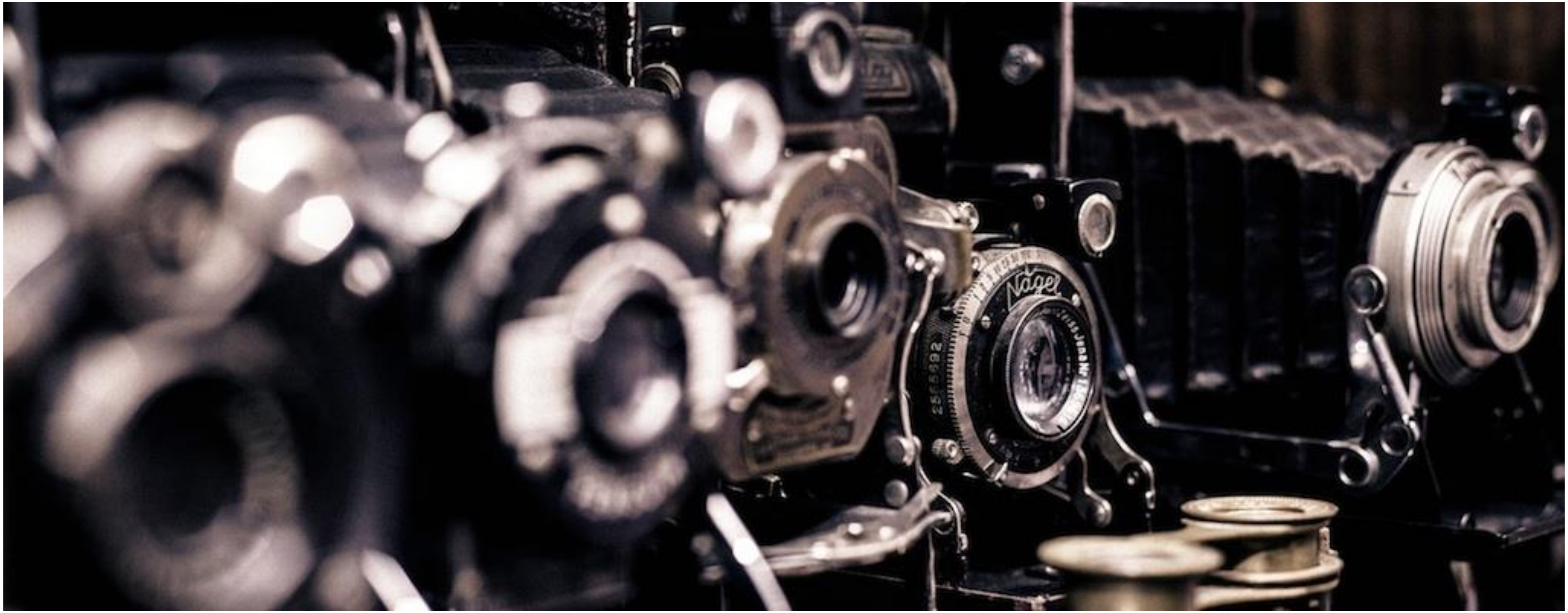
# Geometric camera models



ICS 483 Computer Vision
Fall 2025, Lecture 13

# Course announcements

# Recap

- 2D transformations.
- Projective geometry 101.
- **Transformations in projective geometry.**
- What transformation is this?

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

translation

$$\begin{bmatrix} \boldsymbol{x}' \\ \boldsymbol{y}' \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{s}_{\boldsymbol{x}} & 0 & 0 \\ 0 & \boldsymbol{s}_{\boldsymbol{y}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ 1 \end{bmatrix}$$

scaling

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

rotation

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & \beta_x & 0 \\ \beta_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

shearing

# Recap

- 2D transformations.
- Projective geometry 101.
- **Transformations in projective geometry.**
- What transformation is this?

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

affine

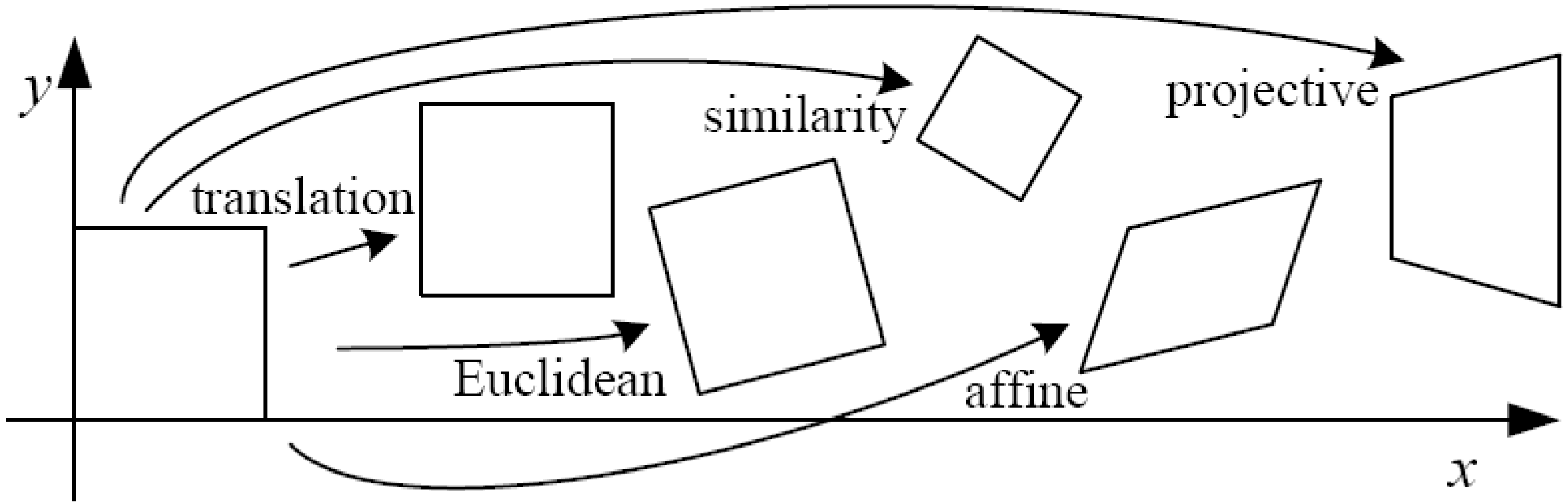
last coordinate w does not change

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

projective

last coordinate w changes?

# Classification of 2D transformations




| Name | Matrix | # D.O.F. |
|---|---|---|
| translation | $\left[\, \boldsymbol{I} \mid \boldsymbol{t} \,\right]_{2\times 3}$ | 2 |
| rigid (Euclidean) | $\left[\, \boldsymbol{R} \mid \boldsymbol{t} \,\right]_{2\times 3}$ | 3 |
| similarity | $\left[\, s\boldsymbol{R} \mid \boldsymbol{t} \,\right]_{2\times 3}$ | 4 |
| affine | $\left[\, \boldsymbol{A} \,\right]_{2\times 3}$ | 6 |
| projective | $\left[\, \tilde{\boldsymbol{H}} \,\right]_{3\times 3}$ | 8 |

# Reminder: Determining affine transformations

Affine transformation:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 \\ p_4 & p_5 & p_6 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Vectorize transformation parameters:

Stack equations from point correspondences:

$$\underbrace{\begin{bmatrix} x' \\ y' \\ x' \\ y' \\ \vdots \\ x' \\ y' \end{bmatrix}}_{\boldsymbol{b}} = \underbrace{\begin{bmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \\ x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \\ & & & \vdots & & \\ x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{bmatrix}}_{\boldsymbol{x}}$$

Notation in system form:

$$\boxed{Ax = b}$$

# Reminder: Determining affine transformations

Convert the system to a linear least-squares problem:

$$E_{\mathrm{LLS}} = \|\mathbf{A}\boldsymbol{x} - \boldsymbol{b}\|^2$$

Expand the error:

$$E_{\mathrm{LLS}} = \boldsymbol{x}^\top(\mathbf{A}^\top\mathbf{A})\boldsymbol{x} - 2\boldsymbol{x}^\top(\mathbf{A}^\top\boldsymbol{b}) + \|\boldsymbol{b}\|^2$$

Minimize the error:

Set derivative to 0 $(\mathbf{A}^\top\mathbf{A})\boldsymbol{x} = \mathbf{A}^\top\boldsymbol{b}$

Solve for x $\boldsymbol{x} = (\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{A}^\top\boldsymbol{b}$ ⟵ Note: You almost <u>never</u> want to compute the inverse of a matrix.

In Python:

```
x = numpy.linalg.
    solve(A, b)
```

# Determining the homography matrix

Write out linear equation for each correspondence:

$$P' = H \cdot P \quad \text{or} \quad \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \alpha \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Expand matrix multiplication:

$$x' = \alpha(h_1 x + h_2 y + h_3)$$
$$y' = \alpha(h_4 x + h_5 y + h_6)$$
$$1 = \alpha(h_7 x + h_8 y + h_9)$$

Divide out unknown scale factor:

$$x'(h_7 x + h_8 y + h_9) = (h_1 x + h_2 y + h_3)$$
$$y'(h_7 x + h_8 y + h_9) = (h_4 x + h_5 y + h_6)$$

*How do you rearrange terms to make it a linear system?*

# Affine

$$x' = p_{11}x + p_{12}y + p_{13}$$

$$y' = p_{21}x + p_{22}y + p_{23}$$

# Projective

$$x'(h_7x + h_8y + h_9) = (h_1x + h_2y + h_3)$$
$$y'(h_7x + h_8y + h_9) = (h_4x + h_5y + h_6)$$

# Determining the homography matrix

Stack together constraints from multiple point correspondences:

$$\mathbf{A}\boldsymbol{h} = \mathbf{0}$$

$$\begin{bmatrix} -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \\ 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \\ -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \\ 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \\ & & & & \vdots & & & & \\ -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \\ 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \\ h_9 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

*Homogeneous* linear least squares problem

- Solve with SVD

Use *numpy.linalg.svd* in assignment. No need to solve by hand.

# Overview of today's lecture

- Some motivational imaging experiments.
- Pinhole camera.
- Accidental pinholes.
- Camera matrix.

# Slide credits

Most of these slides were adapted from:

- Matt O'Toole (CMU, 15-463, Fall 2022).
- Ioannis Gkioulekas (CMU, 15-463, Fall 2020).
- Kris Kitani (15-463, Fall 2016).

Some slides inspired from:

- Fredo Durand (MIT).

# Some motivational imaging experiments

# Let’s say we have a sensor…

digital sensor
(CCD or CMOS)

# … and an object we like to photograph

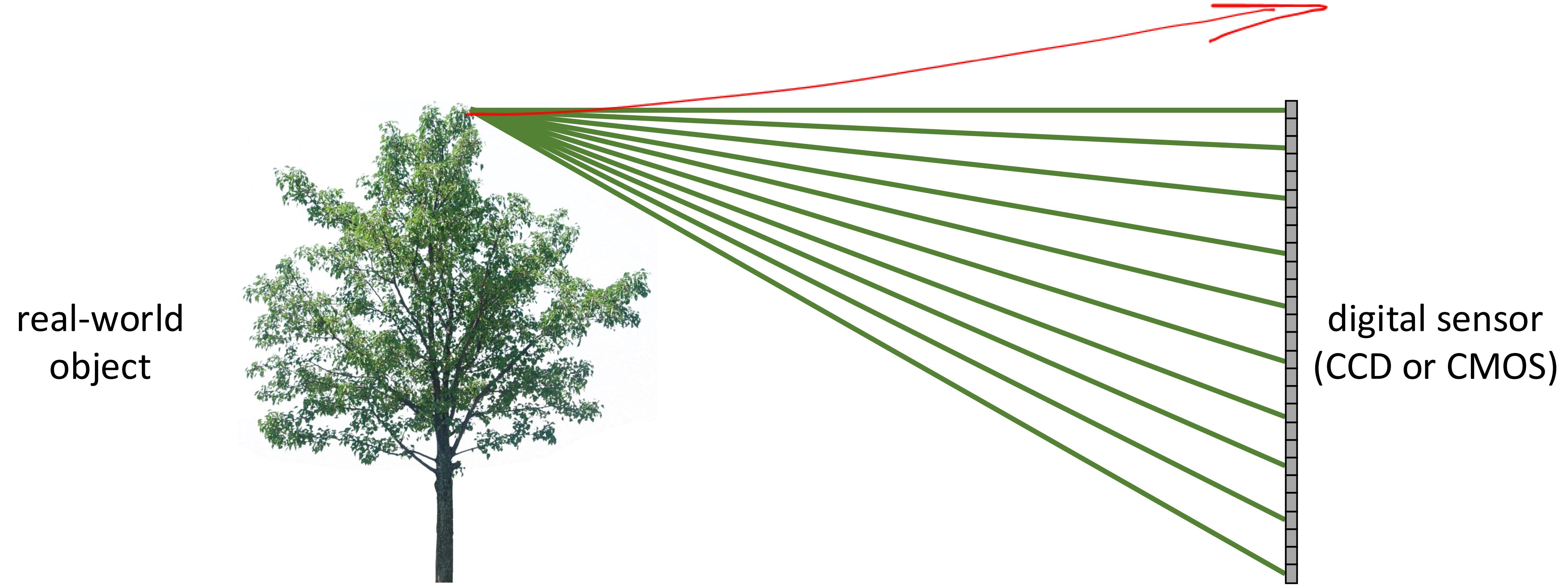
real-world
object

digital sensor
(CCD or CMOS)

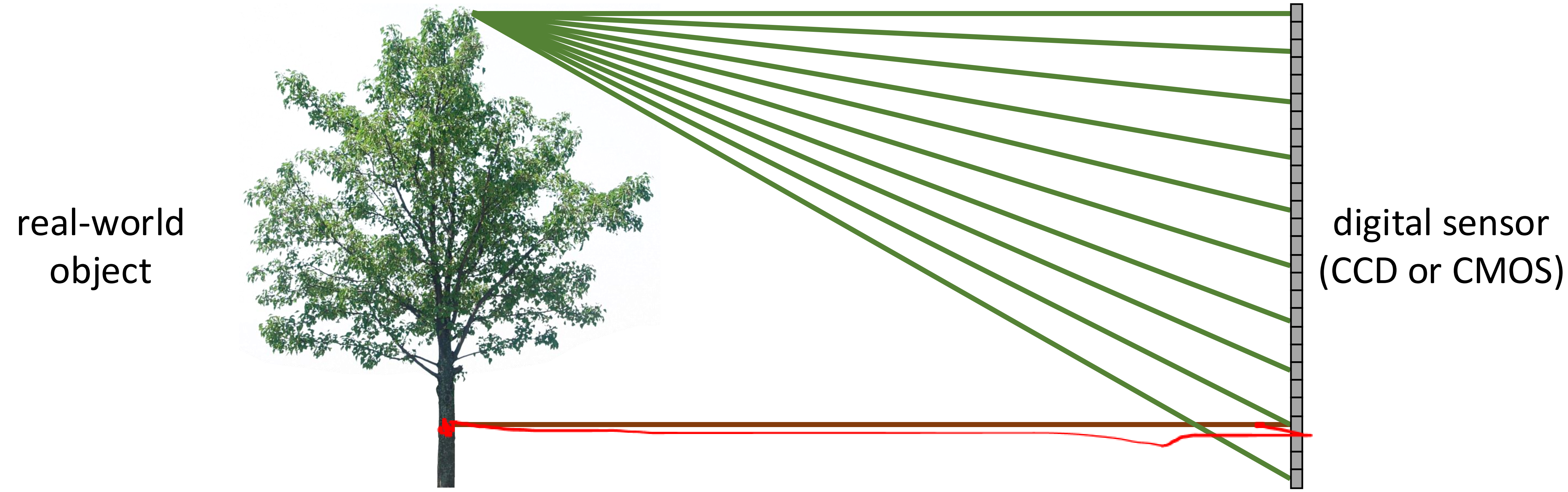
What would an image taken like this look like?

# Bare-sensor imaging

# Bare-sensor imaging

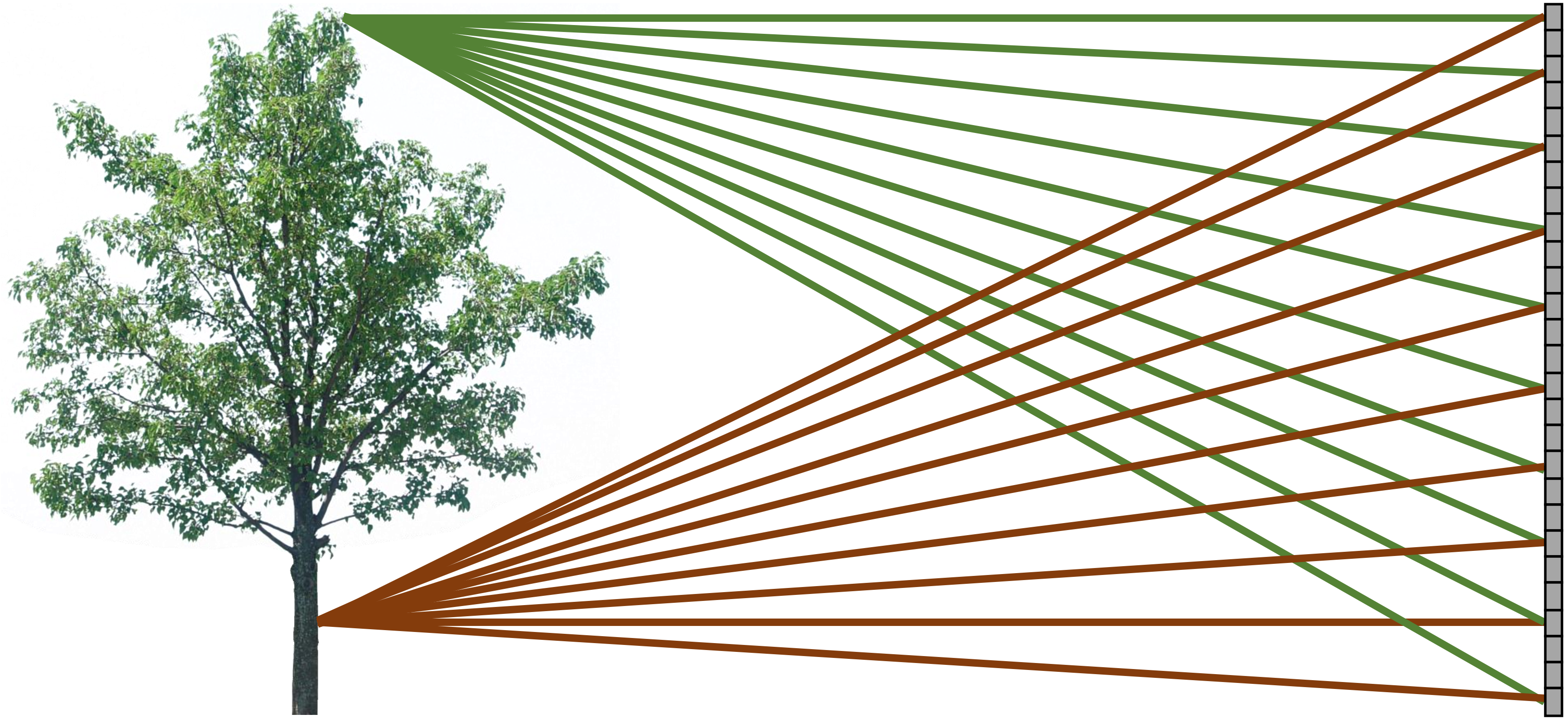


# Bare-sensor imaging



# Bare-sensor imaging

real-world
object



digital sensor
(CCD or CMOS)

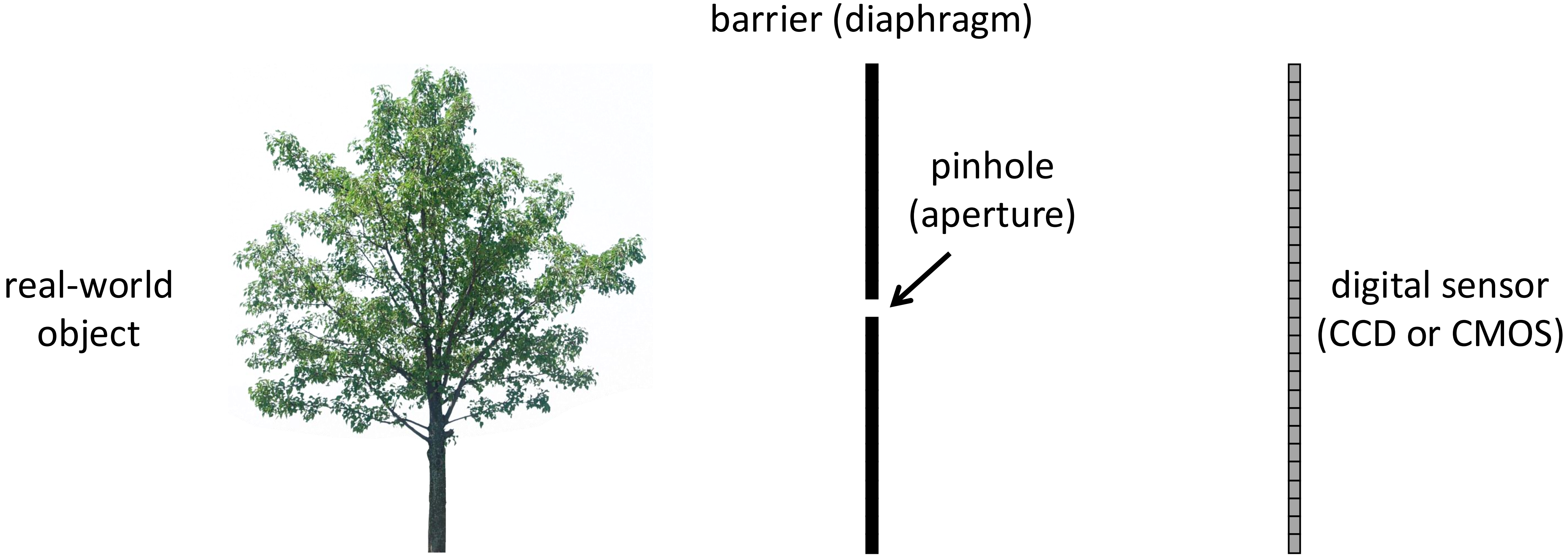
All scene points contribute to all sensor pixels

What does the
image on the
sensor look like?

# Bare-sensor imaging

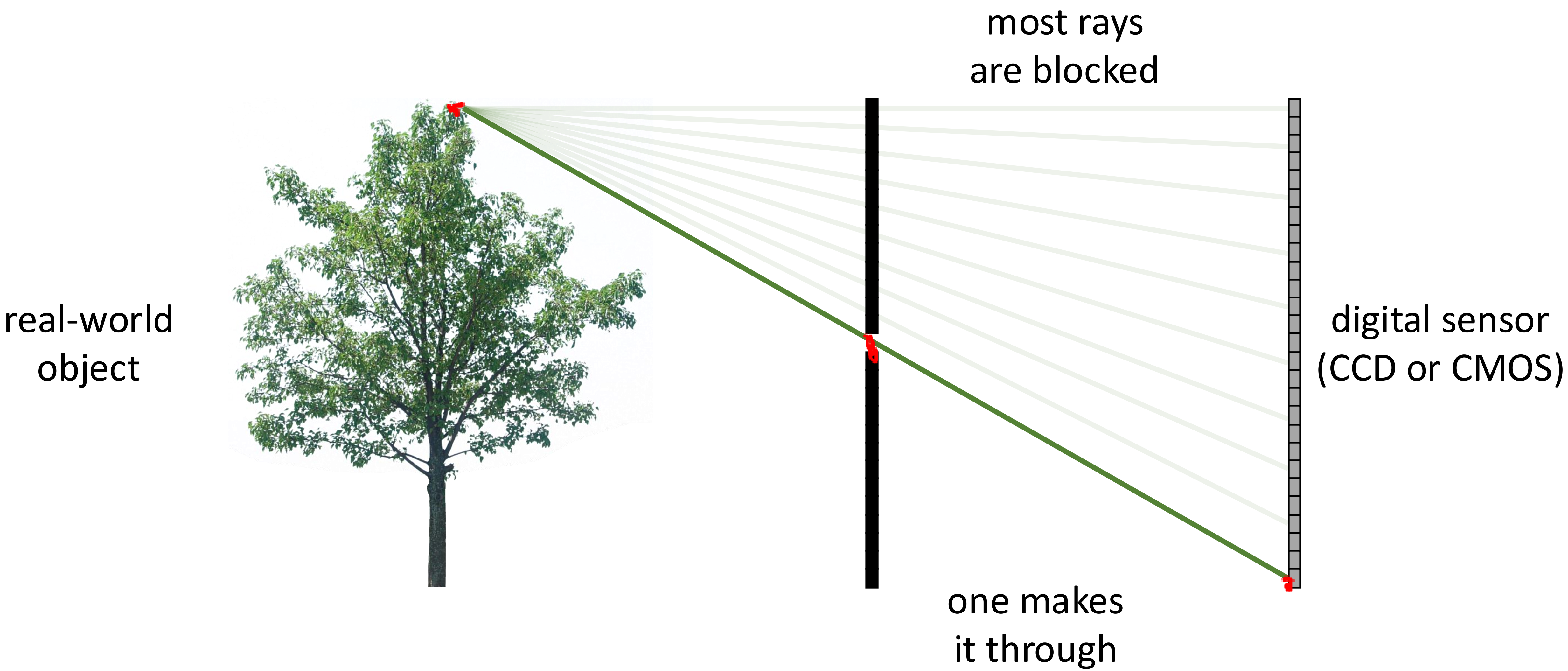
All scene points contribute to all sensor pixels

# Let's add something to this scene

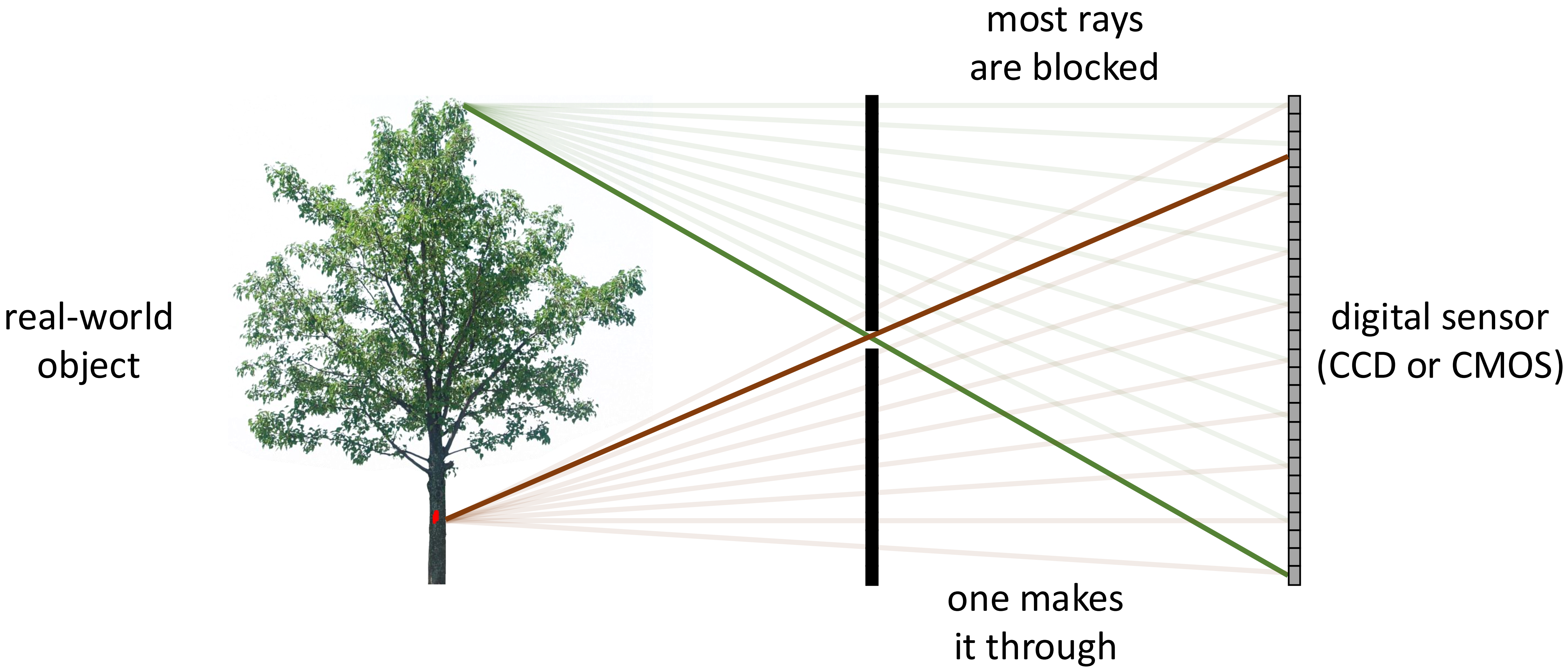



What would an image taken like this look like?

