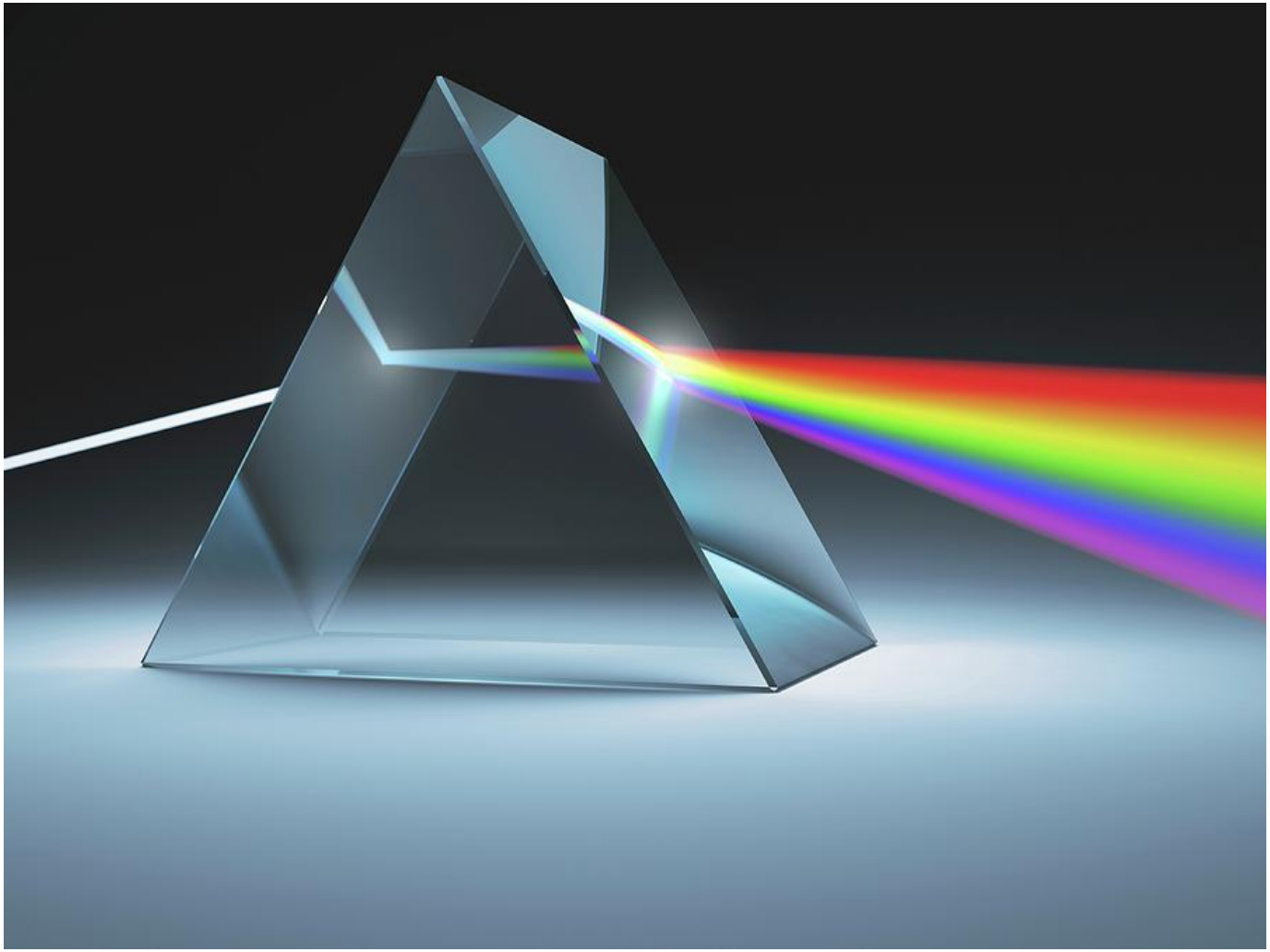
Image formation



References

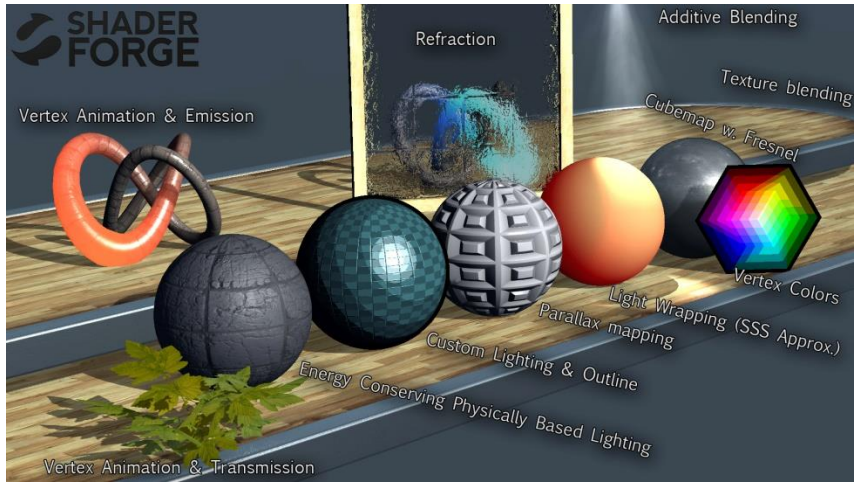
- http://szeliski.org/Book/
- http://www.cs.cornell.edu/courses/cs5670/2019sp/lectures/lectures.html
- http://www.cs.cmu.edu/~16385/

Contents

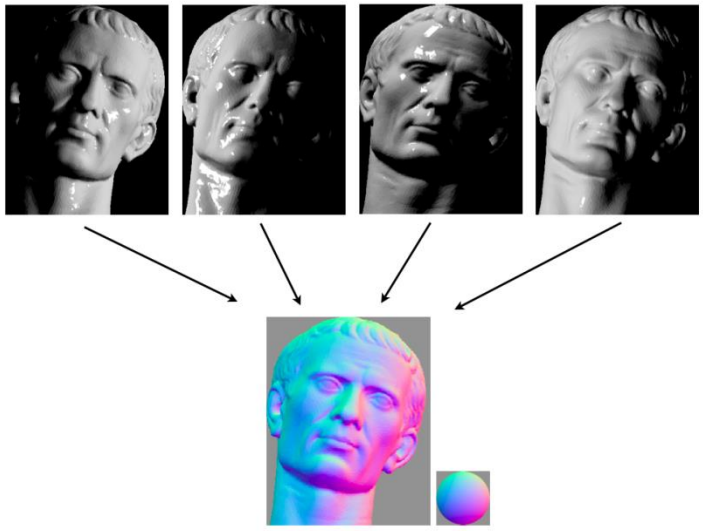
- BRDF
- Pinhole camera
- Digital camera
- The human eye
- Image transformations

Some motivation

Art
(Exposure time)



3D game rendering
(ray tracing)

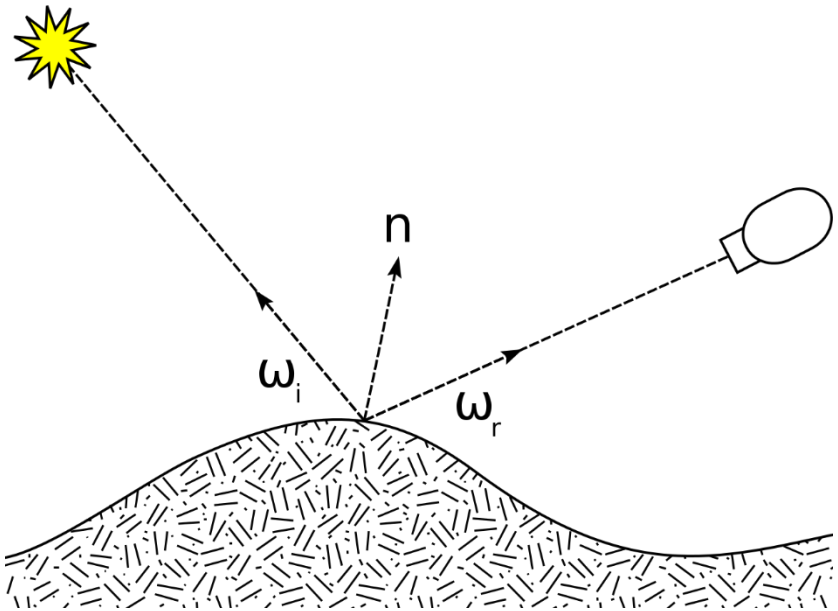


Computational photography
(shape from shading- 3D reconstruction)

Computational photography
(aperture blur shape)

Image formation

- Let's talk about what happens before the image is captured- how it's captured and why do we see it the way the we see it.

