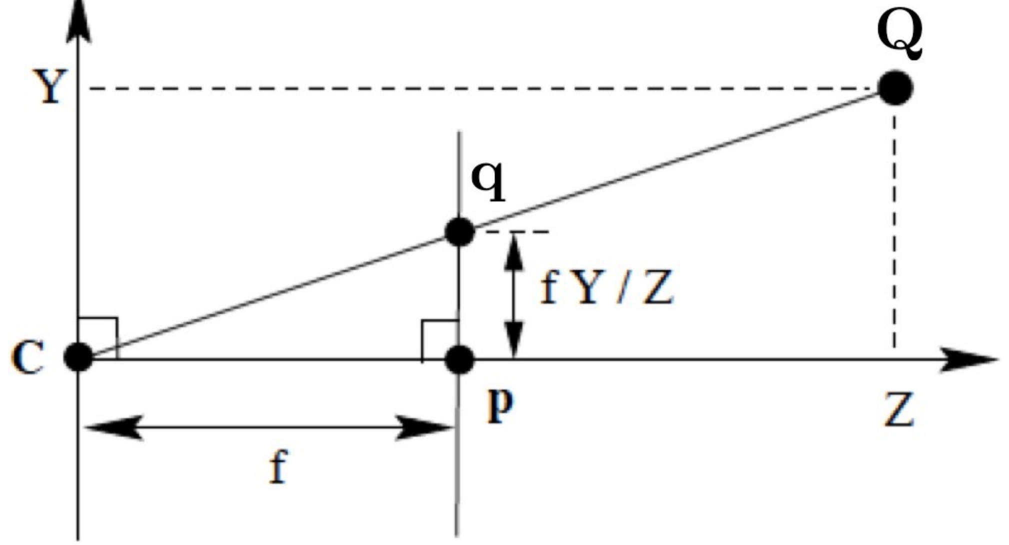
- Why did the detective put a dollar bill next to the footprint?
- Ambiguity: It's impossible to know how far a 3D point is from the camera along the projection line by looking only at the image (point q).
- It's impossible to know the real 3D size of objects just from an image

Modeling Projection


From the movie Bone Collector

- Why did the detective put a dollar bill next to the footprint?
- Ambiguity: It's impossible to know how far a 3D point is from the camera along the projection line by looking only at the image (point q).
- It's impossible to know the real 3D size of objects just from an image
- How would you compute the shoe's dimensions?

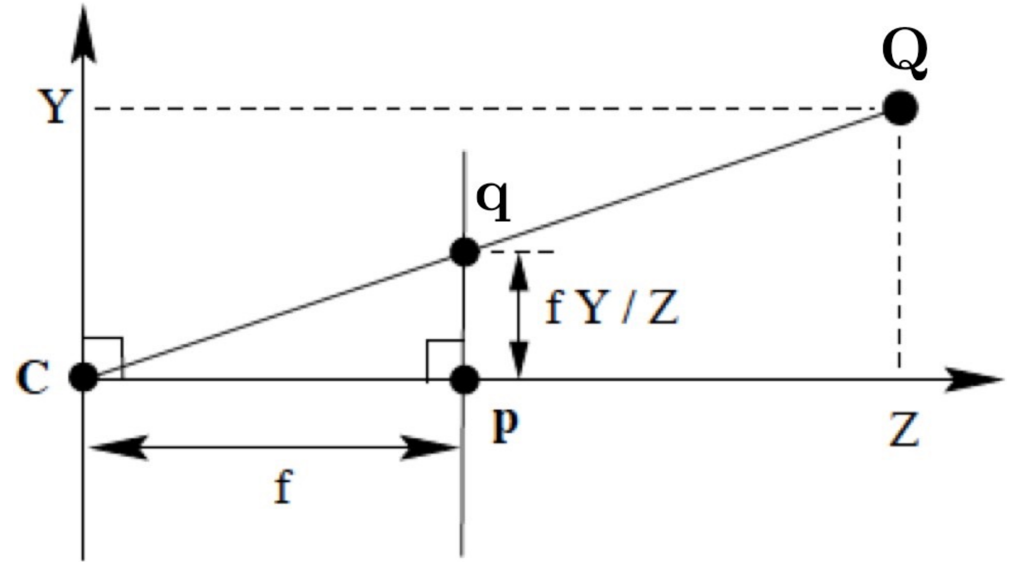
Projection: Ready for Math



Projection Equations

- Using similar triangles to preserve ratios:
- $Q = (X, Y, Z)^T \rightarrow \left(\frac{f \cdot X}{Z}, \frac{f \cdot Y}{Z}, f\right)^T$

Projection: Ready for Math



Projection Equations

- Using similar triangles to preserve ratios:
- $Q = (X, Y, Z)^T \rightarrow \left(\frac{f \cdot X}{Z}, \frac{f \cdot Y}{Z}, f\right)^T$
- This is relative to principal point p. To move the origin to (0, 0) in image:

$q = (X, Y, Z)^T \rightarrow \left(\frac{f \cdot X}{Z} + p_x, \frac{f \cdot Y}{Z} + p_y, 0\right)^T$

Where $p=(p_x, p_y)$ is the principal point.

Projection: Ready for Math

Projection Equations

- Using similar triangles to preserve ratios:

$$Q = (\mathrm{X}, Y, Z)^T \rightarrow \left(\frac{f \cdot X}{Z}, \frac{f \cdot Y}{Z}, f\right)^T$$

- This is relative to principal point p. To move the origin to (0, 0) in image:

$$q = (\mathrm{X}, Y, Z)^T \rightarrow \left(\frac{f \cdot X}{Z} + p_x, \frac{f \cdot Y}{Z} + p_y, 0\right)^T$$

Where p=(p_x, p_y) is the principal point.

Projection: Ready for Math

Projection Equations

- Using similar triangles to preserve ratios:

$$Q = (\mathrm{X}, Y, Z)^T \rightarrow \left(\frac{f \cdot X}{Z}, \frac{f \cdot Y}{Z}, f\right)^T$$

- This is relative to principal point p. To move the origin to (0, 0) in image:

$$q = (\mathrm{X}, Y, Z)^T \rightarrow \left(\frac{f \cdot X}{Z} + p_x, \frac{f \cdot Y}{Z} + p_y, 0\right)^T$$

Where p=(p_x, p_y) is the principal point.

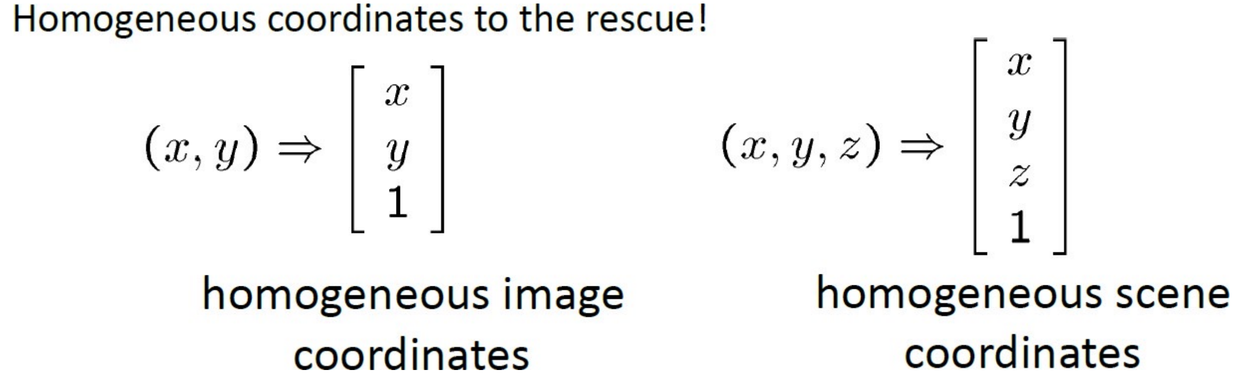
Get the projection by throwing the last coordinate:

$$Q = (\mathrm{X}, Y, Z)^T \rightarrow \mathrm{q} = \left(\frac{f \cdot X}{Z} + p_x, \frac{f \cdot Y}{Z} + p_y, \right)^T$$

This is NOT a linear transformation as a division by Z is non-linear

Homogeneous Coordinates!

- We will use homogeneous coordinates, which simply append a 1 to the vector



Homogeneous coordinates to the rescue!

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

homogeneous image coordinates

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

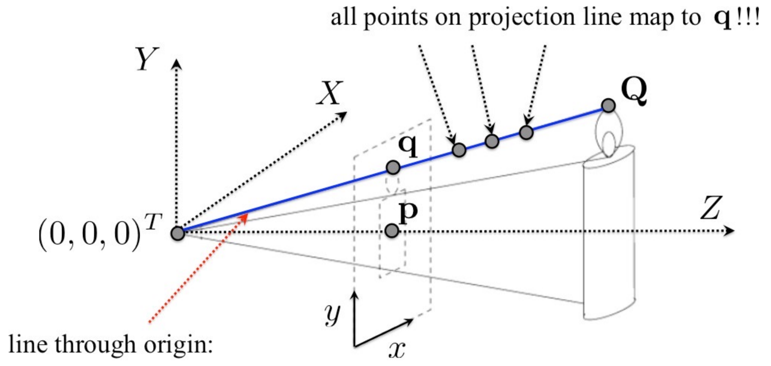
homogeneous scene coordinates

[Source: N. Snavely]

Homogeneous Coordinates!

- We will use homogeneous coordinates, which simply append a 1 to the vector
- In homogeneous coordinates, scaling doesn't affect anything:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} w.x \\ w.y \\ w \end{bmatrix}$$



$$\begin{bmatrix} X_Q \\ Y_Q \\ Z_Q \end{bmatrix} \sim \lambda \cdot \begin{bmatrix} X_Q \\ Y_Q \\ Z_Q \end{bmatrix}$$

- In Projective Geometry, all points are equal under scaling λ along the ray

[Source: N. Snavely]

Back to Perspective Projection

- We currently have this (the nasty division by Z):

$$Q = (\mathrm{X}, Y, Z)^T \rightarrow \mathrm{q} = \begin{bmatrix} \frac{f \cdot X}{Z} + p_x \\ \frac{f \cdot Y}{Z} + p_y \end{bmatrix}$$

Back to Perspective Projection

- We currently have this (the nasty division by Z):

$$Q = (\mathrm{X}, Y, Z)^T \rightarrow \mathrm{q} = \begin{bmatrix} \frac{f \cdot X}{Z} + p_x \\ \frac{f \cdot Y}{Z} + p_y \end{bmatrix}$$

- Write this with homogeneous coordinates:

$$Q = (\mathrm{X}, Y, Z)^T \rightarrow \mathrm{q} = \begin{bmatrix} \frac{f \cdot X}{Z} + p_x \\ \frac{f \cdot Y}{Z} + p_y \\ 1 \end{bmatrix} \sim \begin{bmatrix} f \cdot X + Z \cdot p_x \\ f \cdot Y + Z \cdot p_y \\ Z \end{bmatrix}$$

Back to Perspective Projection

- We currently have this (the nasty division by Z):

$$Q = (\mathrm{X}, Y, Z)^T \rightarrow \mathrm{q} = \begin{bmatrix} \frac{f \cdot X}{Z} + p_x \\ \frac{f \cdot Y}{Z} + p_y \end{bmatrix}$$

- Write this with homogeneous coordinates:

$$Q = (\mathrm{X}, Y, Z)^T \rightarrow \mathrm{q} = \begin{bmatrix} \frac{f \cdot X}{Z} + p_x \\ \frac{f \cdot Y}{Z} + p_y \\ 1 \end{bmatrix} \sim \begin{bmatrix} f \cdot X + Z \cdot p_x \\ f \cdot Y + Z \cdot p_y \\ Z \end{bmatrix}$$

- We can now write this as matrix multiplication:

$$Q = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \begin{bmatrix} f \cdot X + Z . p_x \\ f \cdot Y + Z . p_y \\ Z \end{bmatrix} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Camera Intrinsics

- From Previous Slide:

$$Q = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \begin{bmatrix} f \cdot X + Z \cdot p_x \\ f \cdot Y + Z \cdot p_y \\ Z \end{bmatrix} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Write:

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

This is called a camera calibration matrix or intrinsic parameter matrix. The parameters in K are called internal camera parameters.

Camera Intrinsics

- From Previous Slide:

$$Q = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \begin{bmatrix} f \cdot X + Z.p_x \\ f \cdot Y + Z.p_y \\ Z \end{bmatrix} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Write:

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

This is called a camera calibration matrix or intrinsic parameter matrix. The parameters in K are called intrinsic camera parameters.

Finally $K \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} w \cdot x \\ w \cdot y \\ \mathrm{w} \end{bmatrix} \rightarrow q = \begin{bmatrix} x \\ y \end{bmatrix}$

[Source: Zisserman & Hartley]

Camera Intrinsics

- Camera calibration matrix:

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

Camera Intrinsics

- Camera calibration matrix:

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

- It can be a little more complicated. Pixels may not be square:

$$K = \begin{bmatrix} f_x & 0 & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

Camera Intrinsics

- Camera calibration matrix:

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

- It can be a little more complicated. Pixels may not be square:

$$K = \begin{bmatrix} f_x & 0 & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

- And there might be a skew angle θ between x and y image axis:

$$K = \begin{bmatrix} f_x & -f_x cot\theta & p_x \\ 0 & f_y/sin\theta & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

Camera Intrinsics

- Camera calibration matrix:

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

We'll work with this one

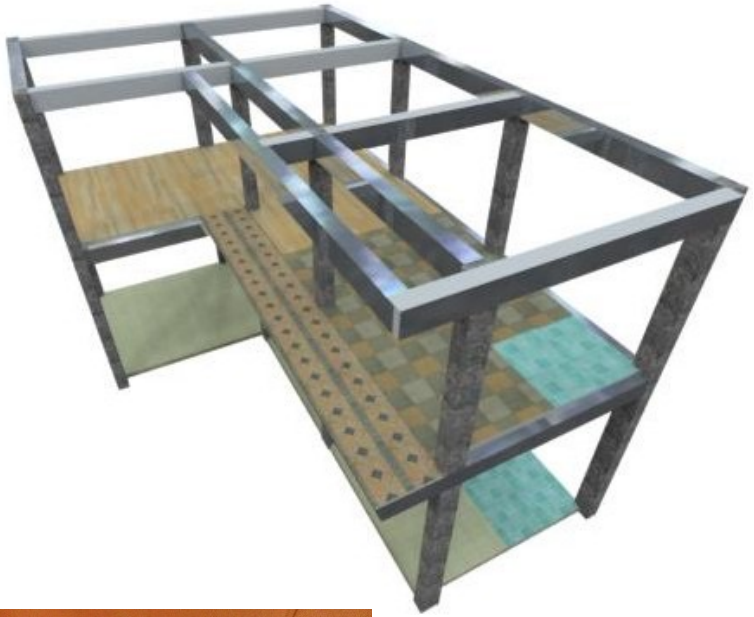
- It can be a little more complicated. Pixels may not be square:

$$K = \begin{bmatrix} f_x & 0 & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

- And there might be a skew angle θ between x and y image axis:

$$K = \begin{bmatrix} f_x & -f_x cot\theta & p_x \\ 0 & f_y / sin\theta & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

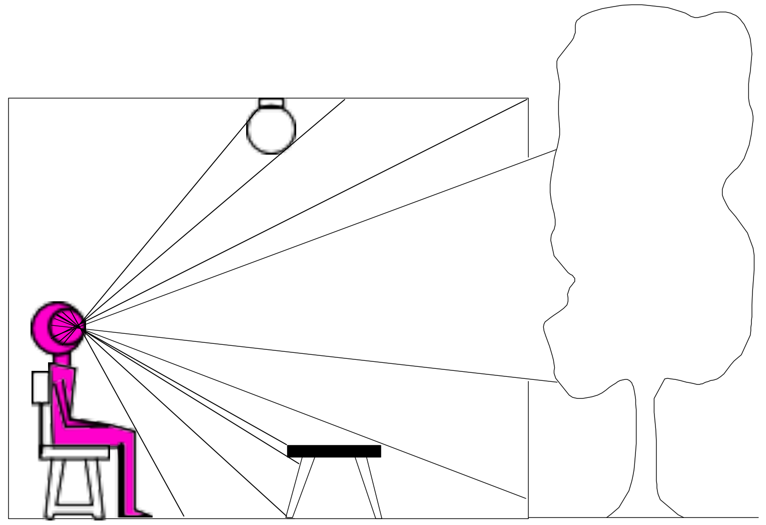
Perspective Projection



[Source: N. Snavely]

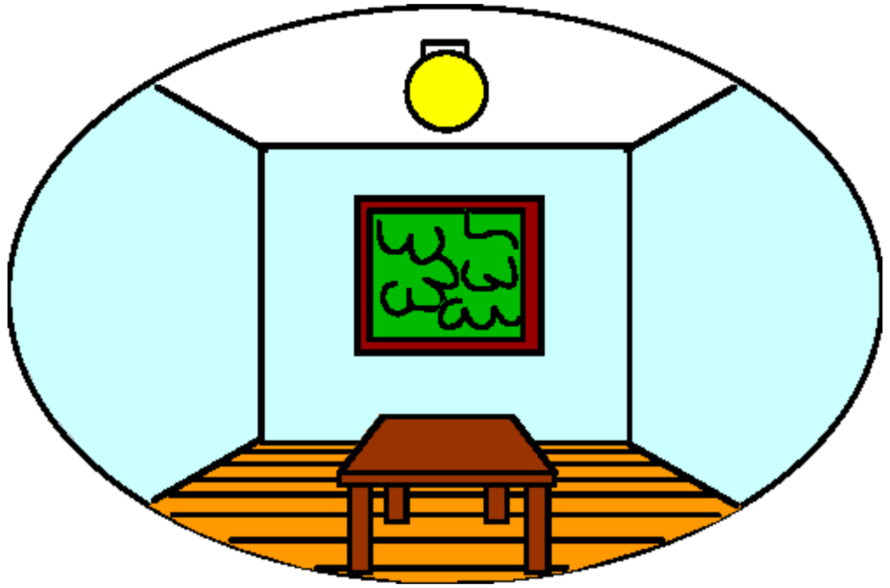
Dimensionality Reduction Machine (3D to 2D)

3D World



Point of Observation

2D Image



- What have we lost?
  - Angles?
  - Distances (lengths)

Slide by A. Efros

Figures © Stephen E. Palmer, 2002

Projection properties

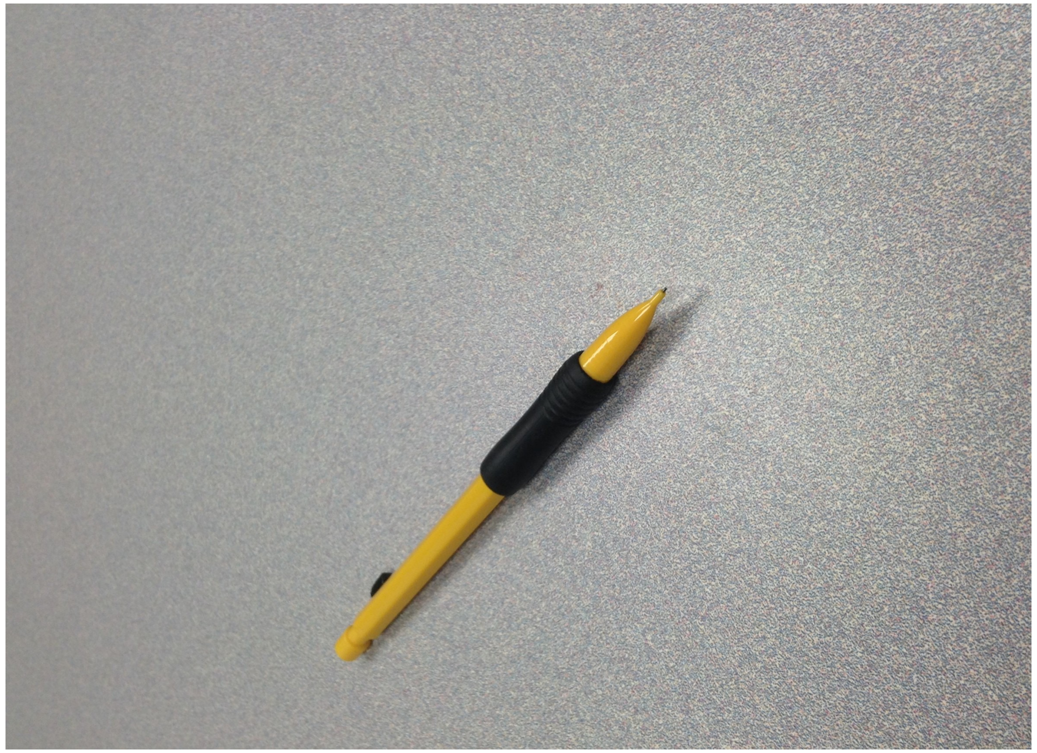
- Many-to-one: any points along same ray map to same point in image

Projection properties

- Many-to-one: any points along same ray map to same point in image
- Points → points

Projection properties

- Many-to-one: any points along same ray map to same point in image
- Points → points
- Lines → lines.