# Pinhole imaging

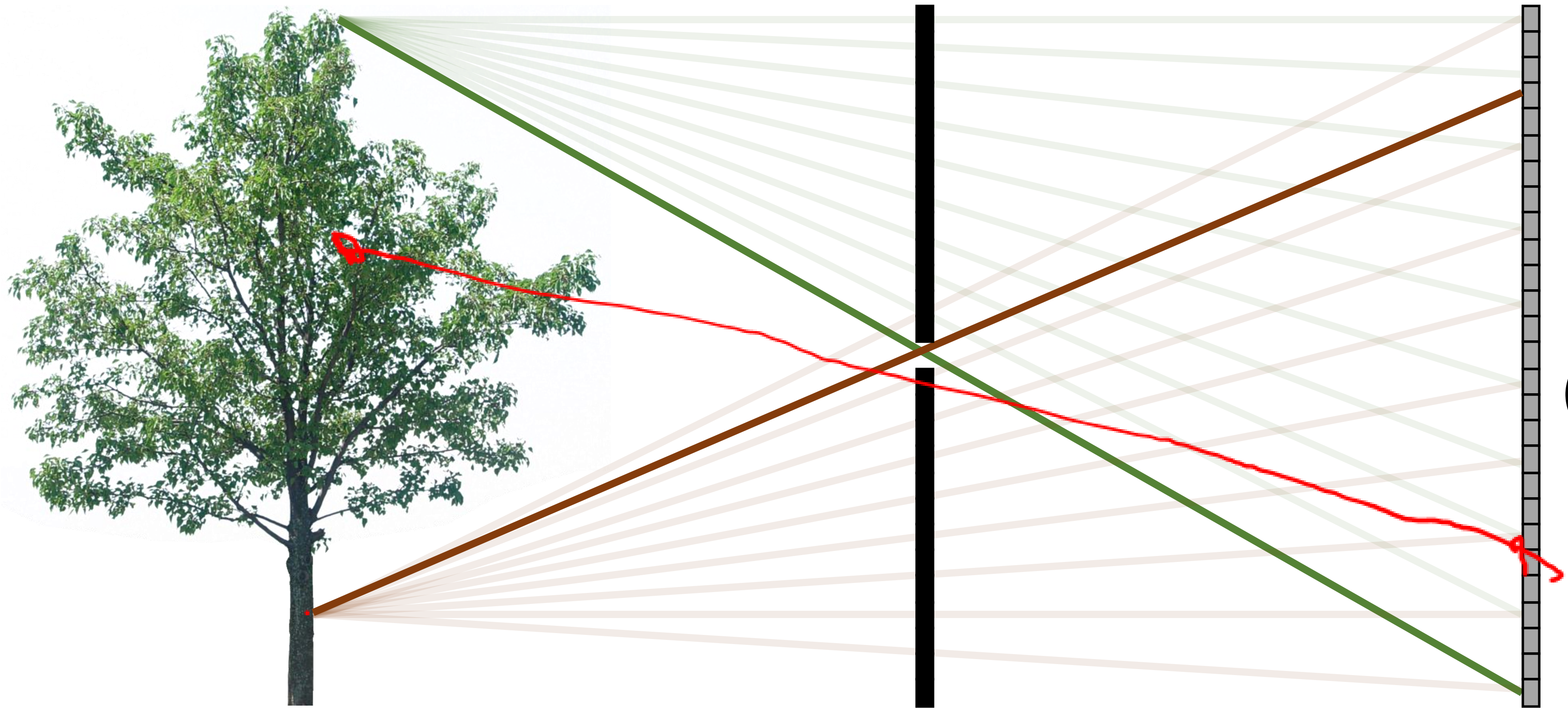


# Pinhole imaging



# Pinhole imaging

real-world object



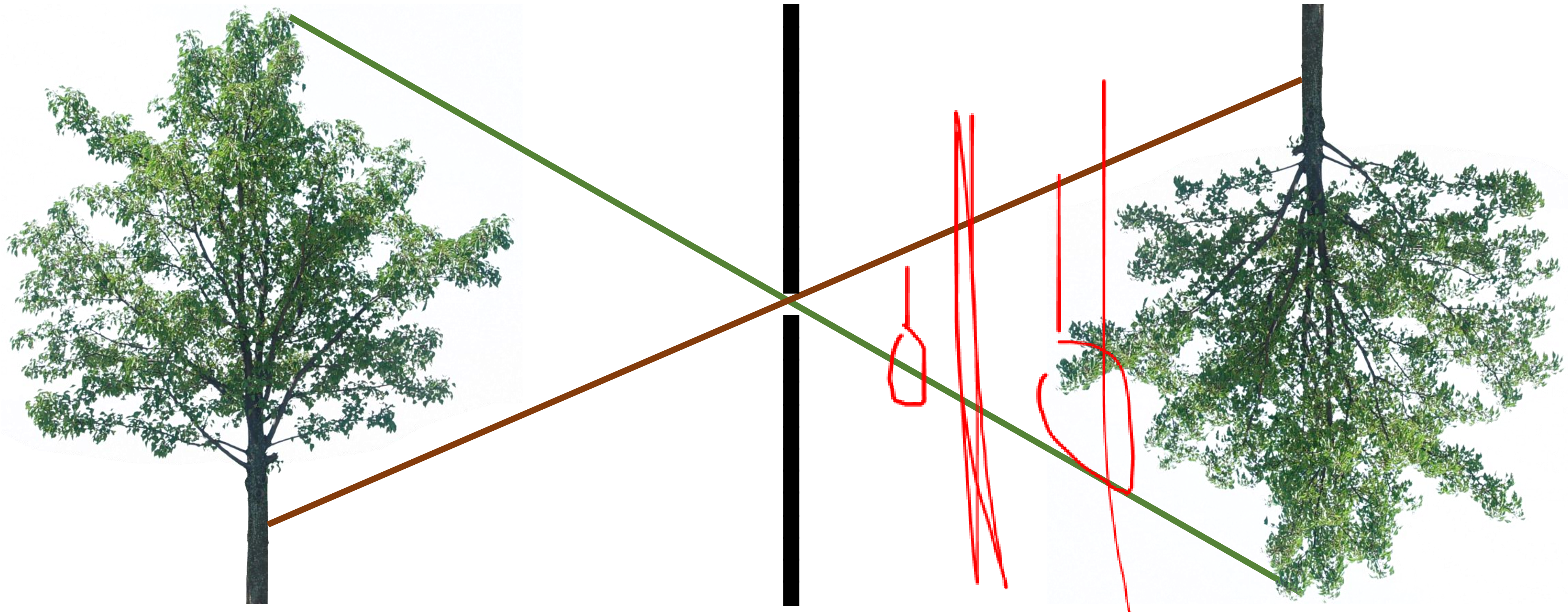
digital sensor (CCD or CMOS)

What does the image on the sensor look like?

Each scene point contributes to only one sensor pixel

# Pinhole imaging

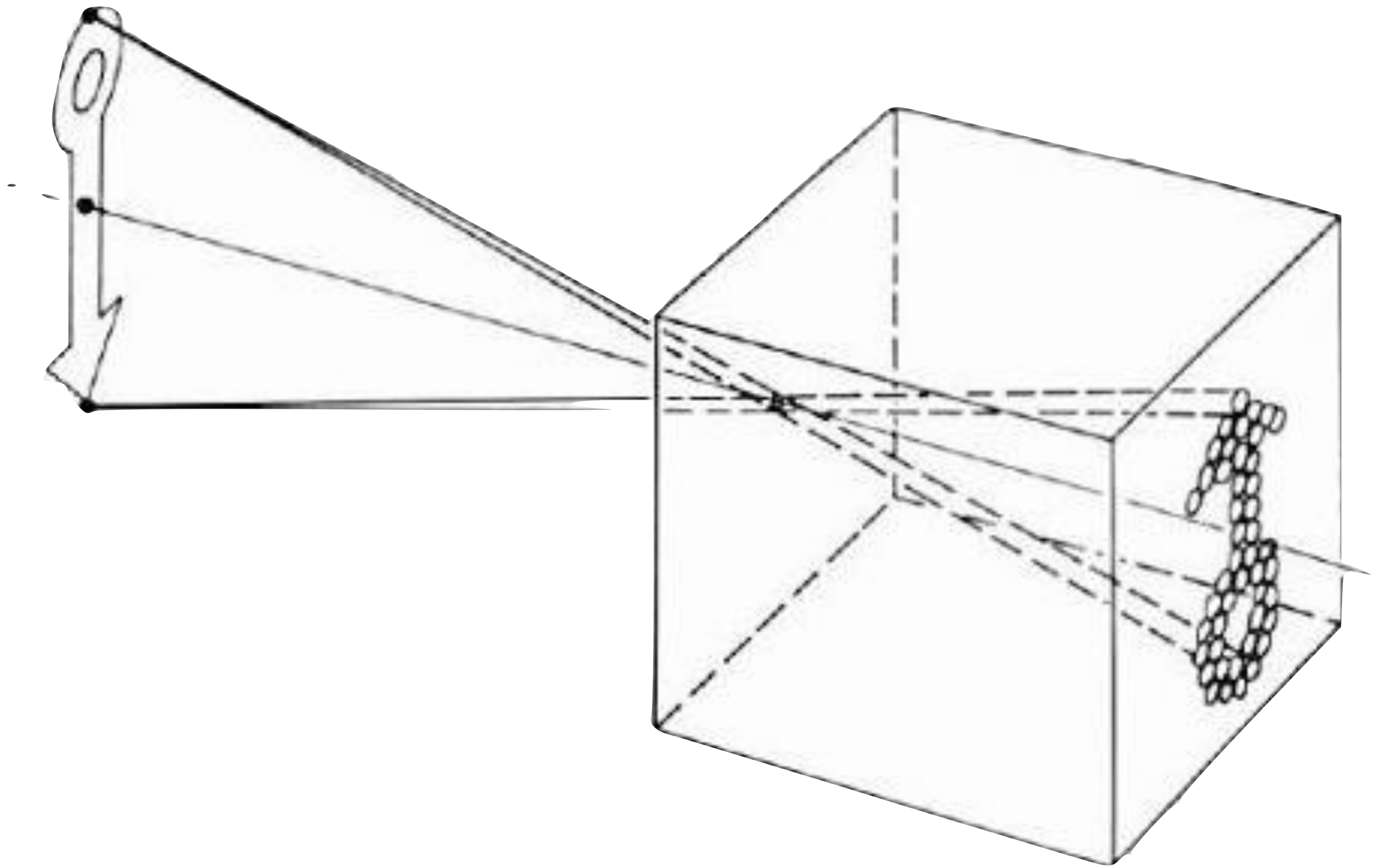
real-world object



copy of real-world object (inverted and scaled)

# Pinhole camera

# Pinhole camera a.k.a. camera obscura

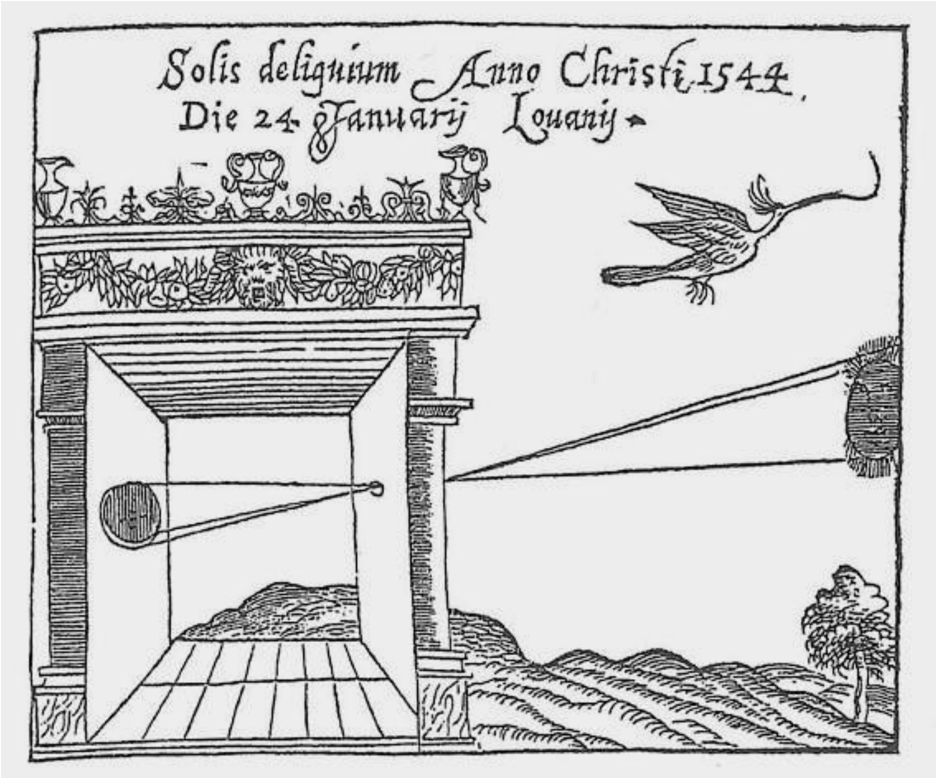


# Pinhole camera a.k.a. camera obscura

First mention …

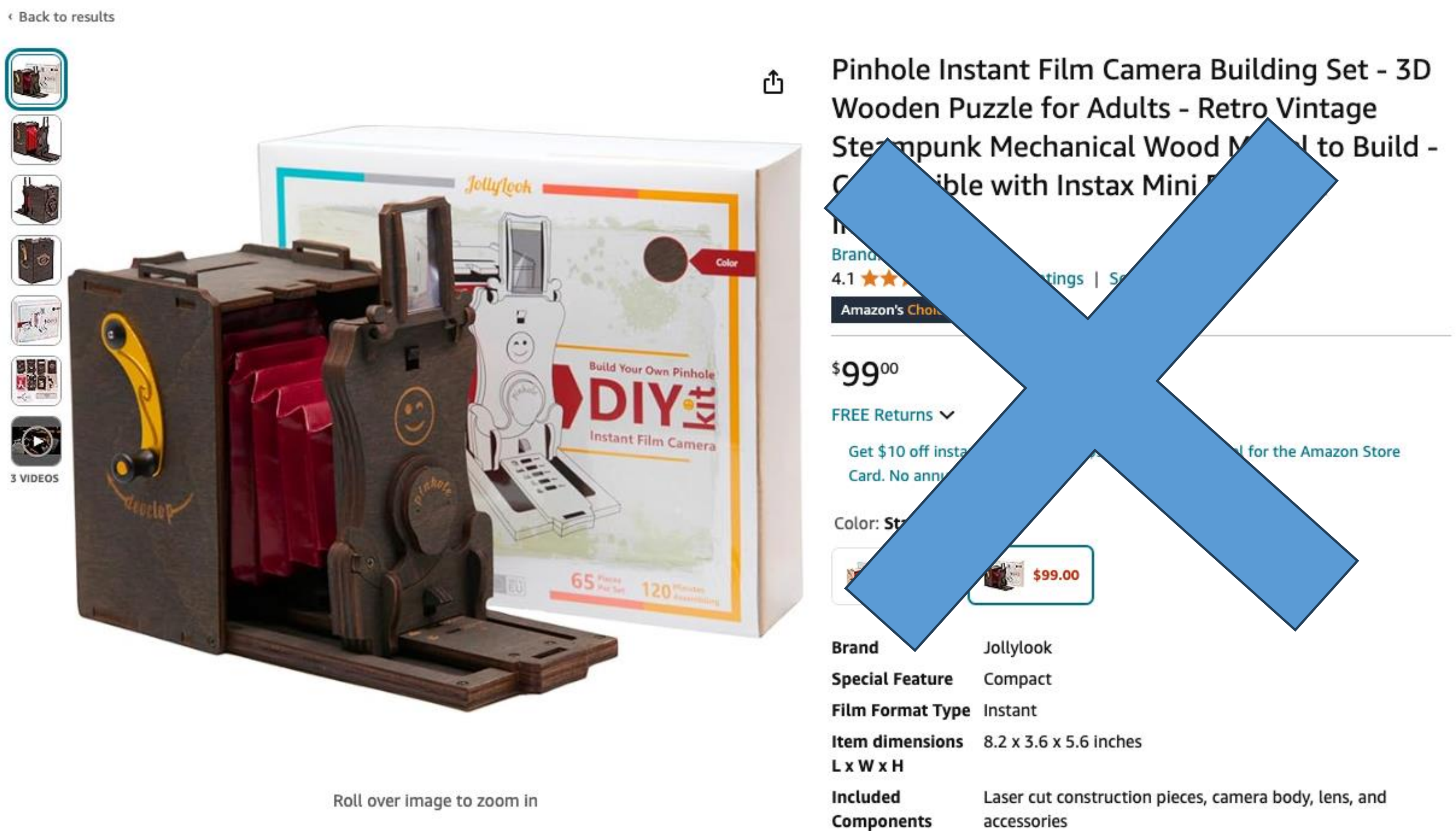
Chinese philosopher Mozi
(470 to 390 BC)

First camera …



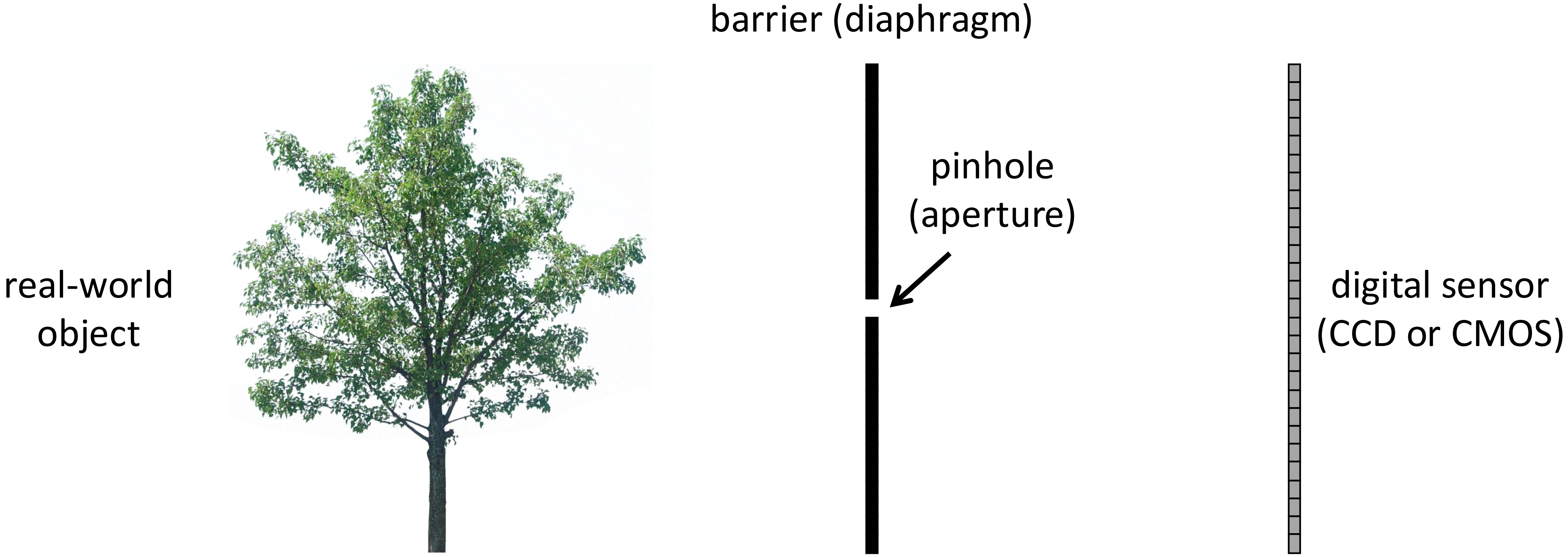

Greek philosopher Aristotle
(384 to 322 BC)

# Let’s try the DIY pinhole camera!

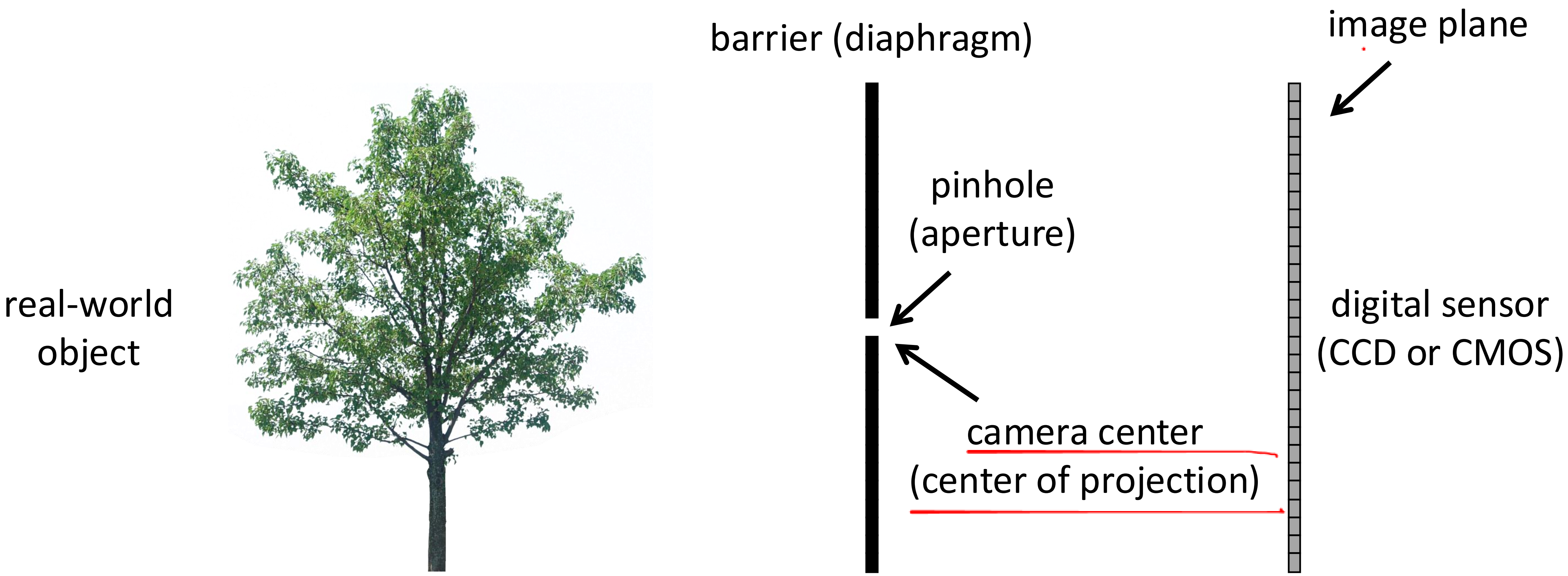


# Let’s try the DIY pinhole camera!


# Pinhole camera terms

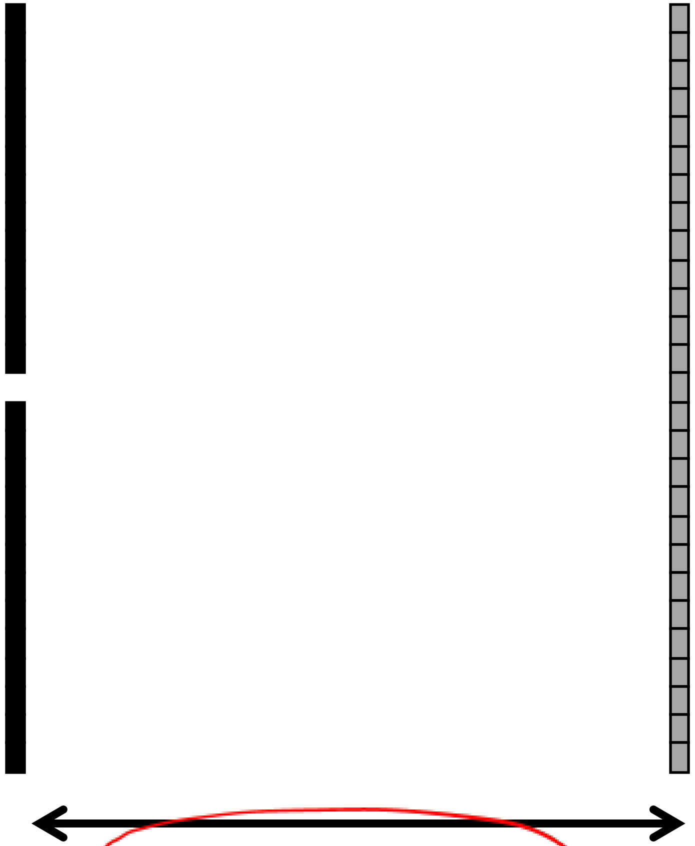


# Pinhole camera terms



# Focal length

real-world
object

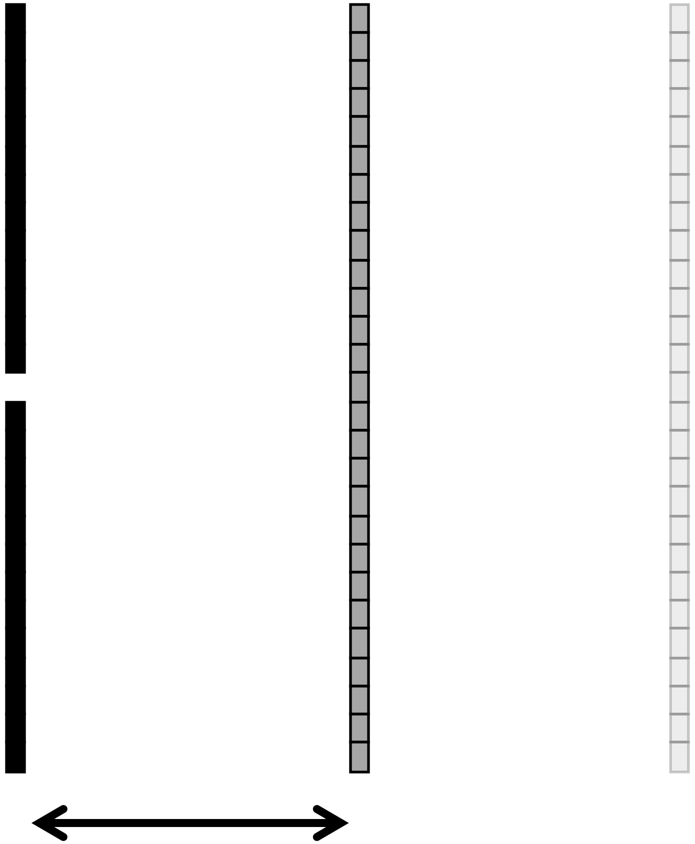


focal length f

# Focal length

What happens to the image as we change the focal length?