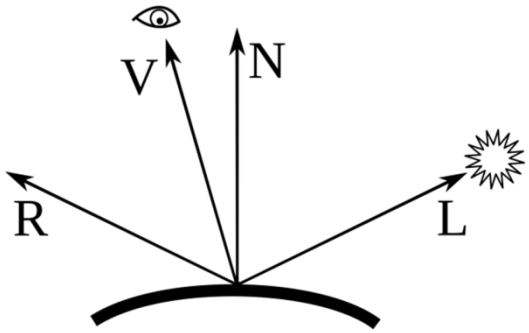


Contents

- **BRDF**
- Pinhole camera
- Digital camera
- The human eye
- Image transformations

BRDF

- **BRDF** or **bidirectional reflectance distribution function** is basically a ratio of output reflected rays' power (I_r) to input rays' power from a light source (I_i).
- The BRDF is a function of the surface which is dependent on the input and output rays' direction relative to the surface normal: $\frac{\mathrm{I_r}}{\mathrm{I_i}} = f_{\hat{N}}(\hat{L}, \hat{V})$
  - [The complete (and more correct) definition is out of scope and can be found here: http://vision.ime.usp.br/~acmt/hakyll/assets/files/wynn.pdf]



BRDF

- BRDF is used extensively in photorealistic rendering, but also in 3D reconstruction and image deblurring.
- When rendering 3D images with ray tracing, actual rays are generated and propagate through the scene using the BRDF of each material.

BRDF of a diffused surface

- A perfectly diffused surface is a surface that light is reflected off it equally in all directions. The light intensity scattered off the surface is only determined by the input ray direction relative to the surface norm:

$$\frac{\mathrm{I_r}}{\mathrm{I_i}} = \kappa_d\left(\hat{L}\cdot\hat{N}\right)$$

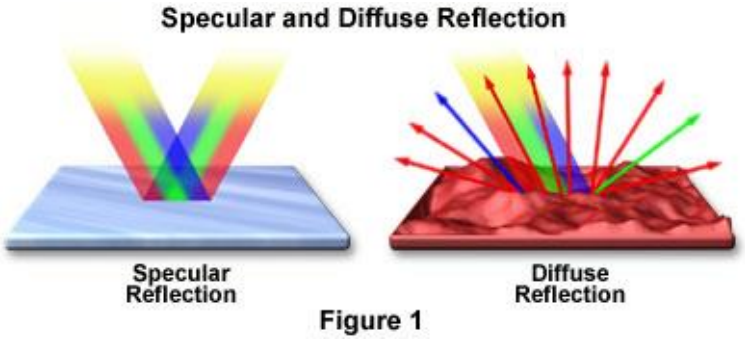
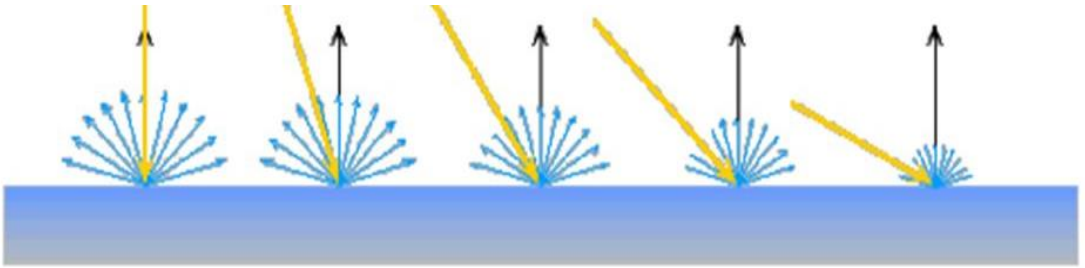


Figure 1



BRDF of a diffused surface

$$\frac{\mathrm{I_r}}{\mathrm{I_i}} = \kappa_d\left(\hat{L}\cdot\hat{N}\right)$$

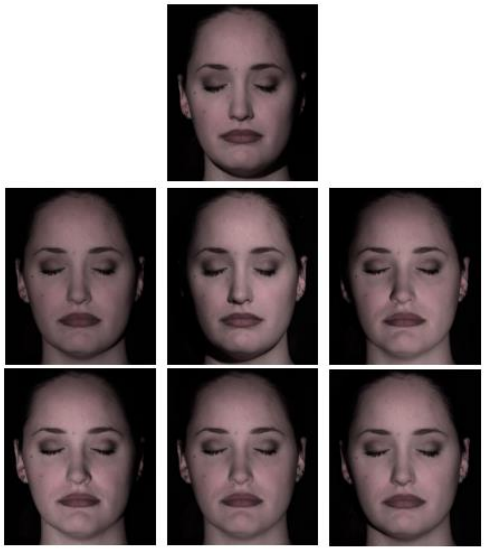
- κ_d is the **diffuse reflectance coefficient** (sometimes named incorrectly **diffuse albedo**)- indicates the proportion of light reflected diffusely to the total input light intensity (some light is absorbed). $\kappa_d \epsilon [0,1]$
- This is known as also **Lambertian reflectance**.

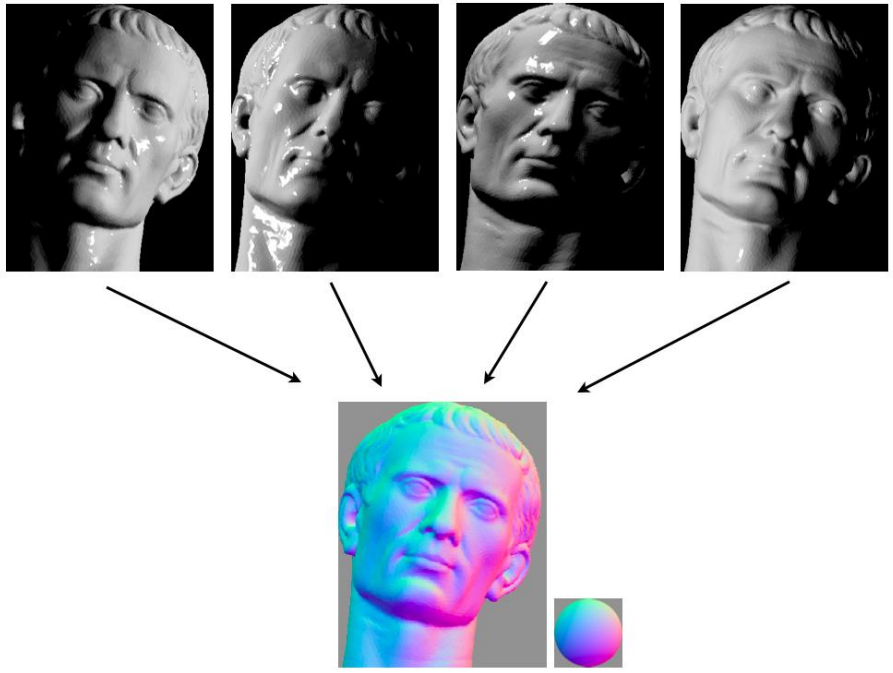
BRDF of a diffused surface

- Diffused surface demo: http://learnwebgl.brown37.net/09_lights/lights_diffuse.html

Side note: photometric stereo

- Estimating surface normals from 2D images of different lighting conditions.
- Assumes Lambertian surface.