Projection properties

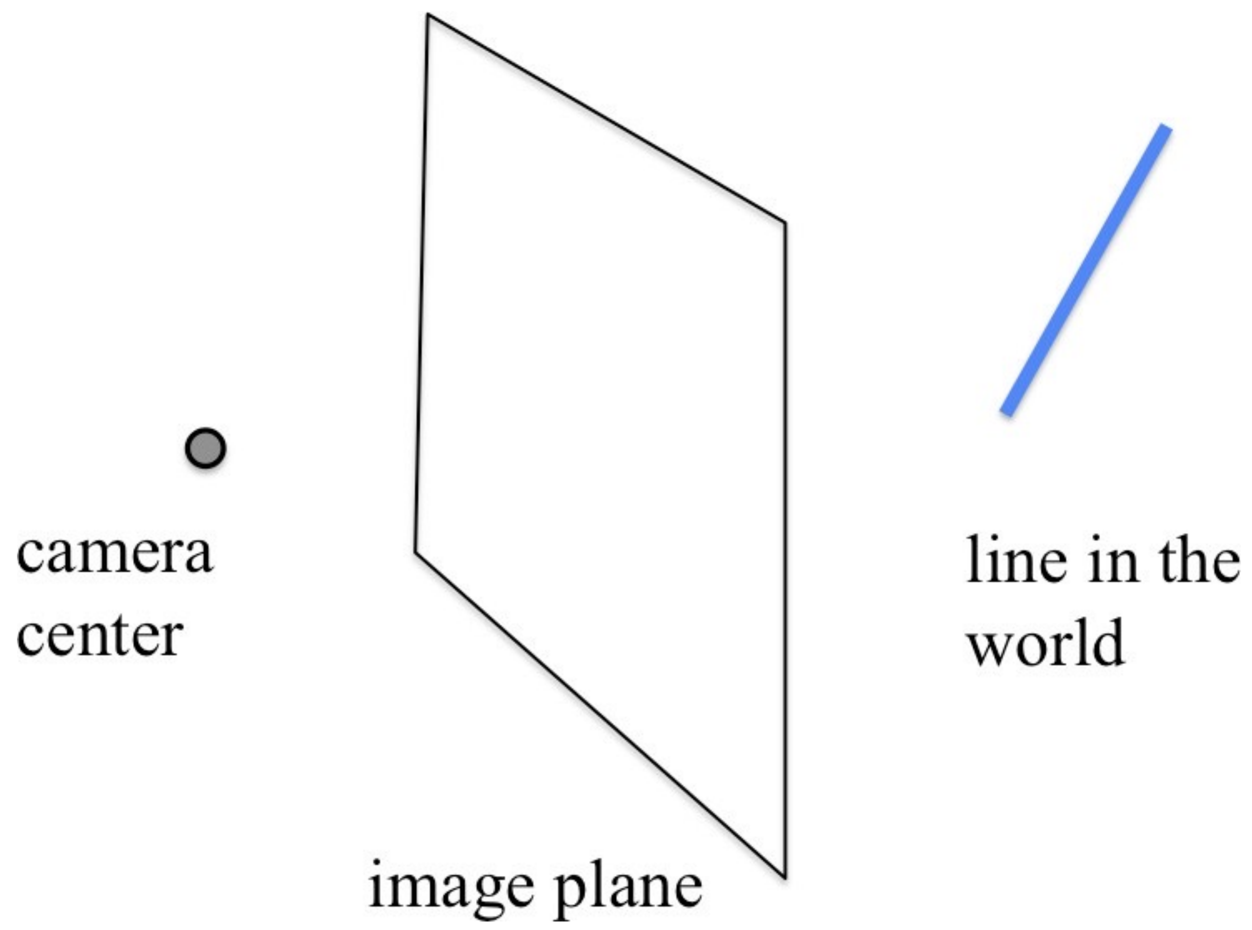
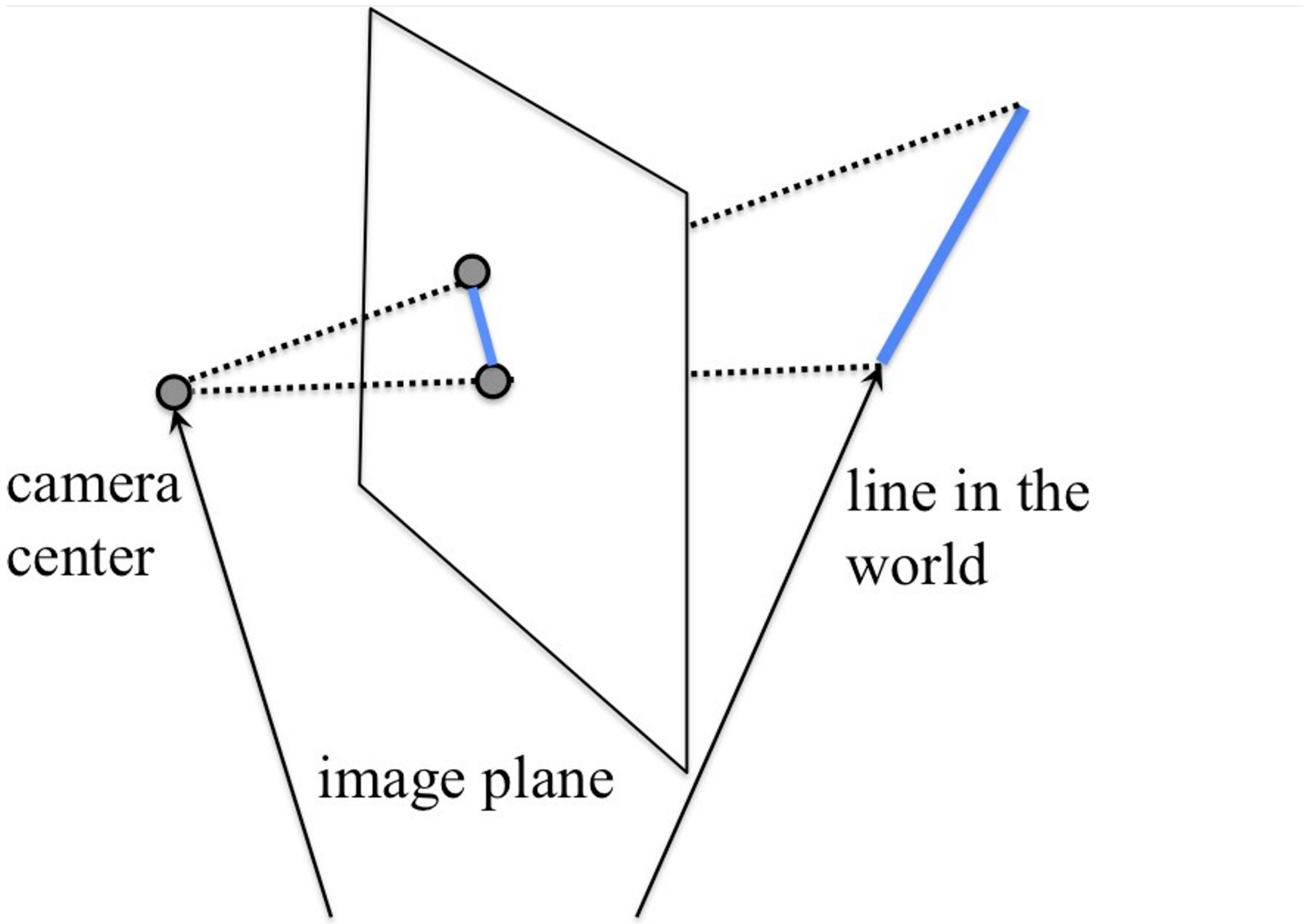


Figure: Proof by drawing

Projection properties



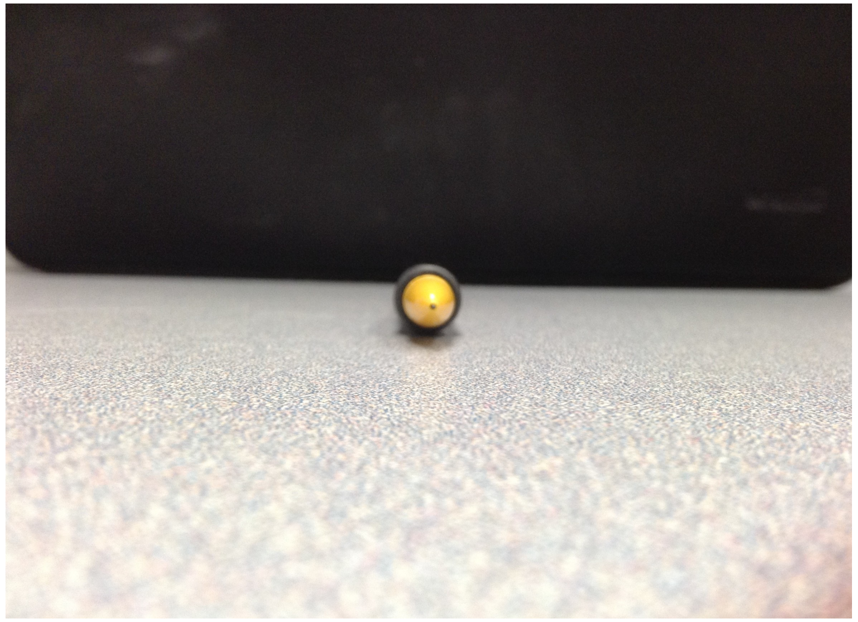
line and a point make a plane

$\rightarrow$ a plane intersects an (image) plane in a line!

Figure: Proof by drawing

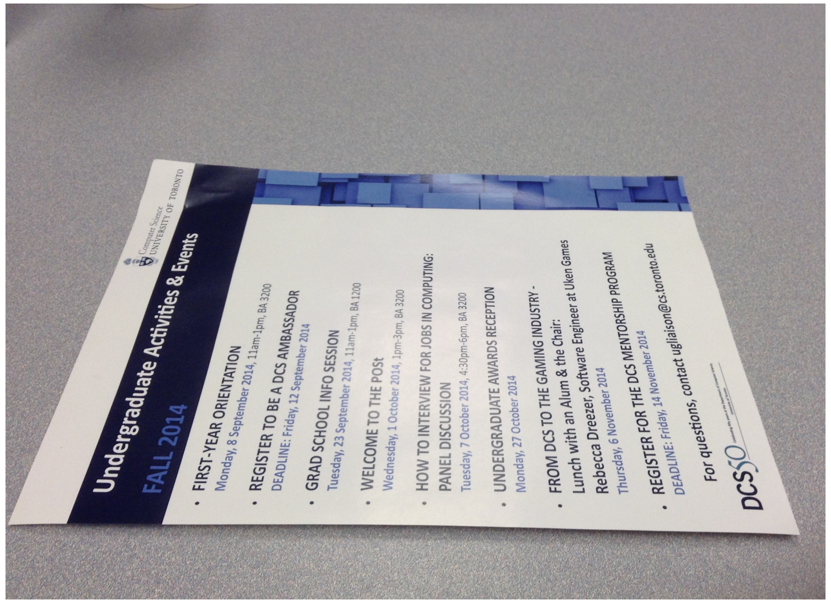
Projection properties

- Many-to-one: any points along same ray map to same point in image
- Points → points
- Lines → lines
- But line through center of projection maps to a point



Projection properties

- Many-to-one: any points along same ray map to same point in image
- Points → points
- Lines → lines
- But line through center of projection maps to a point
- Planes → planes



Computer Science
UNIVERSITY OF TORONTO

Undergraduate Activities & Events
FALL 2014

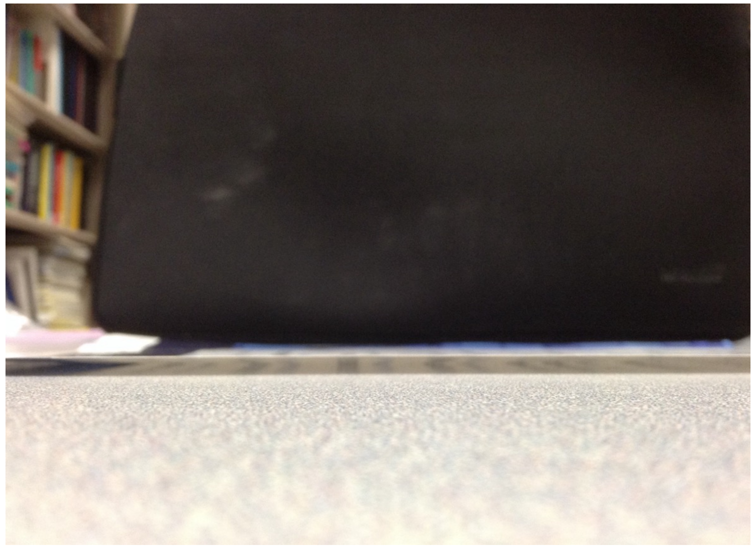
- FIRST-YEAR ORIENTATION
  Monday, 8 September 2014, 11am-1pm, BA 3200
- REGISTER TO BE A DCS AMBASSADOR
  DEADLINE: Friday, 12 September 2014
- GRAD SCHOOL INFO SESSION
  Tuesday, 23 September 2014, 11am-1pm, BA 1200
- WELCOME TO THE POSt
  Wednesday, 1 October 2014, 1pm-3pm, BA 3200
- HOW TO INTERVIEW FOR JOBS IN COMPUTING:
  PANEL DISCUSSION
  Tuesday, 7 October 2014, 4:30pm-6pm, BA 3200
- UNDERGRADUATE AWARDS RECEPTION
  Monday, 27 October 2014
- FROM DCS TO THE GAMING INDUSTRY -
  Lunch with an Alum & the Chair:
  Rebecca Dreezer, Software Engineer at Uken Games
  Thursday, 6 November 2014
- REGISTER FOR THE DCS MENTORSHIP PROGRAM
  DEADLINE: Friday, 14 November 2014

For questions, contact ugliaison@cs.toronto.edu

DCS50

Projection properties

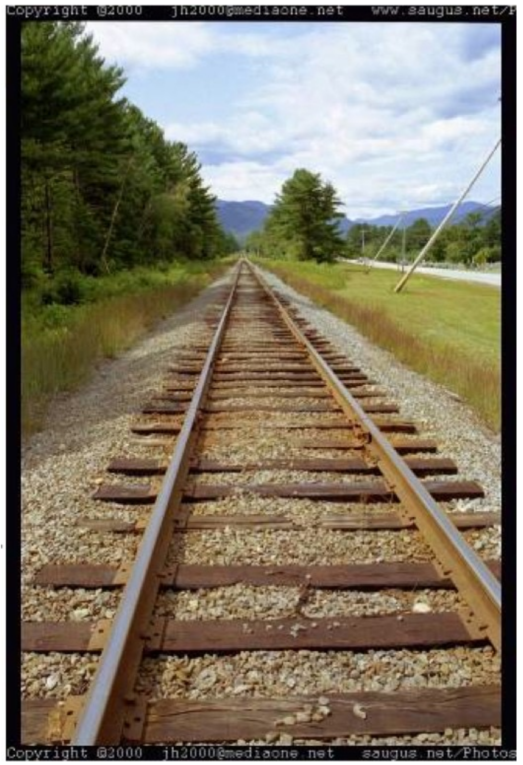
- Many-to-one: any points along same ray map to same point in image
- Points → points
- Lines → lines
- But line through center of projection maps to a point
- Planes → planes
- But plane through the center of projection maps to line

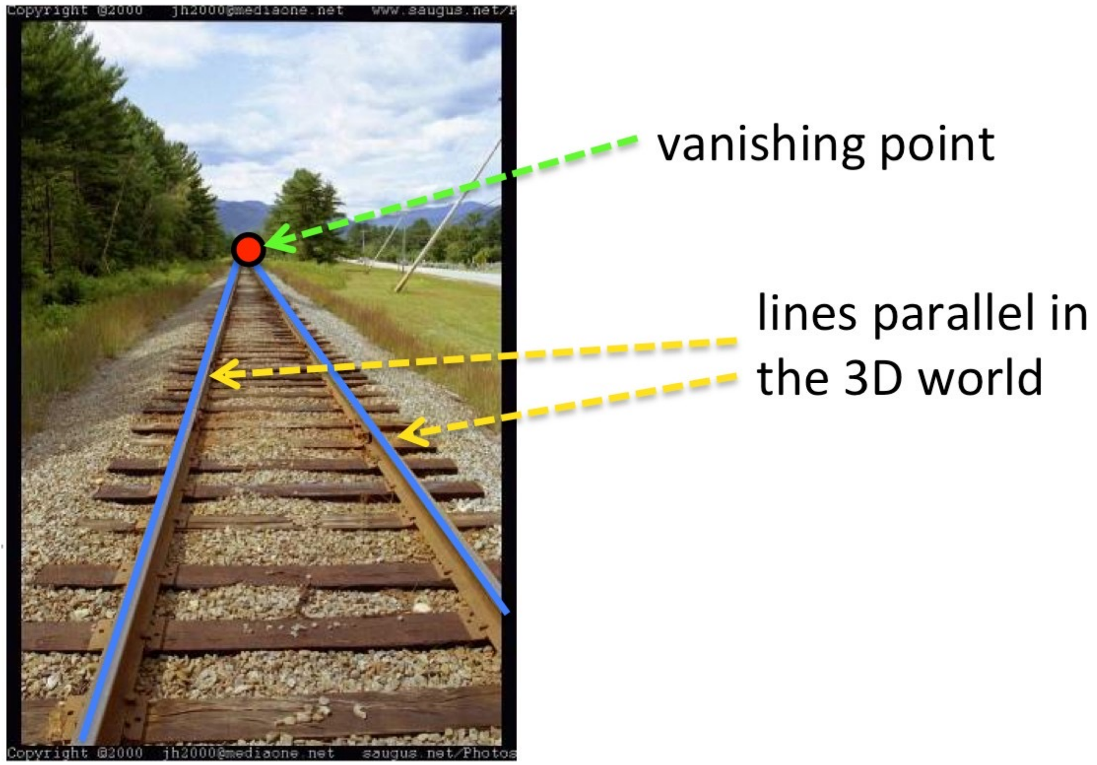


Projection Properties: Cool Facts

Parallel lines converge at a vanishing point

- Each different direction in the world has its own vanishing point


Copyright ©2000 jh2000@mediaone.net www.saugus.net/P
Copyright ©2000 jh2000@mediaone.net saugus.net/Photos

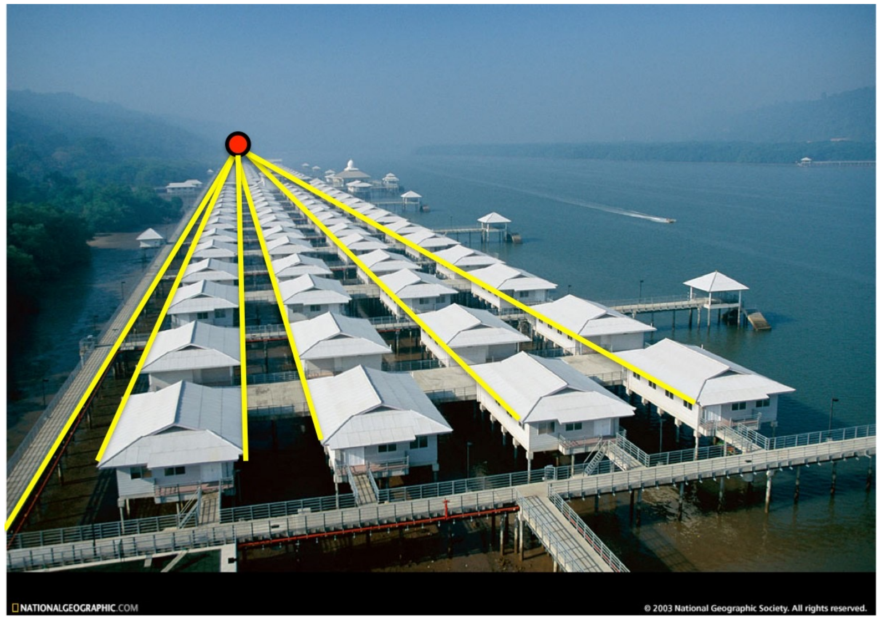

vanishing point
lines parallel in the 3D world
Copyright ©2000 jh2000@mediaone.net www.saugus.net/P
Copyright ©2000 jh2000@mediaone.net saugus.net/Photos

Adopted from: N. Snavely]

Projection Properties: Cool Facts

Parallel lines converge at a vanishing point

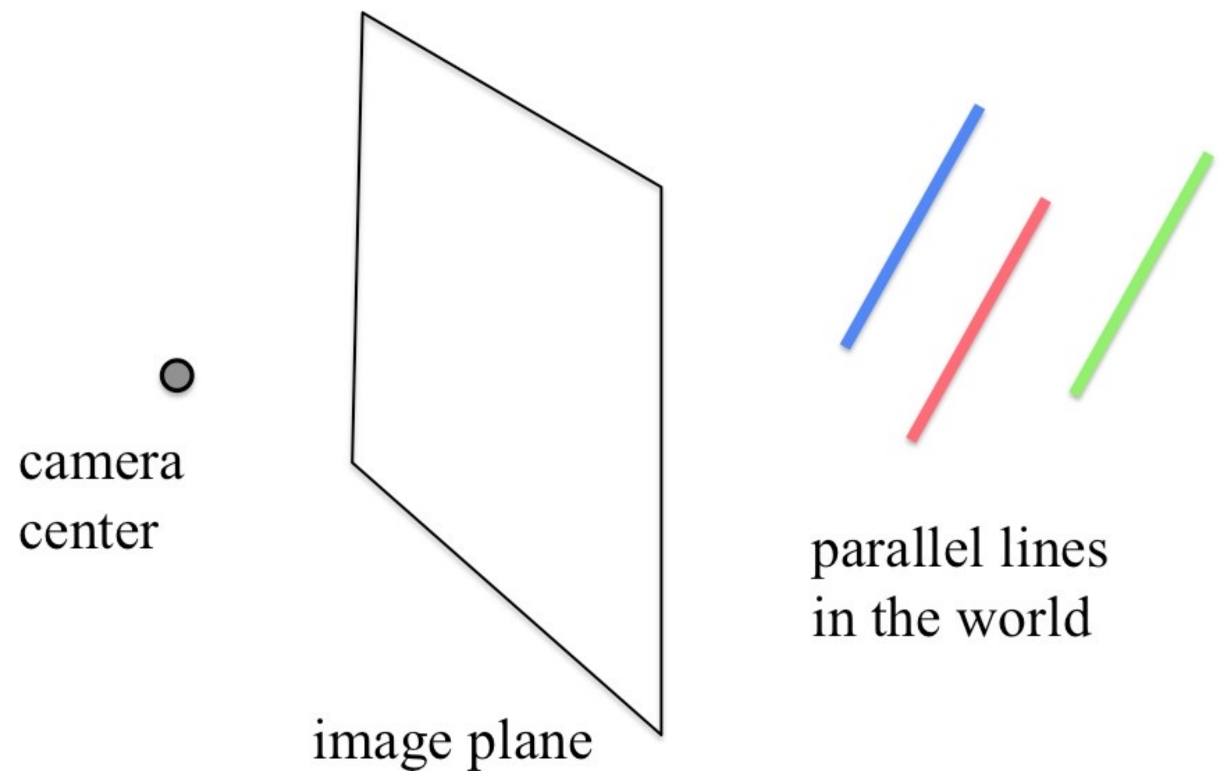
- Each different direction in the world has its own vanishing point
- All lines in the same direction in 3D intersect at the same vanishing point



[Pic: R. Szeliski]

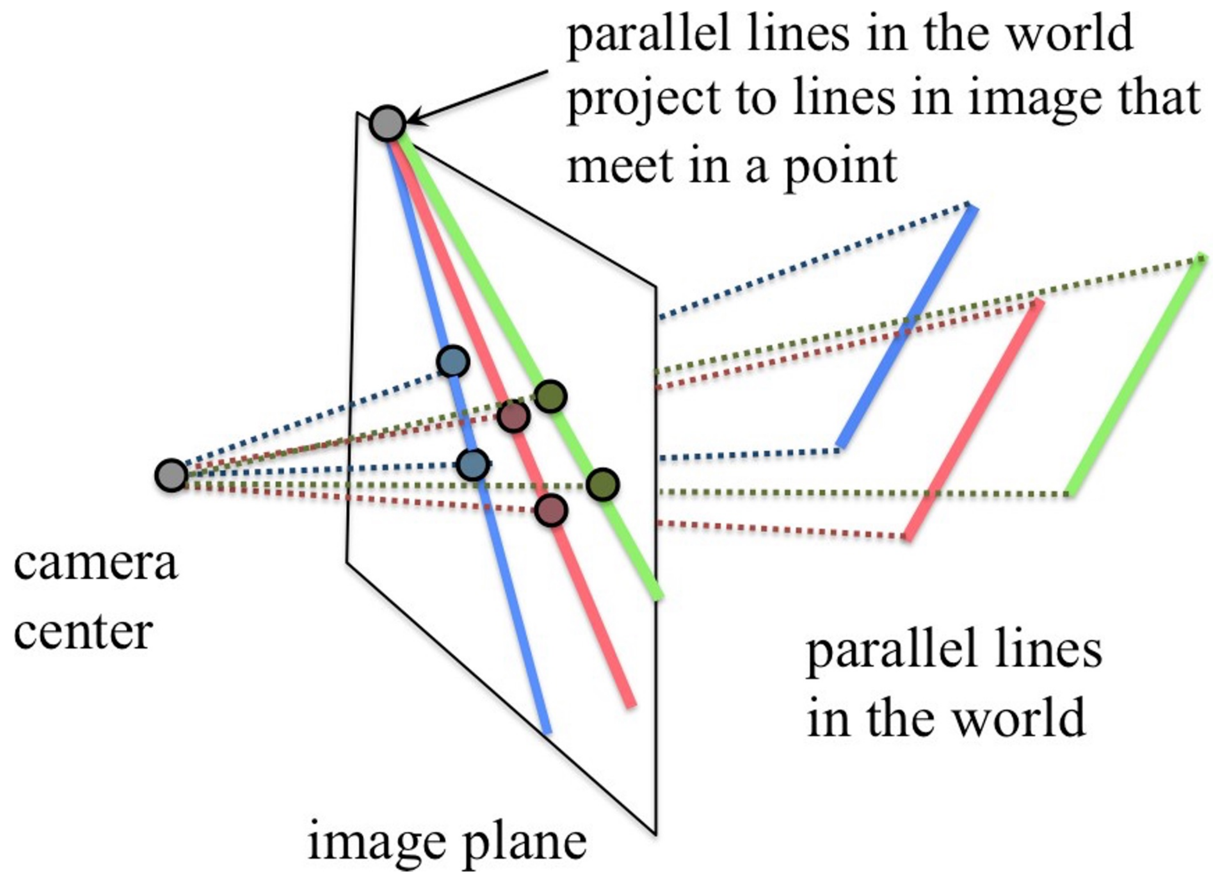
Projection Properties: Vanishing Point

- All lines in the same direction in 3D intersect at the same vanishing point.