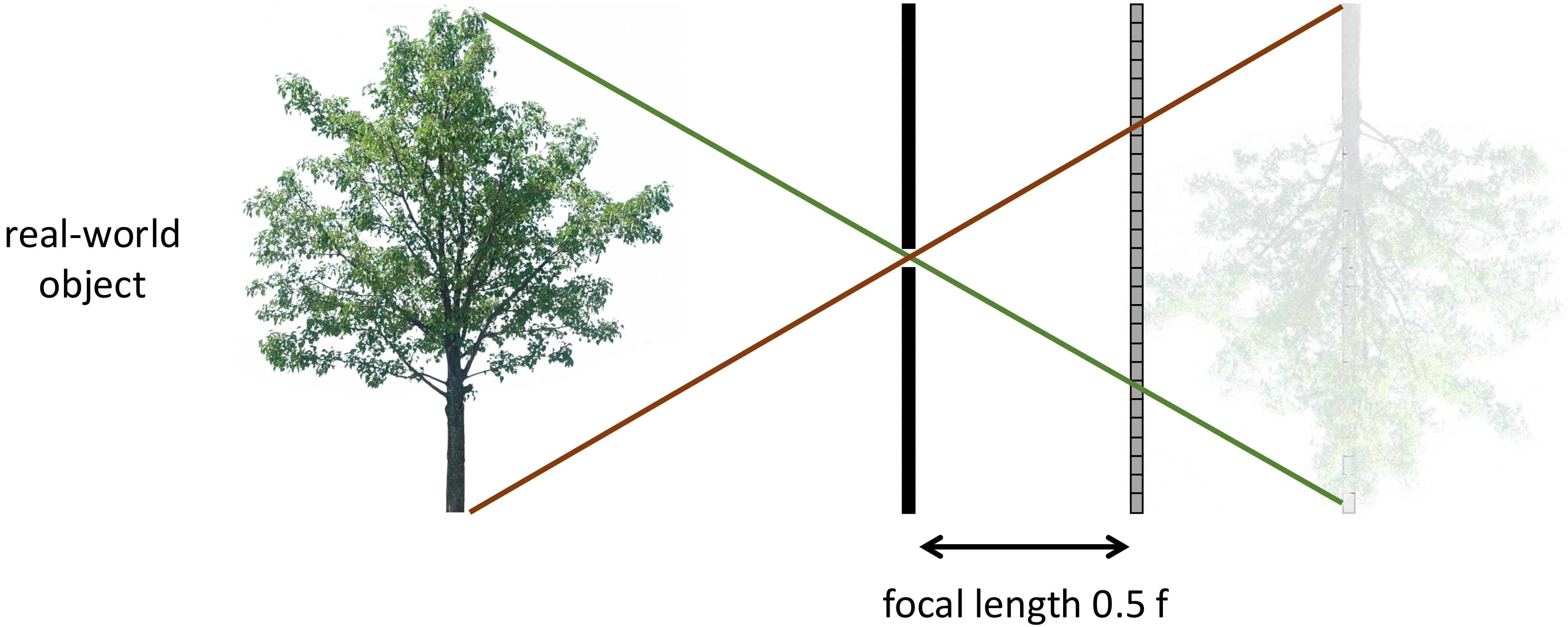
real-world object



focal length 0.5 f

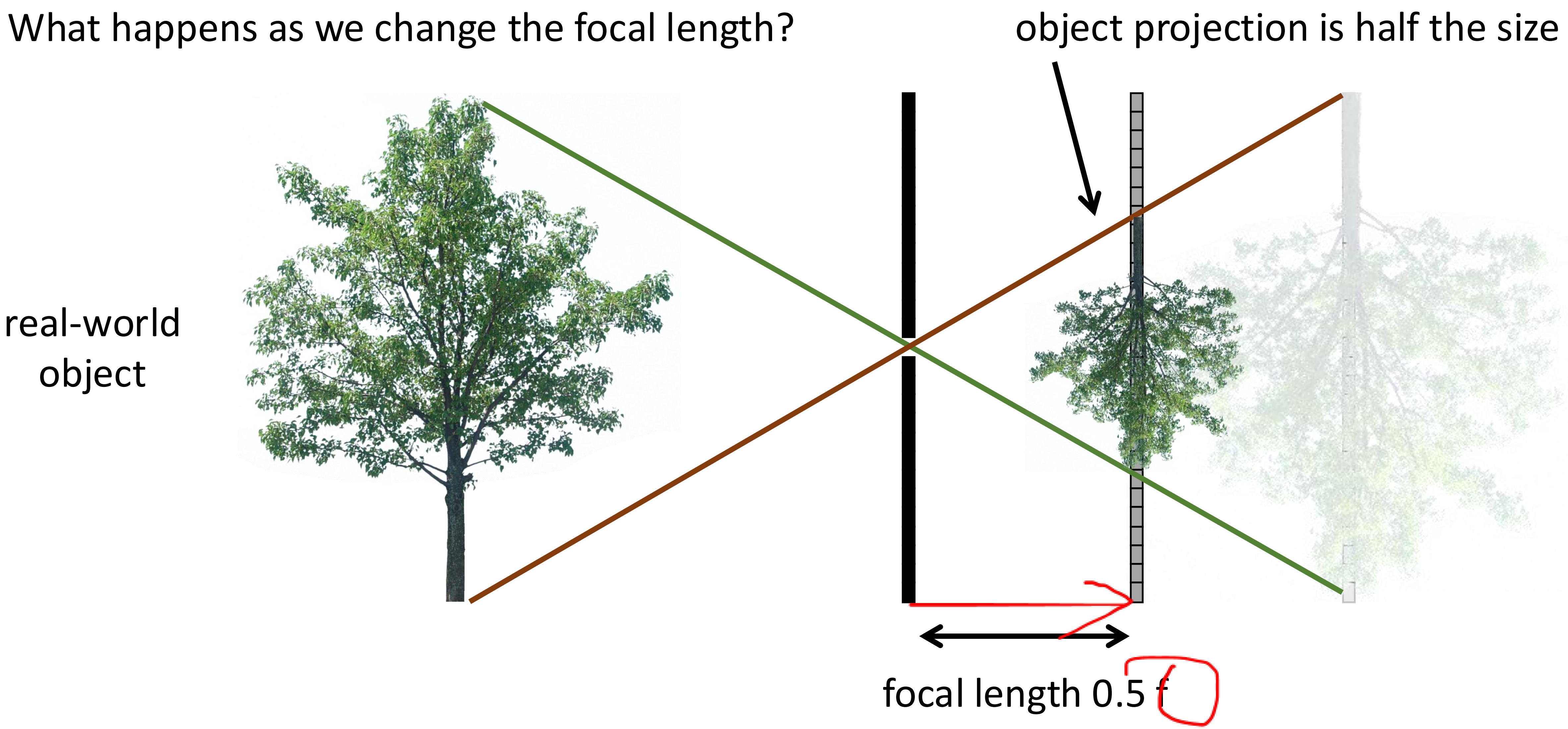
# Focal length

What happens to the image as we change the focal length?

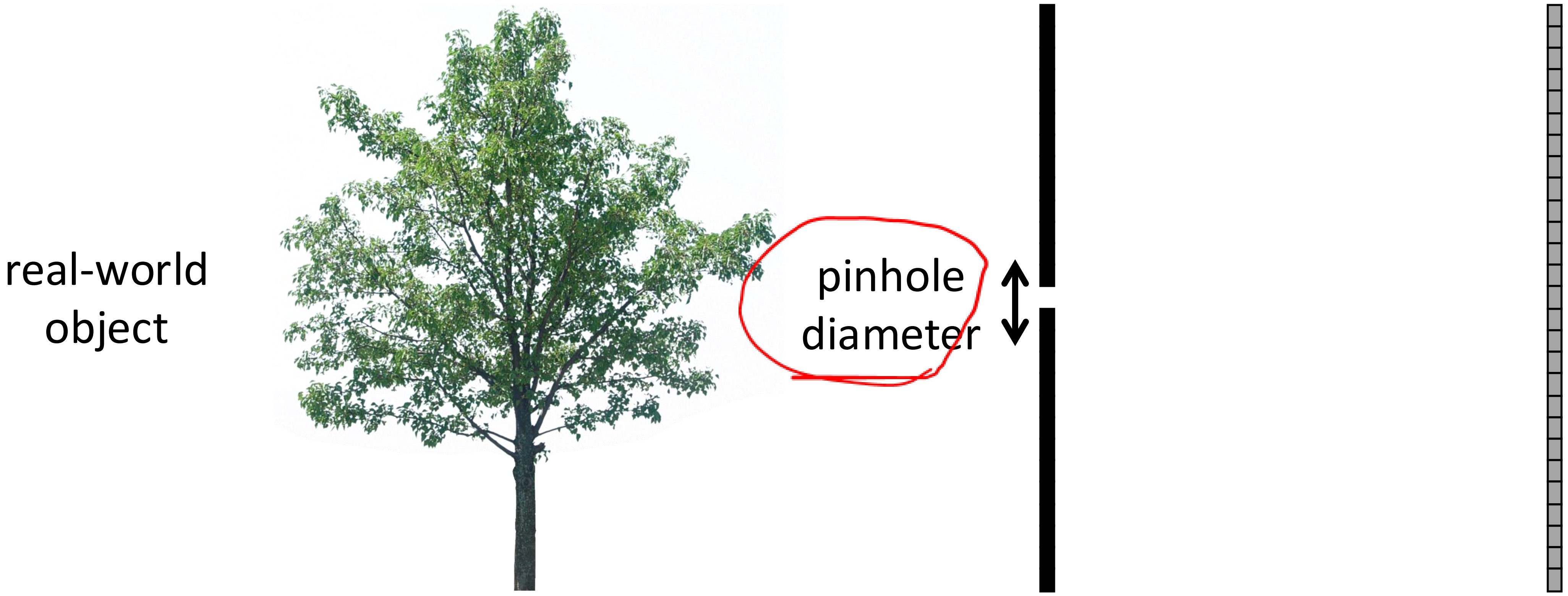


# Focal length

What happens as we change the focal length?



# Pinhole size

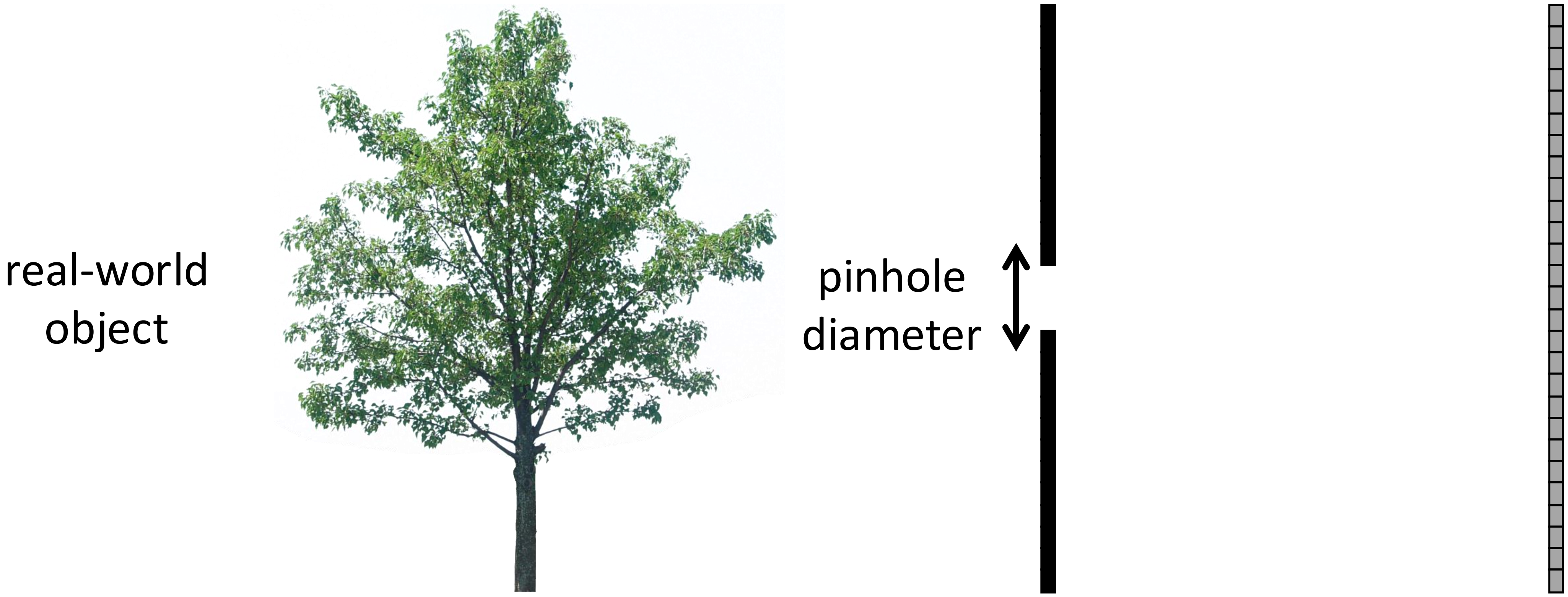



Ideal pinhole has infinitesimally small size

- In practice that is impossible.

# Pinhole size

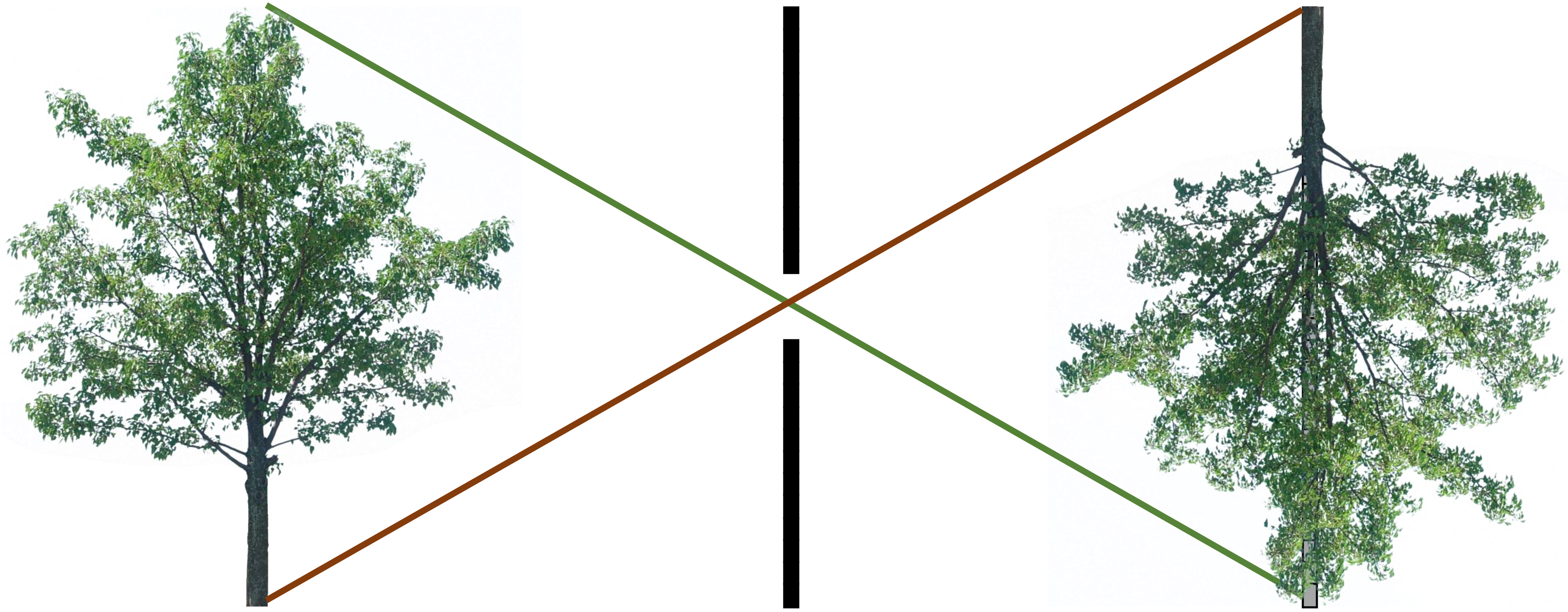
What happens as we change the pinhole diameter?



# Pinhole size

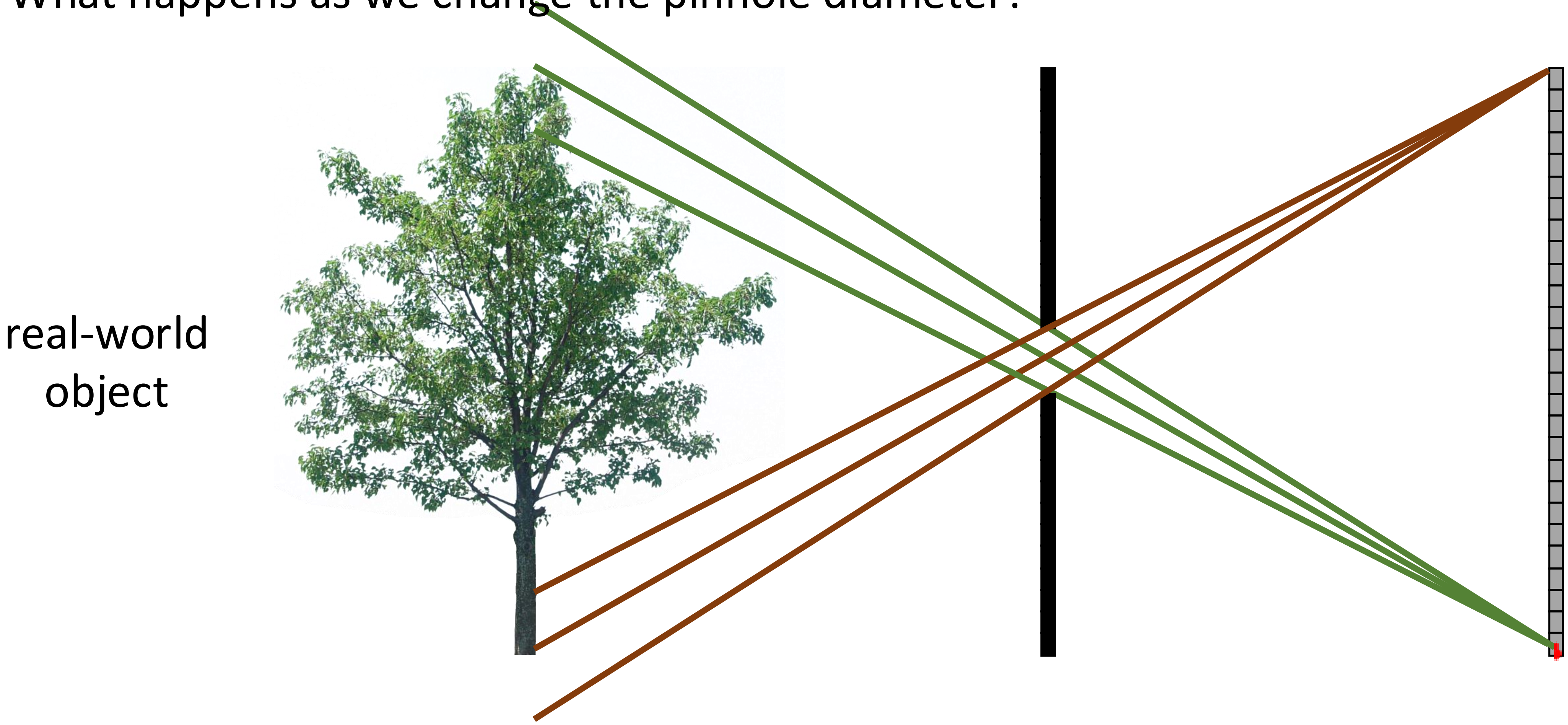
What happens as we change the pinhole diameter?

real-world object



# Pinhole size

What happens as we change the pinhole diameter?

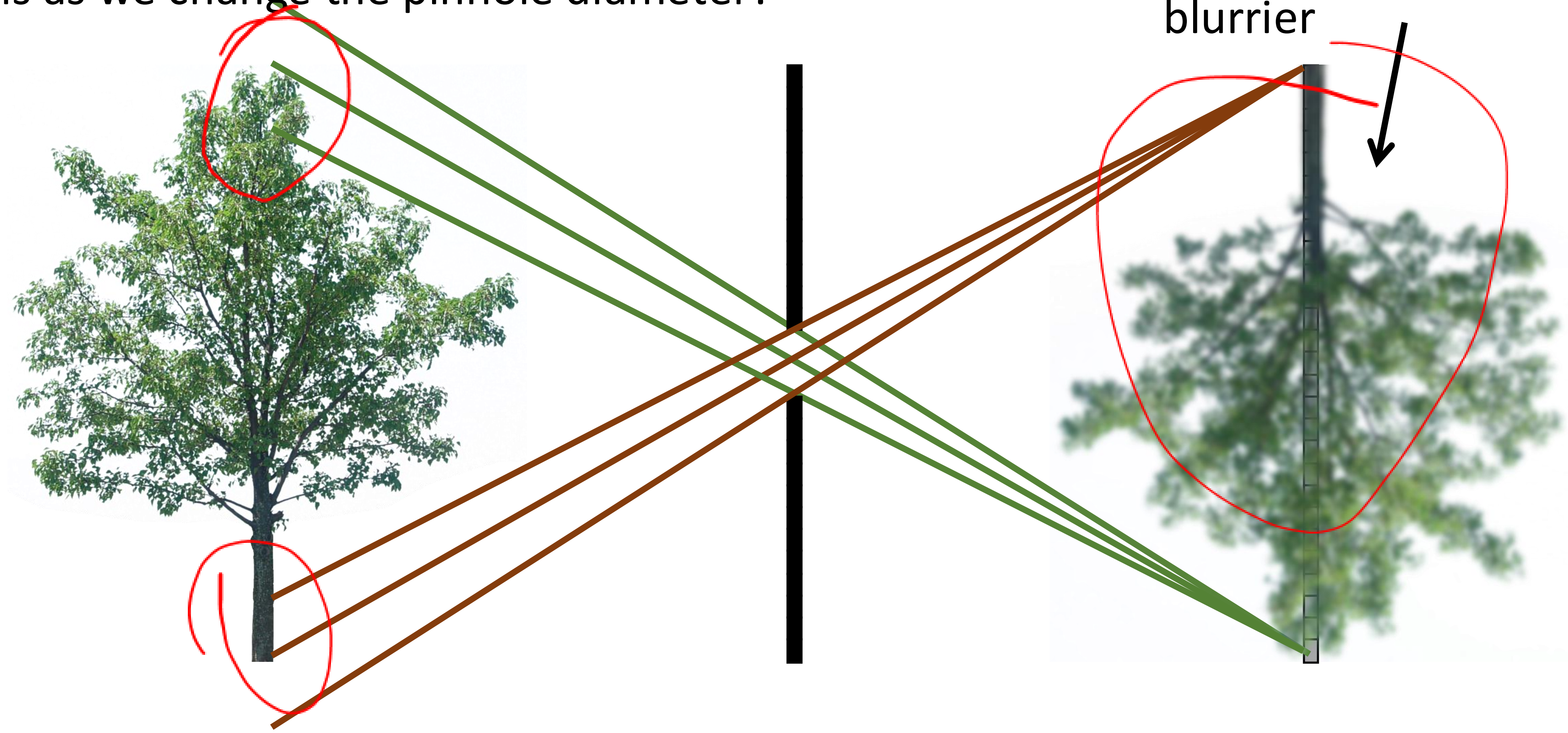


# Pinhole size

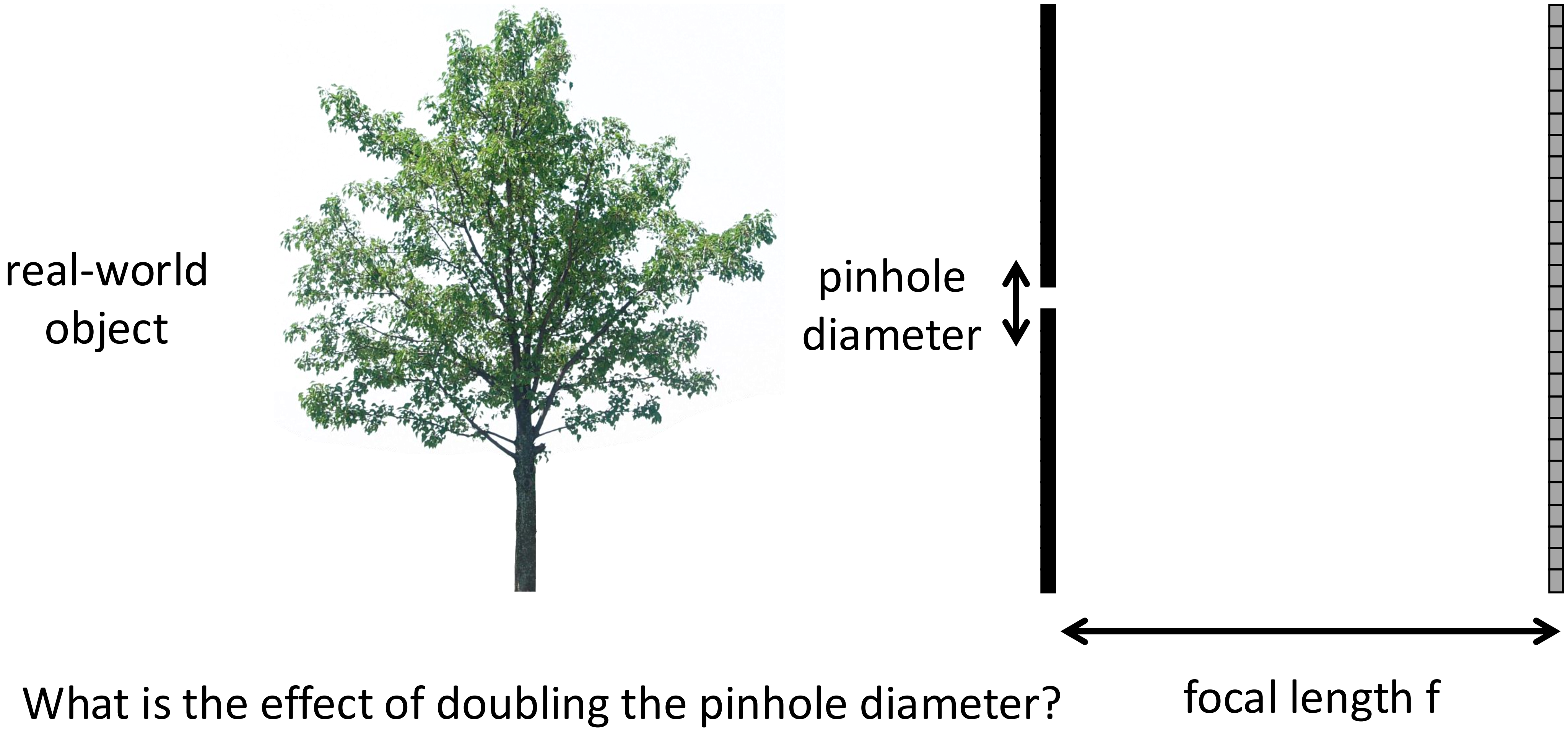
What happens as we change the pinhole diameter?