Side note: photometric stereo

- Let's say we have m images of the same Lambertian object with different lighting scenarios:
  - $I_{m\ x\ 1}$: m different intensities of the same pixel- known.
  - $L_{m\ x\ 3}$: m different light vector directions - known.
  - $N_{3\ x\ 1}$: the pixel's normal – unknown (2 DOFs).
  - $\kappa_{d_{1\ x\ 1}}$: diffuse reflectance coefficient- unknown (1 DOF).

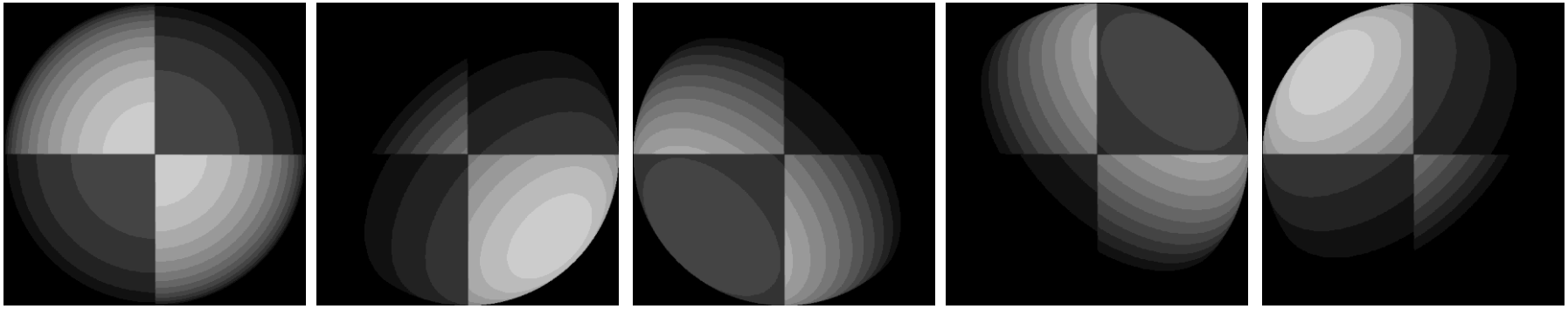
$$I = \kappa_d LN = LG$$

  - Where G is 3 DOFs.
- From here, a simple psaudoinverse (thought in least squares class) can be applied the get N:

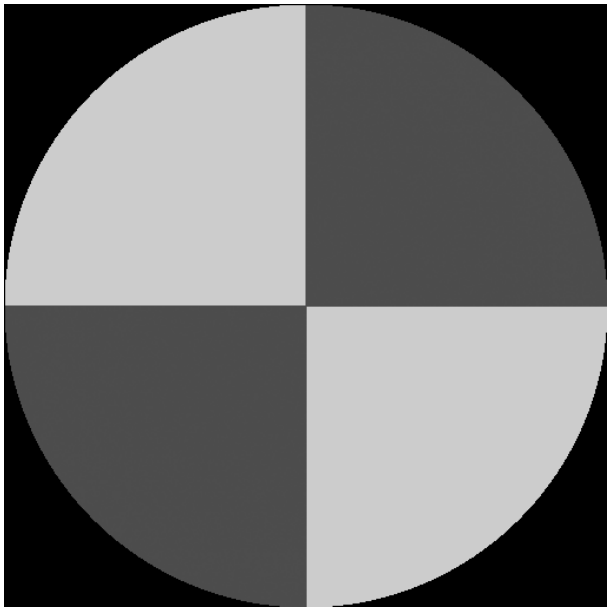
$$(L^T L)^{-1} L^T I = G$$

$$||G|| = \kappa_d, \frac{G}{||G||} = N$$

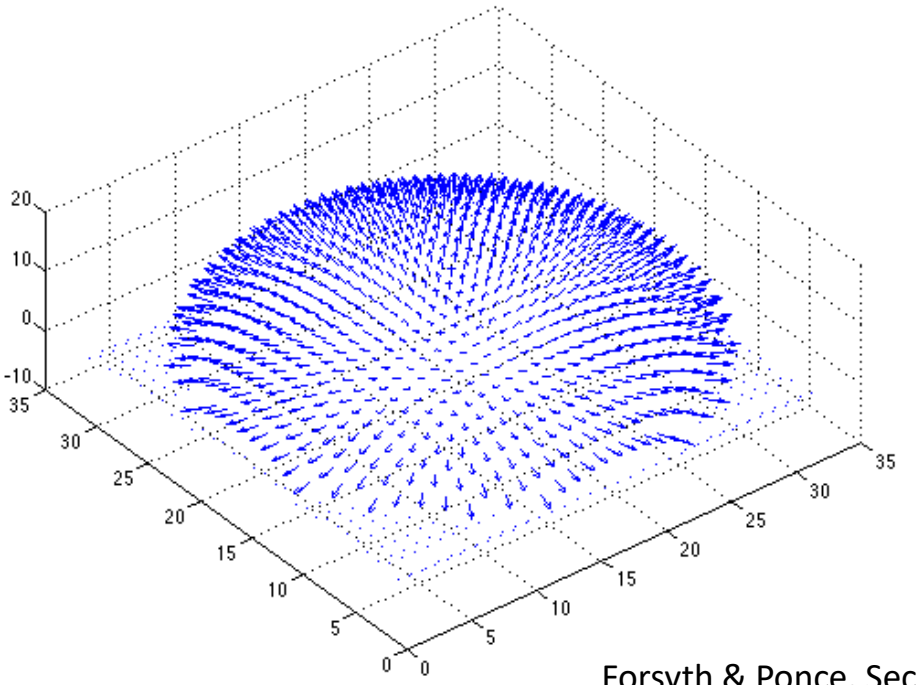
Side note: photometric stereo



Recovered albedo



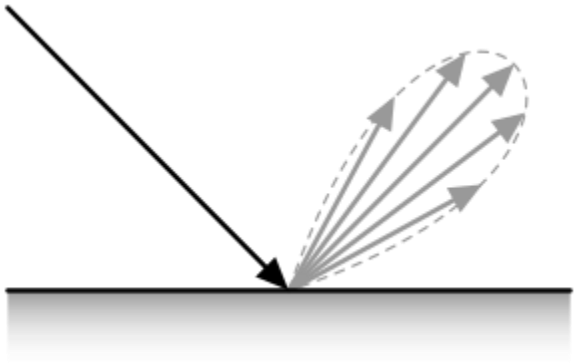
Recovered normal field

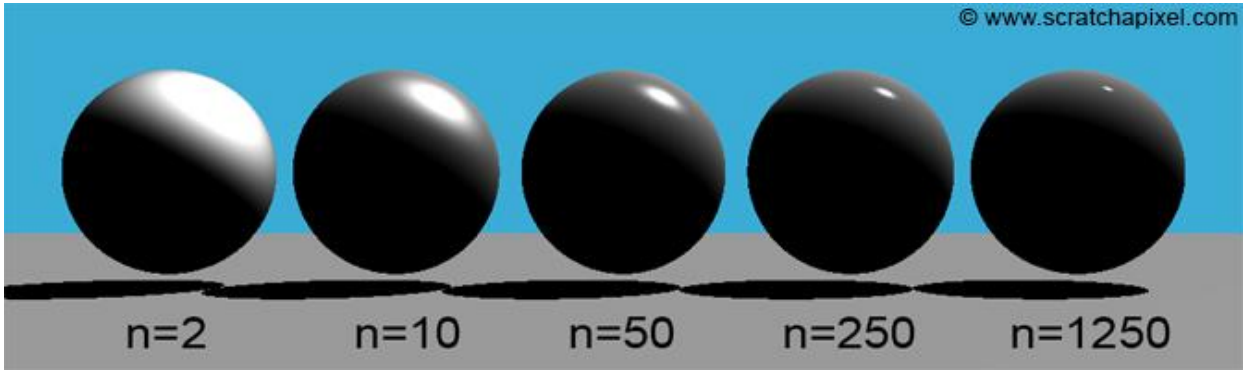


Forsyth & Ponce, Sec. 5.4

BRDF of a specular surface

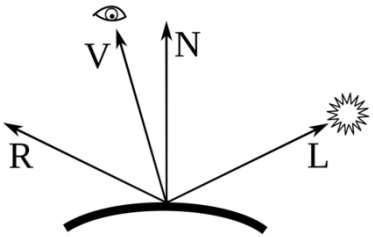
- A specular surface is one that reflected light is returned mainly on the mirrored reflected vector $\hat{R}$.
  - A mirror is a kind of specular surface where the ray is reflected only in the $\hat{R}$ Direction.