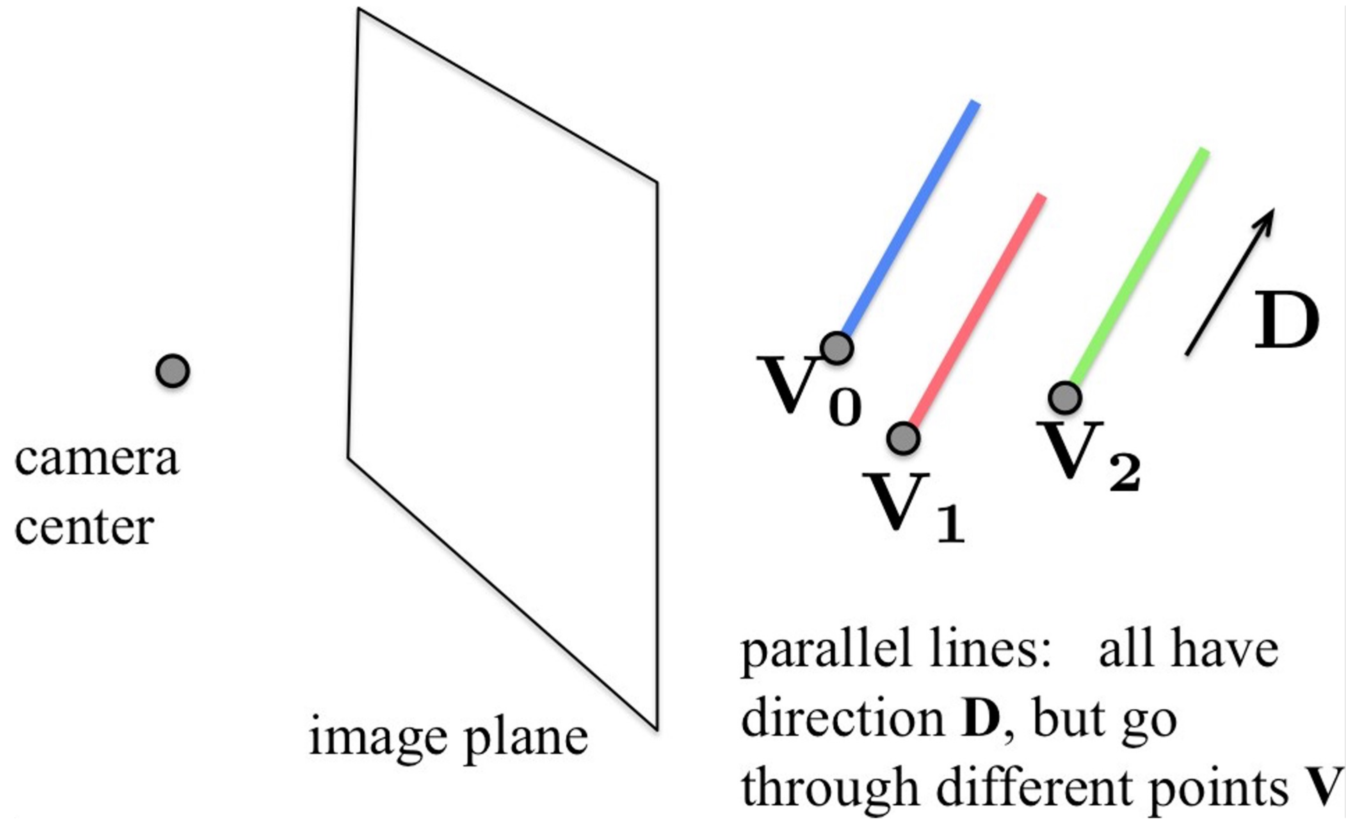
Projection Properties: Vanishing Point

- All lines in the same direction in 3D intersect at the same vanishing point.


parallel lines in the world project to lines in image that meet in a point
camera center
parallel lines in the world
image plane

Projection Properties: Vanishing Point

- All lines in the same direction in 3D intersect at the same vanishing point.


camera center
image plane
$\mathbf{V_0}$
$\mathbf{V_1}$
$\mathbf{V_2}$
$\mathbf{D}$
parallel lines: all have direction **D**, but go through different points **V**

Projection Properties: Vanishing Point

- All lines with the same 3D direction intersect at the same vanishing point.
- Line that passes through V with direction D: X = V + t D.

Projection Properties: Vanishing Point

- All lines with the same 3D direction intersect at the same vanishing point.
- Line that passes through V with direction D: X = V + t D.
- Project it:

$$\begin{bmatrix} wx \\ wy \\ w \end{bmatrix} = K\mathbf{X} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} V_x + tD_x \\ V_y + tD_y \\ V_z + tD_z \end{bmatrix} = \begin{bmatrix} fV_x + ftD_x + p_xV_z + tp_xD_z \\ fV_y + ftD_y + p_yV_z + tp_yD_z \\ V_z + tD_z \end{bmatrix}$$

Projection Properties: Vanishing Point

- All lines with the same 3D direction intersect at the same vanishing point.
- Line that passes through V with direction D: X = V + t D.
- Project it:

$$\begin{bmatrix} wx \\ wy \\ w \end{bmatrix} = K\mathbf{X} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} V_x + tD_x \\ V_y + tD_y \\ V_z + tD_z \end{bmatrix} = \begin{bmatrix} fV_x + ftD_x + p_xV_z + tp_xD_z \\ fV_y + ftD_y + p_yV_z + tp_yD_z \\ V_z + tD_z \end{bmatrix}$$

Move infinitely far from the camera by taking t → ∞ and compute x and y:

$$x = \lim_{t\to\infty} \frac{fV_x + ftD_x + p_xV_z + tp_xD_z}{V_z + tD_z} = \frac{fD_x + p_xD_z}{D_z}$$

$$y = \lim_{t\to\infty} \frac{fV_y + ftD_y + p_yV_z + tp_yD_z}{V_z + tD_z} = \frac{fD_y + p_yD_z}{D_z}$$

Projection Properties: Vanishing Point

- All lines with the same 3D direction intersect at the same vanishing point.
- Line that passes through V with direction D: X = V + t D.
- Project it:

$$\begin{bmatrix} wx \\ wy \\ w \end{bmatrix} = K\mathbf{X} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} V_x + tD_x \\ V_y + tD_y \\ V_z + tD_z \end{bmatrix} = \begin{bmatrix} fV_x + ftD_x + p_xV_z + tp_xD_z \\ fV_y + ftD_y + p_yV_z + tp_yD_z \\ V_z + tD_z \end{bmatrix}$$

Move infinitely far from the camera by taking t → ∞ and compute x and y:

$$x = \lim_{t\to\infty} \frac{fV_x + ftD_x + p_xV_z + tp_xD_z}{V_z + tD_z} = \frac{fD_x + p_xD_z}{D_z}$$

$$y = \lim_{t\to\infty} \frac{fV_y + ftD_y + p_yV_z + tp_yD_z}{V_z + tD_z} = \frac{fD_y + p_yD_z}{D_z}$$

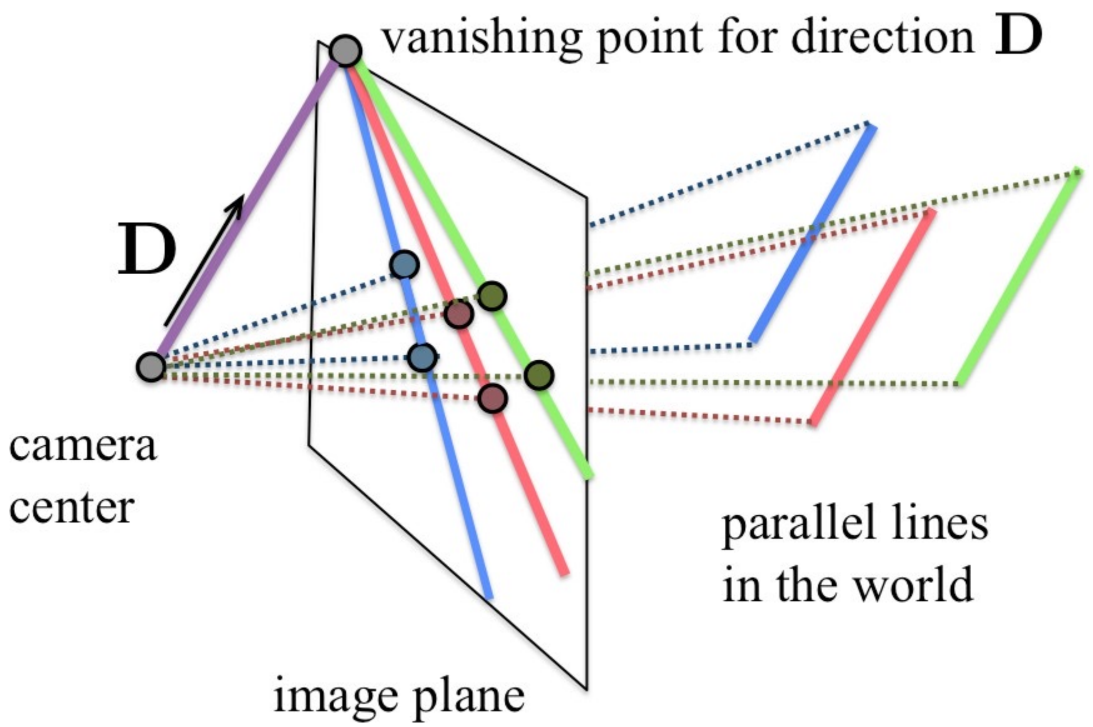
This doesn’t depend on V! So all lines with direction D go to this point!

Projection Properties: Vanishing Point

- All lines with the same 3D direction intersect at the same vanishing point.

Projection Properties: Vanishing Point

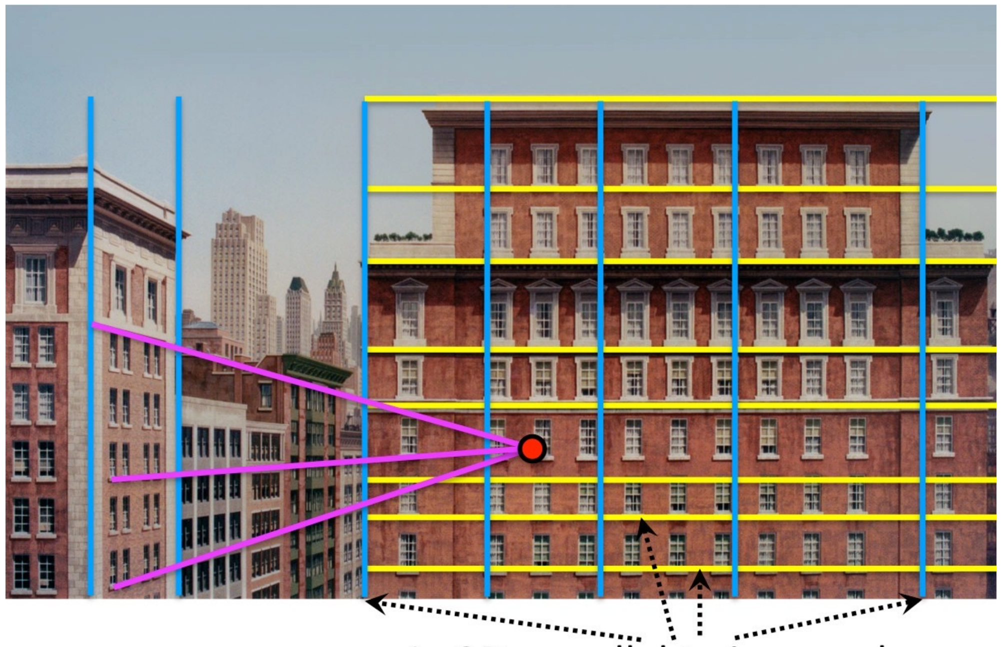
- All lines with the same 3D direction intersect at the same vanishing point.
- The easiest way to find this point: Translate line with direction D to the camera center. This line intersects the image plane in the vanishing point corresponding to direction D!


vanishing point for direction **D**
D
camera center
image plane
parallel lines in the world

Projection Properties: Cool Facts

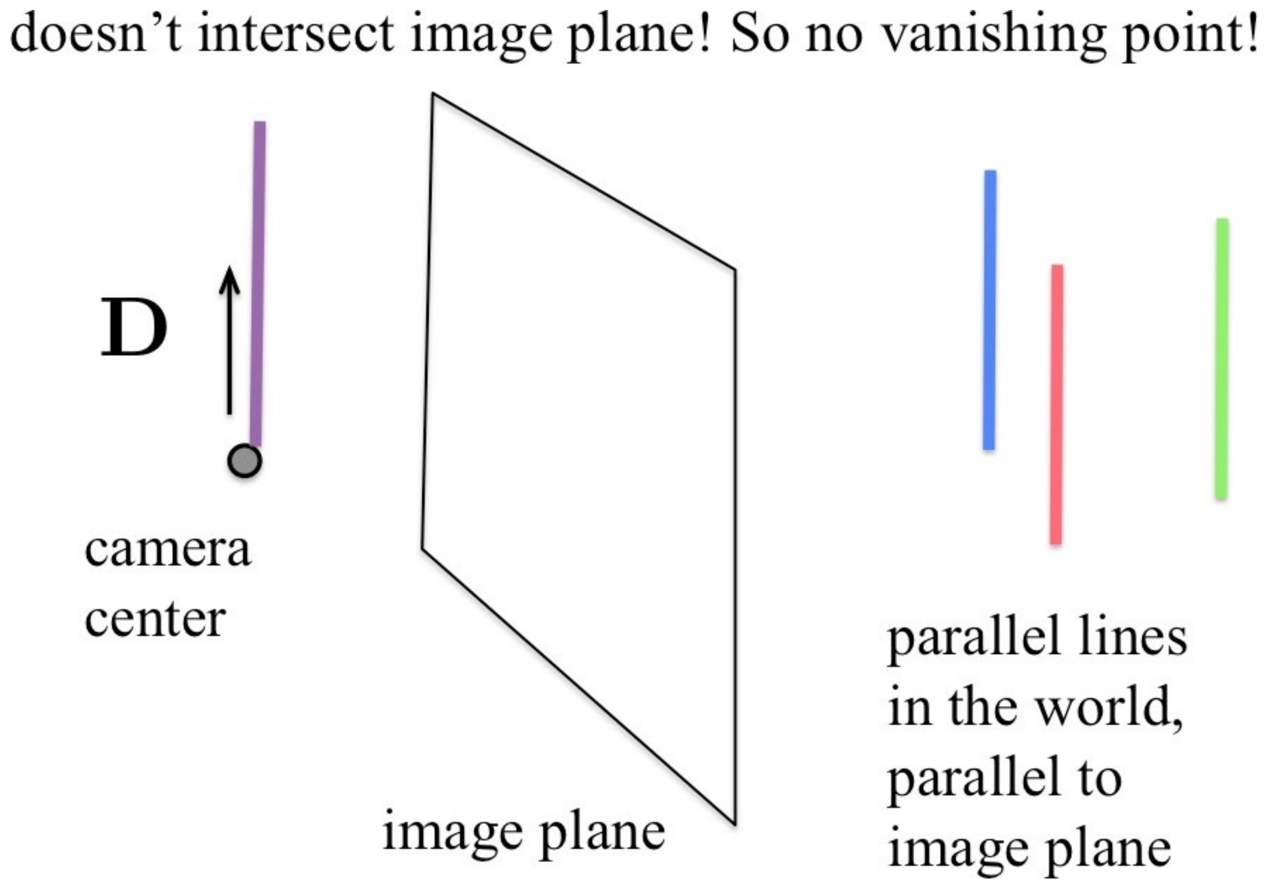
Parallel lines converge at a vanishing point

- Each different direction in the world has its own vanishing point
- Lines parallel to image plane are also parallel in the image (we say that they intersect at infinity).



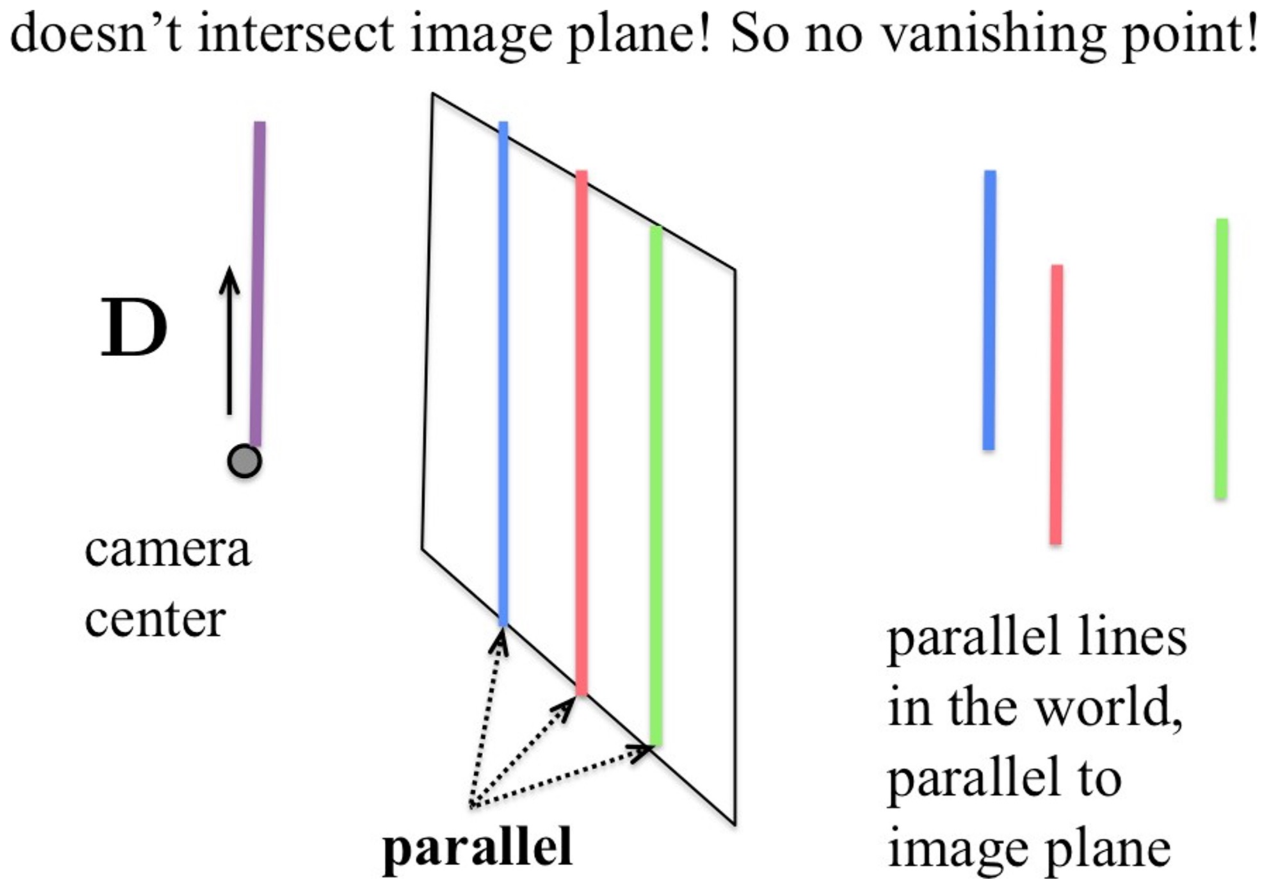
Projection Properties: Cool Facts

- Lines parallel to image plane are also parallel in the image. We say that they intersect at infinity.


doesn't intersect image plane! So no vanishing point!
D
camera center
image plane
parallel lines in the world, parallel to image plane

Projection Properties: Cool Facts

- Lines parallel to image plane are also parallel in the image. We say that they intersect at infinity.


doesn't intersect image plane! So no vanishing point!
D
camera center
parallel
parallel lines in the world, parallel to image plane

Projection Properties: Vanishing Point

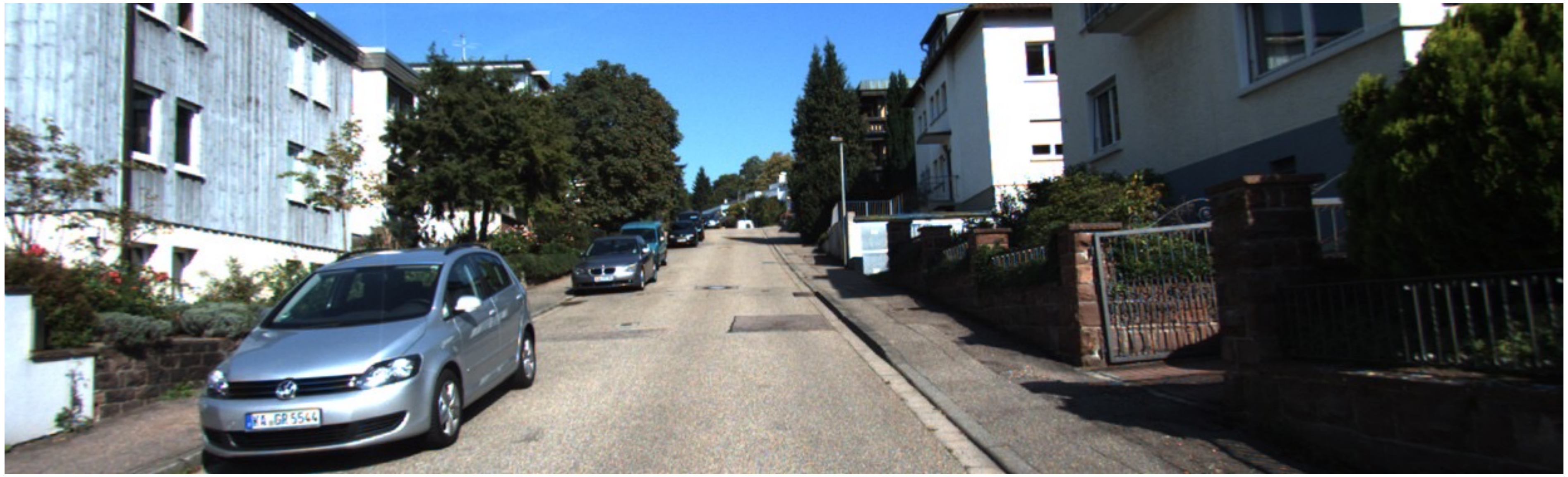
- Line that passes through V with direction D: X = V + t D.
- Project a second line passing through *R* and take the distance:

$$d_x = \frac{fV_x + ftD_x + p_xV_z + tp_xD_z}{V_z + tD_z} - \frac{fR_x + ftD_x + p_xR_z + tp_xD_z}{R_z + tD_z}$$

- Simplify with $D_z = 0$ and $V_z = R_z = 1$
- Can you prove the above distance remains constant in the image plane regardless of t i.e., the projected versions of the lines are also parallel.