object projection becomes blurrier

real-world object

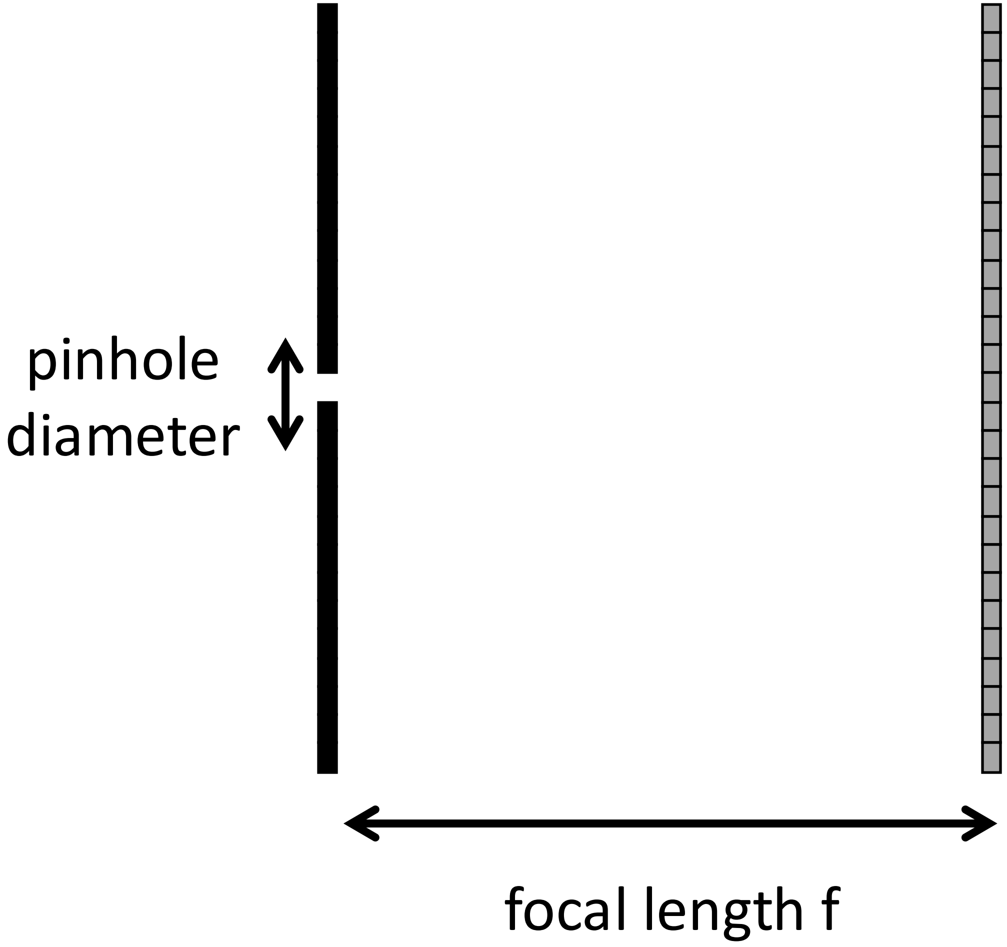


# What about light efficiency?



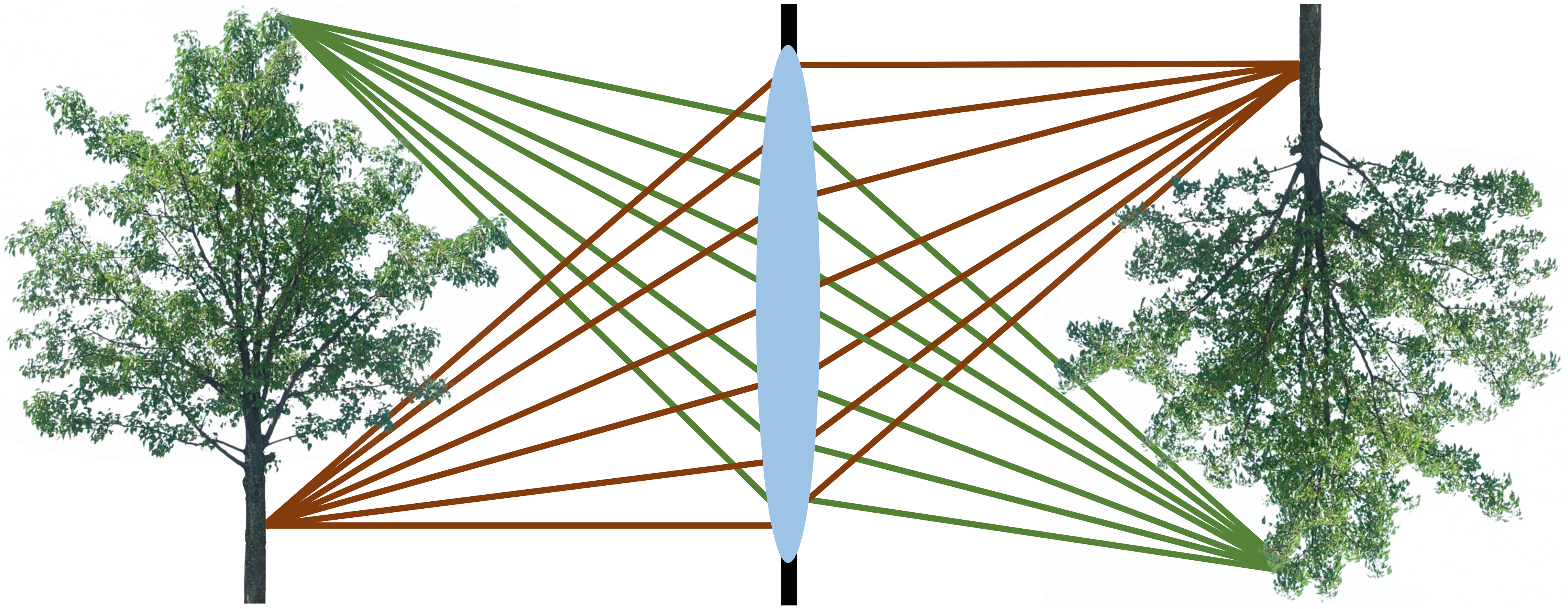

- What is the effect of doubling the pinhole diameter?
- What is the effect of doubling the focal length?

# What about light efficiency?




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

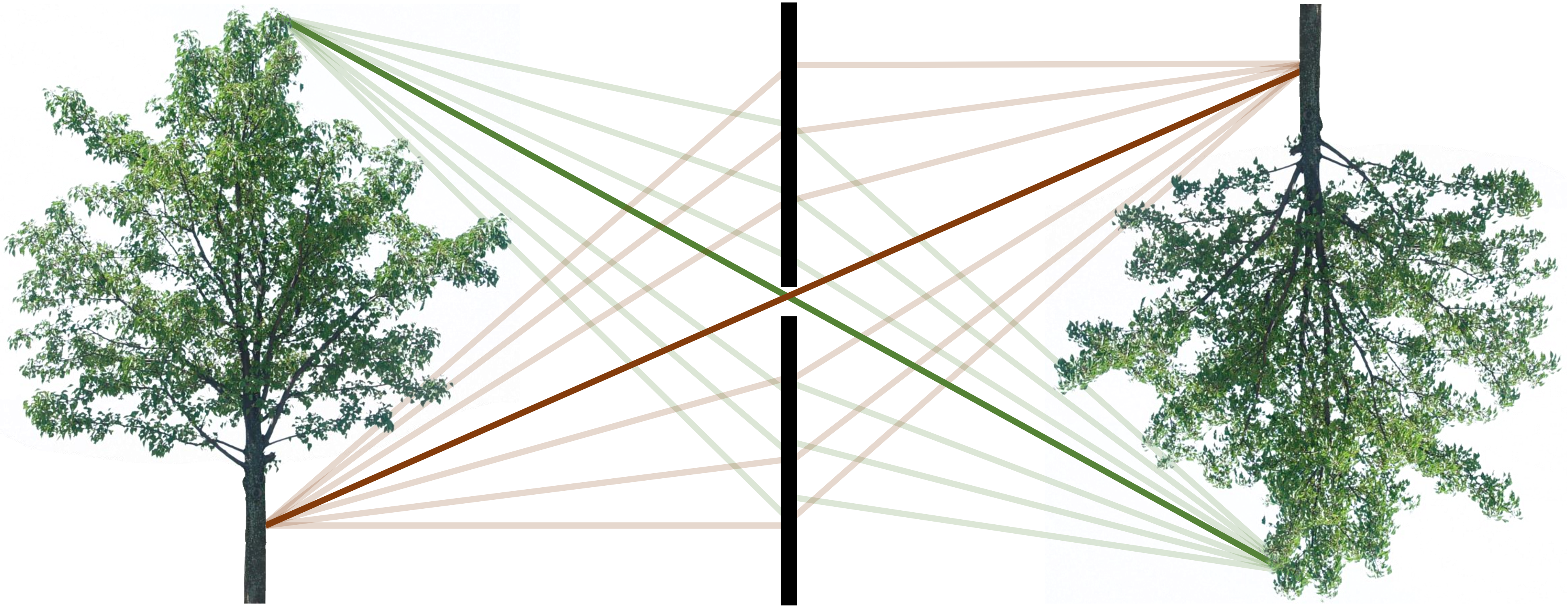
# The lens camera



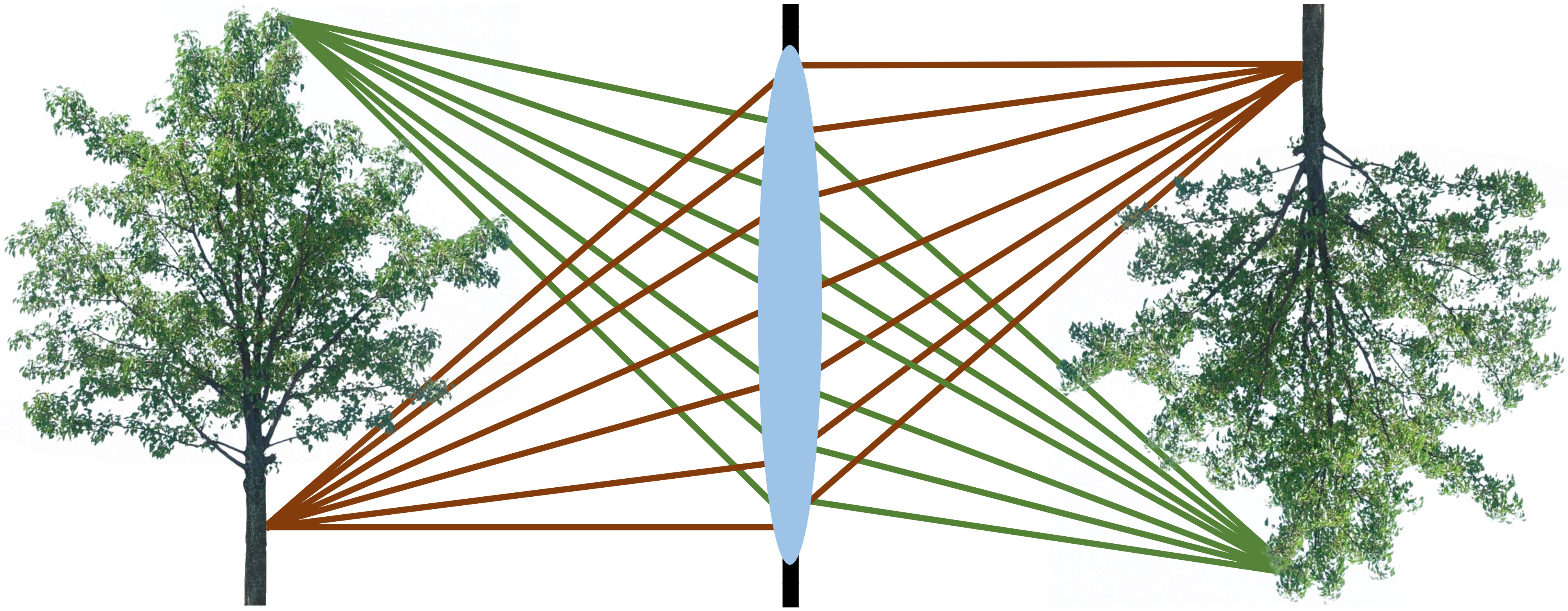
Lenses map "bundles" of rays from points on the scene to the sensor.

How does this mapping work exactly?

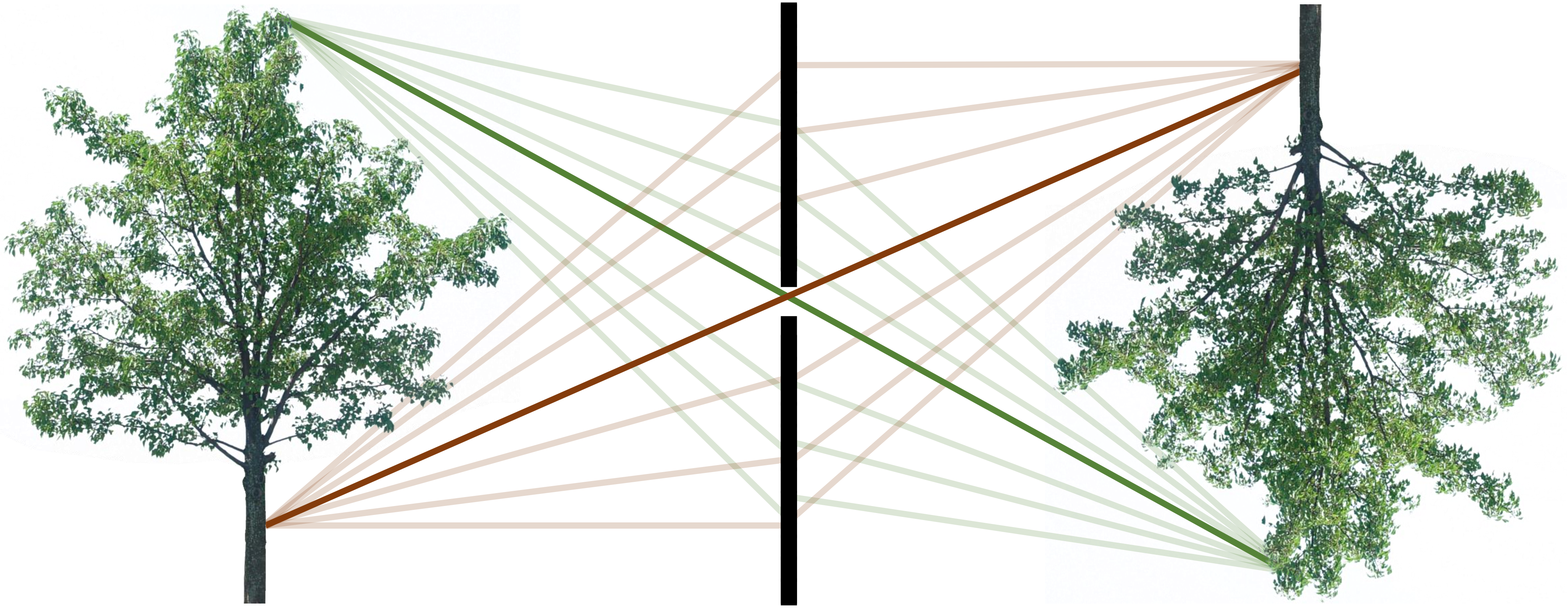
# The pinhole camera



# The lens camera

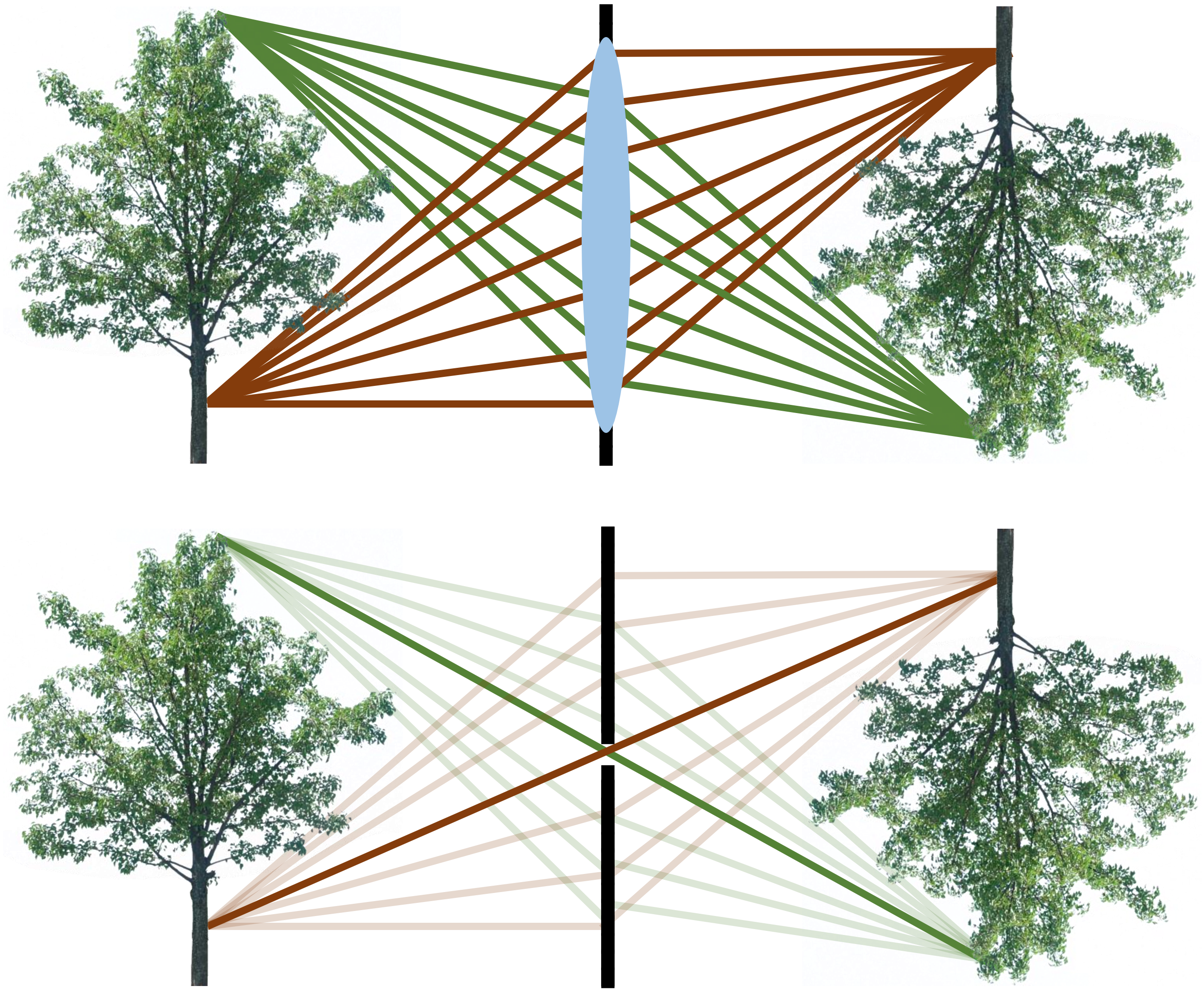


# The pinhole camera



Central rays propagate in the same way for both models!

# Describing both lens and pinhole cameras



We can derive properties and descriptions that hold for both camera models if:

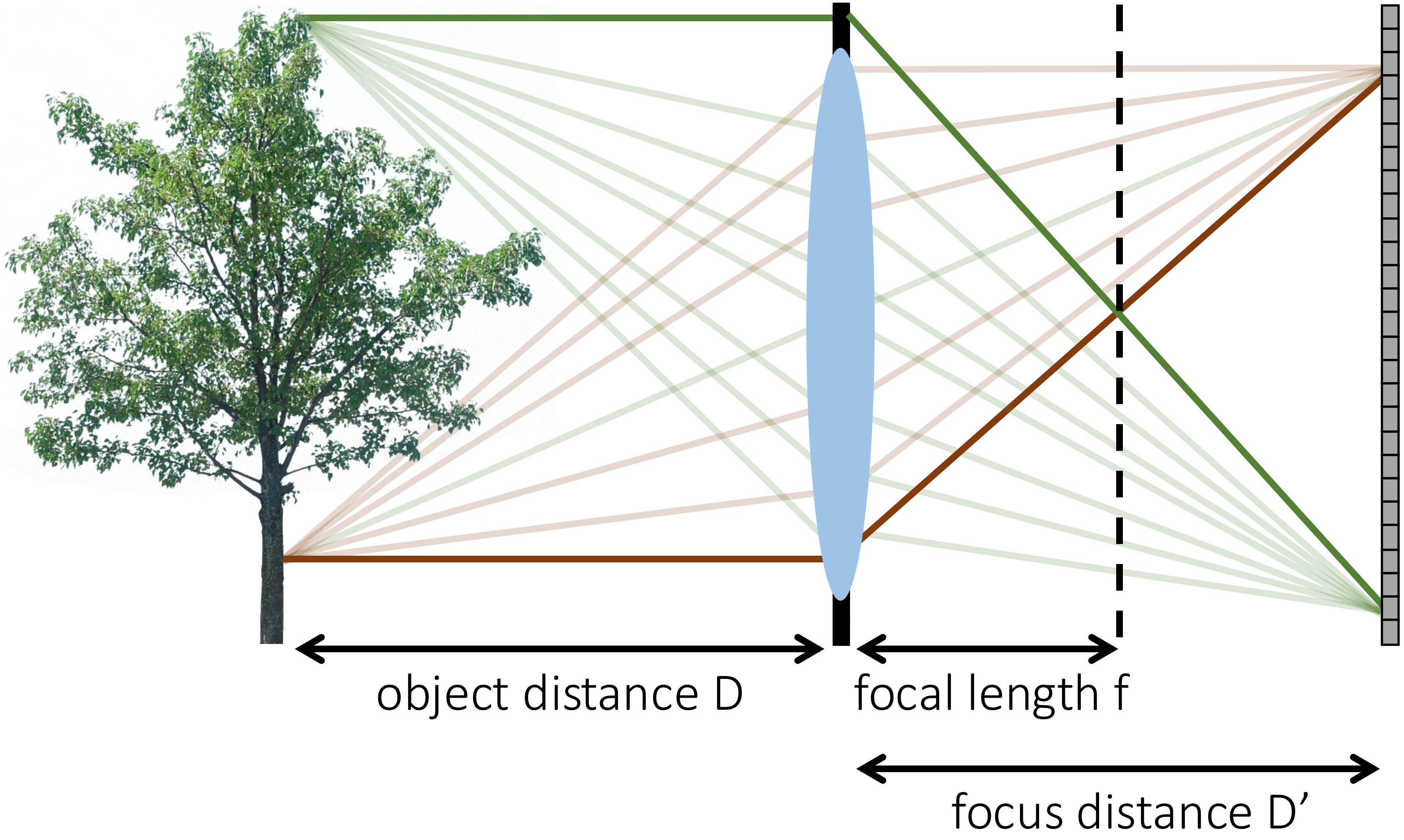
- We use only central rays.
- We assume the lens camera is in focus.

# Important difference: focal length

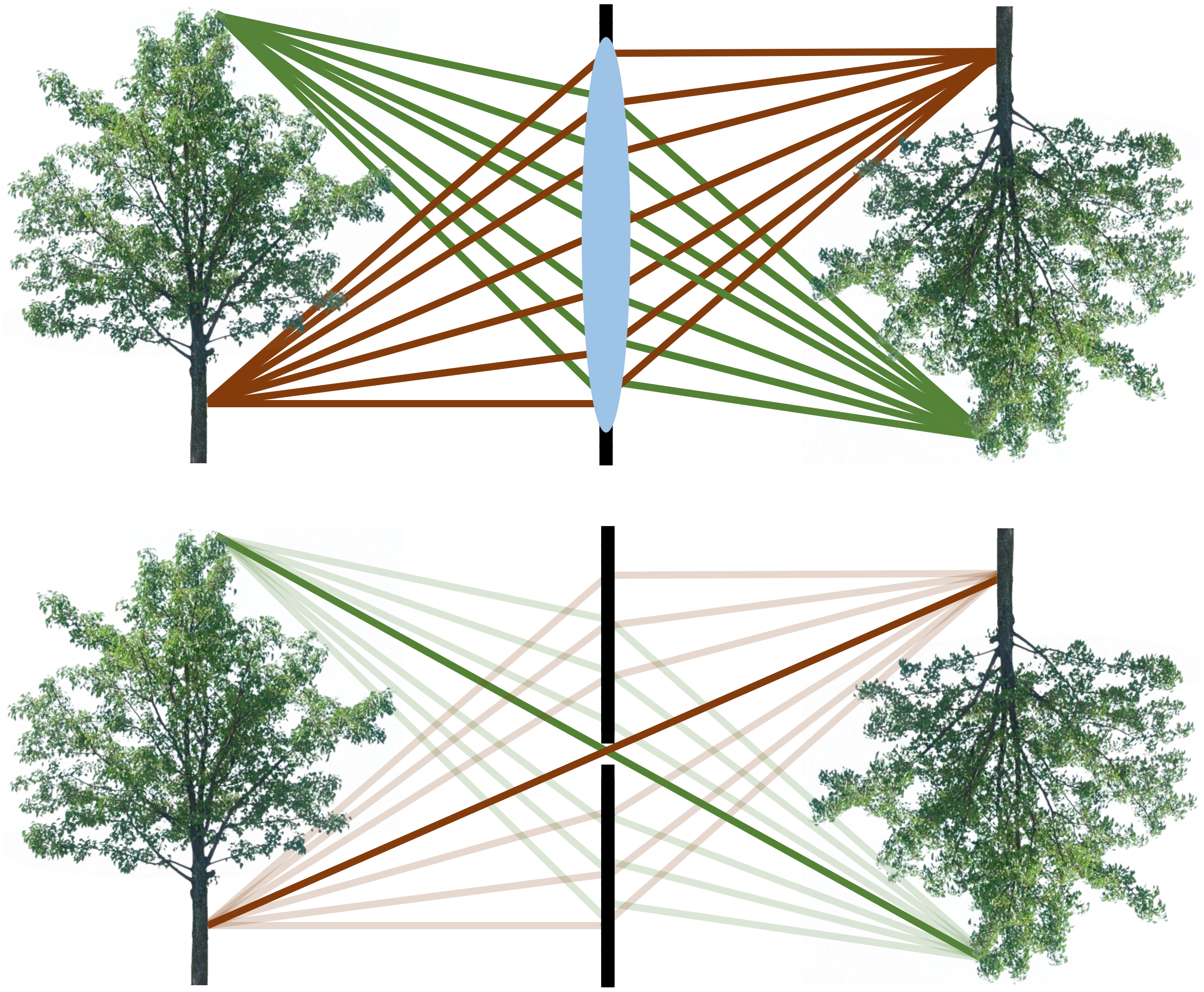
In a pinhole camera, focal length is distance between aperture and sensor

# Important difference: focal length

In a lens camera, focal length is distance where parallel rays intersect



# Describing both lens and pinhole cameras



We can derive properties and descriptions that hold for both camera models if:

- We use only central rays.
- We assume the lens camera is in focus.
- We assume that the focus distance of the lens camera is equal to the focal length of the pinhole camera.