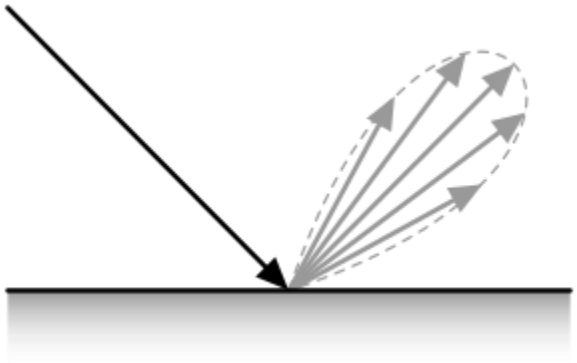
BRDF of a specular surface

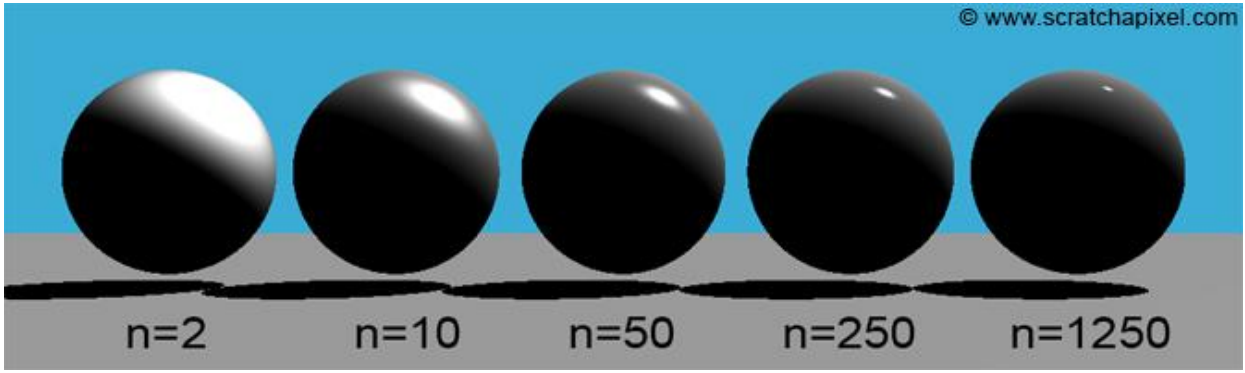
- A common model of a specular surface is:



$$\frac{\mathrm{I_r}}{\mathrm{I_i}} = \kappa_s(\hat{R} \cdot \hat{V})^n$$

  - n is a **shininess** constant for this material, which is larger for surfaces that are smoother and more mirror-like.
  - κ_s is the specular reflectance coefficient. $\kappa_s \epsilon [0,1]$

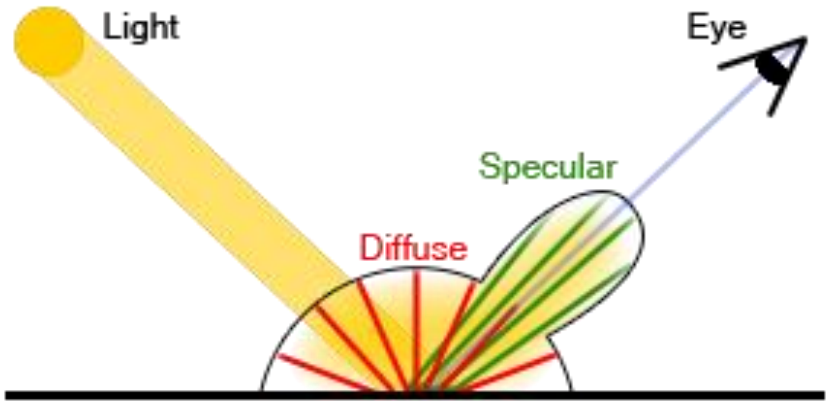




Diffused and specular surface

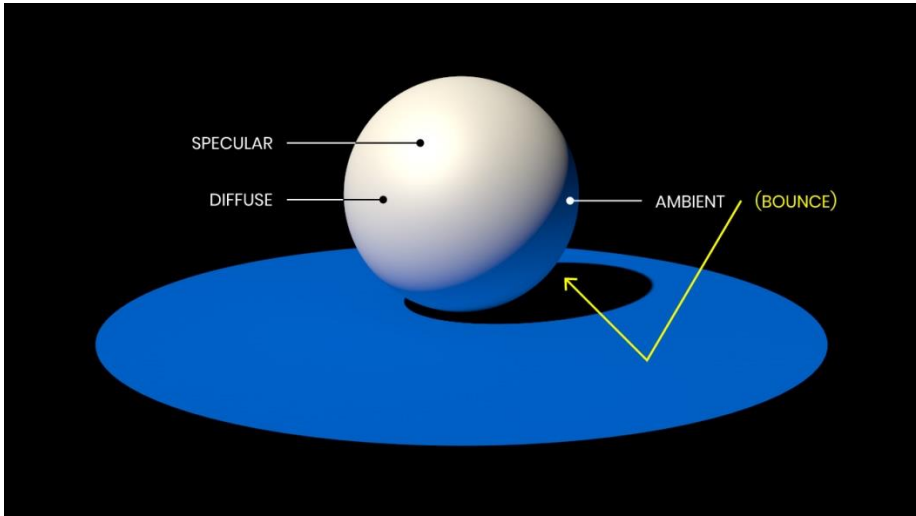
- Surfaces can combine both features with some kind of weighting such as:

$$\frac{\mathrm{I_r}}{\mathrm{I_i}} = \kappa_d\left(\hat{L}\cdot\hat{N}\right) + \kappa_s(\hat{R}\cdot\hat{V})^n$$



Ambient light

- In some rendering techniques reflection of light only as one that directly comes from the light source.
- It does not compute secondary reflection of light that is being reflected at one object and indirectly illuminates another object.
- To account for secondary reflection, a general light I_a is added that distributes homogeneously through the entire 3D scene, which is called **ambient light**

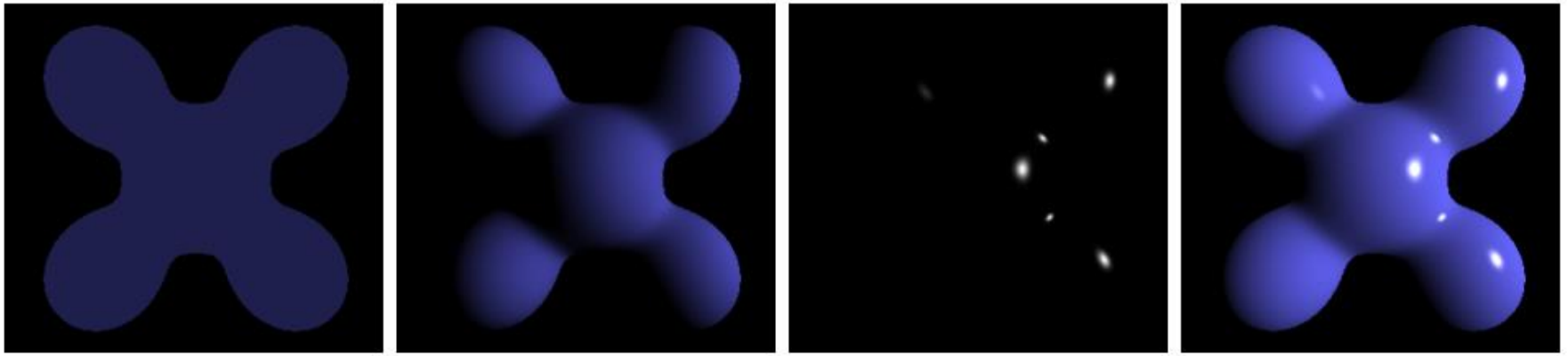


Phong reflection model

- Bui Tuong Phong used ambient, diffused and specular reflections in a well known illumination model known as **Phong reflection model:**

$$I_{\mathrm{r}} = k_{\mathrm{a}} I_{\mathrm{a}} + \sum_{m \in \text{lights}} \left(k_{\mathrm{d}} (\hat{L}_m \cdot \hat{N}) I_m + k_{\mathrm{s}} (\hat{R}_m \cdot \hat{V})^n I_m \right)$$