Projection Properties: Cool Tricks

- This picture has been recorded from a car with a camera on top. We know the camera intrinsic matrix K.
- Can we tell the incline of the hill we are driving on?
- How?



Projection Properties: Cool Tricks

- Can we tell the incline of the hill we are driving on?

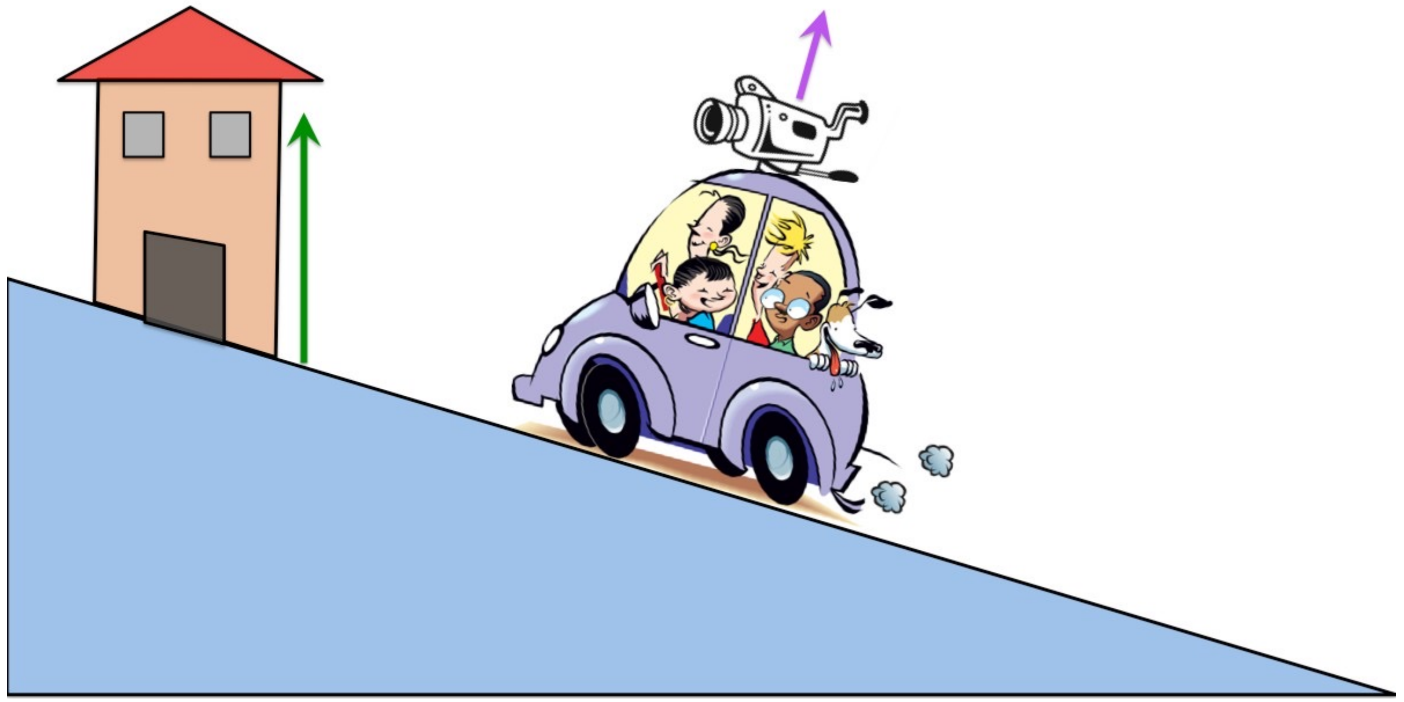


Figure: This is the 3D world behind the picture.

Projection Properties: Cool Tricks

- Can we tell the incline of the hill we are driving on?

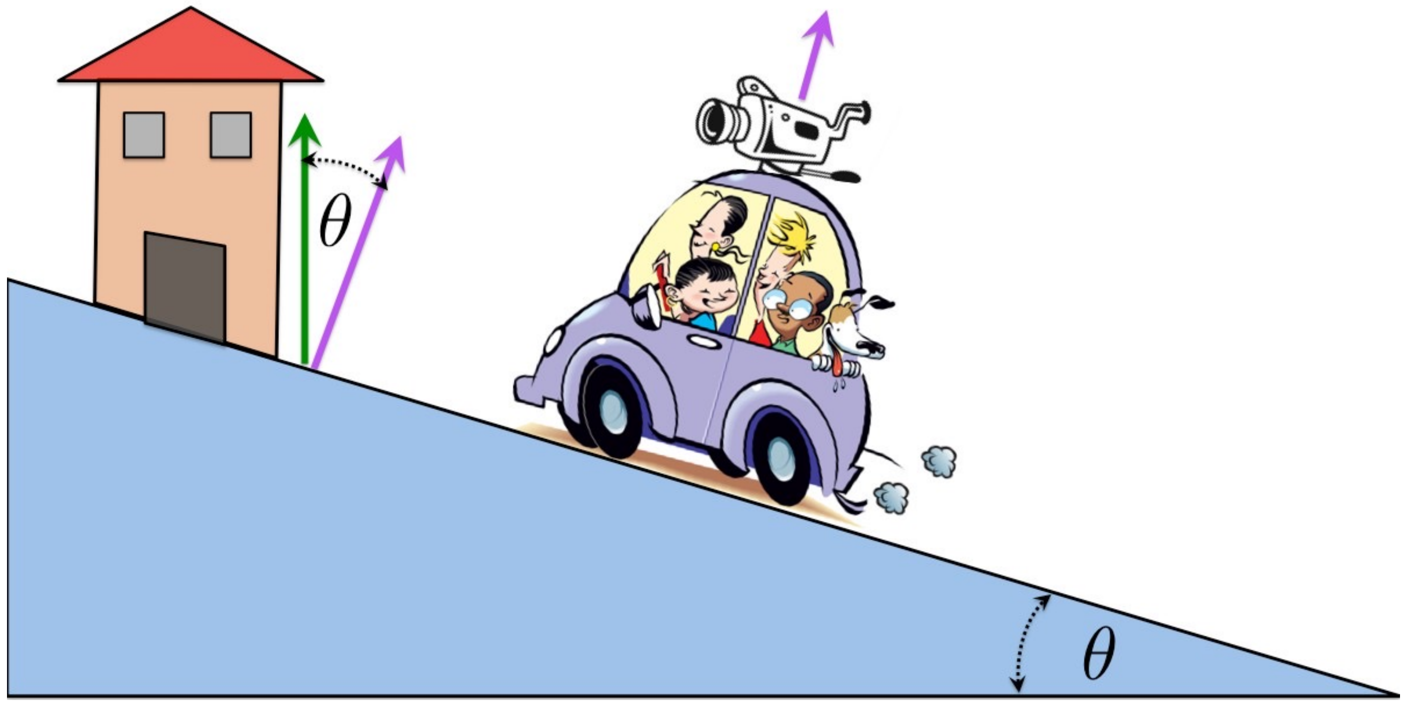


Figure: If we compute the 3D direction of the house's vertical lines relative to camera, we have the incline! How can we do that?

Projection Properties: Cool Tricks

- Can we tell the incline of the hill we are drivina on?

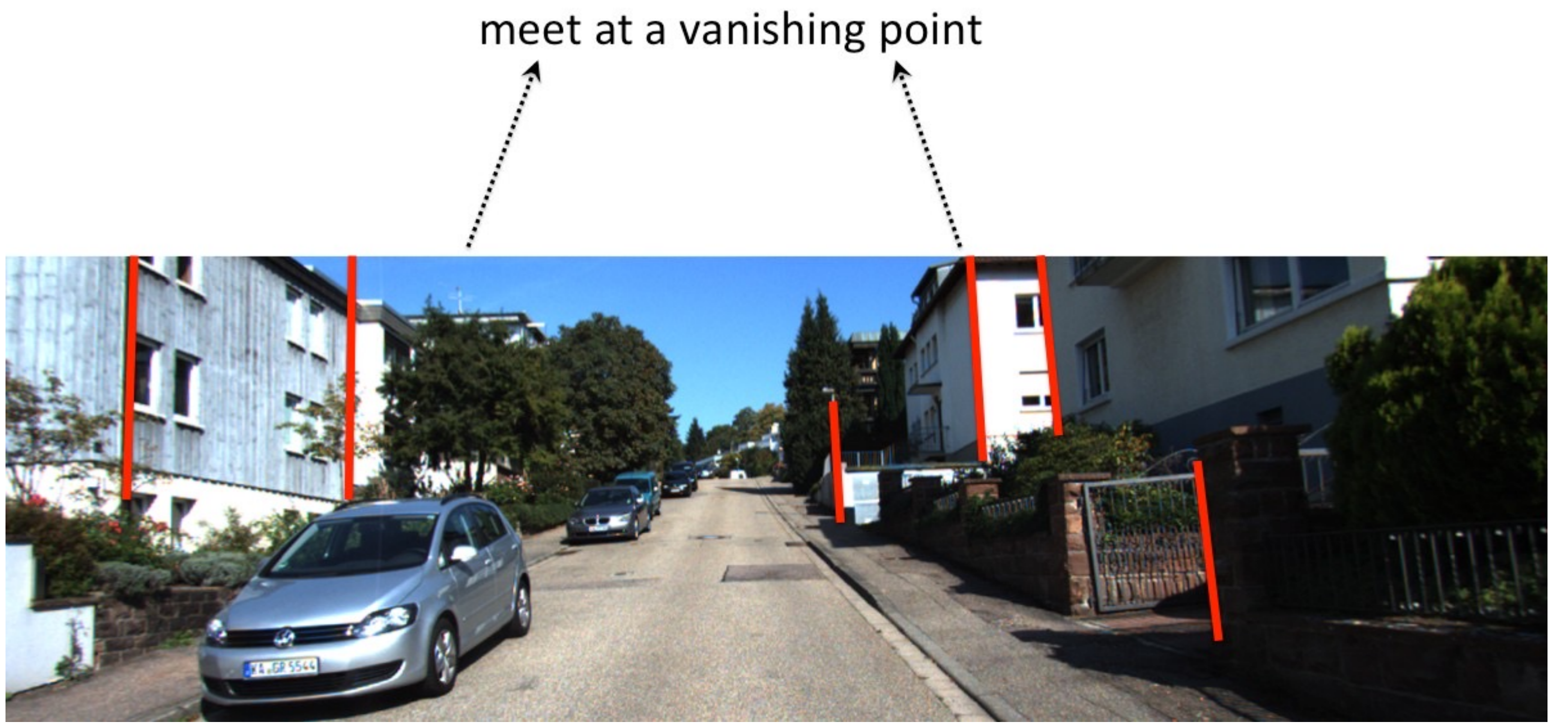


Figure: Extract "vertical" lines and compute vanishing point in 2D via intersections. Ho can we compute direction in 3D from vanishing point (if we have K)?

Projection Properties: Cool Tricks

- Can we tell the incline of the hill we are driving on?

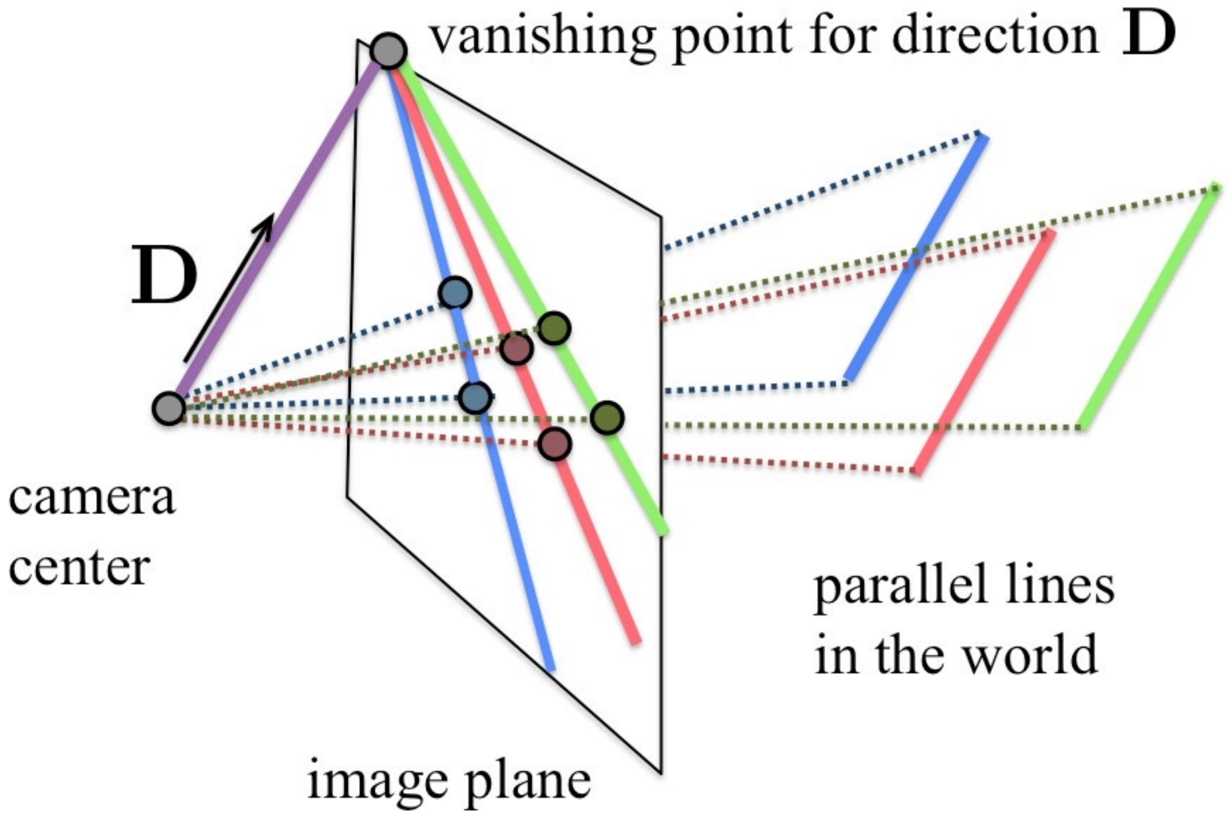
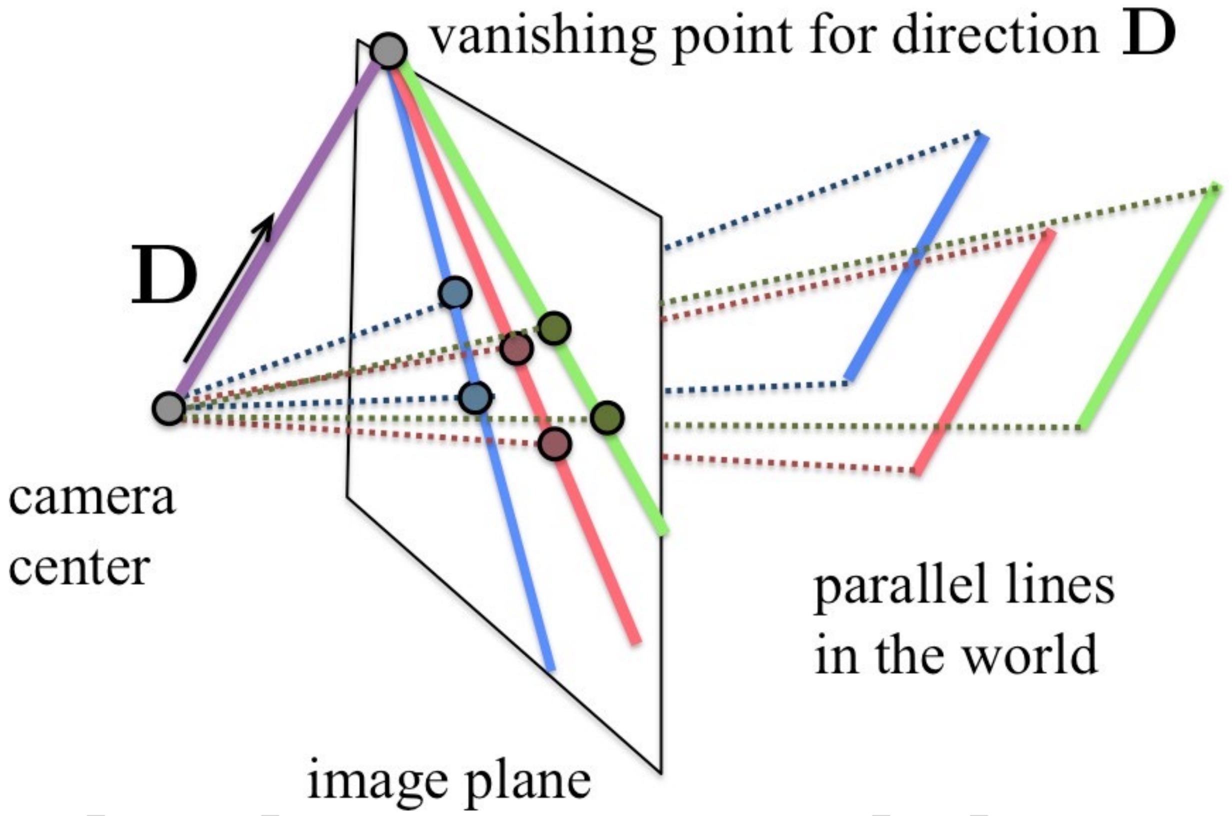

vanishing point for direction $\mathbf{D}$
$\mathbf{D}$
camera center
parallel lines in the world
image plane

Figure: This picture should help.

Projection Properties: Cool Tricks

- Can we tell the incline of the hill we are driving on?

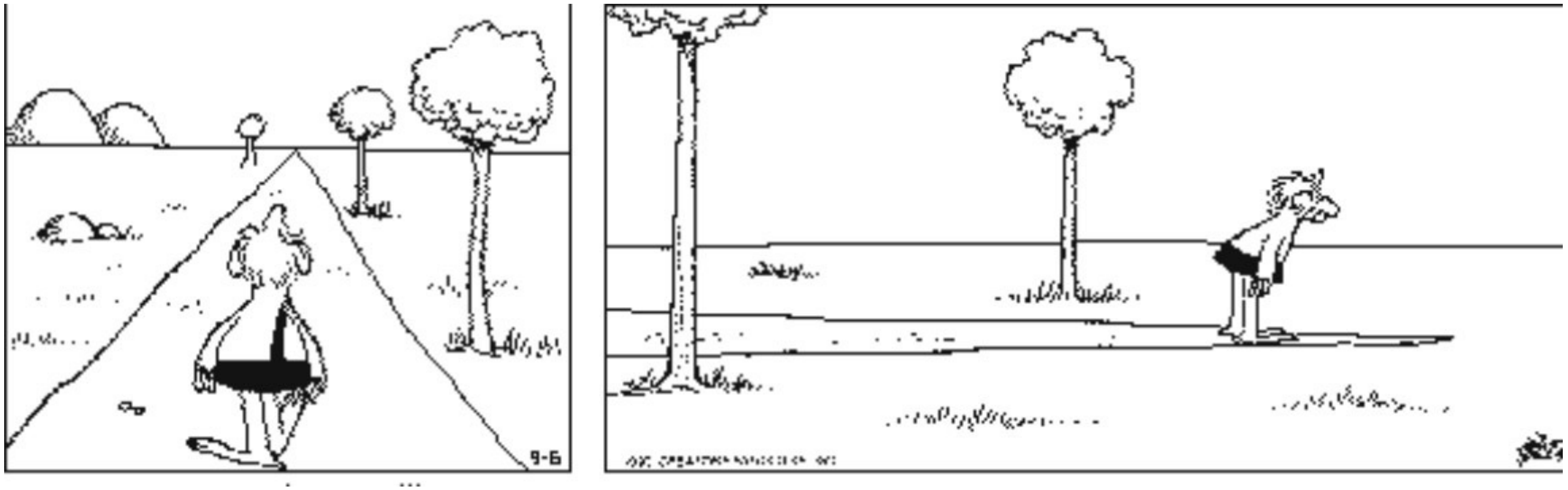


- We have $\begin{bmatrix} w \cdot vp_x \\ w \cdot vp_y \\ w \end{bmatrix} = K\mathbf{D} \rightarrow \mathbf{D} = wK^{-1}\begin{bmatrix} vp_x \\ vp_y \\ 1 \end{bmatrix} \rightarrow$ normalize **D** to norm 1

Vanishing Points Can be Deceiving

Parallel lines converge at a vanishing point.

- But intersecting lines in 2D are not necessarily parallel in 3D.