Remember: *focal length* f refers to different things for lens and pinhole cameras.

- In this lecture, we use it to refer to the aperture-sensor distance, as in the pinhole camera case.

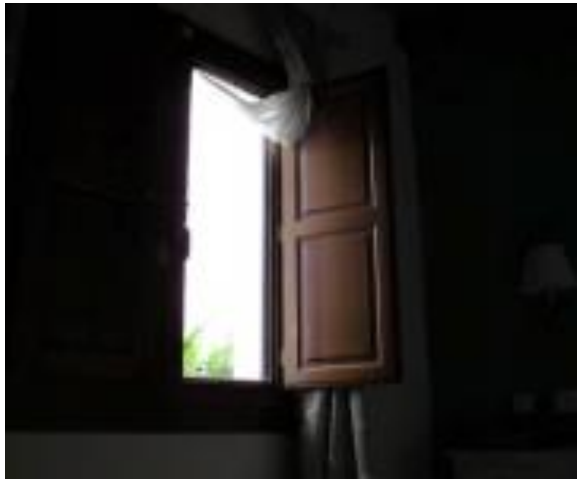
# Accidental pinholes

# What does this image say about the world outside?

# Accidental pinhole camera

Antonio Torralba, William T. Freeman
Computer Science and Artificial Intelligence Laboratory (CSAIL)
MIT
torralba@mit.edu, billf@mit.edu

# Accidental pinhole camera



window is an aperture

projected pattern on the wall

upside down

window with smaller gap

view outside window

# Pinhole cameras

What are we imaging here?

# Deep Camera Obscura: An Image Restoration Pipeline for Lensless Pinhole Photography

Joshua D. Rego[1,2*,3], Huaijin Chen[2], Shuai Li[2], Jinwei Gu[2], Suren Jayasuriya[1,3]

*Optica Optics Express 2021*

1 School of Electrical, Computer and Energy Engineering,
Arizona State University
2 SenseBrain Technology
3 School of Arts, Media and Engineering,
Arizona State University

Arizona State University

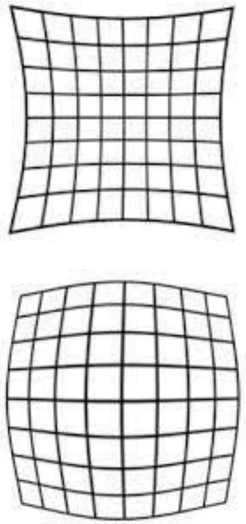
SenseBrain

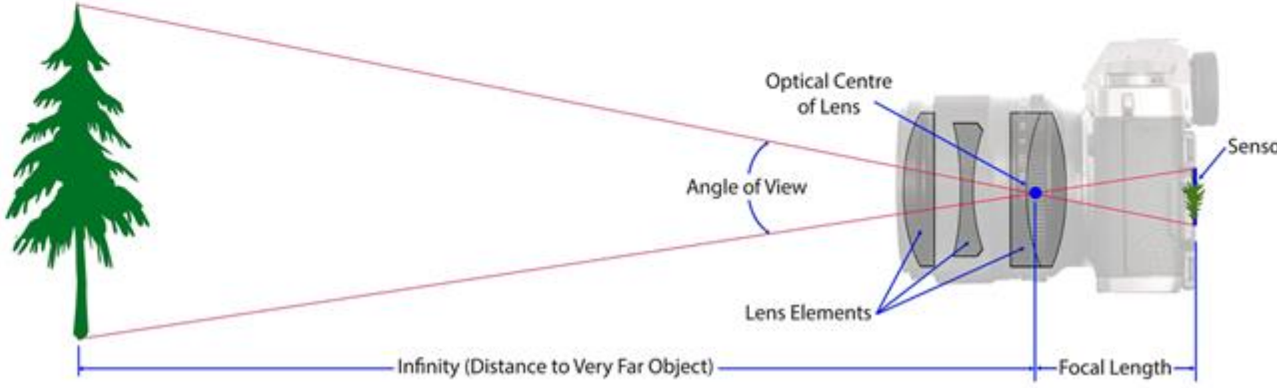
*Work completed during internship at SenseBrain

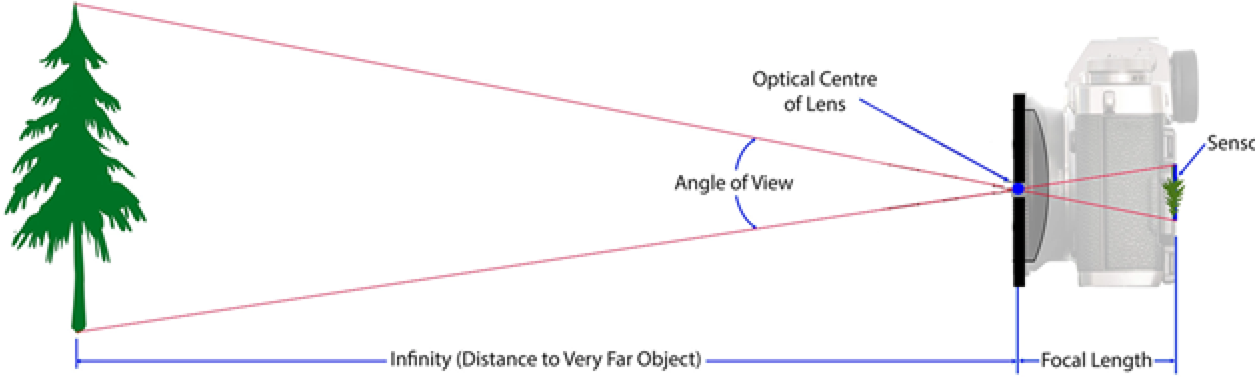
# Pinhole Photography

## Advantages

1. Simple construction
2. More compact than lens
3. No geometric lens distortion
4. Infinite depth-of-field





# Pinhole Photography

## Disadvantages

1. Low light throughput – high noise
2. Diffraction artifacts - blurry

*“ With modern light sources, films, and detectors, even the pinhole’s low aperture need not be an important limitation”*

- M. Young, Pinhole Optics, **1971**

# Pinhole Photography

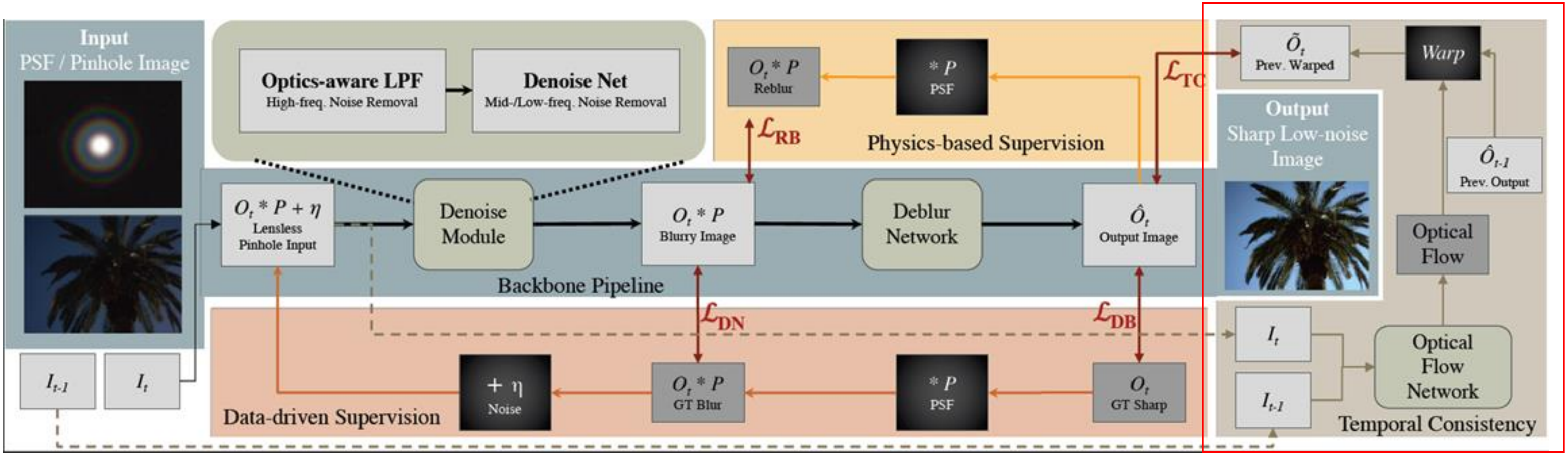
What has changed since 1971?

1. Higher sensitivity sensors for low-light photography
2. Better denoise algorithms to balance higher-gain
3. Better de-blur algorithms to improve sharpness
4. Data-driven/ML methods to further improve all these limitations

*“ With modern light sources, films, and detectors, even the pinhole’s low aperture need not be an important limitation”*

- M. Young, Pinhole Optics, **1971**

# Proposed Pipeline




**Backbone Steps:**

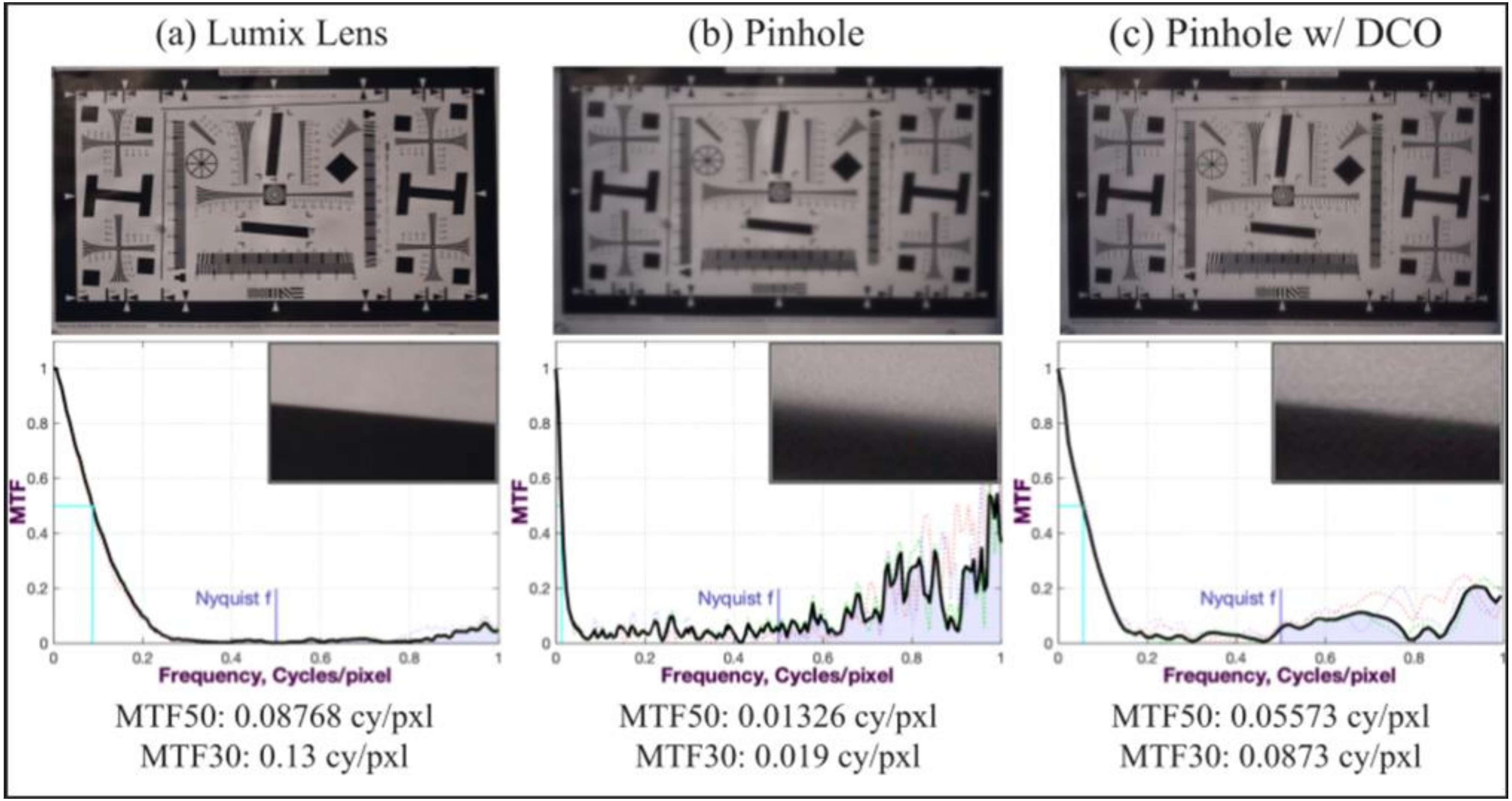
1. Pre-Process RAW Image w/ **Black Level Correction & Demosaicing**
2. High-Frequency Denoising w/ **PSF-informed Low Pass Filter**
3. Mid/Low-Frequency Denoising further w/ **Denoise Network**
4. Sharpen Image w/ **Deblur Network**
5. Post-Process Image w/ **AWB, Gamma Correction, & Exposure Enhancement**

**Training:**

1. **Data-driven Supervision** on synthetic data for **Denoise ($\mathcal{L}_{DN}$) & Deblur ($\mathcal{L}_{DB}$) Losses**
2. **Physics-based Supervision** by rebluring output image for a **Reblur Loss ($\mathcal{L}_{RB}$)**
3. Additional **Temporal Consistency training** for Video data uses Optical Flow method to get a **Temporal Consistency Loss ($\mathcal{L}_{TC}$)**
4. Train each network from scratch + Joint-training to fine-tune

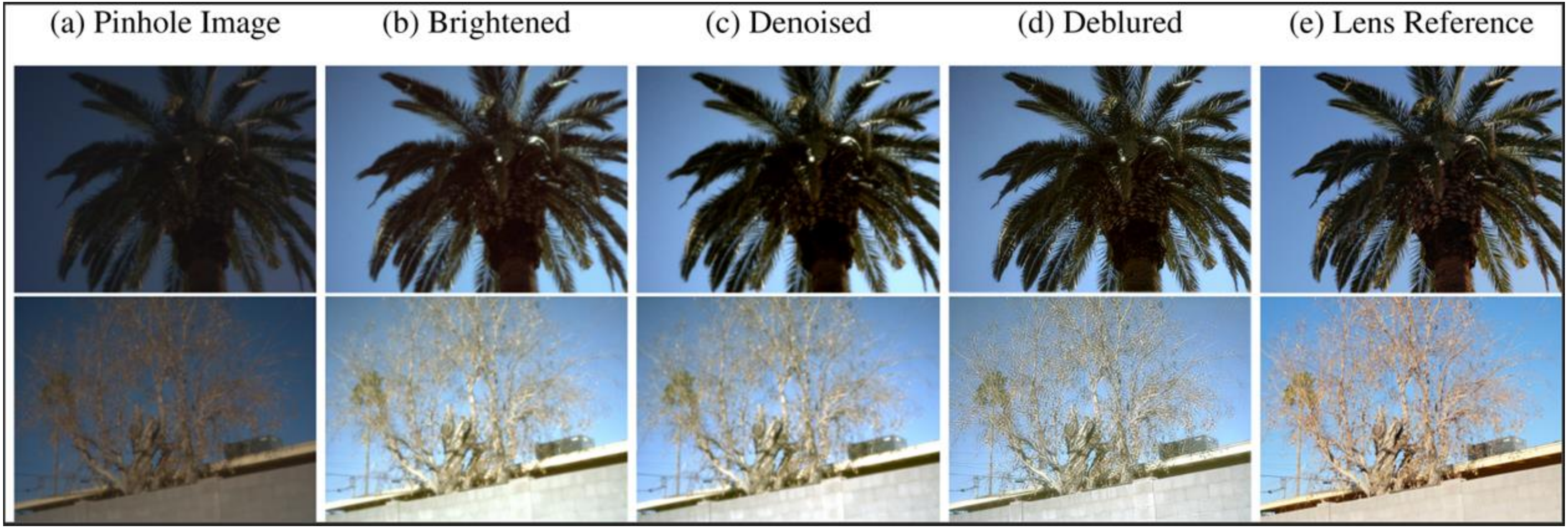

Input
DCO with multi-frame input
DCO trained with Temporal Consistency
DCO to seperate Temporal Consistency module

# Comparisons with Lens Camera

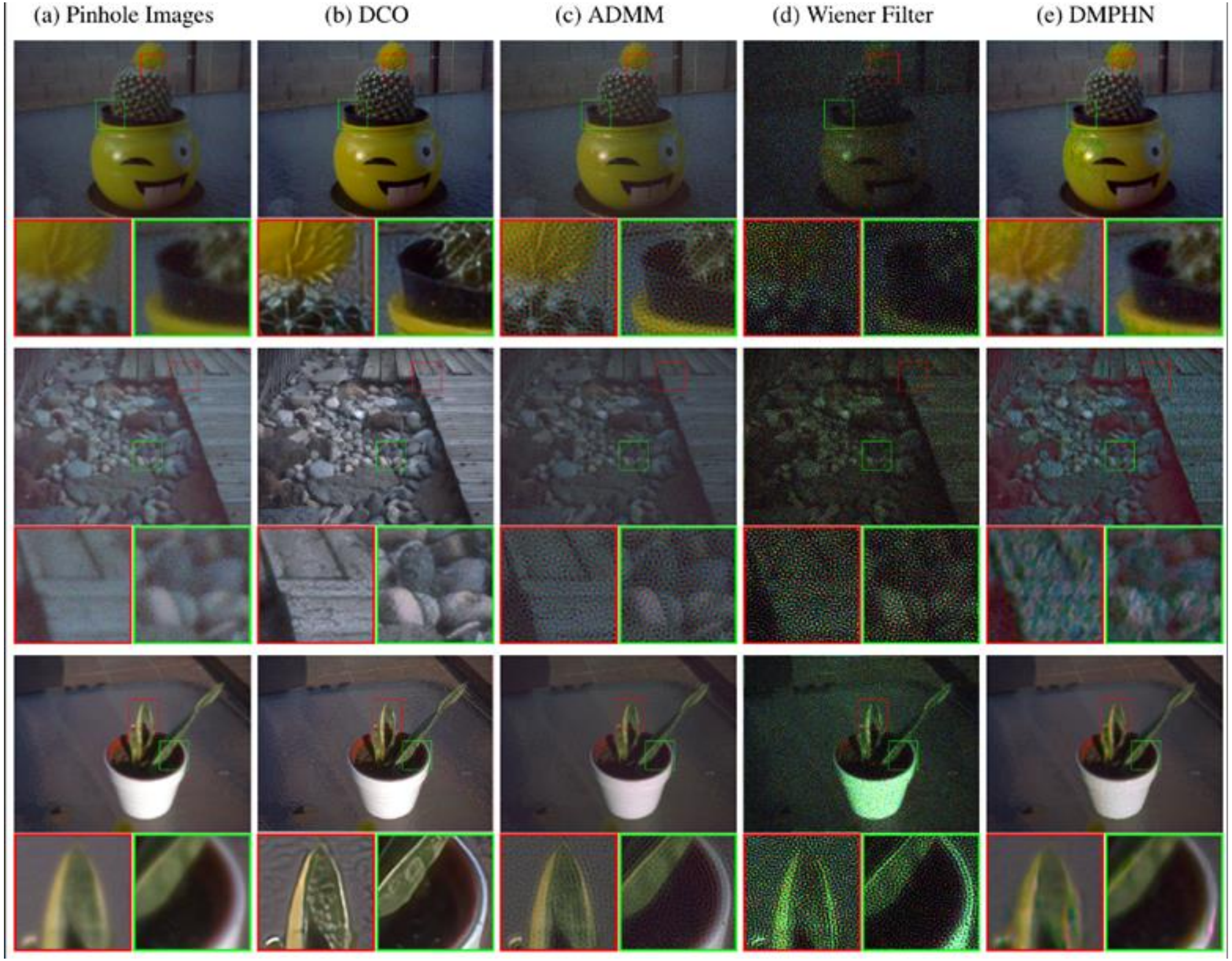


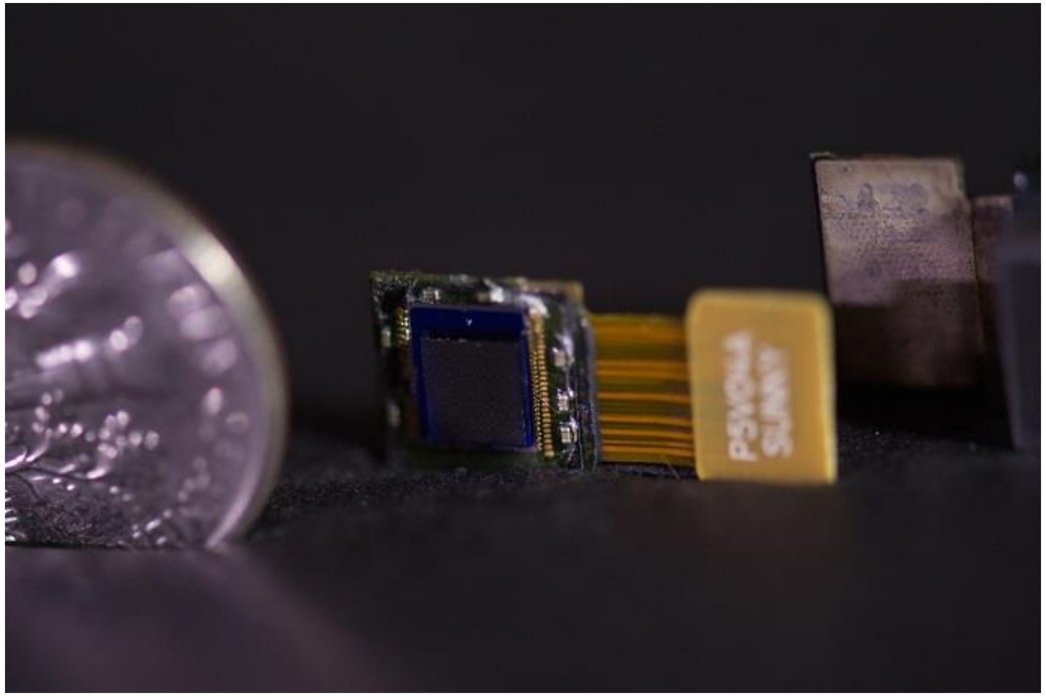
# Real-world Results

Step-wise output visualizations



# Deblur Methods Comparison





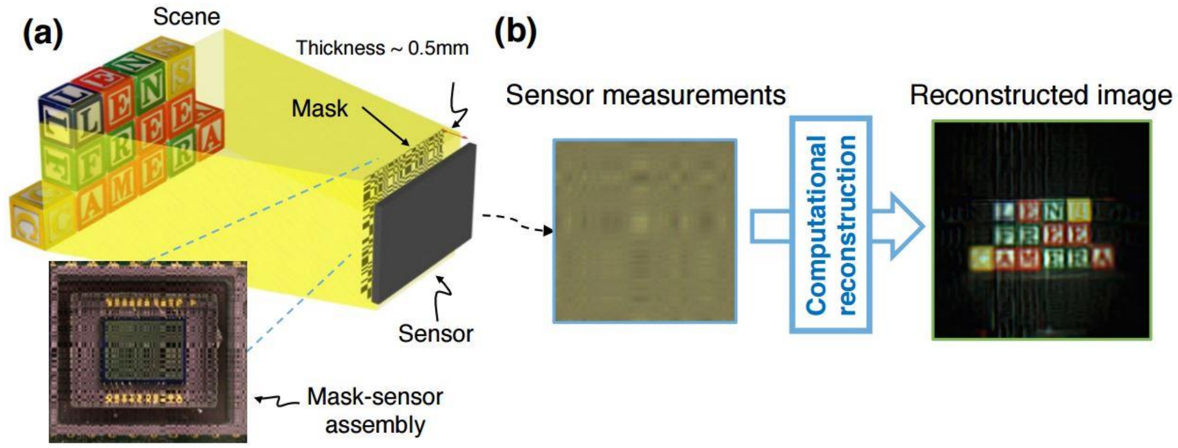
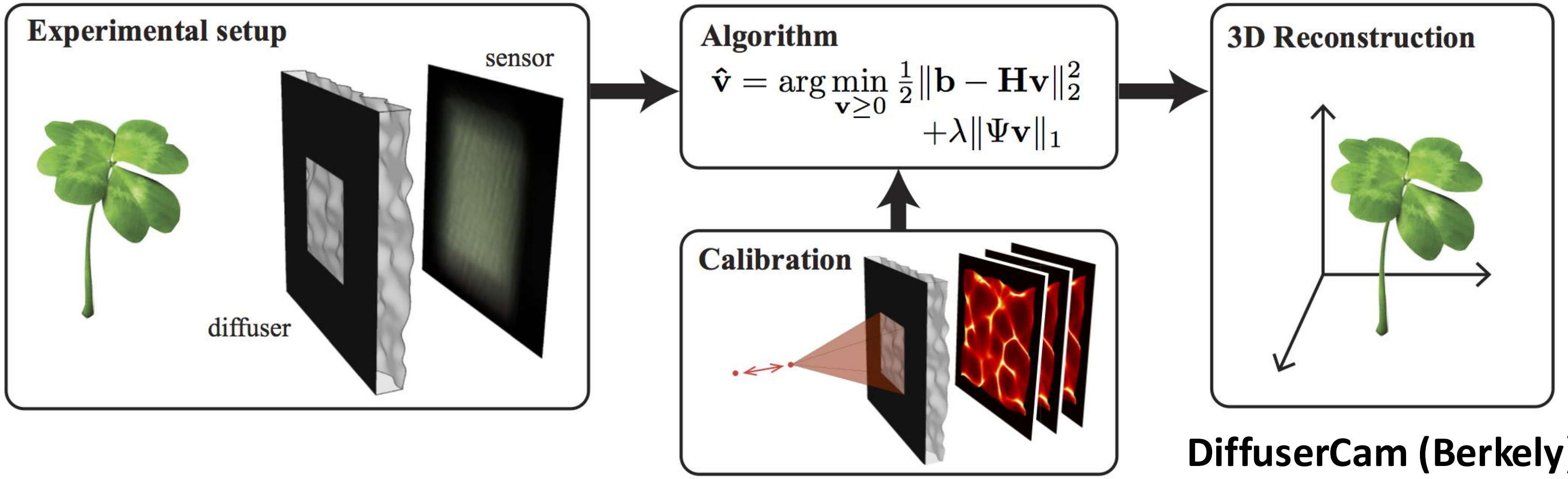


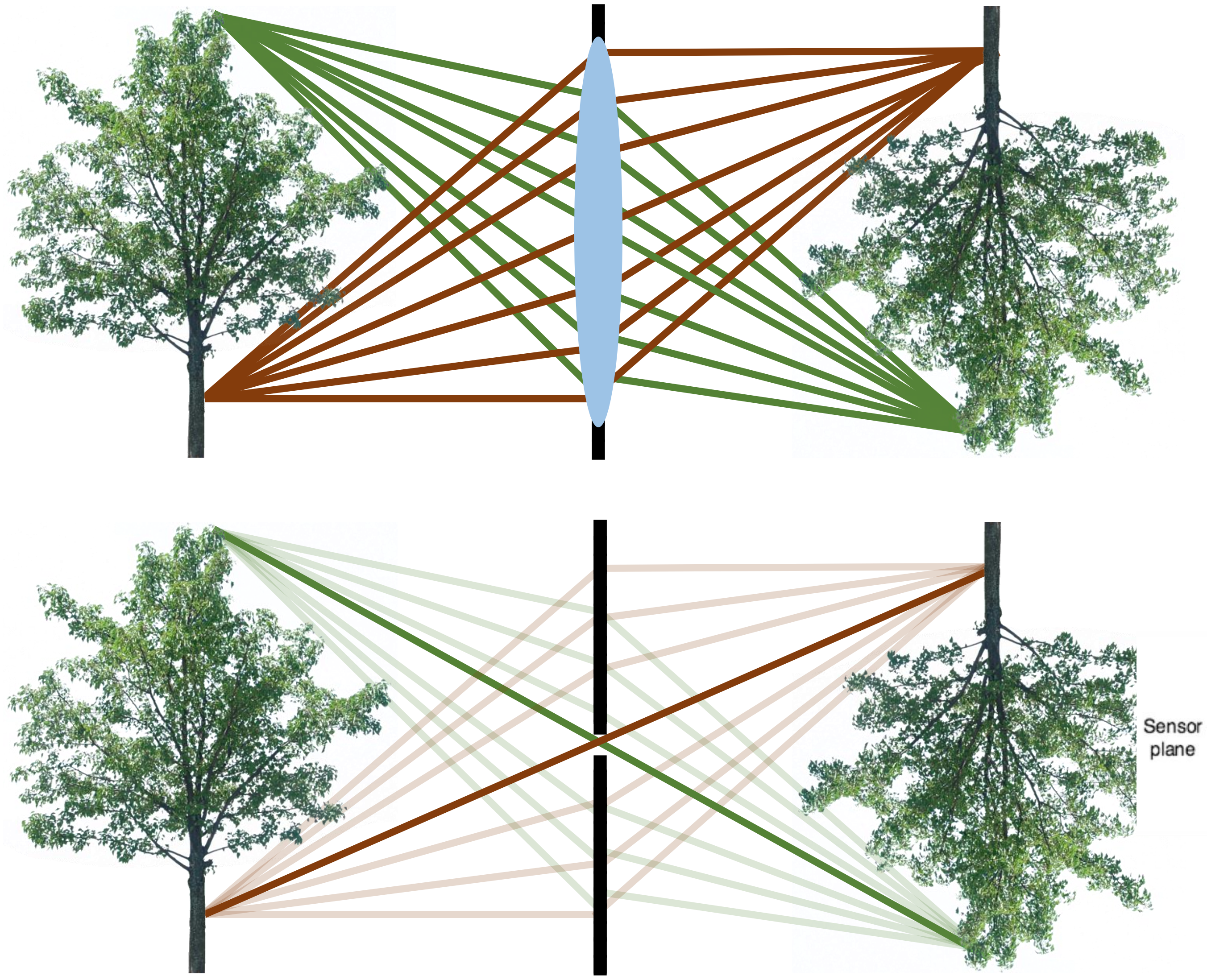

**Figure 1: FlatCam architecture.** **(a)** Every point in the scene casts an image on the sensor, all of which add up to produce the sensor measurements. **(b)** An example of sensor measurements and the reconstructed image.