- Demo: http://www.cs.toronto.edu/~jacobson/phong-demo/

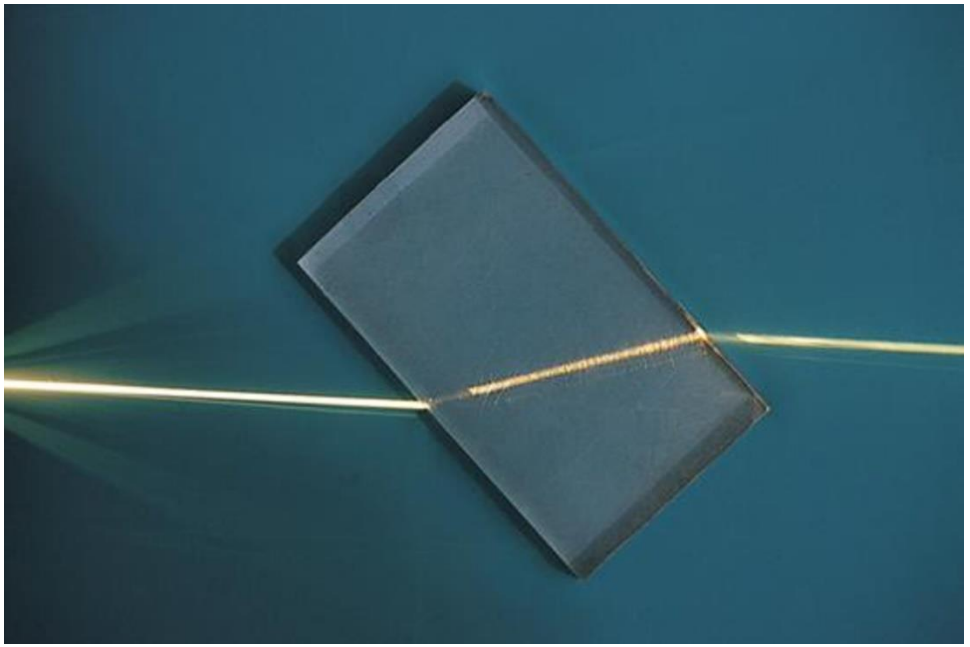


Phong model limitations

- Phong model doesn’t account for refraction and Interreflections.

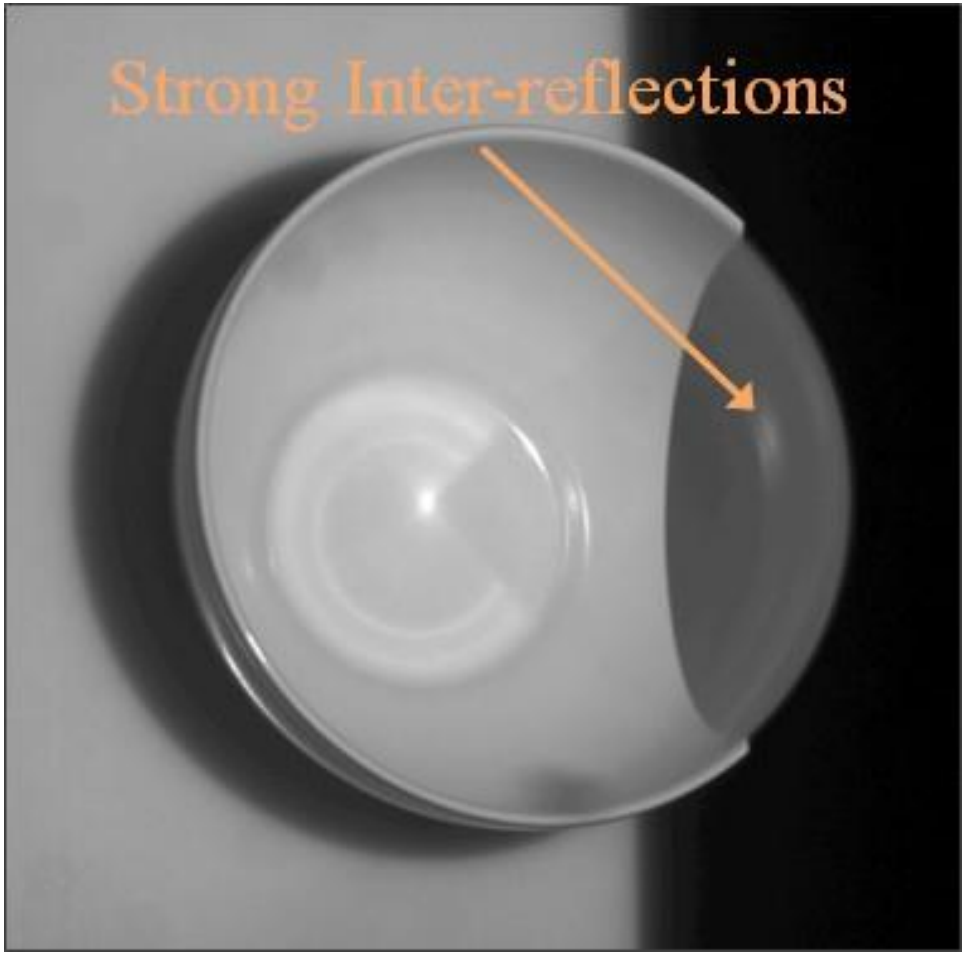
Phong model limitations

- **Refraction** is the change in direction of a wave passing from one medium to another or from a gradual change in the medium.



Phong model limitations

- **interreflection** - reciprocal reflection between two reflecting surfaces



BRDF limitations

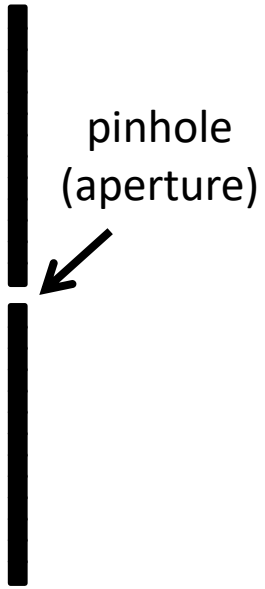
- BRDF assumes single point of ray entering and single point for existing. It doesn't account for light rays that scatters through a medium
- **Subsurface scattering** is a mechanism of light transport in which light that penetrates the surface of a translucent object, is scattered by interacting with the material, and exits the surface at a different point.