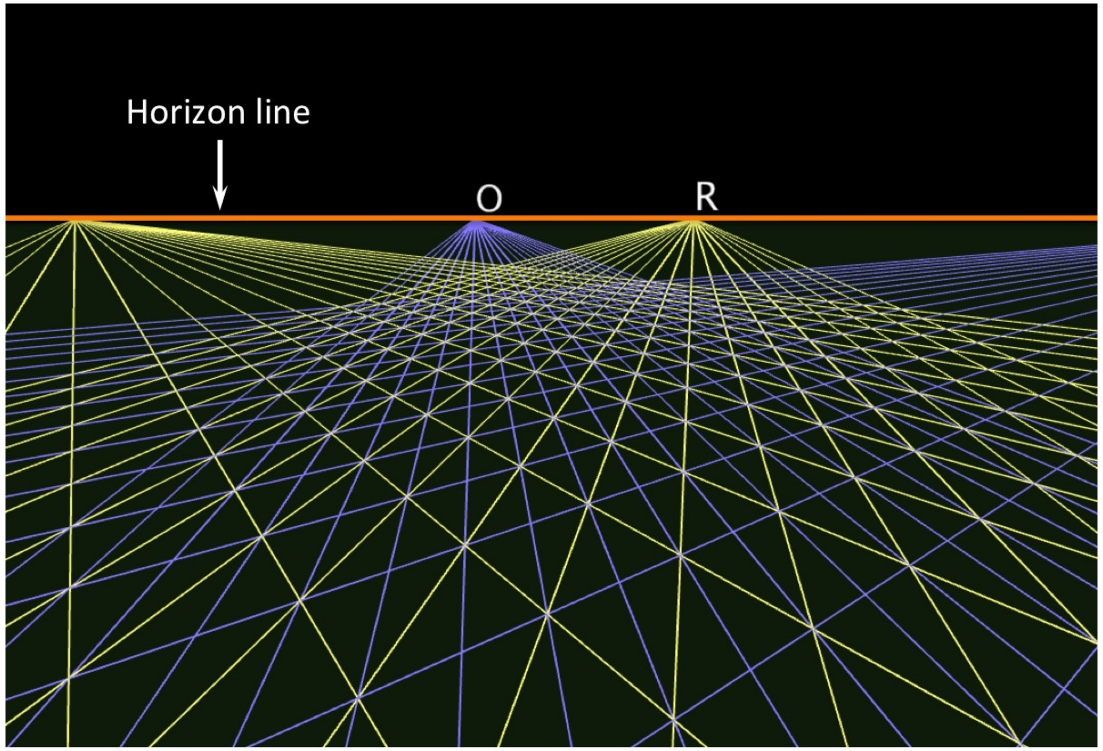


[Source: A. Jepson]

Vanishing Points Can be Deceiving

Parallel lines converge at a vanishing point

- Each different direction in the world has its own vanishing point
- For lines on the same 3D plane, the vanishing points lie on a line. We call it a vanishing line. Vanishing line for the ground plane is a horizon line.



http://4.bp.blogspot.com/-0Jm9d9j35Tc/T5ESbVpKl7I/AAAAAAAACEk/nVAiTxBuiyc/s1600/perspectiveGrid-01.png

Projection Properties: Cool Facts

Parallel lines converge at a vanishing point

- For lines on the same 3D plane, the vanishing points lie on a line. We call it a vanishing line. Vanishing line for the ground plane is a horizon line.
- Some horizon lines are nicer than others ;)

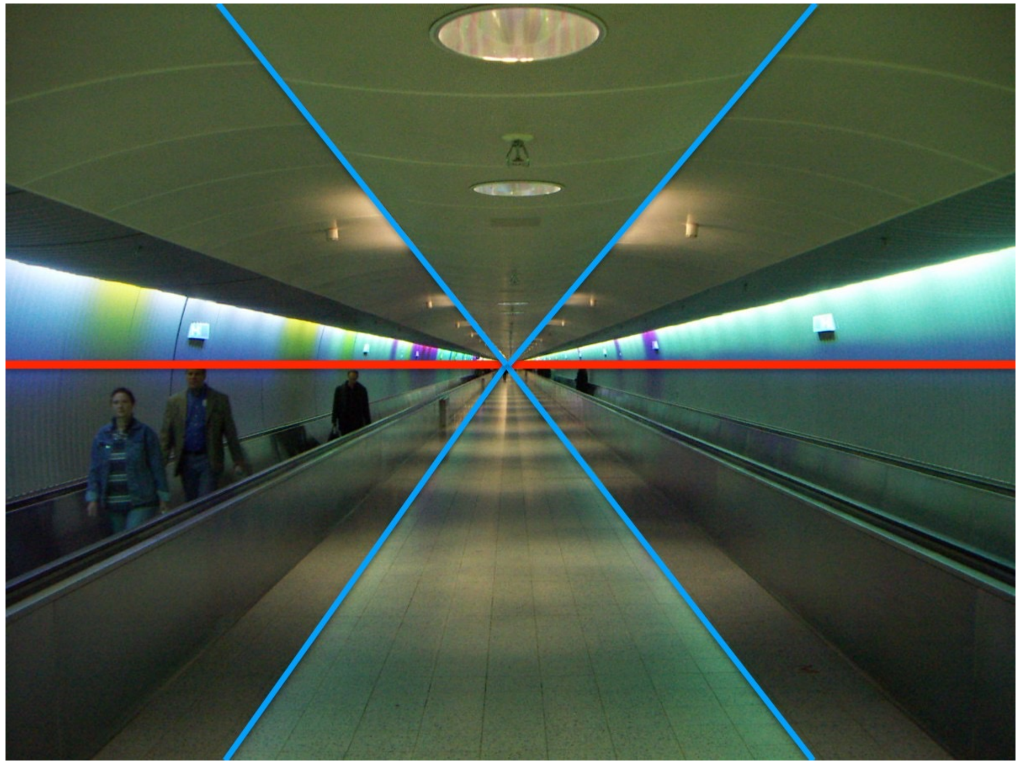
[Punta Cana]

Projection Properties: Cool Facts

Parallel lines converge at a vanishing point

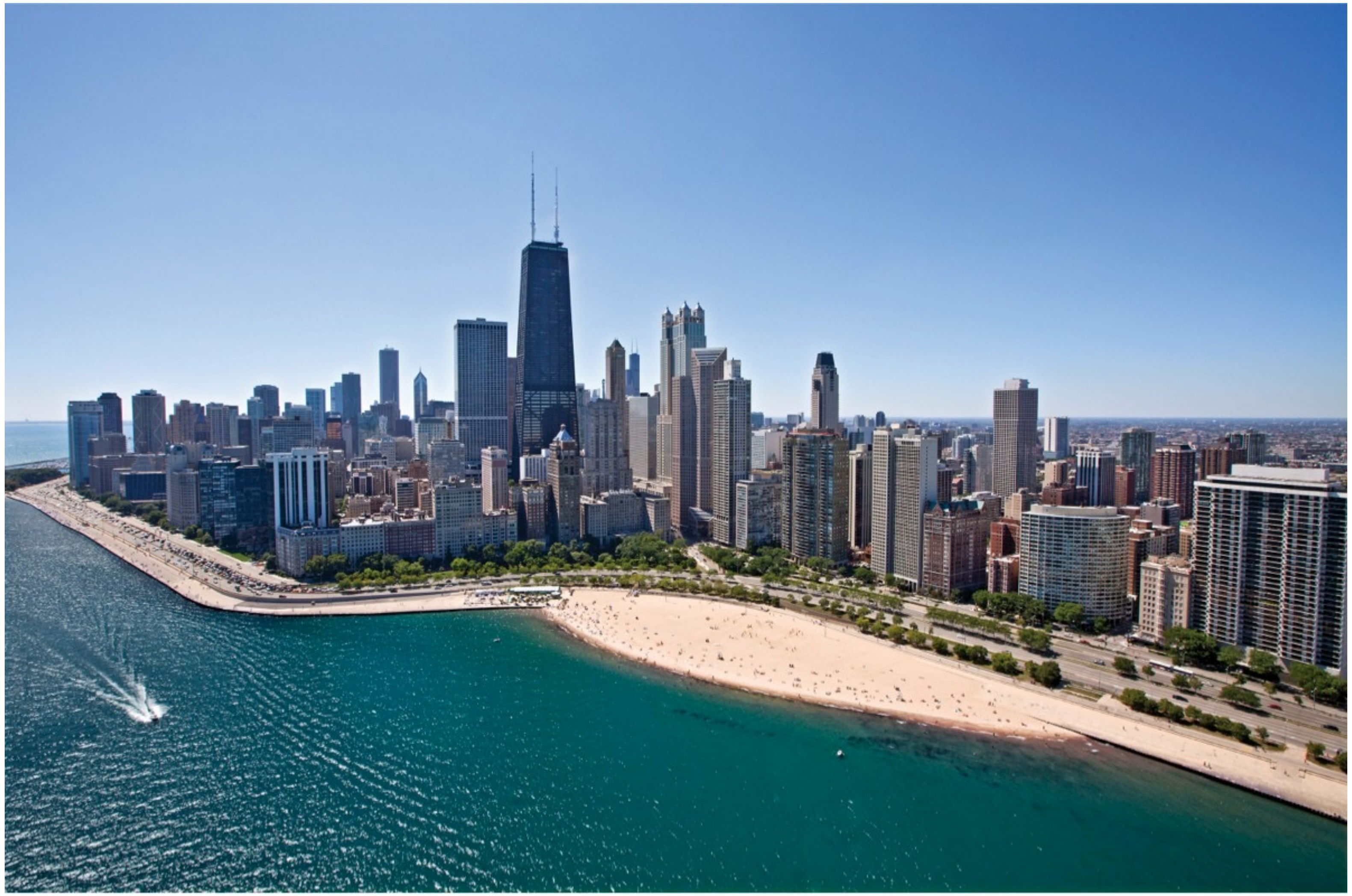
- For lines on the same 3D plane, the vanishing points lie on a line. We call it a vanishing line. Vanishing line for the ground plane is a horizon line.
- Parallel planes in 3D have the same horizon line in the image.

http://upload.wikimedia.org/wikipedia/commons/d/d8/Frankfurt_Airport_tunnel.JPG



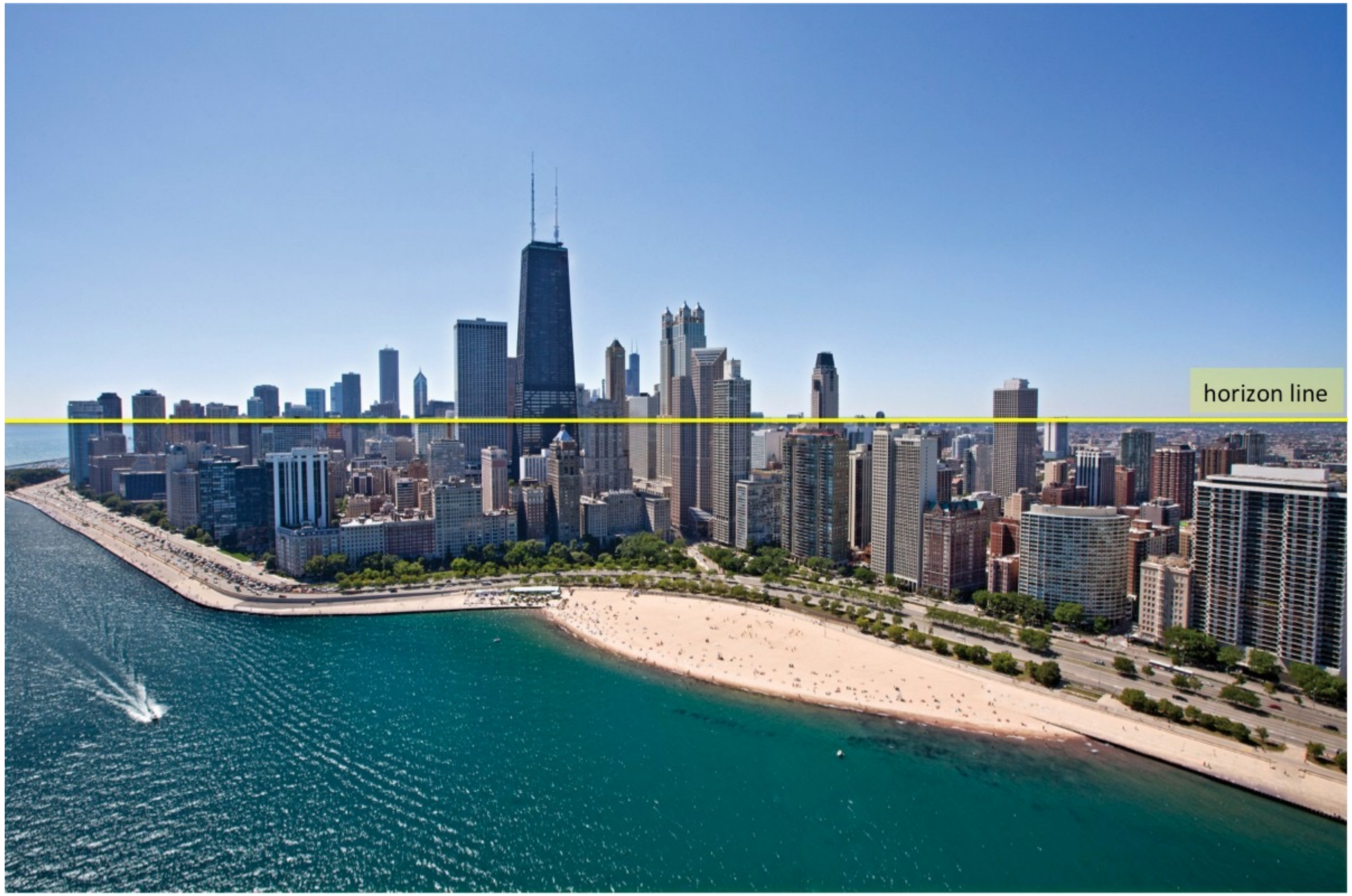
Projection Properties: Cool Facts

- Can I tell how much above ground this picture was taken?



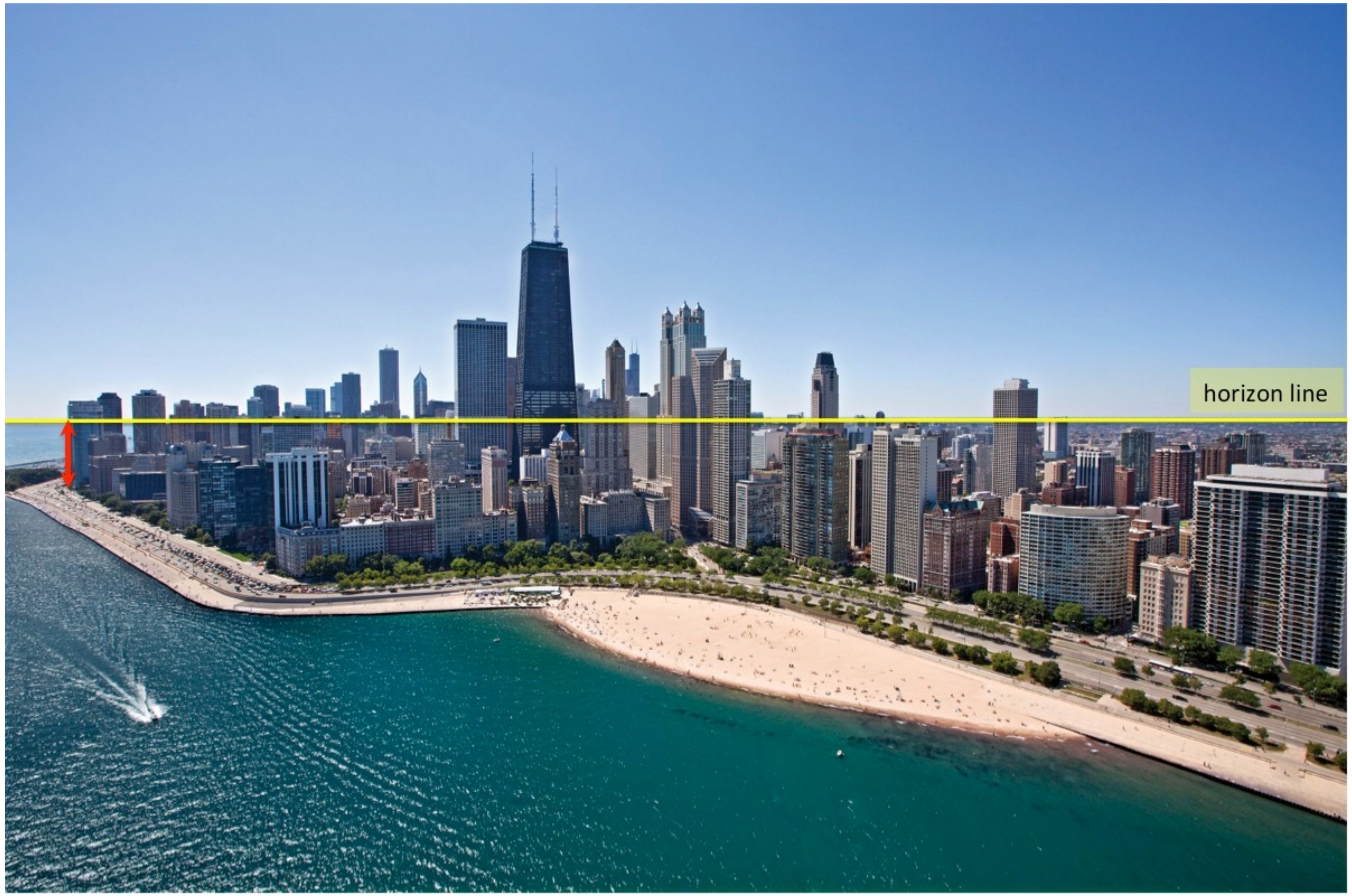
Projection Properties: Cool Facts

- Can I tell how much above ground this picture was taken?



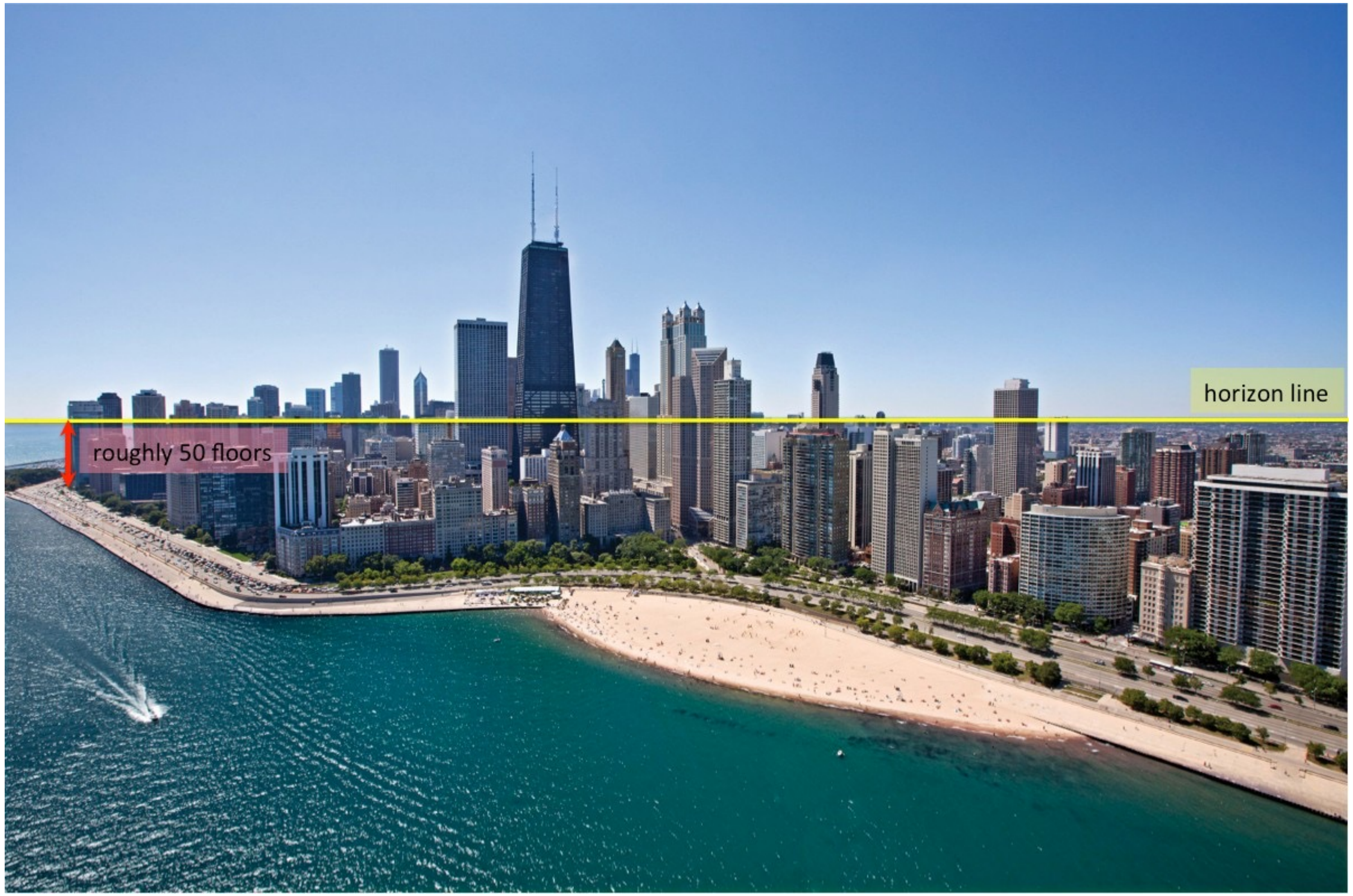
Projection Properties: Cool Facts

- Same distance as where the horizon line intersects the scene



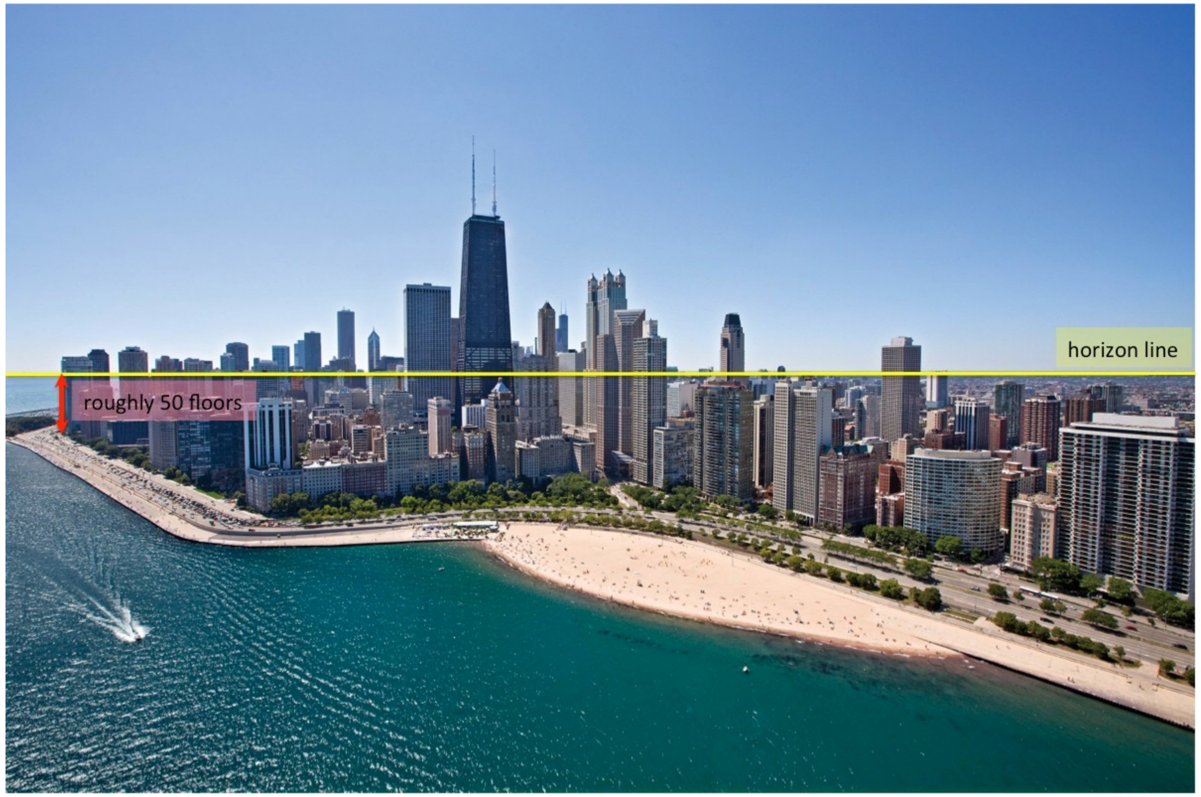
Projection Properties: Cool Facts

- Same distance as where the horizon line intersects the scene



Projection Properties: Cool Facts

- This is only true when the camera (image plane) is orthogonal to the ground plane. And the ground plane is flat.
- A very nice explanation of this phenomena can be find by Derek Hoiem here: https://courses.engr.illinois.edu/cs543/sp2011/materials/3dscene_book_svg.pdf

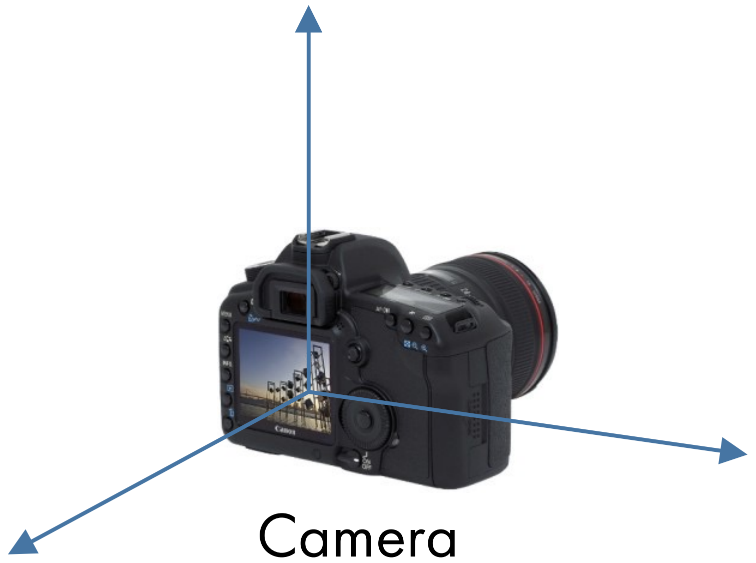


Camera Parameters

- We are not yet done with projection. To fully specify projection, we need to:
- Describe its internal parameters (we know this, this is our K)