**FlatCam (Rice)**



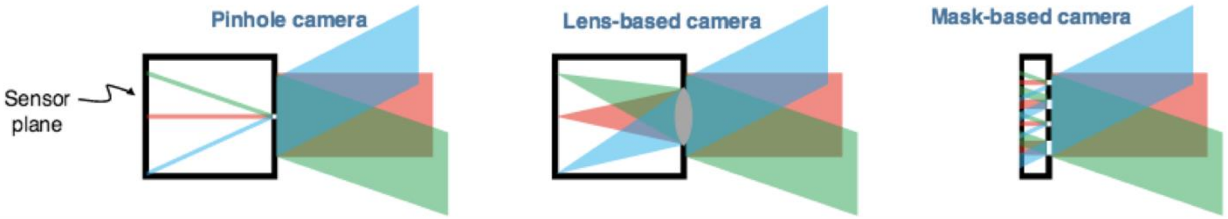

**DiffuserCam (Berkely)**

# Describing both lens and pinhole cameras

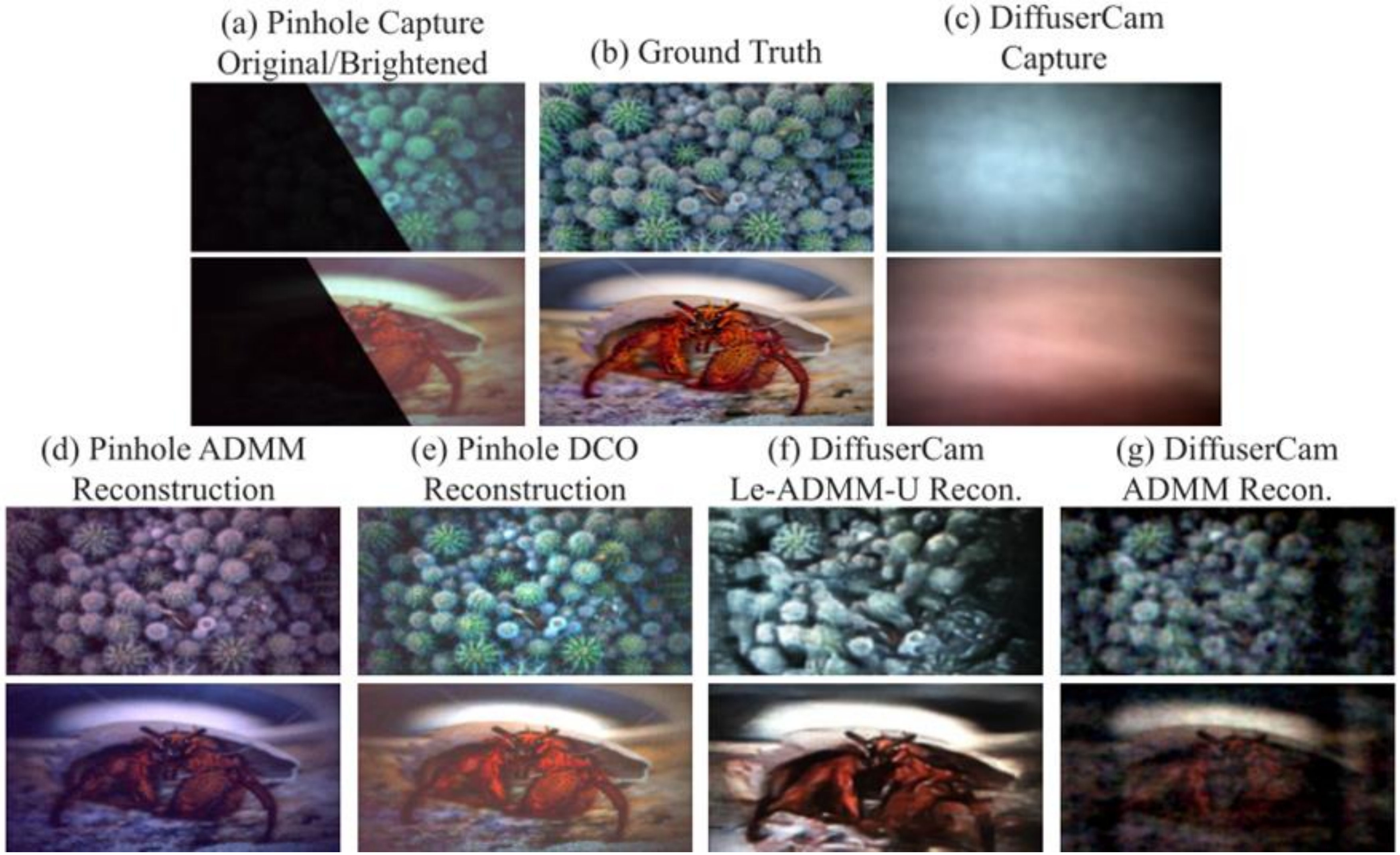


Key ideas for lensless camera: hack the light rays without lens then use computation to back-calculate.

Rays can be quite unstructured so calibration is needed.



# Comparisons with Other Lensless Cameras



# Comparisons with Other Lensless Cameras

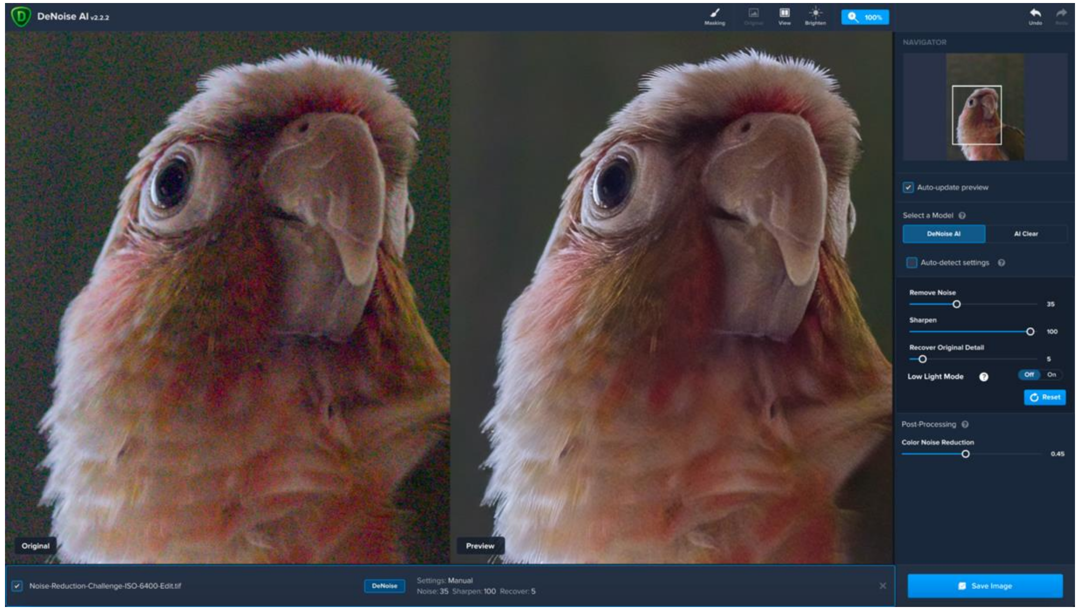
# Conclusion

- An end-to-end imaging pipeline for practical pinhole photography with joint denoising and deblurring for low-light capture leveraging both a data-driven network and imaging physics.

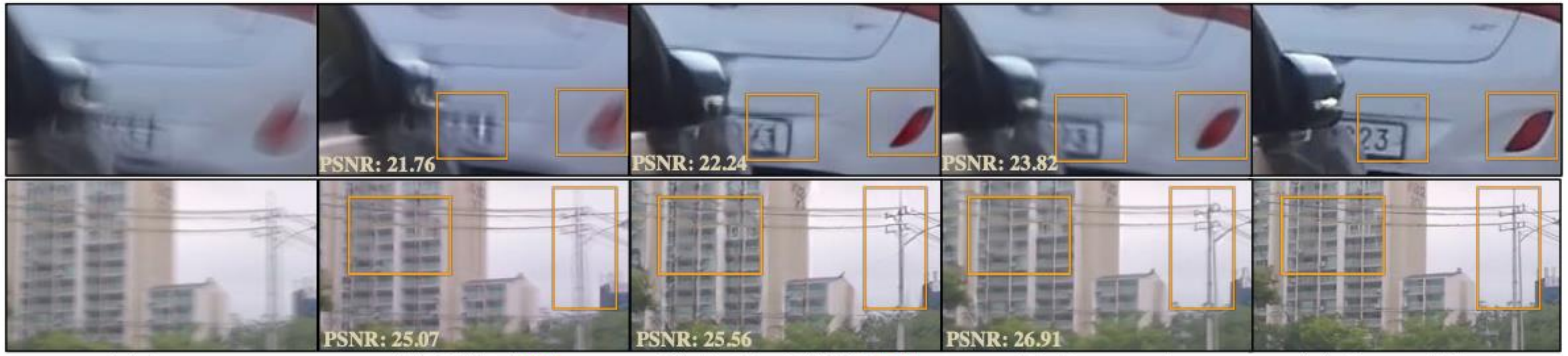
- A pinhole image dataset with measured HDR PSF suitable for generating synthetic data.

- Detailed tradeoff analysis and ablation studies including pinhole size, light loss, choice of denoise and deblur networks, effects of ISO and exposure time.

# Ideas for Final Projects?




- Undistorting images?
- Denoising?
- Deblurring?




(a) Input (b) DVD [22] (c) DeblurGAN [13] (d) Proposed (e) Ground Truth

Figure 1. We propose a novel method for video motion deblur with self-supervised learning. Compared to prior method such as DVD [22] and DeblurGAN [13], we enforce a physics-based blur formation model during Deep Neural Network (DNN) learning, which effectively reduces image artifacts and improves generalization ability of DNN-based video deblurring.

# Camera matrix

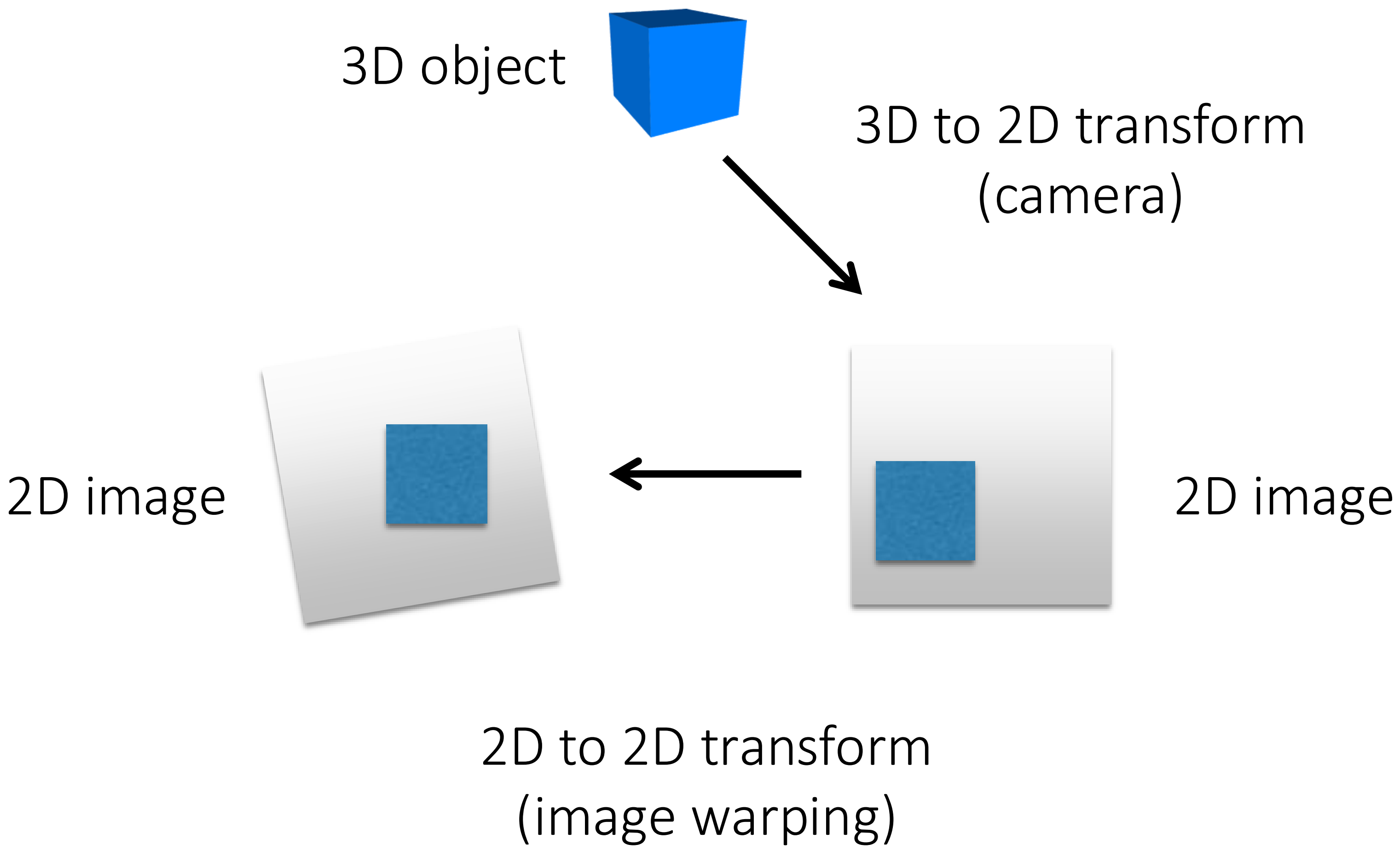
# The camera as a coordinate transformation

A camera is a mapping from:

the 3D world

to:

a 2D image



# The camera as a coordinate transformation

A camera is a mapping from:

the 3D world

to:

a 2D image

homogeneous coordinates

$$\mathbf{x} = \mathbf{P}\mathbf{X}$$

2D image point

camera matrix

3D world point

What are the dimensions of each variable?

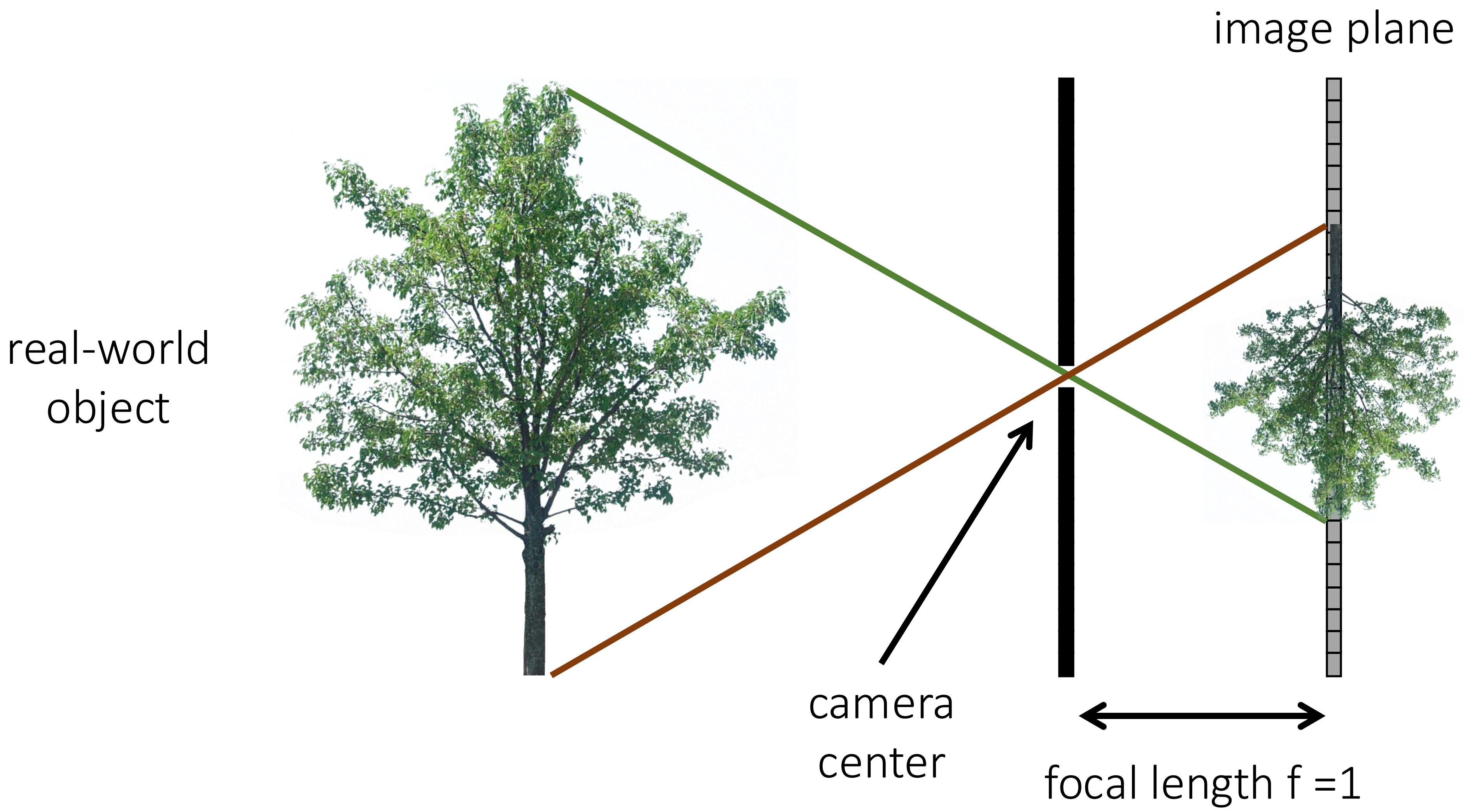
# The camera as a coordinate transformation

$$\mathbf{x} = \mathbf{P}\mathbf{X}$$

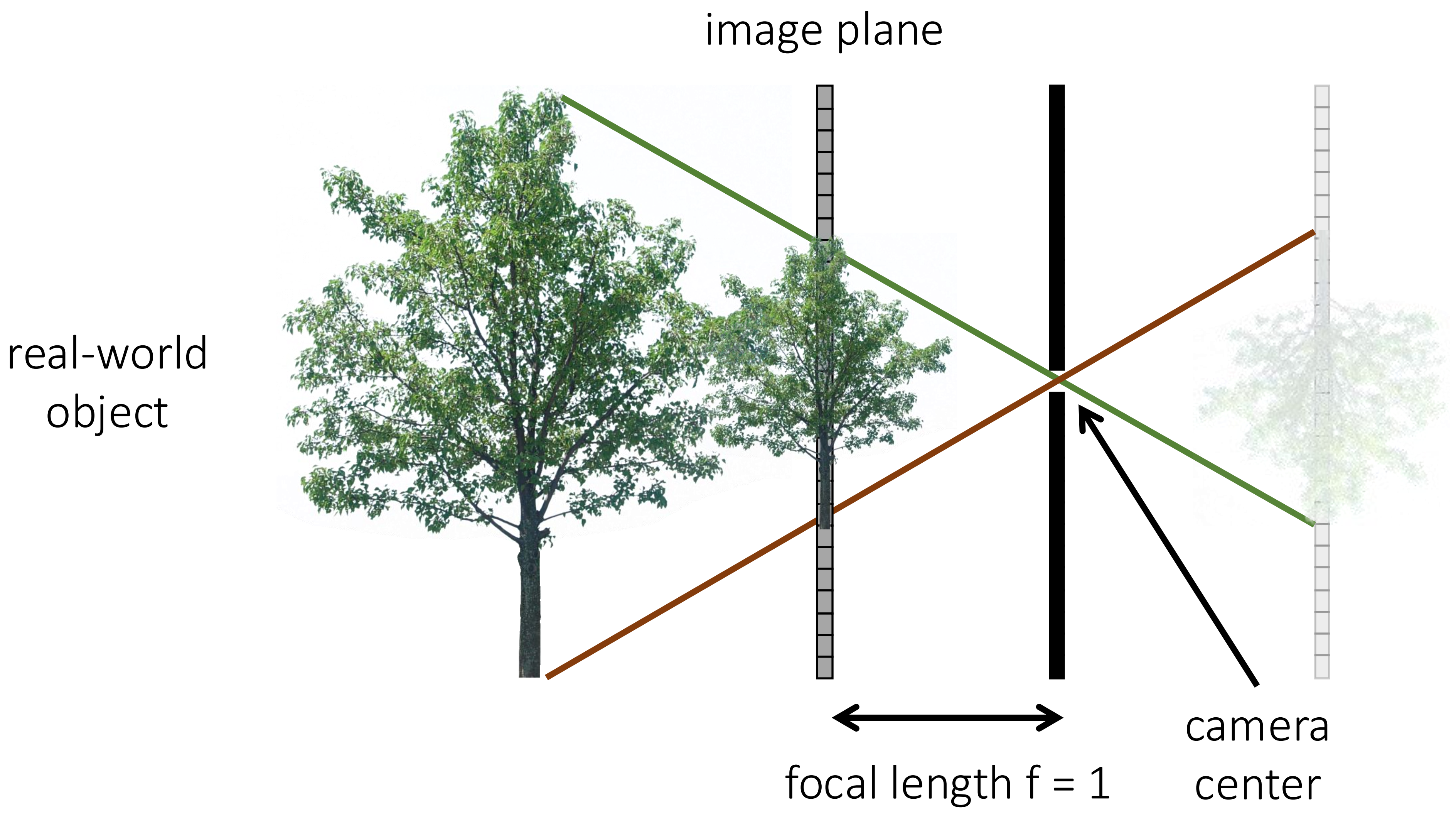
$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

| homogeneous image coordinates | camera matrix | homogeneous world coordinates |
|---|---|---|
| 3 x 1 | 3 x 4 | 4 x 1 |