Contents

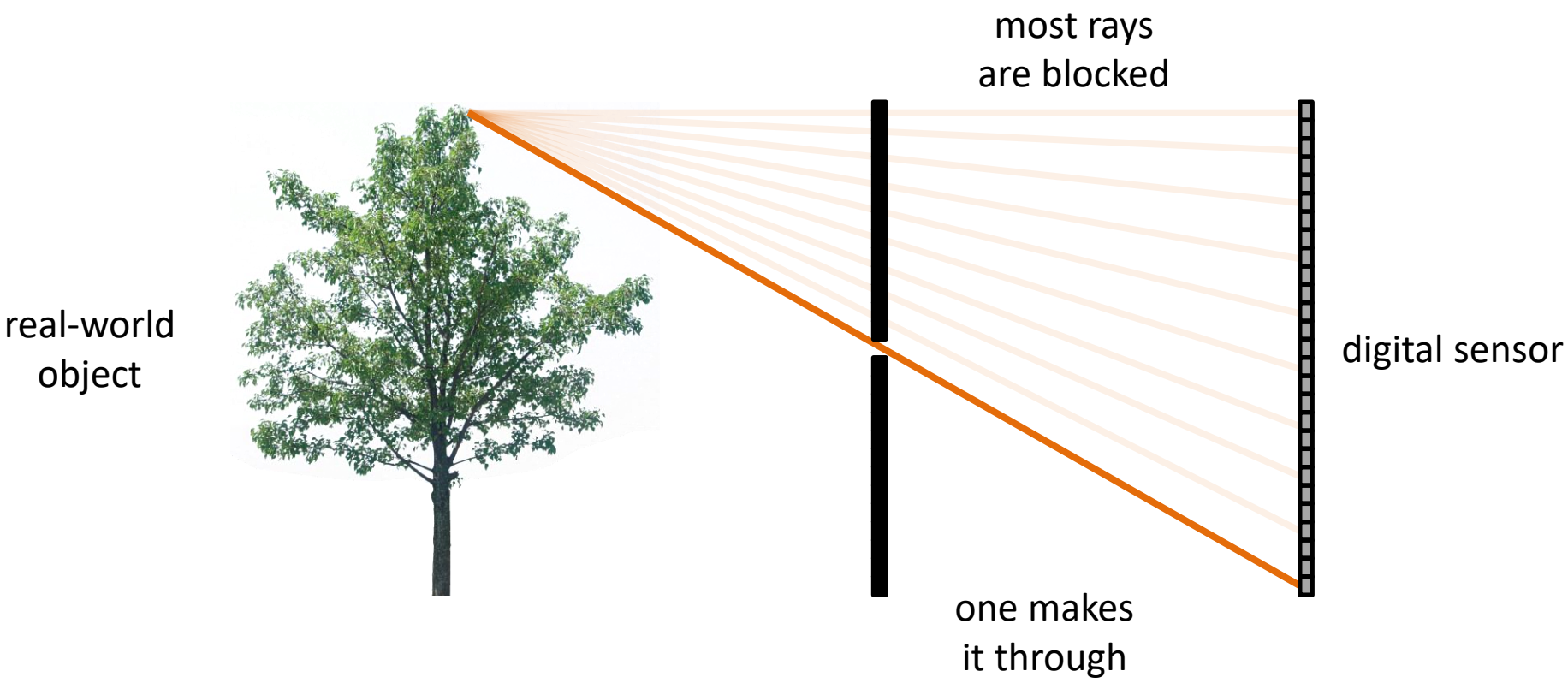
- BRDF
- **Pinhole camera**
- Digital camera
- The human eye
- Image transformations

Pinhole imaging

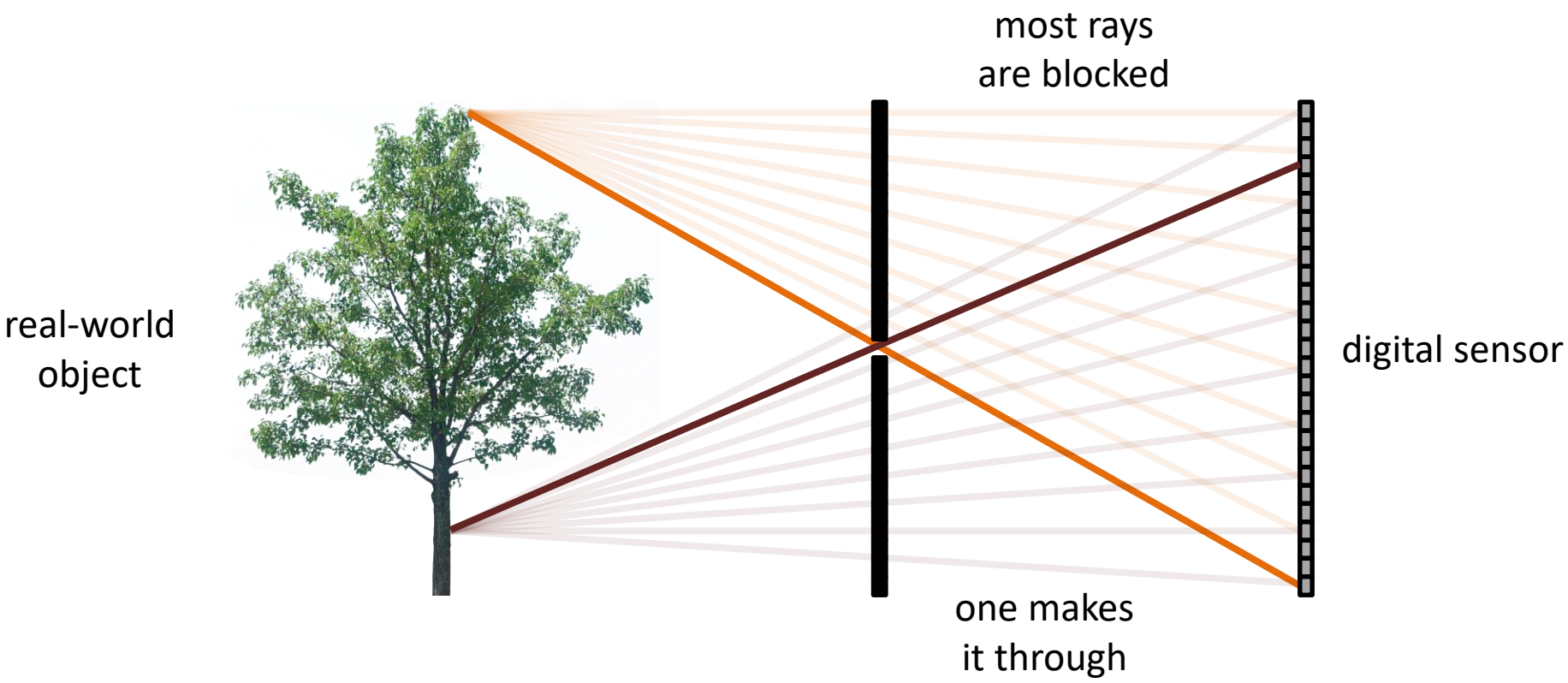
- What would an image taken like this look like?