Camera Parameters

- We are not yet done with projection. To fully specify projection, we need to:
- Describe its internal parameters (we know this, this is our K)
- Describe its pose in the world. Two important coordinate systems:
  - World coordinate system
  - Camera coordinate system

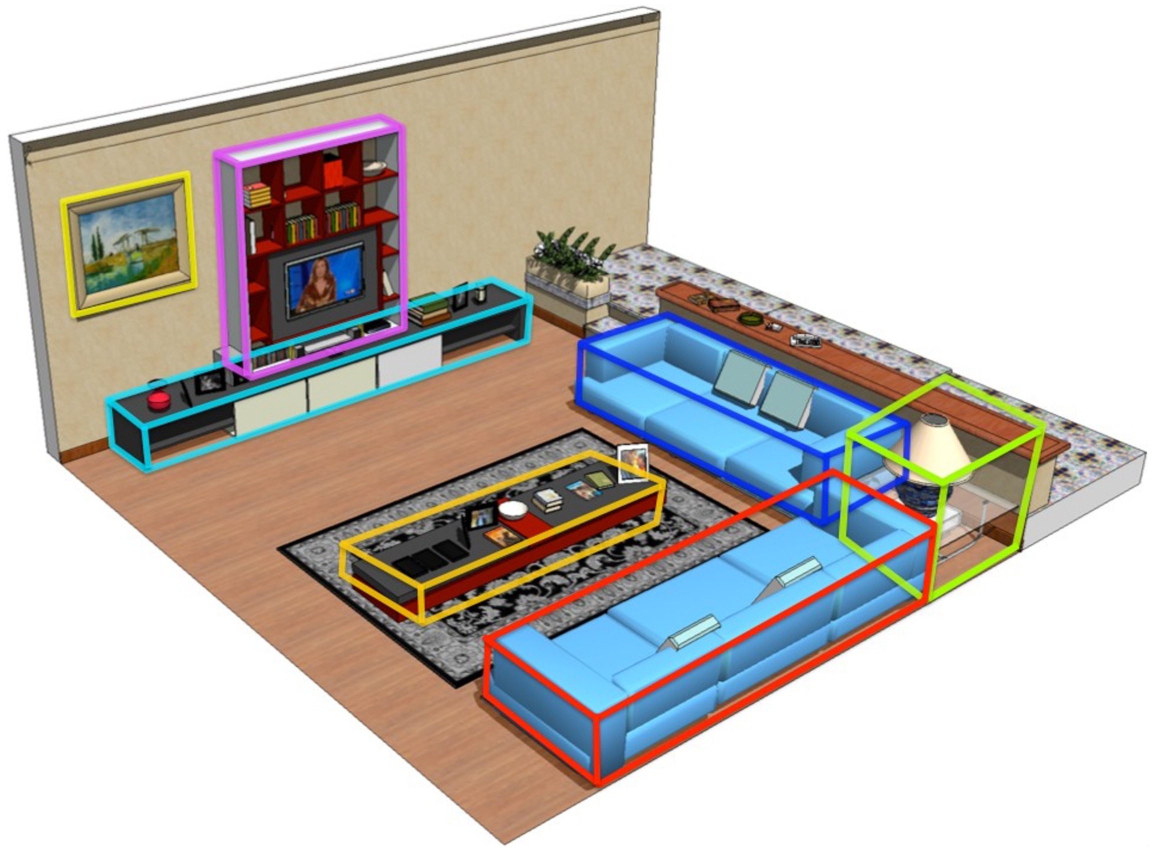

Camera

"The World"

[Source: N. Snavely]

Camera Parameters

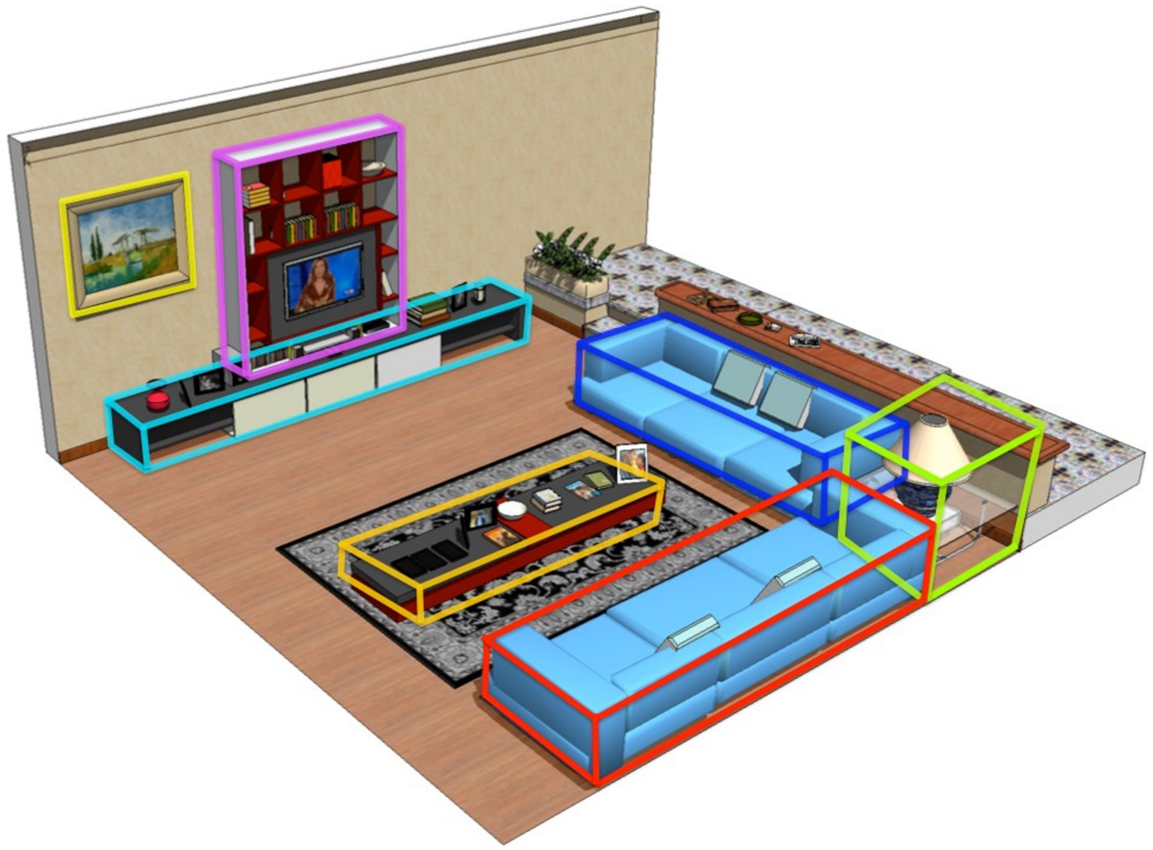
- Why two coordinate systems?



Imagine this is your room.

Camera Parameters

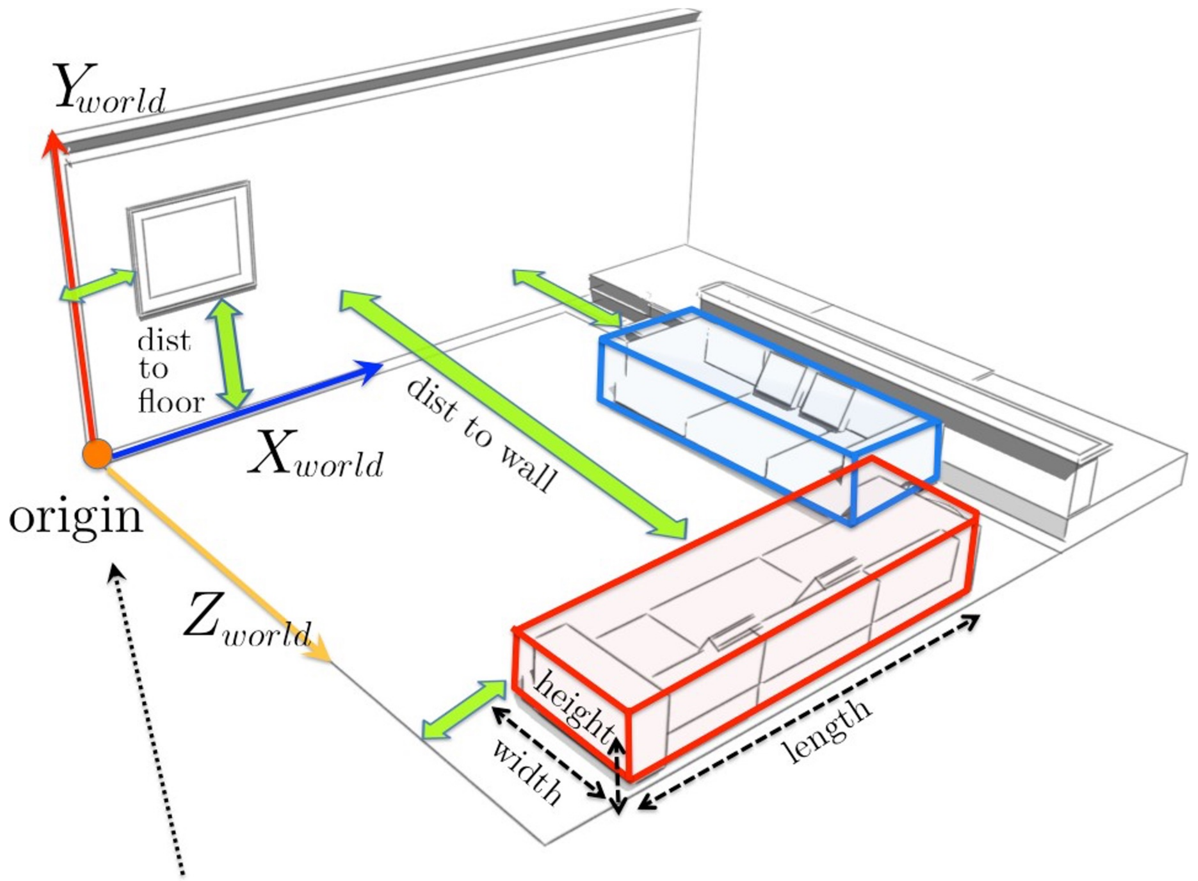
- Why two coordinate systems?



When you were furnishing you measured everything in detail.

Camera Parameters

- Why two coordinate systems?

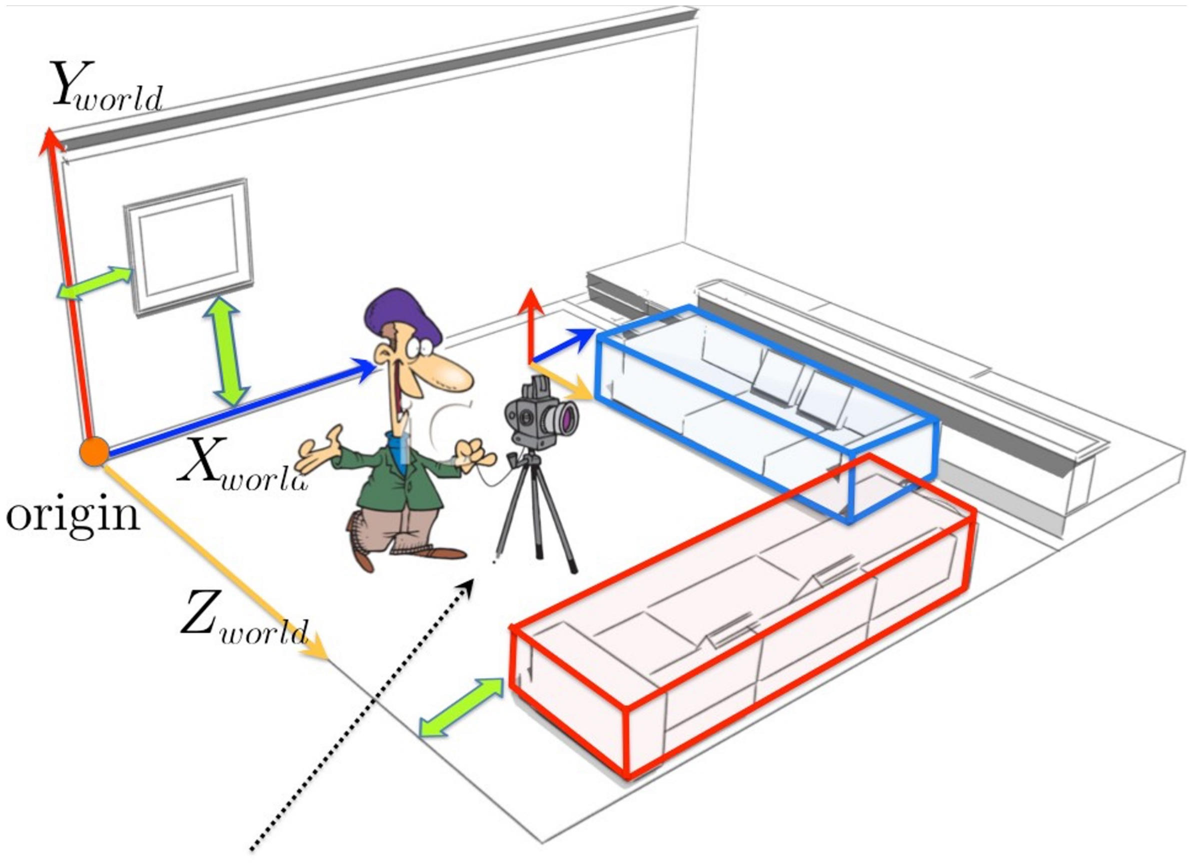


I measured everything in my room relative to this point

Thus you know all coordinates relative to a special point (origin) and coordinate system in the room. This is your room's (world) coordinate system.

Camera Parameters

- Why two coordinate systems?

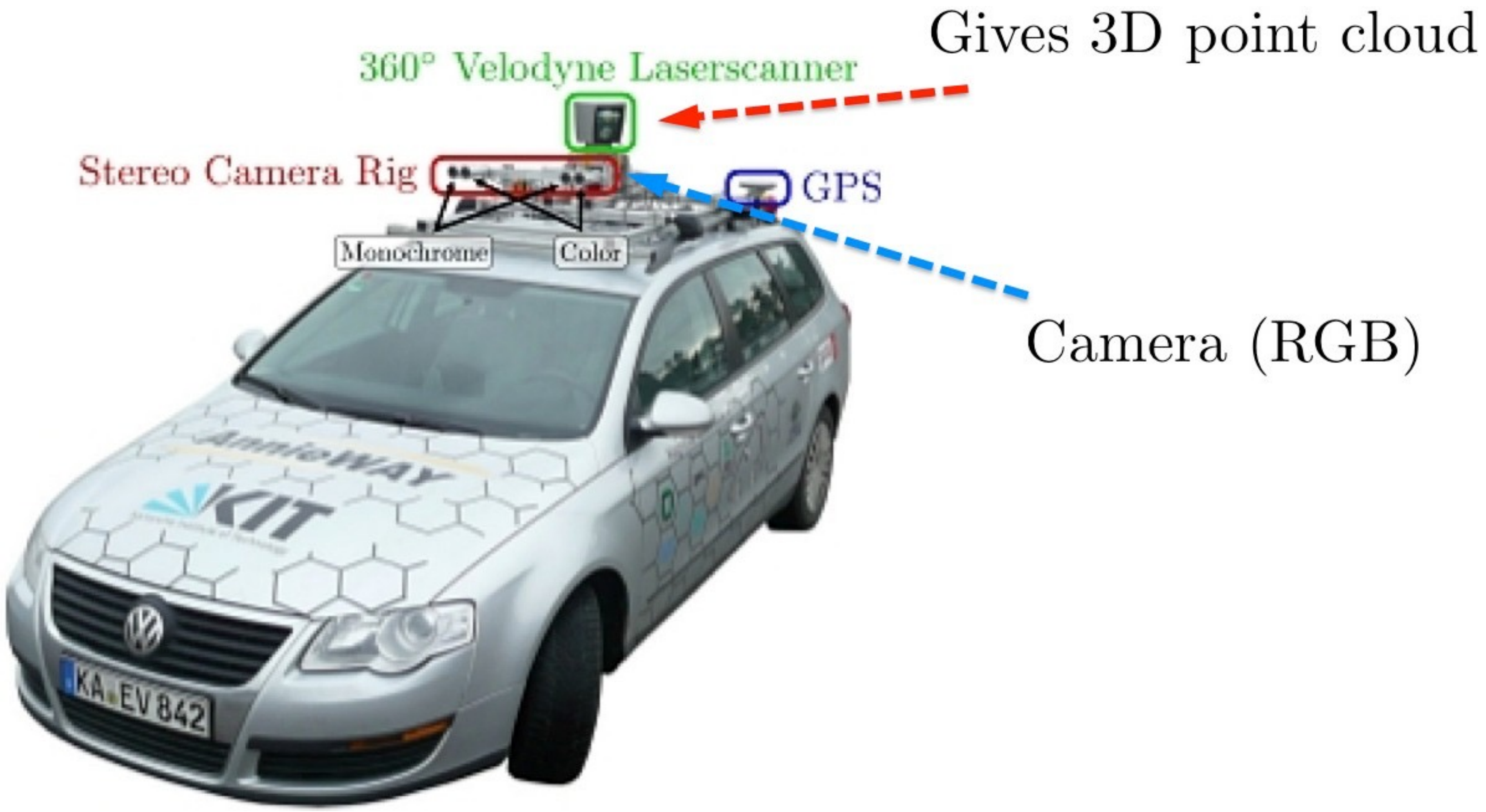


But to project my room to camera, I need to have the room in the camera coordinate system!

Now you take a picture and you wonder how points project to camera. In order to project, you need all points in camera's coordinate system.

Camera Parameters

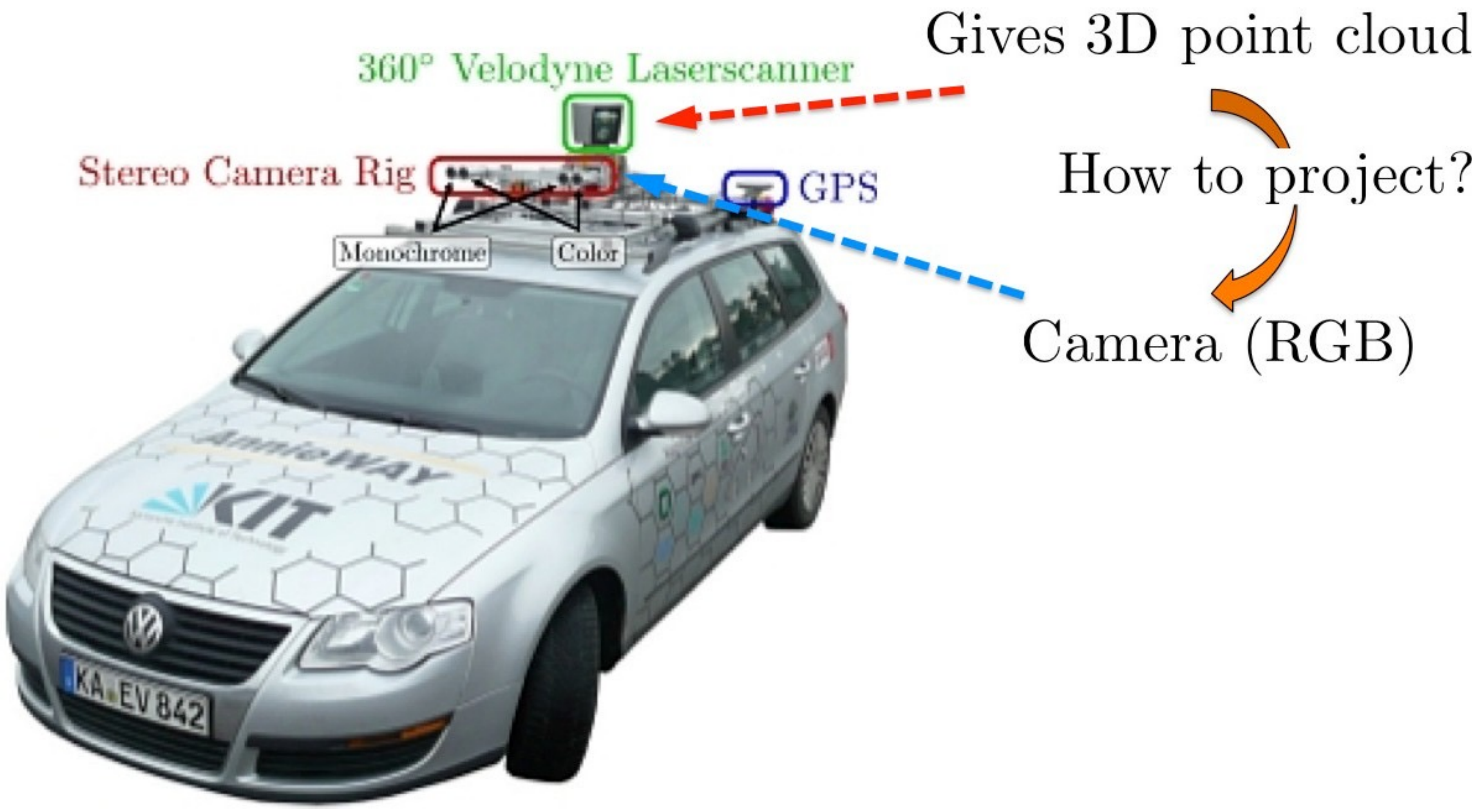
- Why two coordinate systems?



For e.g. self-driving cars, 3D points are typically measured with Velodyne.