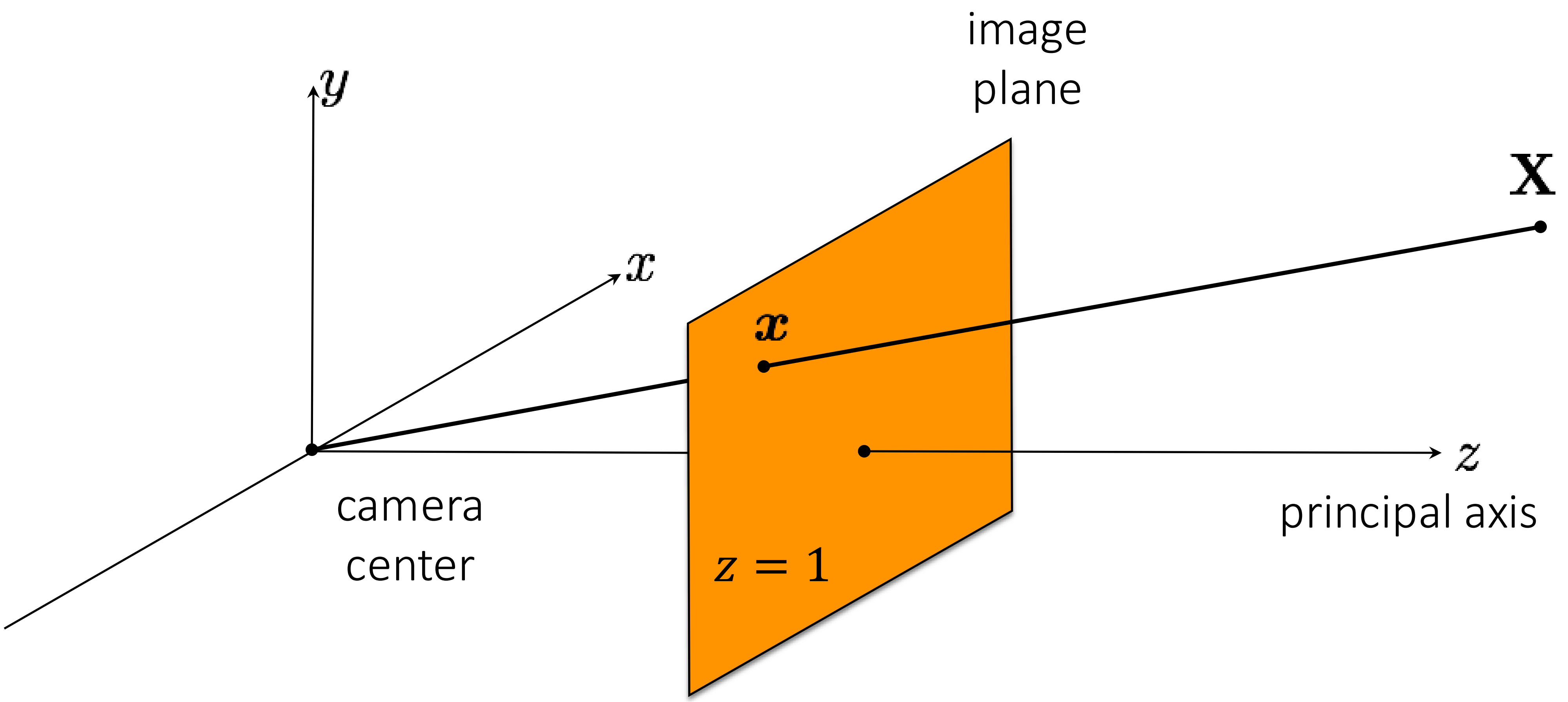
# The pinhole camera



# The (rearranged) pinhole camera

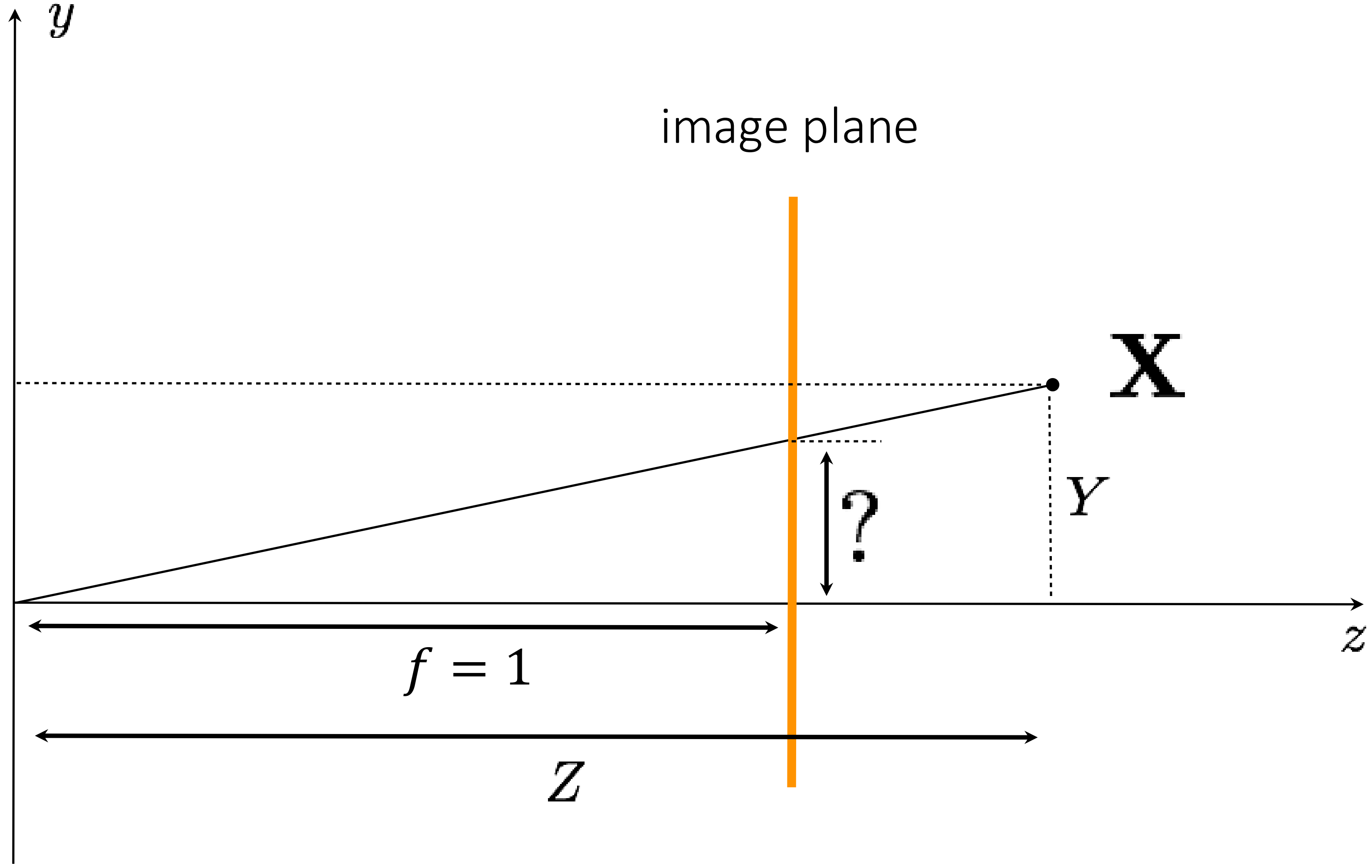


# The (rearranged) pinhole camera



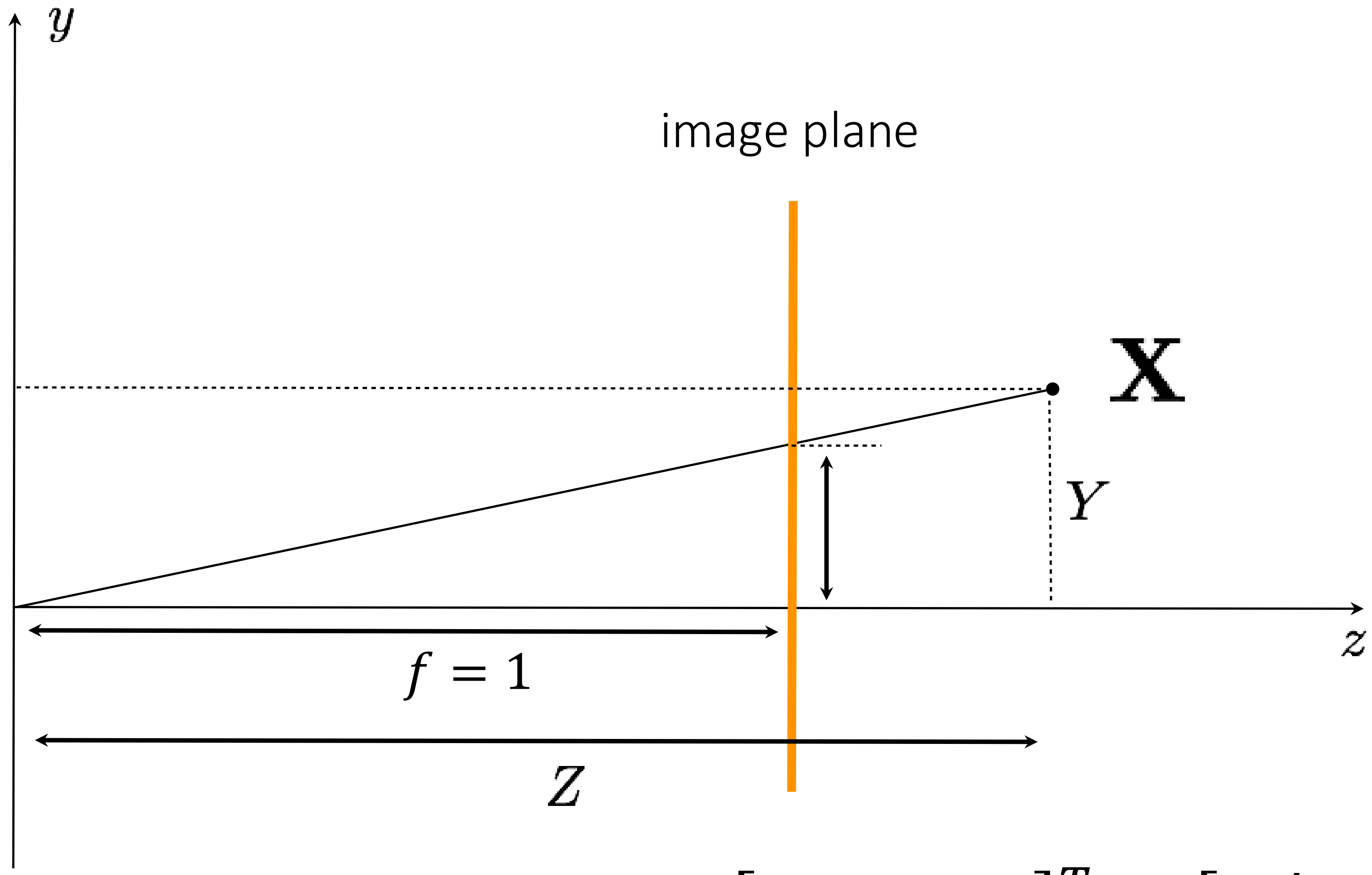

What is the equation for image coordinate **x** in terms of **X**?

# The 2D view of the (rearranged) pinhole camera



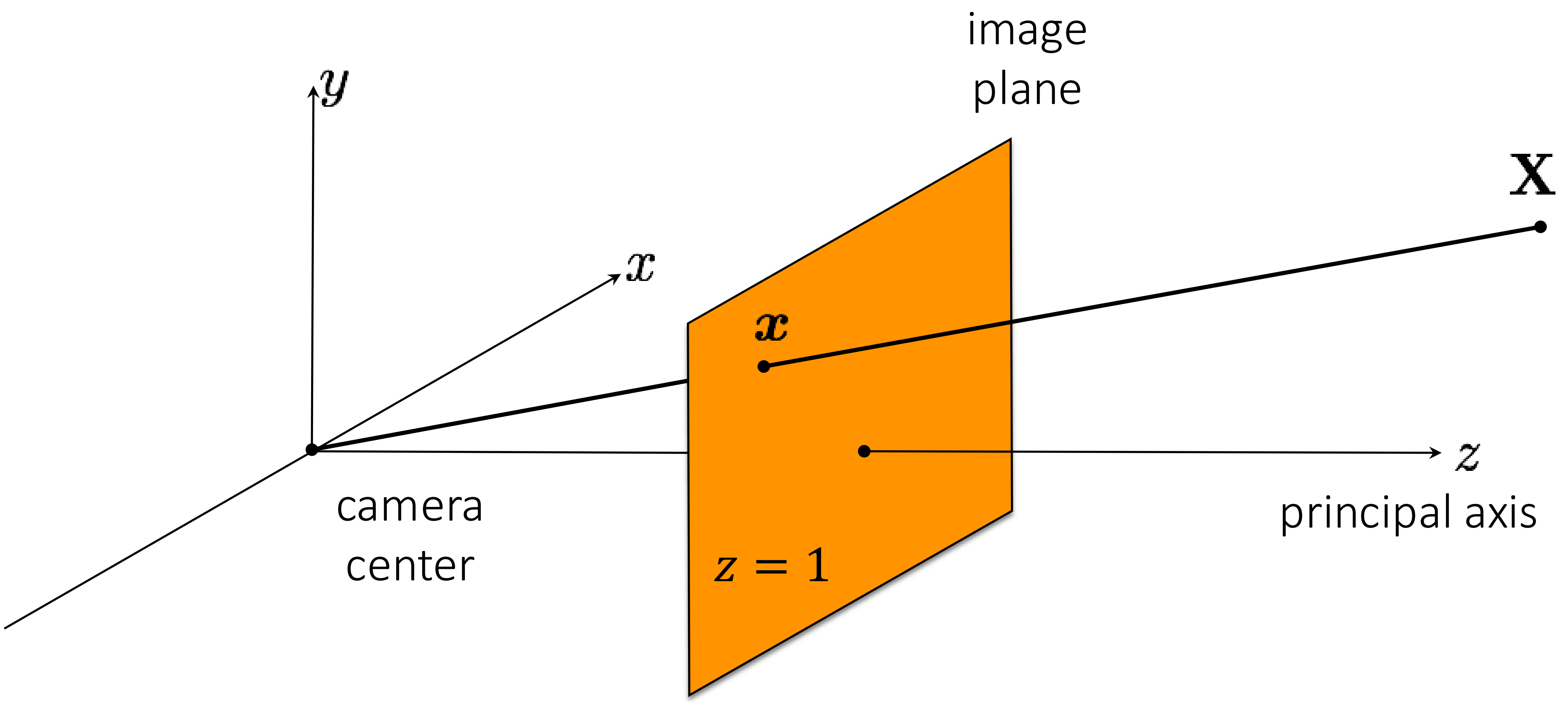

What is the equation for image coordinate **x** in terms of **X**?

# The 2D view of the (rearranged) pinhole camera




$$[X \quad Y \quad Z]^T \rightarrow [X/Z \quad Y/Z]$$

# The (rearranged) pinhole camera




What is the camera matrix **P** for a pinhole camera?

$$\boldsymbol{x} = \mathbf{P}\mathbf{X}$$

# The pinhole camera matrix

Relationship from similar triangles:

$$[X \quad Y \quad Z]^T \rightarrow [X/Z \quad Y/Z]$$

General camera model in *homogeneous coordinates*:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

What does the pinhole camera projection look like?

$$\mathbf{P} = \begin{bmatrix} ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{bmatrix}$$

# The pinhole camera matrix

Relationship from similar triangles:

$$[X \quad Y \quad Z]^T \rightarrow [X/Z \quad Y/Z]$$

General camera model *in homogeneous coordinates*:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

What does the pinhole camera projection look like?

The perspective projection matrix

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

# The pinhole camera matrix

Relationship from similar triangles:

$$[X \quad Y \quad Z]^T \rightarrow [X/Z \quad Y/Z]$$

General camera model *in homogeneous coordinates*:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

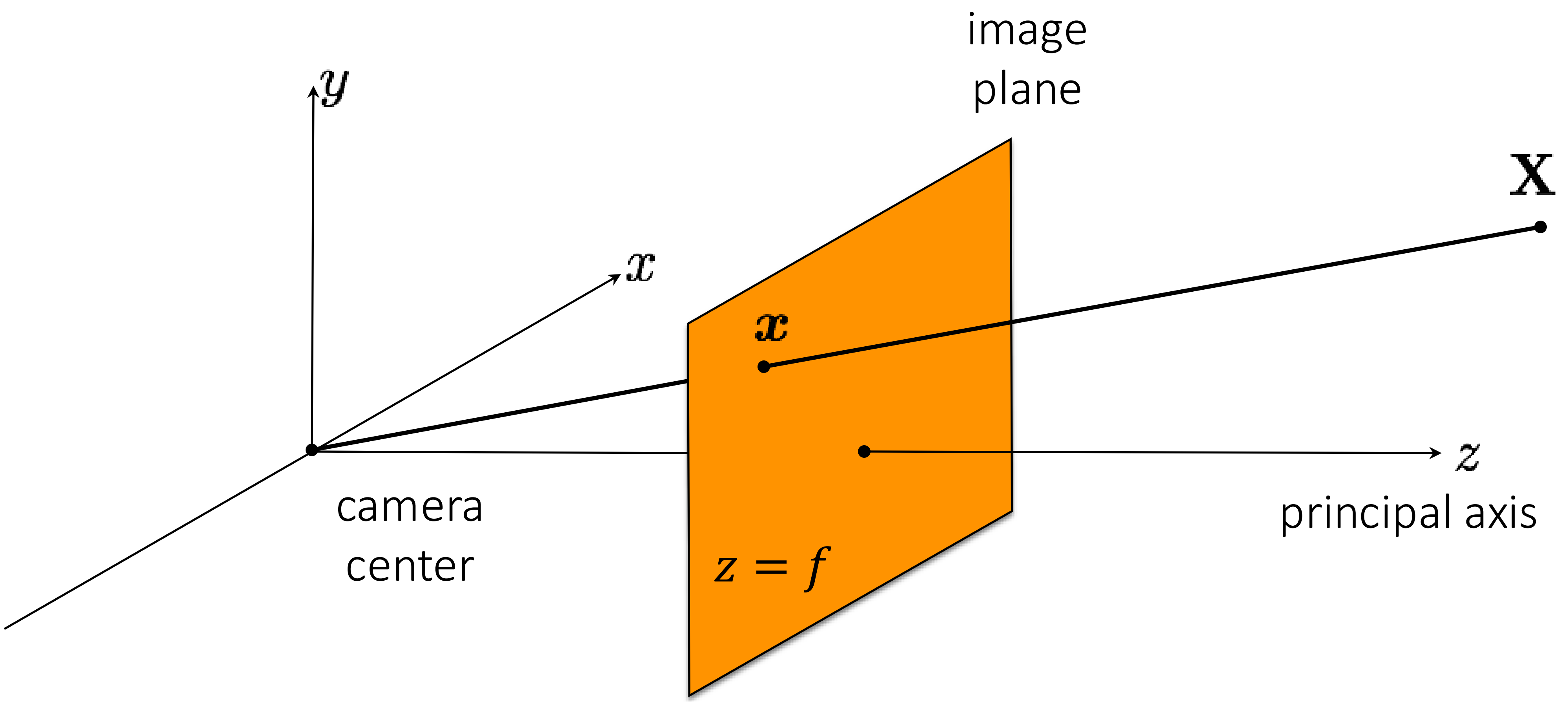
What does the pinhole camera projection look like?

The perspective projection matrix

$$\mathbf{P} = \left[\begin{array}{ccc:c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right] = [\mathbf{I} \quad | \quad \mathbf{0}]$$

alternative way to write the same thing

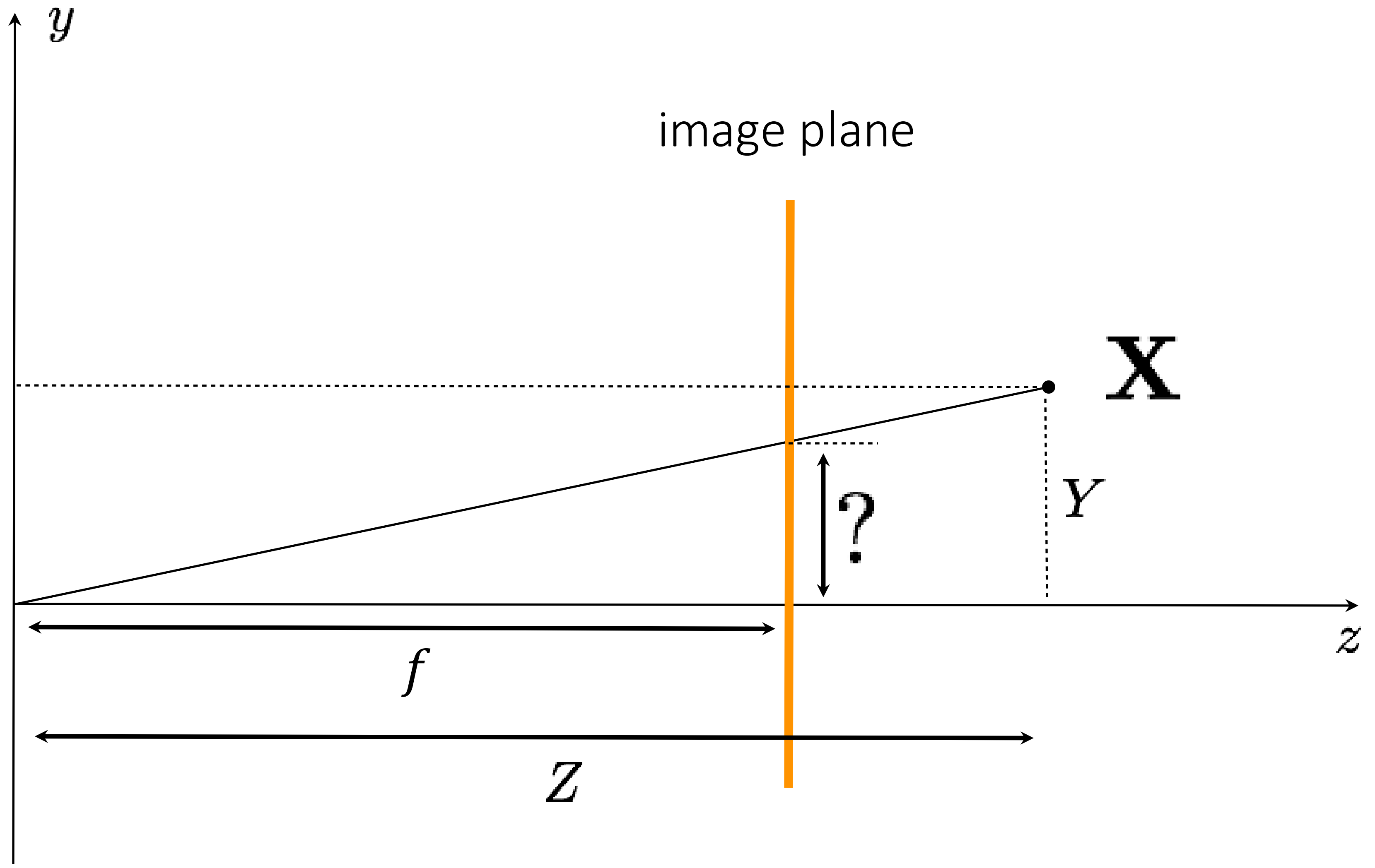
# More general case: arbitrary focal length




What is the camera matrix **P** for a pinhole camera?

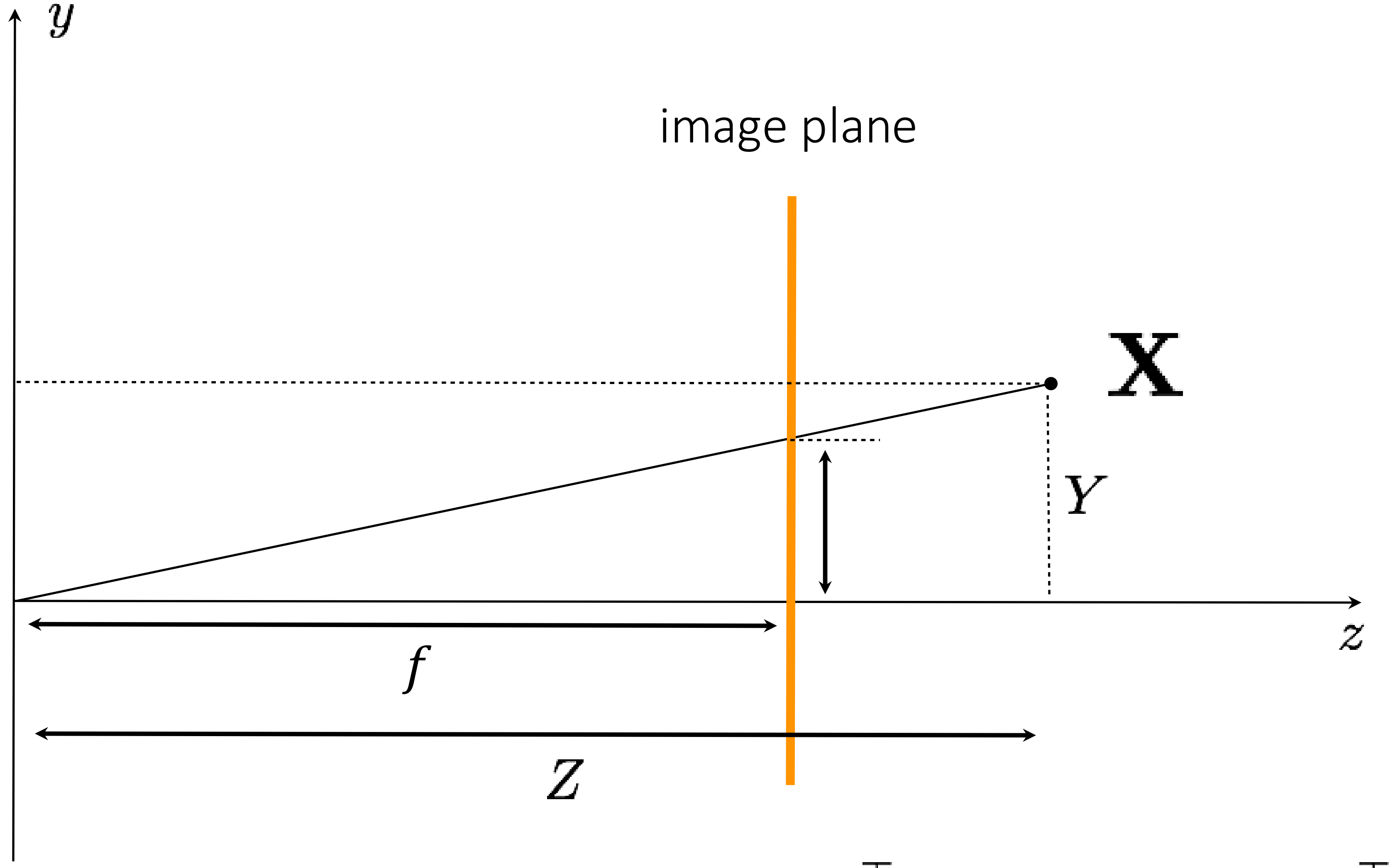
$$\boldsymbol{x} = \mathbf{P}\mathbf{X}$$

# More general (2D) case: arbitrary focal length




What is the equation for image coordinate **x** in terms of **X**?

# More general (2D) case: arbitrary focal length




$$[X \quad Y \quad Z]^\top \mapsto [fX/Z \quad fY/Z]^\top$$

# The pinhole camera matrix for arbitrary focal length

Relationship from similar triangles:

$$[X \quad Y \quad Z]^\top \mapsto [fX/Z \quad fY/Z]^\top$$

General camera model *in homogeneous coordinates*:

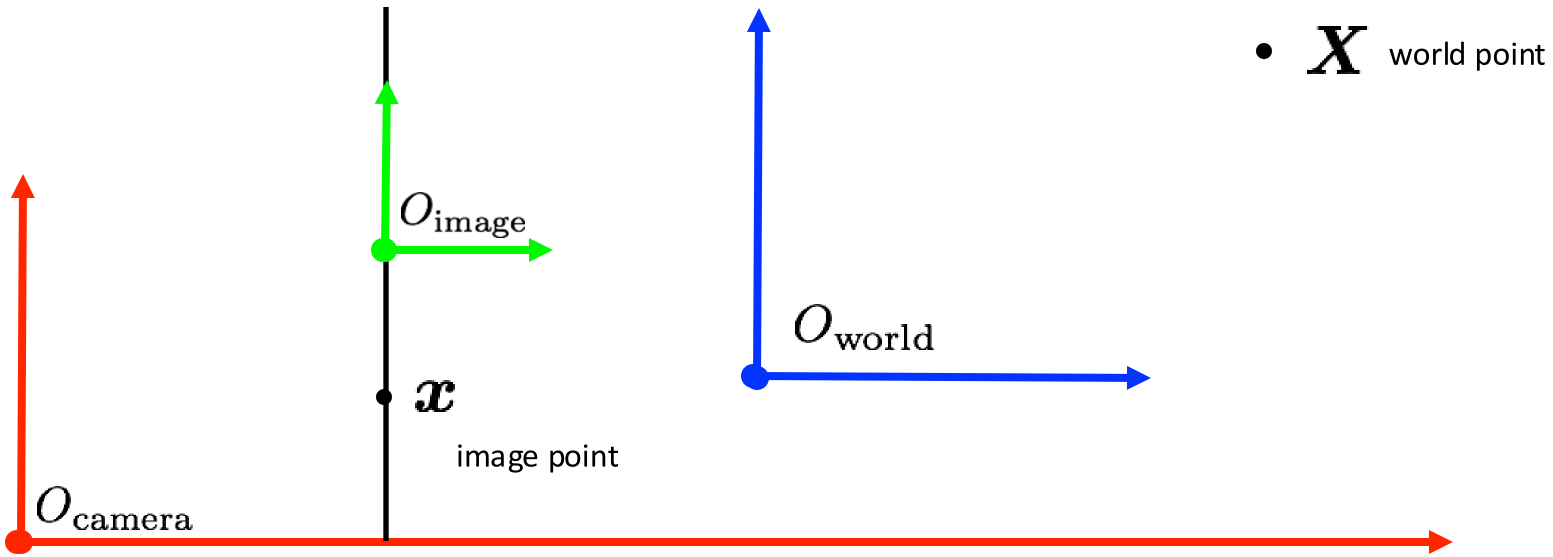
$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

What does the pinhole camera projection look like?

$$\mathbf{P} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

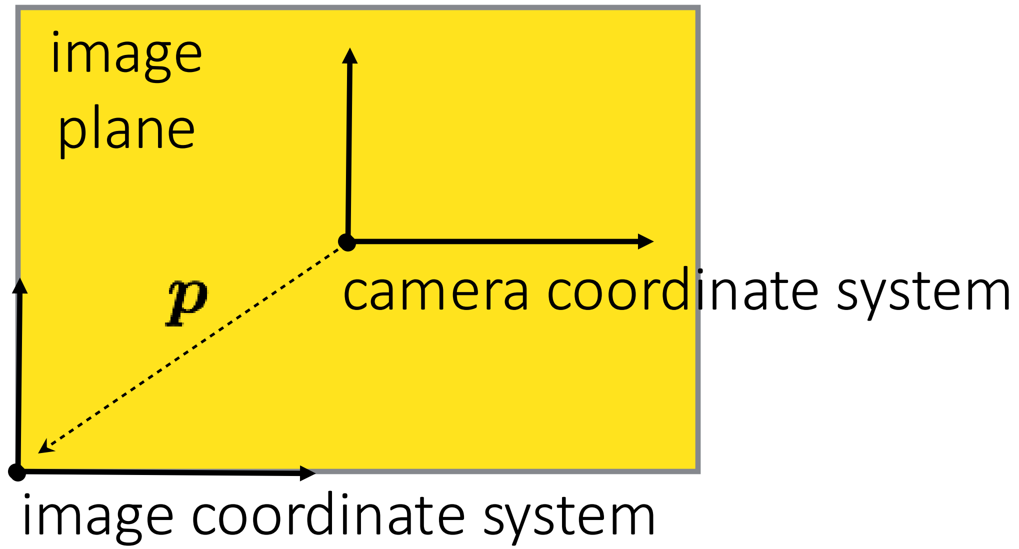
# Generalizing the camera matrix

In general, the camera and image have *different* coordinate systems.