Pinhole imaging

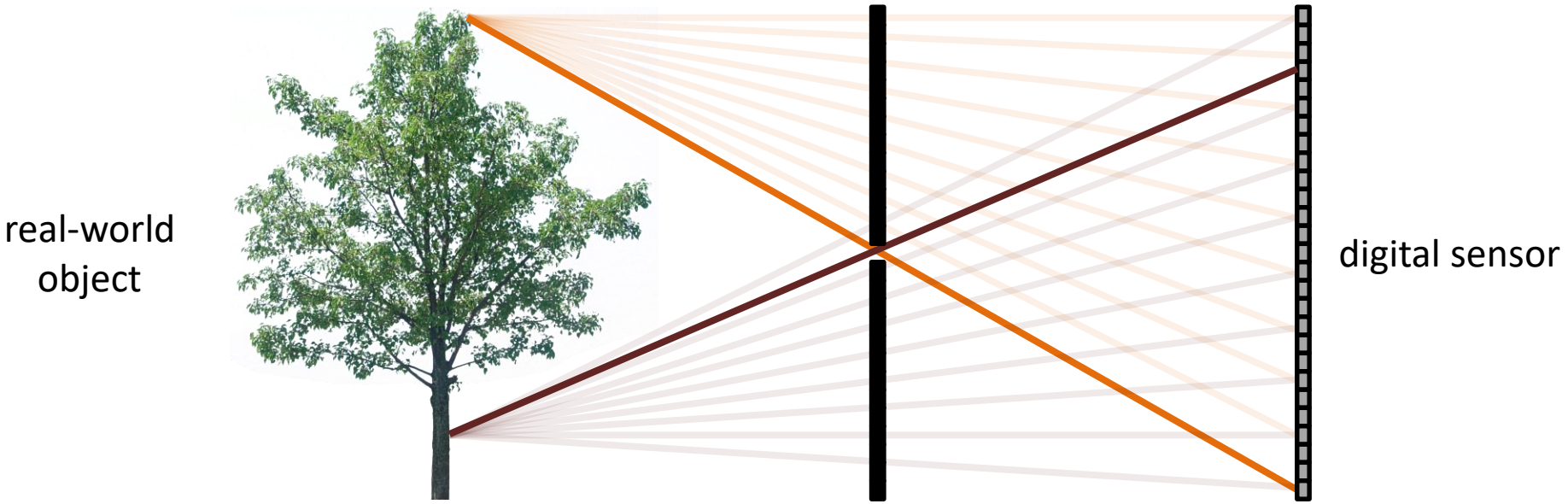


Pinhole imaging



Pinhole imaging

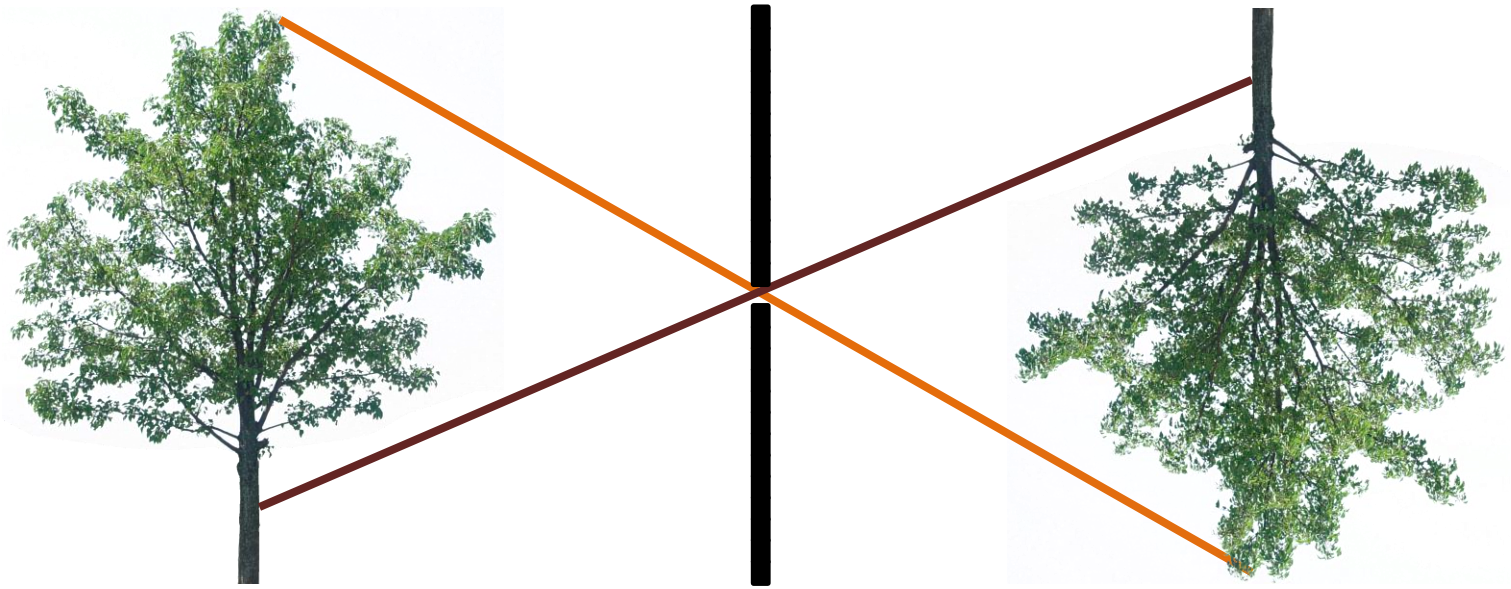
- Each scene point contributes to **only one** sensor pixel.



Pinhole imaging

- Imaged object is inverted.

real-world object



Blur in pinhole cameras

- Since only one ray reaches every pixel in the pinhole camera, the image is always in focus.
  - (blur can come only from movement of camera or objects in time of exposure)

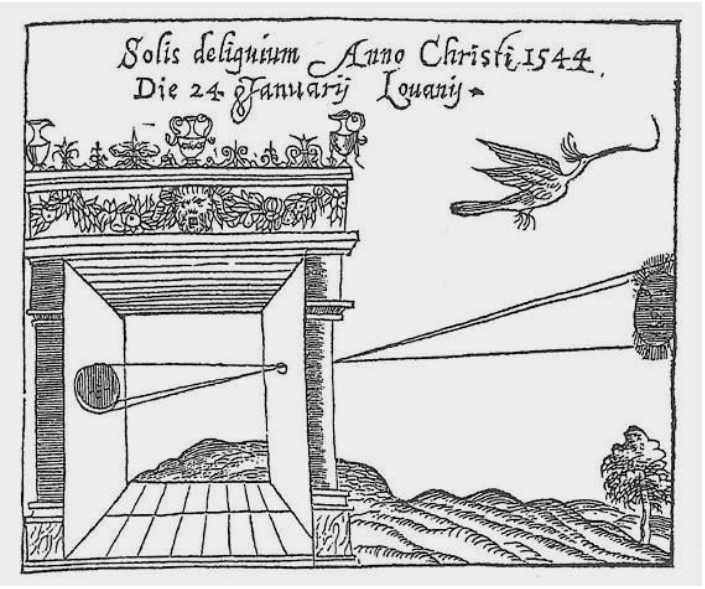
Pinhole camera a.k.a. camera obscura

- **Camera obscura** - from Latin, meaning "dark room“.

First mention …

Chinese philosopher Mozi
(470 to 390 BC)

First camera …



Greek philosopher Aristotle
(384 to 322 BC)