Camera Parameters

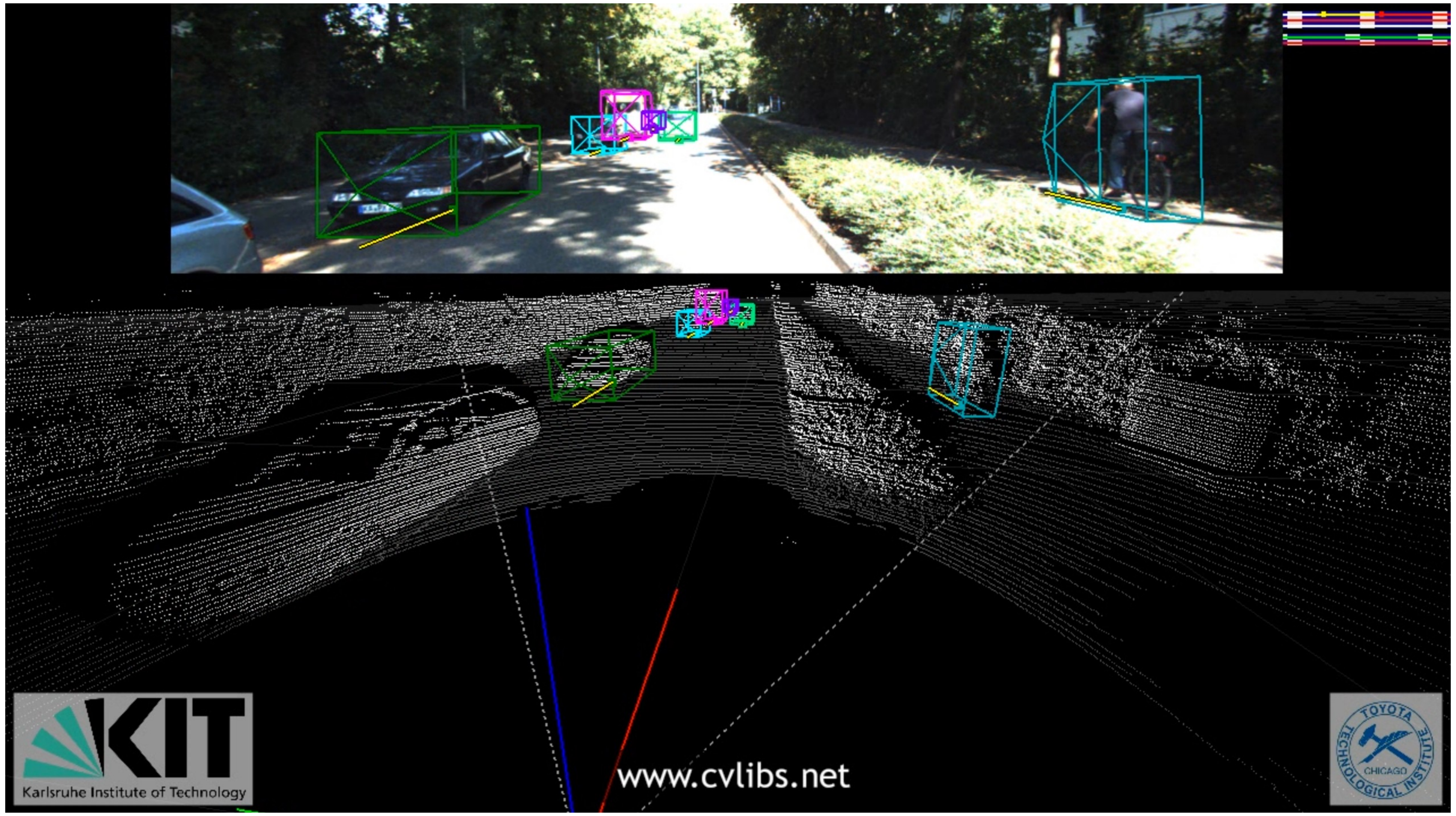
- Why two coordinate systems?



We want to be able to project the 3D points in Velodyne's coordinate system onto an image captured by a camera.

Camera Parameters

- Why two coordinate systems?


Karlsruhe Institute of Technology
www.cvlibs.net
TOYOTA TECHNOLOGICAL INSTITUTE
CHICAGO

We want to be able to project the 3D points in Velodyne's coordinate system onto an image captured by a camera.

Projection

To project a point (X, Y, Z) in world coordinates on the image plane, we need to:

[Source: N. Snavely]

Projection

To project a point (X, Y, Z) in world coordinates on the image plane, we need to:

- Transform (X, Y, Z) into camera coordinates. We thus need:

[Source: N. Snavely]

Projection

To project a point (X, Y, Z) in world coordinates on the image plane, we need to:

- Transform (X, Y, Z) into camera coordinates. We thus need:
  - Camera position (in world coordinates)

[Source: N. Snavely]

Projection

To project a point (X, Y, Z) in world coordinates on the image plane, we need to:

- Transform (X, Y, Z) into camera coordinates. We thus need:
  - Camera position (in world coordinates)
  - Camera orientation (in world coordinates)

[Source: N. Snavely]

Projection

To project a point (X, Y, Z) in world coordinates on the image plane, we need to:

- Transform (X, Y, Z) into camera coordinates. We thus need:
  - Camera position (in world coordinates)
  - Camera orientation (in world coordinates)
- To project into the image plane
  - Need to know camera intrinsics

[Source: N. Snavely]

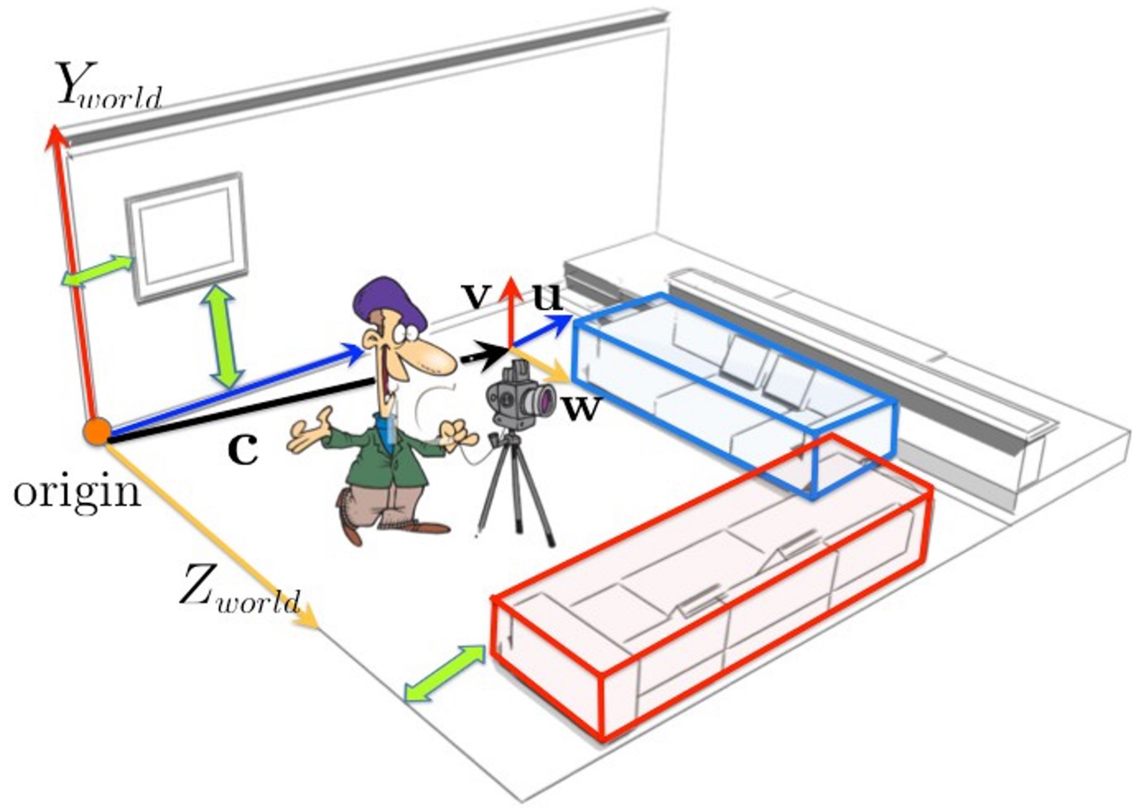
Projection

To project a point (X, Y, Z) in world coordinates on the image plane, we need to:

- Transform (X, Y, Z) into camera coordinates. We thus need:
  - Camera position (in world coordinates)
  - Camera orientation (in world coordinates)
- To project into the image plane
  - Need to know camera intrinsics
- These can all be described with matrices!

[Source: N. Snavely]

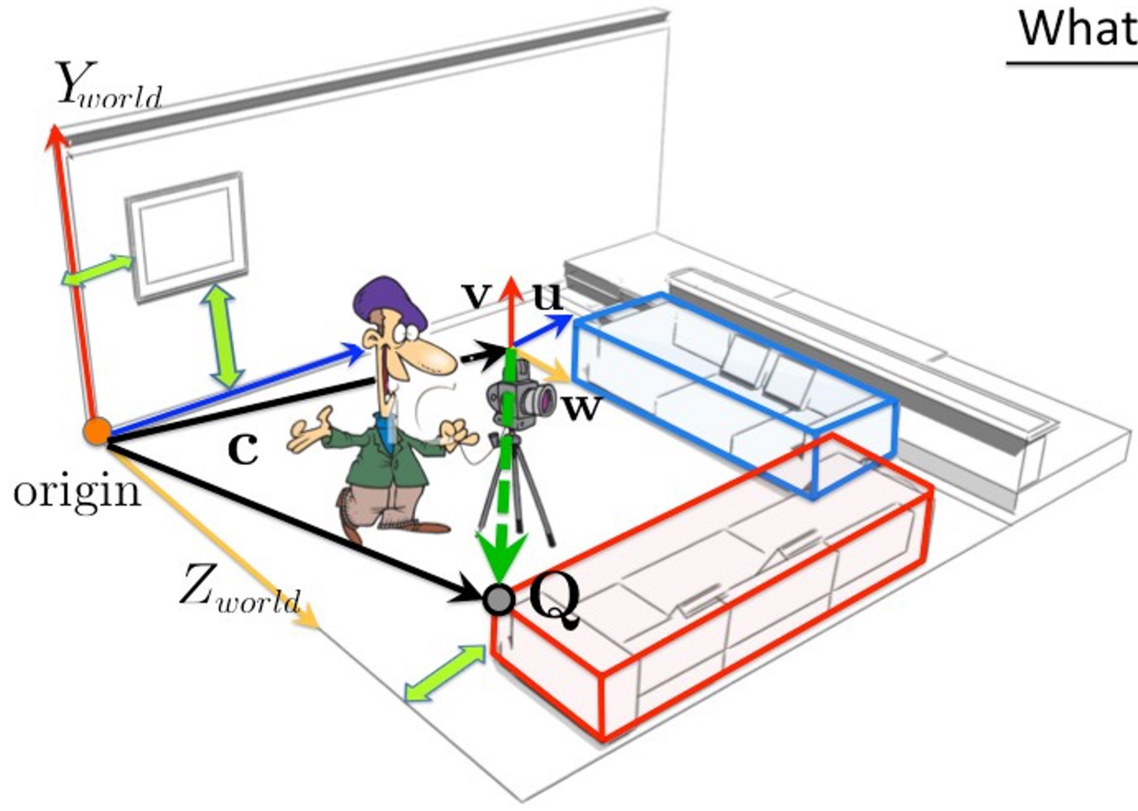
Camera Extrinsics



$\mathbf{c}$	...	camera position in room coordinate system
$\mathbf{u}, \mathbf{v}, \mathbf{w}$	...	3 orthogonal directions of camera in room coordinate system

We first need our camera position and orientation in the room's world.

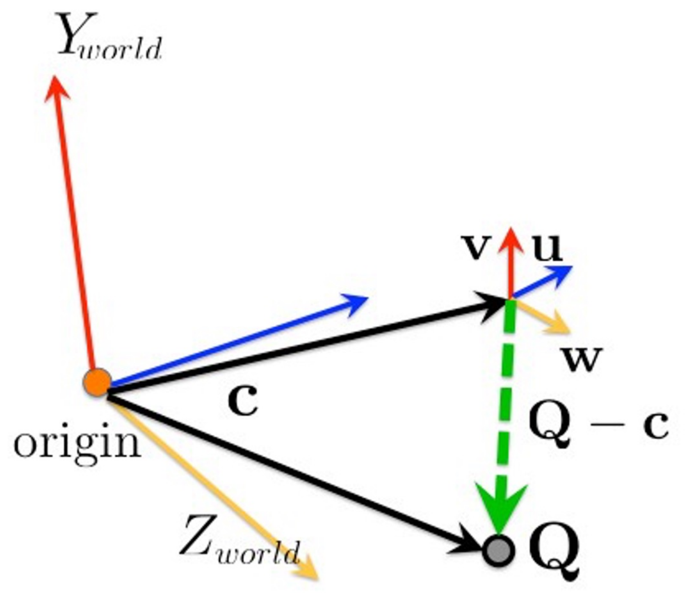
Camera Extrinsics



What is Q in camera's coordinate system??

$\mathbf{c}$	...	camera position in room coordinate system
$\mathbf{u}, \mathbf{v}, \mathbf{w}$	...	3 orthogonal directions of camera in room coordinate system

Camera Extrinsics

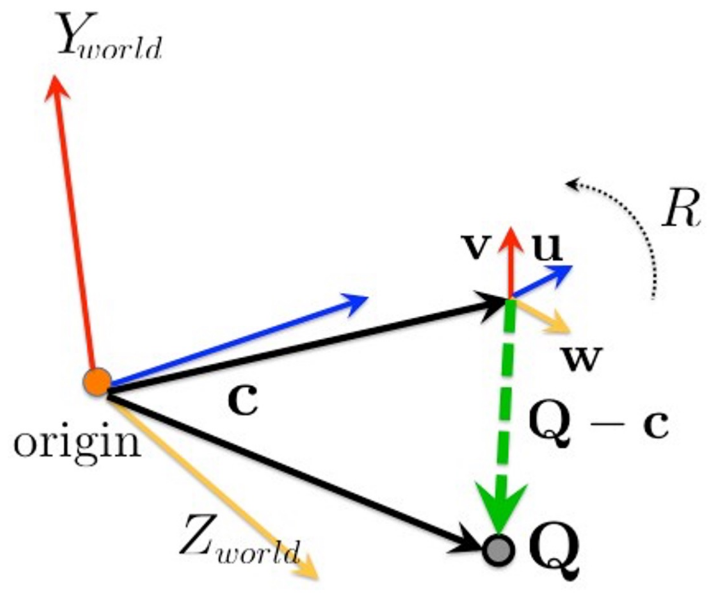


$\mathbf{c}$	…	camera position in room coordinate system
$\mathbf{u}, \mathbf{v}, \mathbf{w}$	…	3 orthogonal directions of camera in room coordinate system

What is Q in camera's coordinate system??

$\mathbf{Q} - \mathbf{c}$ … makes **position** relative to camera

Camera Extrinsics



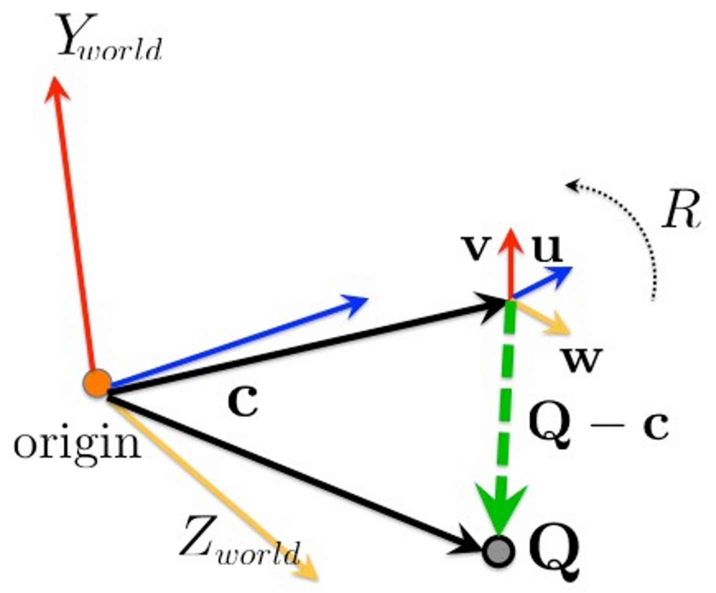
What is Q in camera's coordinate system??

$\mathbf{Q} - \mathbf{c}$ ... makes **position** relative to camera

$R\begin{bmatrix}\mathbf{u} & \mathbf{v} & \mathbf{w}\end{bmatrix} = I$ (looking for rotation to canonical orientation)

$\mathbf{c}$	...	camera position in room coordinate system
$\mathbf{u}, \mathbf{v}, \mathbf{w}$	...	3 orthogonal directions of camera in room coordinate system

Camera Extrinsics



$\mathbf{c}$	...	camera position in room coordinate system
$\mathbf{u}, \mathbf{v}, \mathbf{w}$	...	3 orthogonal directions of camera in room coordinate system

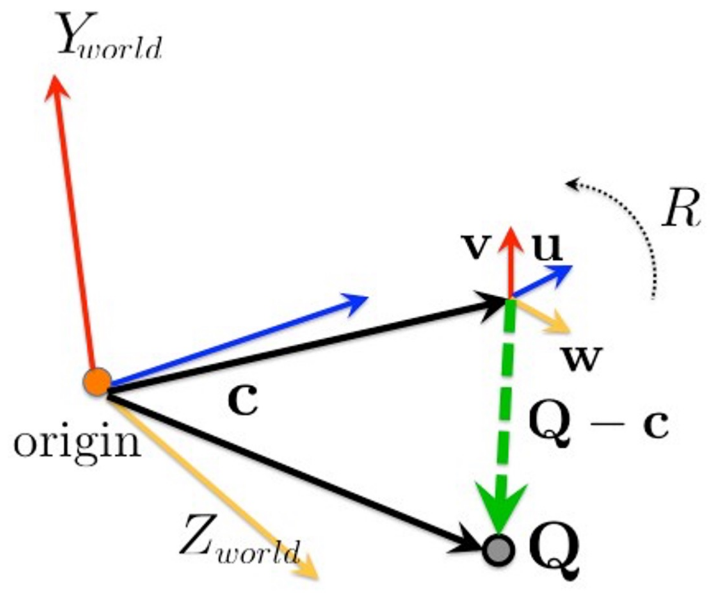
What is Q in camera's coordinate system??

$\mathbf{Q} - \mathbf{c}$ ... makes **position** relative to camera

$R \begin{bmatrix} \mathbf{u} & \mathbf{v} & \mathbf{w} \end{bmatrix} = I$ (looking for rotation to canonical orientation)

$R \cdot R^T = I$ (since orientation is orthogonal matrix)

Camera Extrinsics



What is Q in camera's coordinate system??

$\mathbf{Q} - \mathbf{c}$ ... makes **position** relative to camera

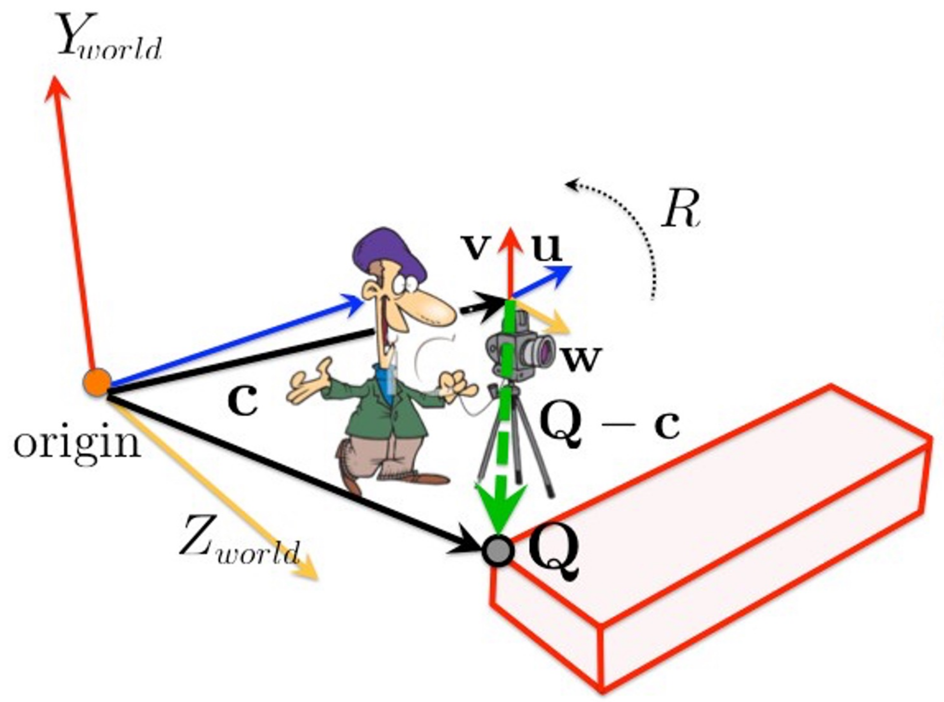
$R \begin{bmatrix} \mathbf{u} & \mathbf{v} & \mathbf{w} \end{bmatrix} = I$ (looking for rotation to canonical orientation)

$R \cdot R^T = I$ (since orientation is orthogonal matrix)

$R = \begin{bmatrix} \mathbf{u}^T & \mathbf{v}^T & \mathbf{w}^T \end{bmatrix}$

$\mathbf{c}$	...	camera position in room coordinate system
$\mathbf{u}, \mathbf{v}, \mathbf{w}$	...	3 orthogonal directions of camera in room coordinate system

Camera Extrinsics



$\mathbf{c}$ ... camera position in room coordinate system

$\mathbf{u}, \mathbf{v}, \mathbf{w}$ ... 3 orthogonal directions of camera in room coordinate system

What is Q in camera's coordinate system??

$\mathbf{Q} - \mathbf{c}$ ... makes **position** relative to camera

$R\,[\mathbf{u} \quad \mathbf{v} \quad \mathbf{w}] = I$ (looking for rotation to canonical orientation)

$R \cdot R^T = I$ (since orientation is orthogonal matrix)

$R = [\mathbf{u}^T \quad \mathbf{v}^T \quad \mathbf{w}^T]$

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = R\left(\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} - \mathbf{c} \right) = [R \quad -Rc] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

camera coordinates

room (world) coordinates

Projection Equations

- Projection matrix P models the cumulative effect of all intrinsic and extrinsic parameters. We use homogeneous coordinates for 2D and 3D:

$$\mathbf{q} = \begin{bmatrix} ax \\ ay \\ a \end{bmatrix} = \mathbf{P} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Projection Equations

- Projection matrix P models the cumulative effect of all intrinsic and extrinsic parameters. We use homogeneous coordinates for 2D and 3D:

$$\mathbf{q} = \begin{bmatrix} ax \\ ay \\ a \end{bmatrix} = \mathbf{P} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

- It can be computed as

$$\mathbf{P} = \underbrace{\begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix}}_{\text{intrinsics } K} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{projection}} \underbrace{\begin{bmatrix} \mathbf{R}_{3\times3} & 0_{3\times1} \\ 0_{1\times3} & 1 \end{bmatrix}}_{\text{rotation}} \underbrace{\begin{bmatrix} \mathbf{I}_{3\times3} & \mathbf{T}_{3\times1} \\ 0_{1\times3} & 1 \end{bmatrix}}_{\text{translation}}$$

Projection Equations

- Projection matrix P models the cumulative effect of all intrinsic and extrinsic parameters. We use homogeneous coordinates for 2D and 3D:

$$\mathbf{q} = \begin{bmatrix} ax \\ ay \\ a \end{bmatrix} = \mathbf{P} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

- It can be computed as

$$\mathbf{P} = \underbrace{\begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix}}_{\text{intrinsics } K} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{projection}} \underbrace{\begin{bmatrix} \mathbf{R}_{3\times3} & 0_{3\times1} \\ 0_{1\times3} & 1 \end{bmatrix}}_{\text{rotation}} \underbrace{\begin{bmatrix} \mathbf{I}_{3\times3} & \mathbf{T}_{3\times1} \\ 0_{1\times3} & 1 \end{bmatrix}}_{\text{translation}}$$

- To get a point **q** in the image plane, I need to compute **P** $(X, Y, Z, 1)^T$, where P is a 3×4 matrix. This gives me a 3×1 vector. Now I divide all coordinates with the third coordinate (making the third coordinate equal to 1), and then drop the last coordinate.