# Generalizing the camera matrix

In particular, the camera origin and image origin may be different:

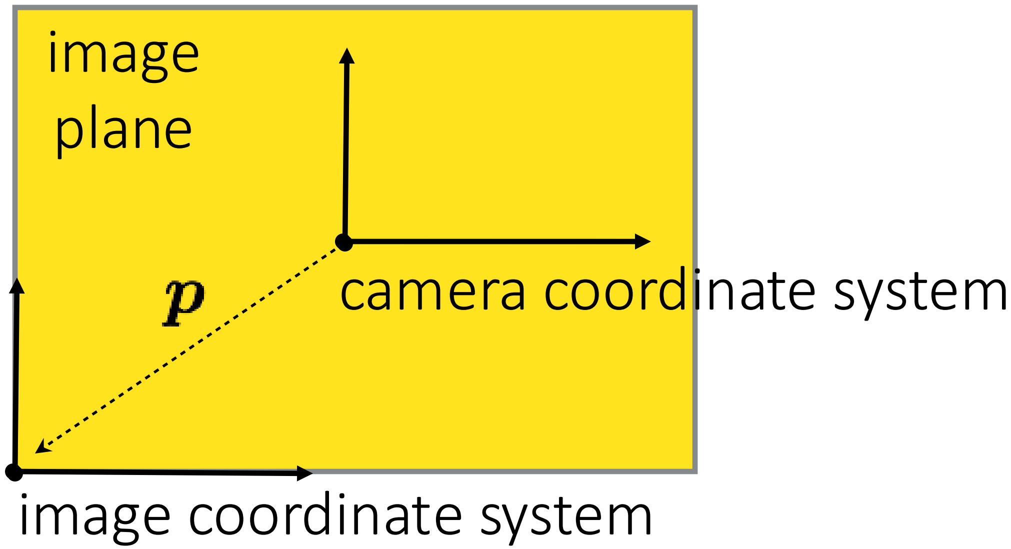



How does the camera matrix change?

$$\mathbf{P} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

# Generalizing the camera matrix

In particular, the camera origin and image origin may be different:




How does the camera matrix change?

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

shift vector transforming camera origin to image origin

# Camera matrix decomposition

We can decompose the camera matrix like this:

$$\mathbf{P} = \left[\begin{array}{ccc} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{array}\right] \left[\begin{array}{ccc:c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right]$$

What does each part of the matrix represent?

# Camera matrix decomposition

We can decompose the camera matrix like this:

$$\mathbf{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[ \begin{array}{ccc:c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array} \right]$$

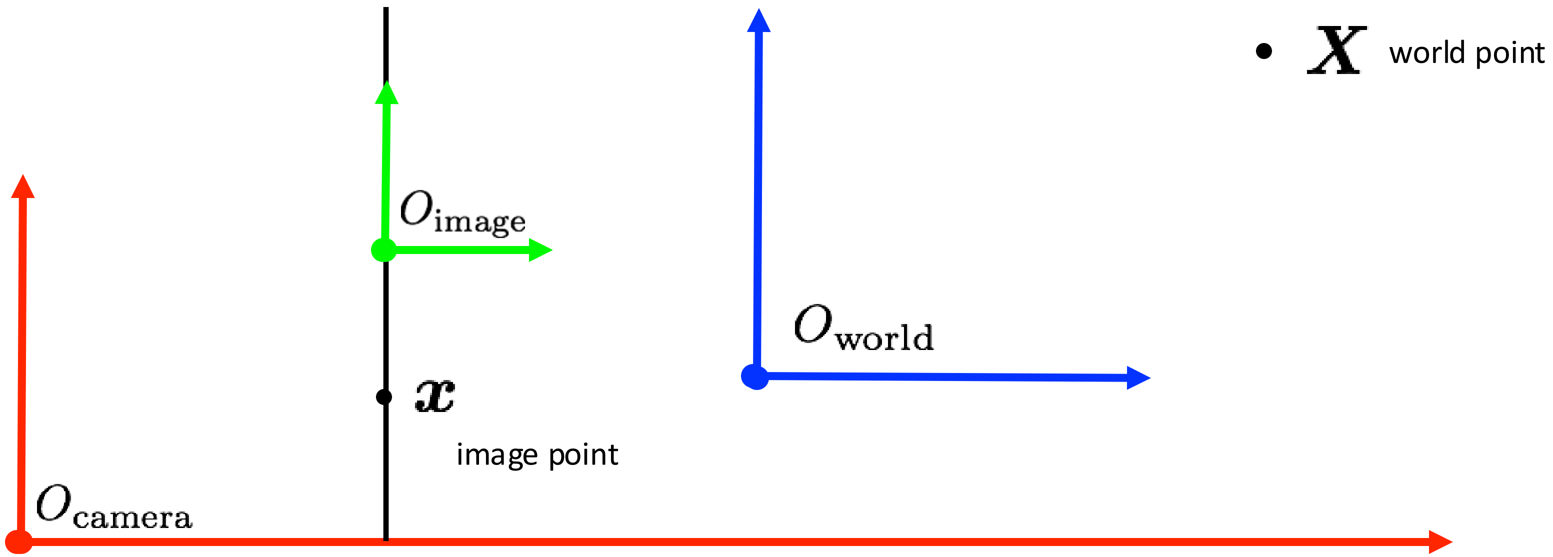
(homogeneous) transformation from 2D to 2D, accounting for not unit focal length and origin shift

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

Also written as: $\mathbf{P} = \mathbf{K}[\mathbf{I}|\mathbf{0}]$ where $\mathbf{K} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix}$

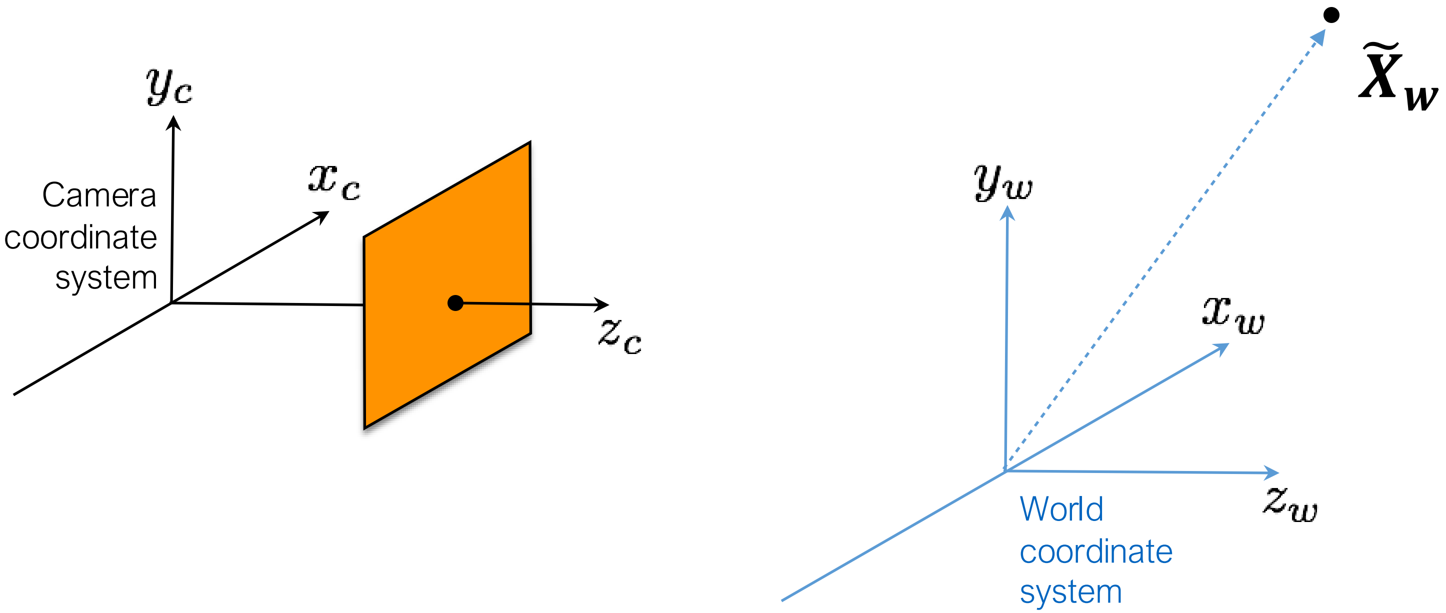
# Generalizing the camera matrix

In general, there are *three*, generally different, coordinate systems.



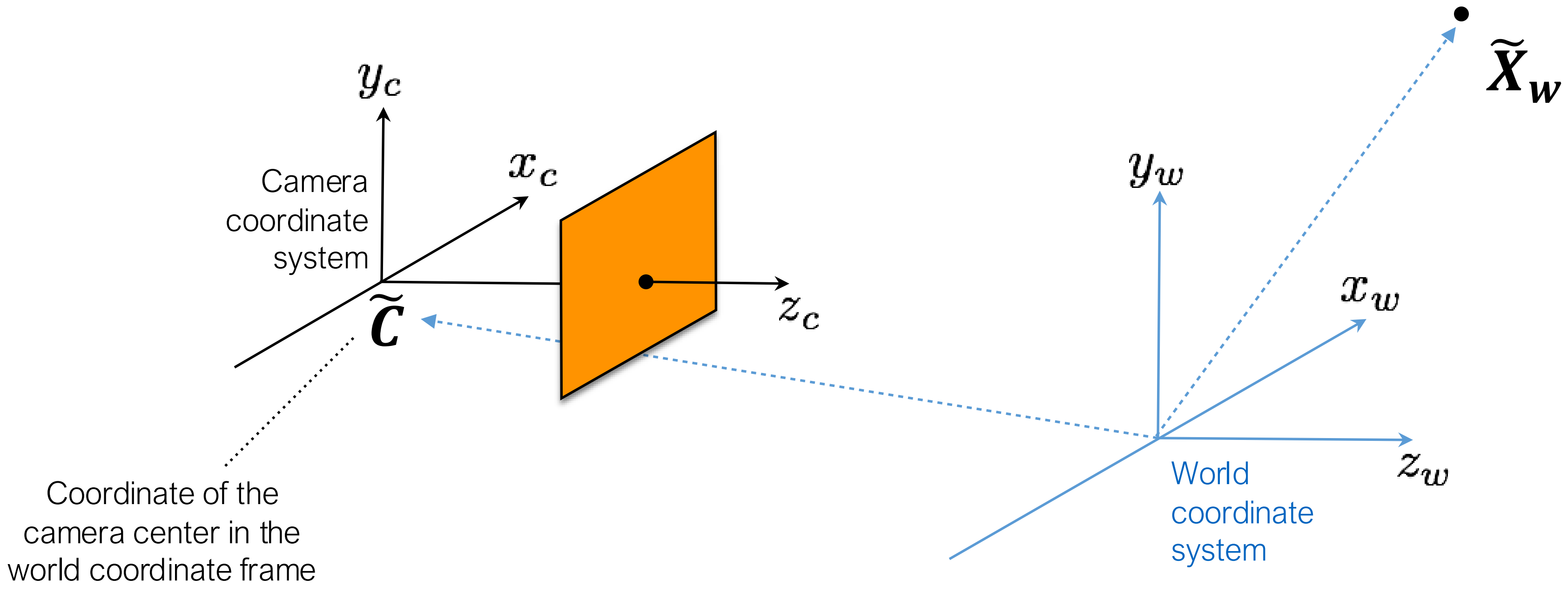

We need to know the transformations between them.

# World-to-camera coordinate system transformation

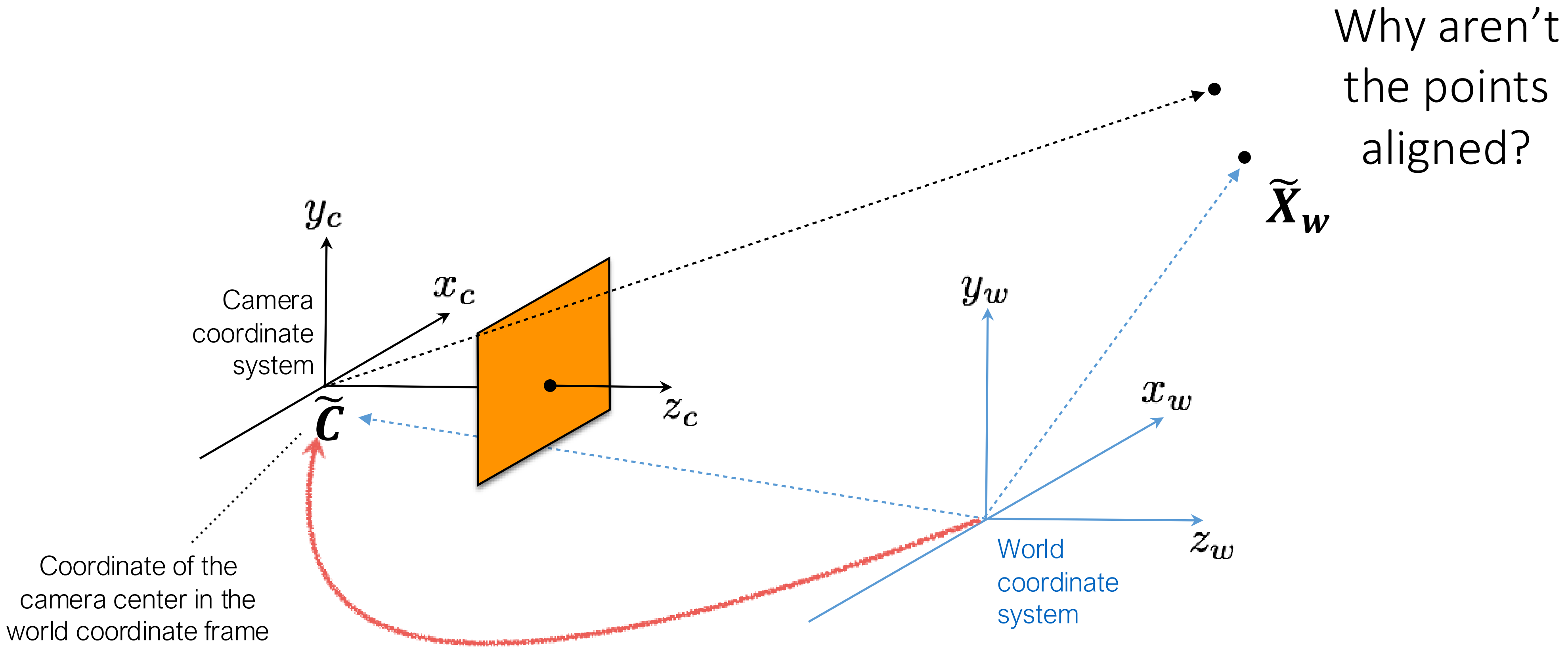



tilde means *heterogeneous* coordinates

# World-to-camera coordinate system transformation



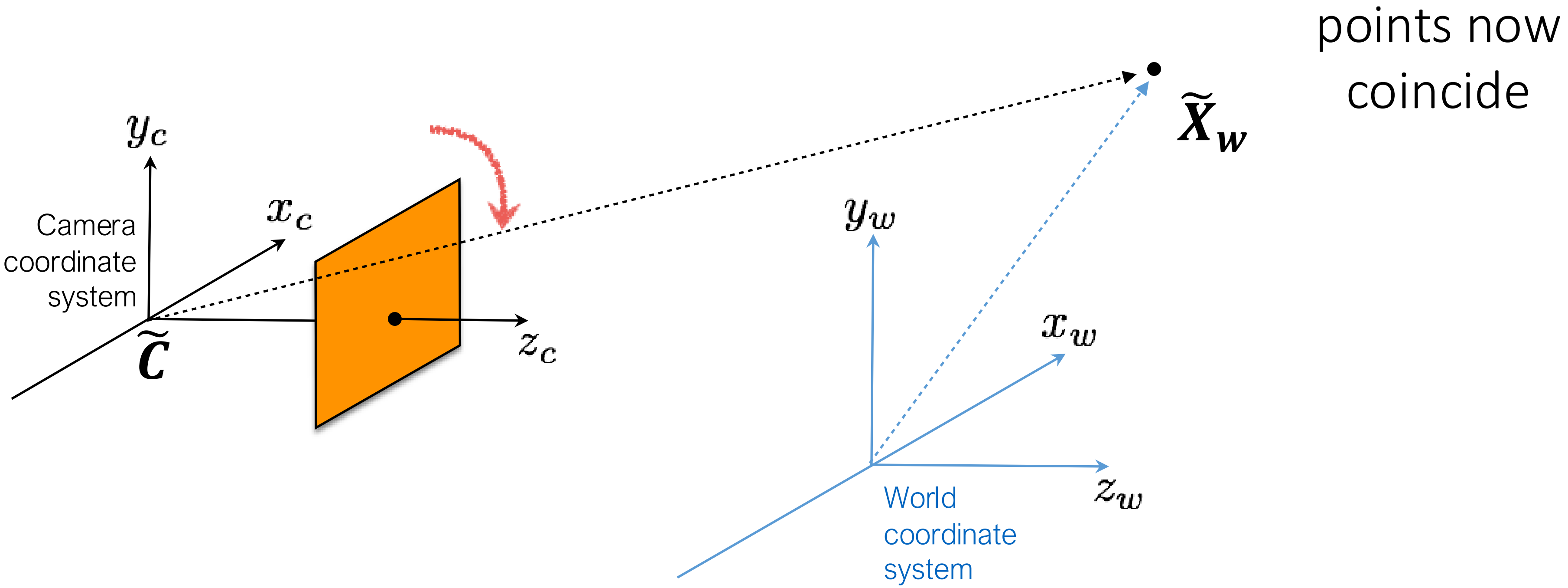
# World-to-camera coordinate system transformation




$$(\tilde{X}_w - \tilde{C})$$

translate

# World-to-camera coordinate system transformation




$$\boldsymbol{R} \cdot \left(\widetilde{\boldsymbol{X}}_{\boldsymbol{w}} - \widetilde{\boldsymbol{C}}\right)$$

rotate translate

# Modeling the coordinate system transformation

In heterogeneous coordinates, we have:

$$\widetilde{\mathbf{X}}_{\mathbf{c}} = \mathbf{R} \cdot \left(\widetilde{\mathbf{X}}_{\mathbf{w}} - \tilde{\mathbf{C}}\right)$$

How do we write this transformation in homogeneous coordinates?

# Modeling the coordinate system transformation

In heterogeneous coordinates, we have:

$$\widetilde{\mathbf{X}}_{\mathbf{c}} = \mathbf{R} \cdot \left( \widetilde{\mathbf{X}}_{\mathbf{w}} - \tilde{\mathbf{C}} \right)$$

In homogeneous coordinates, we have:

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

or

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

# Incorporating the transform in the camera matrix

The previous camera matrix is for homogeneous 3D coordinates in camera coordinate system:

$$\mathbf{x} = \mathbf{P}\mathbf{X_c} = \mathbf{K}[\mathbf{I}|\mathbf{0}]\mathbf{X_c}$$

We also just derived:

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

# Putting it all together

We can write everything into a single projection:

$$\mathbf{x} = \mathbf{P}\mathbf{X}_\mathbf{w}$$

The camera matrix now looks like:

$$\mathbf{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} [\mathbf{I} \quad | \quad \mathbf{0}] \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1 \end{bmatrix}$$

*intrinsic parameters* (3 x 3): correspond to camera internals (image-to-image transformation)

*perspective projection* (3 x 4): maps 3D to 2D points (camera-to-image transformation)

*extrinsic parameters* (4 x 4): correspond to camera externals (world-to-camera transformation)

# Putting it all together

We can write everything into a single projection:

$$\mathbf{x} = \mathbf{P}\mathbf{X}_{\mathbf{w}}$$

The camera matrix now looks like:

$$\mathbf{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[ \mathbf{R} \;\middle|\; -\mathbf{RC} \right]$$

*intrinsic parameters* (3 x 3):
correspond to camera internals
(sensor not at f = 1 and origin shift)

*extrinsic parameters* (3 x 4):
correspond to camera externals
(world-to-image transformation)

# General pinhole camera matrix

We can decompose the camera matrix like this:

$$\mathbf{P} = \mathbf{KR}[\mathbf{I}| - \mathbf{C}]$$

(translate first then rotate)

Another way to write the mapping:

$$\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}]$$

where $\mathbf{t} = -\mathbf{RC}$

(rotate first then translate)

# General pinhole camera matrix

$$\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}]$$

$$\mathbf{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc:c} r_1 & r_2 & r_3 & t_1 \\ r_4 & r_5 & r_6 & t_2 \\ r_7 & r_8 & r_9 & t_3 \end{array}\right]$$

intrinsic parameters

extrinsic parameters

$$\mathbf{R} = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} \qquad \mathbf{t} = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}$$

3D rotation

3D translation

# Recap

What is the size and meaning of each term in the camera matrix?

$$\mathbf{P} = \mathbf{K}\mathbf{R}[\mathbf{I}| - \mathbf{C}]$$

? ? ? ?

# Recap

What is the size and meaning of each term in the camera matrix?

$$\mathbf{P} = \mathbf{K}\mathbf{R}[\mathbf{I}| - \mathbf{C}]$$

3x3
intrinsics

?

?

?

# Recap

What is the size and meaning of each term in the camera matrix?

$$\mathbf{P} = \mathbf{K}\mathbf{R}[\mathbf{I}| - \mathbf{C}]$$

3x3
intrinsics

3x3
3D rotation

?

?

# Recap

What is the size and meaning of each term in the camera matrix?

$$\mathbf{P} = \mathbf{K}\mathbf{R}[\mathbf{I}| - \mathbf{C}]$$

3x3
intrinsics

3x3
3D rotation

3x3
identity

?

# Recap

What is the size and meaning of each term in the camera matrix?

$$\mathbf{P} = \mathbf{K}\mathbf{R}[\mathbf{I}| - \mathbf{C}]$$

| | | | |
|---|---|---|---|
| 3x3<br>intrinsics | 3x3<br>3D rotation | 3x3<br>identity | 3x1<br>3D translation |

# Quiz

The camera matrix relates what two quantities?

# Quiz

The camera matrix relates what two quantities?

$$\boldsymbol{x} = \mathbf{P}\mathbf{X}$$

homogeneous 3D points to 2D image points

# Quiz

The camera matrix relates what two quantities?

$$\boldsymbol{x} = \mathbf{P}\mathbf{X}$$

homogeneous 3D points to 2D image points

The camera matrix can be decomposed into?

# Quiz

The camera matrix relates what two quantities?

$$\boldsymbol{x} = \mathbf{P}\mathbf{X}$$

homogeneous 3D points to 2D image points

The camera matrix can be decomposed into?

$$\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}]$$

intrinsic and extrinsic parameters

# More general camera matrices

The following is the standard camera matrix we saw.

$$\mathbf{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \quad \left[\mathbf{R} \;\middle|\; -\mathbf{RC}\right]$$

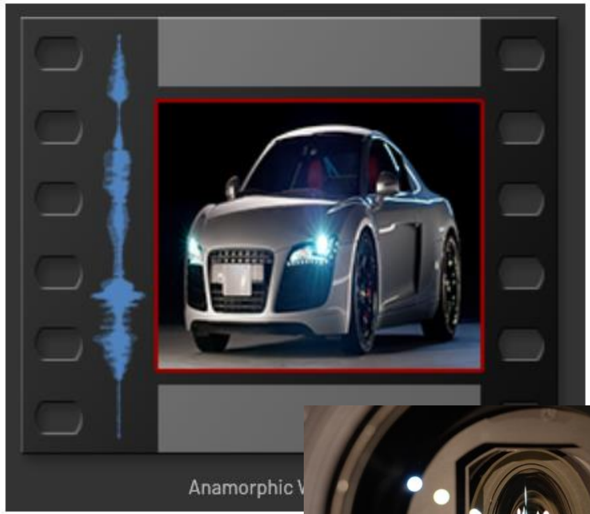
# More general camera matrices

CCD camera: pixels may not be square.
Lens: focal length along x and y may be different

$$\mathbf{P} = \begin{bmatrix} \alpha_x & 0 & p_x \\ 0 & \alpha_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[ \mathbf{R} \;\vdots\; -\mathbf{RC} \right]$$

How many degrees of freedom?

# Anamorphic Lens


Lens flares are scattered light that you usually try to avoid having in photos. It's usually caused when a bright light shines into the lens. Paramount Pictures

Abrams loves them. Paramount Pictures

You can see them across his rebooted "Star Trek" movies. Paramount Pictures

# More general camera matrices

CCD camera: pixels may not be square.
Lens: focal length along x and y may be different

$$\mathbf{P} = \begin{bmatrix} \alpha_x & 0 & p_x \\ 0 & \alpha_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[ \mathbf{R} \mid -\mathbf{RC} \right]$$

How many degrees of freedom?

10 DOF

# More general camera matrices

Finite projective camera: sensor be skewed.

$$\mathbf{P} = \begin{bmatrix} \alpha_x & s & p_x \\ 0 & \alpha_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[ \mathbf{R} \;\vdots\; -\mathbf{RC} \right]$$

How many degrees of freedom?

# More general camera matrices

Finite projective camera: sensor be skewed.

$$\mathbf{P} = \begin{bmatrix} \alpha_x & s & p_x \\ 0 & \alpha_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[ \mathbf{R} \;\vdots\; -\mathbf{RC} \right]$$

How many degrees of freedom?

11 DOF