The Projection Matrix

- The projection matrix is defined as

$$\mathsf{P} = \underbrace{\mathsf{K}}_{\text{intrinsics}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{projection}} \underbrace{\underbrace{\begin{bmatrix} \mathsf{R}_{3\times3} & 0_{3\times1} \\ 0_{1\times3} & 1 \end{bmatrix}}_{\text{rotation}} \underbrace{\begin{bmatrix} \mathsf{I}_{3\times3} & \mathsf{T}_{3x1} \\ 0_{1\times3} & 1 \end{bmatrix}}_{\text{translation}}}_{\begin{bmatrix} \mathsf{R} & \mathsf{t} \end{bmatrix}}$$

The Projection Matrix

- The projection matrix is defined as

$$\mathsf{P} = \underbrace{\mathsf{K}}_{\text{intrinsics}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{projection}} \underbrace{\underbrace{\begin{bmatrix} \mathsf{R}_{3\times 3} & 0_{3\times 1} \\ 0_{1\times 3} & 1 \end{bmatrix}}_{\text{rotation}} \underbrace{\begin{bmatrix} \mathsf{I}_{3\times 3} & \mathsf{T}_{3\times 3} \\ 0_{1\times 3} & 1 \end{bmatrix}}_{\text{translation}}}_{\begin{bmatrix} \mathsf{R} & \mathsf{t} \end{bmatrix}}$$

- More compactly

$$\mathsf{P} = \mathsf{K} \begin{bmatrix} \mathsf{R} & \mathsf{t} \end{bmatrix}$$

- Sometimes you will see notation:

$$\mathsf{P} = \mathsf{K} \left[\mathsf{R} \mid \mathsf{t} \right]$$

- It's the same thing.

The Projection Matrix

- The projection matrix is defined as

$$\mathbf{P} = \underbrace{\mathbf{K}}_{\text{intrinsics}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{projection}} \underbrace{\underbrace{\begin{bmatrix} \mathbf{R}_{3\times 3} & 0_{3\times 1} \\ 0_{1\times 3} & 1 \end{bmatrix}}_{\text{rotation}} \underbrace{\begin{bmatrix} \mathbf{I}_{3\times 3} & \mathbf{T}_{3\times 3} \\ 0_{1\times 3} & 1 \end{bmatrix}}_{\text{translation}}}_{\begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix}}$$

- More compactly

$$\mathbf{P} = \mathbf{K}\begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix}$$

- Sometimes you will see notation:

$$\mathbf{P} = \mathbf{K}\left[\mathbf{R} \mid \mathbf{t}\right]$$

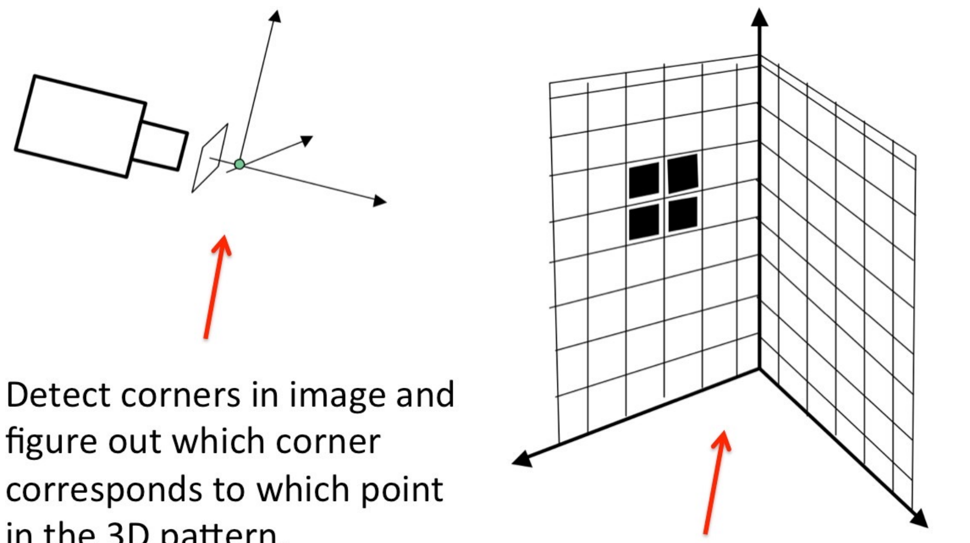
- It’s the same thing.

- This might look complicated. Truth is, in most cases you don’t have P at all, so you can’t really compute any projections. When you have a calibrated camera, then someone typically gives you P. And then projection is easy.

A Short Note on Camera Calibration

The general procedure:

- Place a 3D pattern (for which you know all distances) in front of camera.



Detect corners in image and figure out which corner corresponds to which point in the 3D pattern.

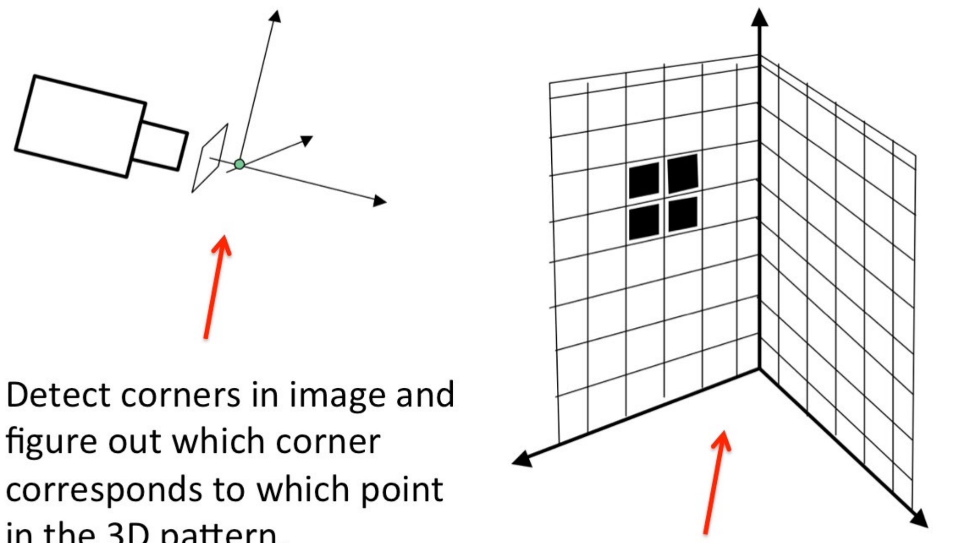
You measured all distances for this pattern.

[Pic from: R. Duraiswami]

A Short Note on Camera Calibration

The general procedure:

- Place a 3D pattern (for which you know all distances) in front of camera.
- Take a picture. Detect corners in image and find correspondences with the points in the pattern.



Detect corners in image and figure out which corner corresponds to which point in the 3D pattern.

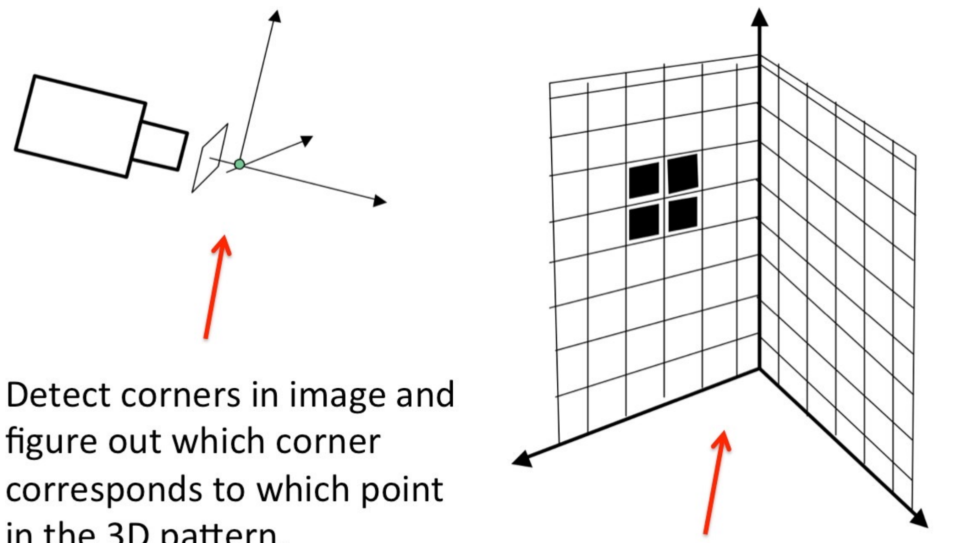
You measured all distances for this pattern.

[Pic from: R. Duraiswami]

A Short Note on Camera Calibration

The general procedure:

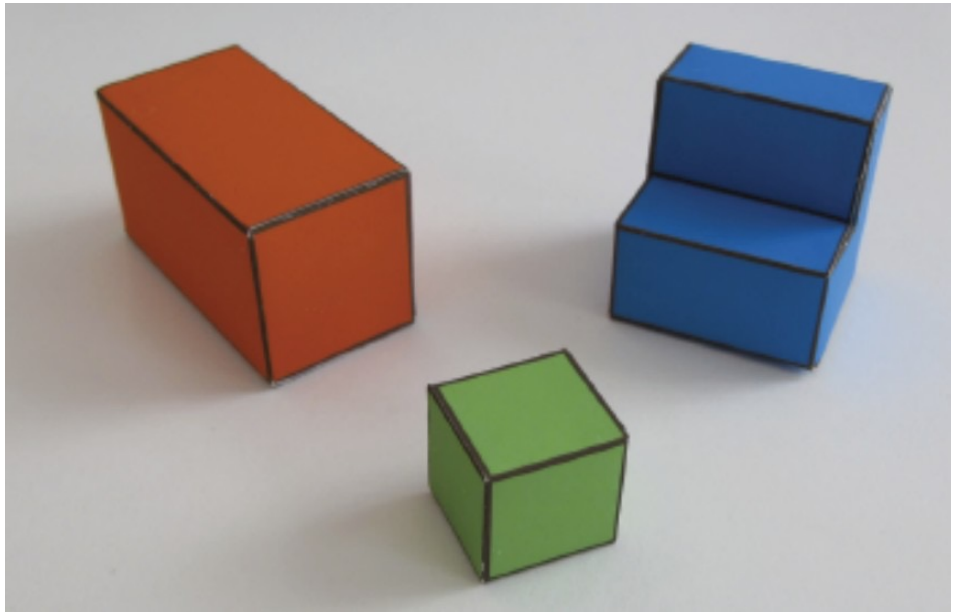
- Place a 3D pattern (for which you know all distances) in front of camera.
- Take a picture. Detect corners in image and find correspondences with the points in the pattern.
- Go to the internet and check out the math that tells you how to compute K from these 2D-3D correspondences. We won't cover in class.


Detect corners in image and figure out which corner corresponds to which point in the 3D pattern.
You measured all distances for this pattern.

[Pic from: R. Duraiswami]

Camera Calibration: Interesting Fact

- Let's say you have an image but you don't know anything about the camera (for example, image downloaded from the web).
- For images where you see lines corresponding to 3 orthogonal directions, like cubes or rooms, you can compute the camera matrix K as well as R and t!



- How to do this is explained in the Zisserman & Hartley book (section 8).

Camera Calibration: Interesting Fact

- As a consequence, for scenes with lots of lines (e.g. man-made scenes) one can reconstruct the scene in 3D from a single image!

Camera Calibration: Interesting Fact

- As a consequence, for scenes with lots of lines (e.g. man-made scenes) one can reconstruct the scene in 3D from a single image!

Camera Calibration: Interesting Fact

- As a consequence, for scenes with lots of lines (e.g. man-made scenes) one can reconstruct the scene in 3D from a single image!
- For those interested, check out the math here:

A. Criminisi, I. Reid, and A. Zisserman

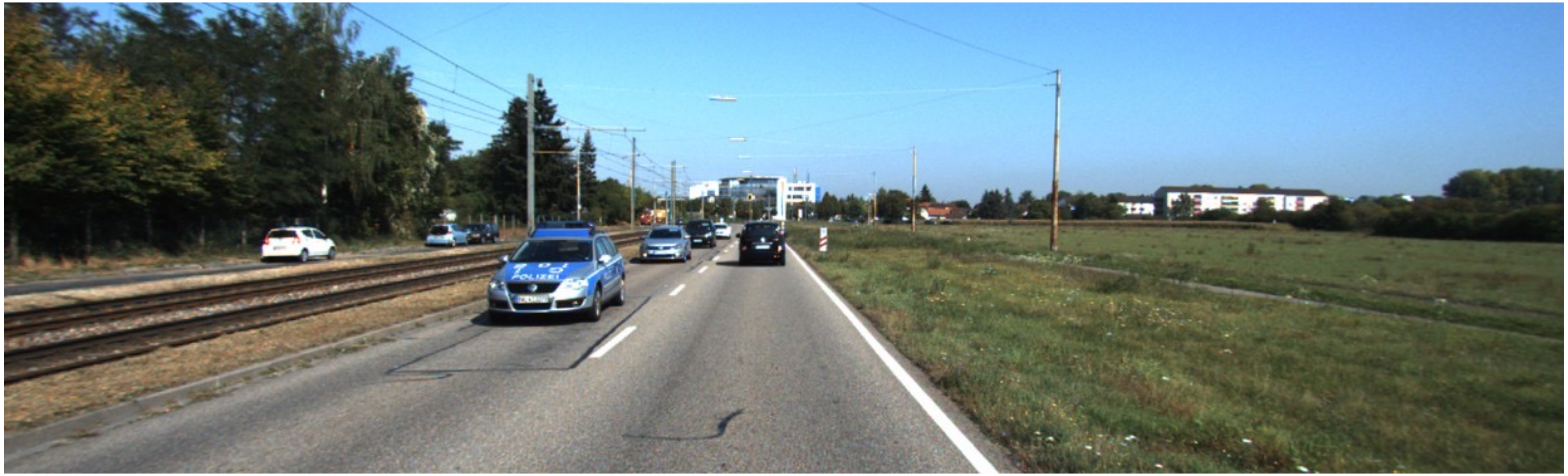
Single View Metrology

International Journal of Computer Vision, vol 40, num 2, 2000

https://www.cs.cmu.edu/~ph/869/papers/Criminisi99.pdf

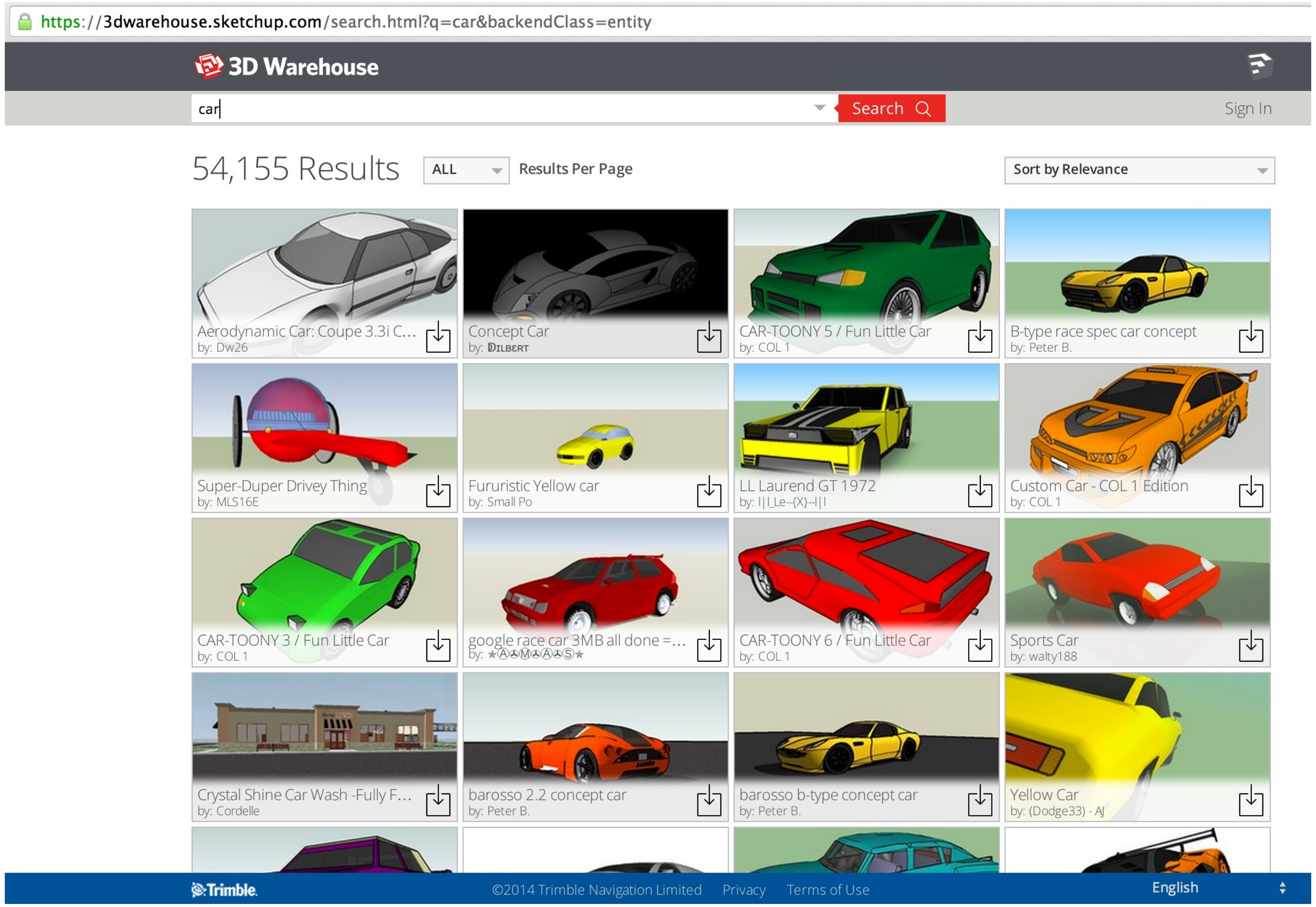
Exercise (Not Very Easy, But Fun)

- We want to render (project) a 3D CAD model of a car to this image in a realistic way.
- How?



Exercise

- First get a CAD model. There are tones of them, e.g. 3D Warehouse (free)



Exercise

- We downloaded this model. Now what?

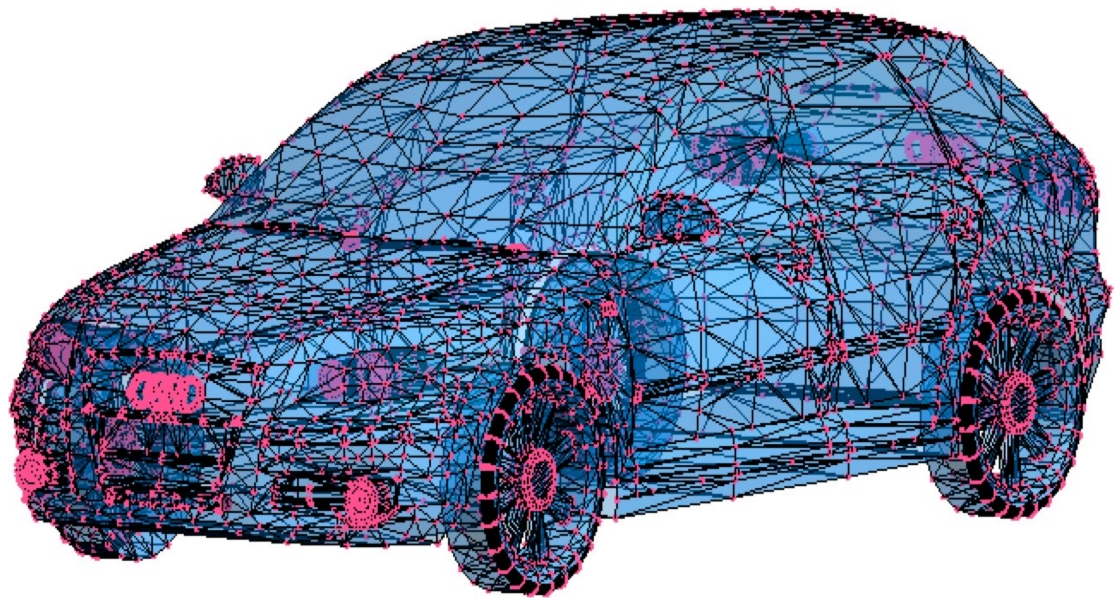
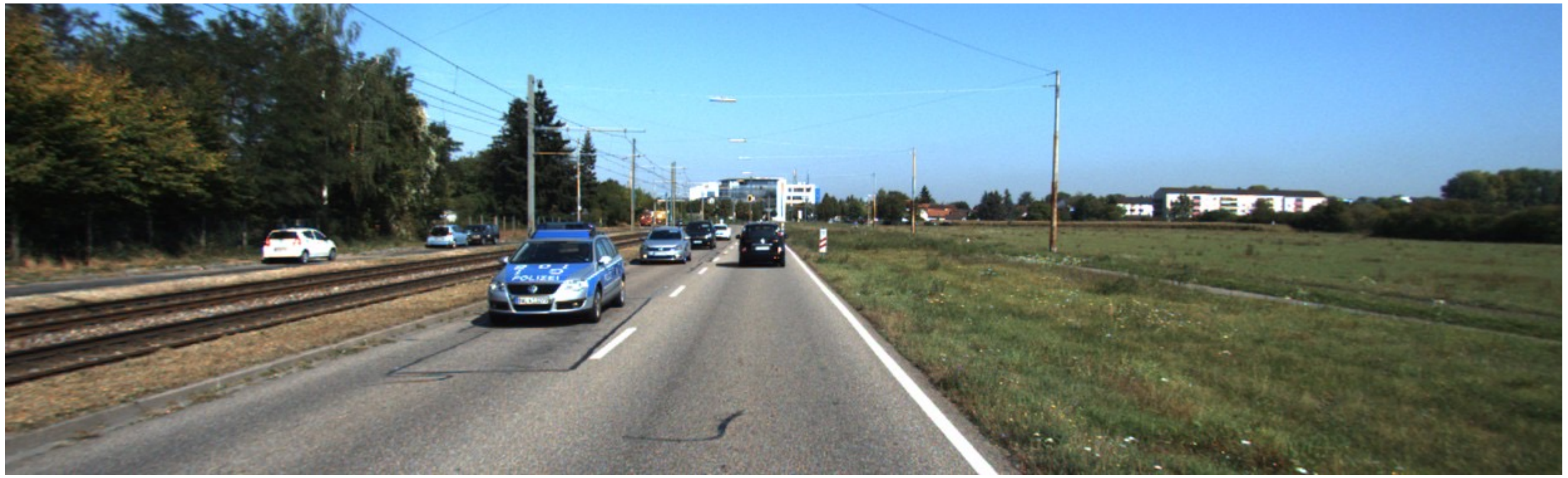


Figure: A CAD model is a collection of 3D vertices and faces that connect the vertices. Each face represents a small triangle. It typically has color.

Exercise

- Our image was collected with a car on the road:
  - A camera was on top of the car, approximately 1.7m above ground
  - Image plane is orthogonal to the ground
  - We have the internal parameters of the camera, K.



Exercise

- Our image was collected with a car on the road:
  - A camera was on top of the car, approximately 1.7m above ground
  - Image plane is orthogonal to the ground
  - We have the internal parameters of the camera, K.

Exercise

- Our image was collected with a car on the road:
    - A camera was on top of the car, approximately 1.7m above ground
    - Image plane is orthogonal to the ground
    - We have the internal parameters of the camera, K.
- With a little bit of math, we can compute the ground plane in 3D, relative to camera.
- With a bit more math we can compute which point on the ground plane projects to an image point (x, y).

Exercise

- Our image was collected with a car on the road:
  - A camera was on top of the car, approximately 1.7m above ground
  - Image plane is orthogonal to the ground
  - We have the internal parameters of the camera, K.
- With a little bit of math, we can compute the ground plane in 3D, relative to camera.
- With a bit more math we can compute which point on the ground plane projects to an image point (x, y).

How?

- We can now "place" our CAD model to this point (compute R and t)
- Rendering:
- Compute $[ax, ay, a]^T = K [R \mid t] [X, Y, Z, 1]^T$ for each CAD vertex $[X, Y, Z]^T$. Divide $[ax, ay, a]^T$ with third coordinate and drop it.

Exercise

- That’s it. Make a video for more fun

Exercise

- That's it. Make a video for more fun

Next time: Stereo