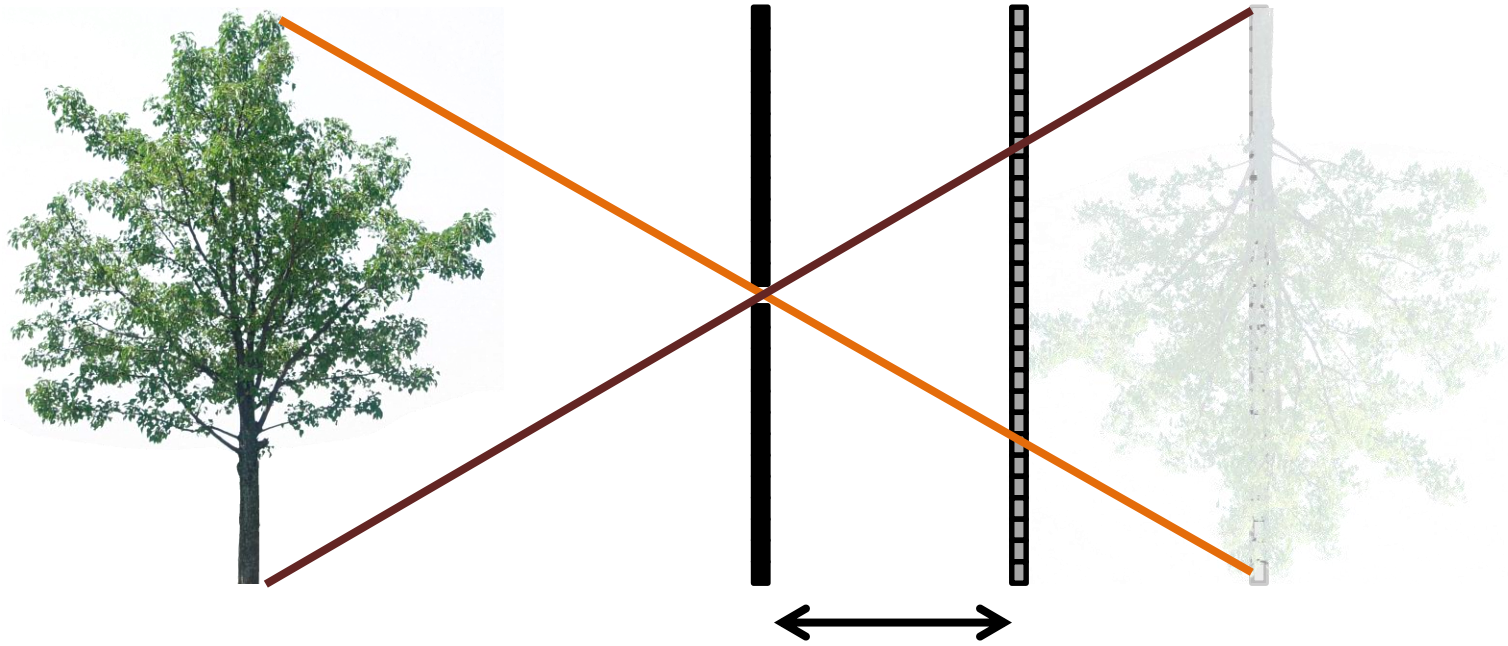
Examples of camera obscura

Examples of camera obscura

Sensor place

- What happens as we change the place of the sensor?

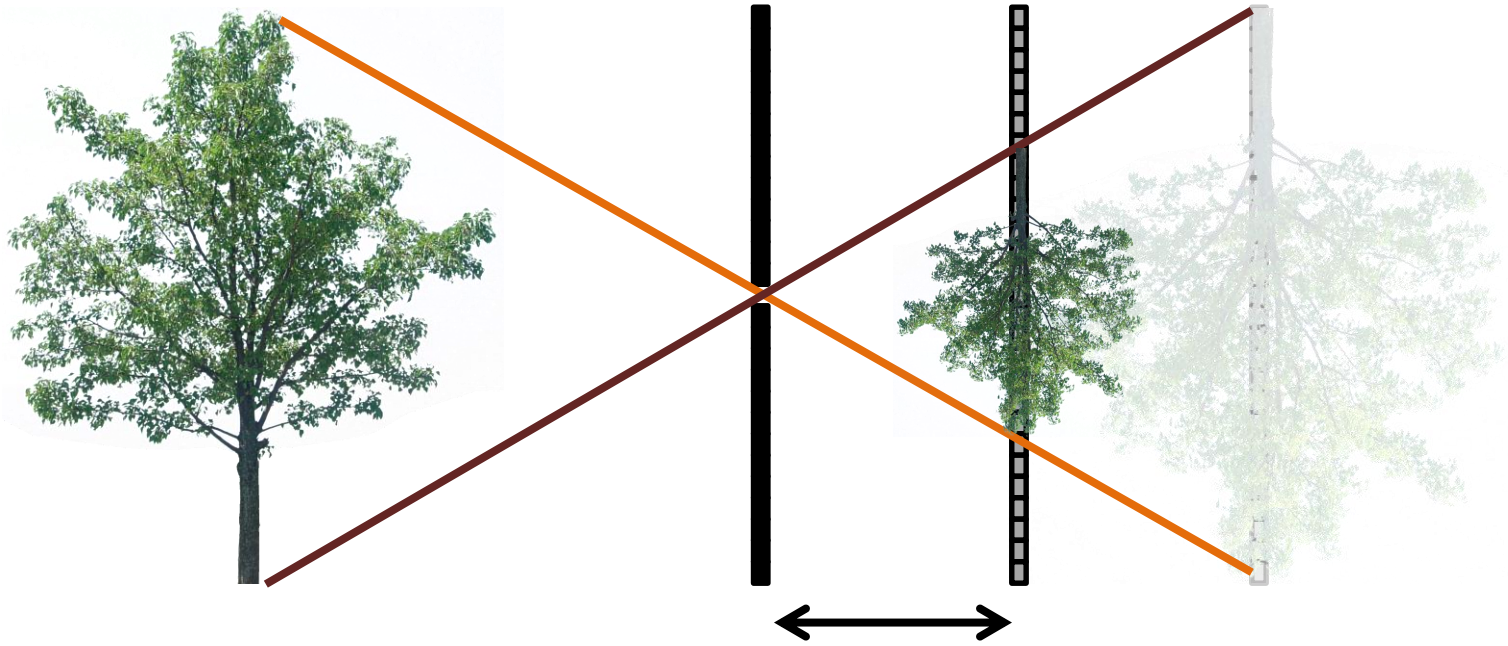
real-world object



Sensor place

- object projection is half the size

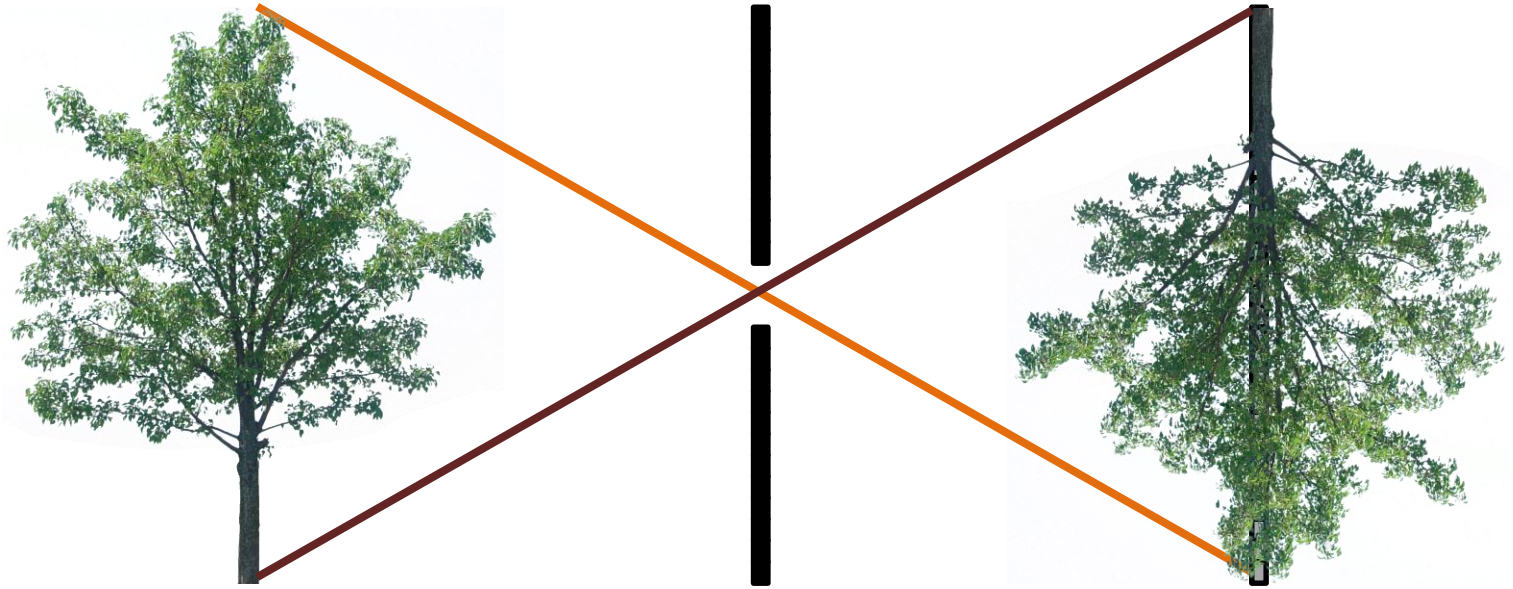
real-world object



Pinhole size

- What happens as we change the pinhole diameter?

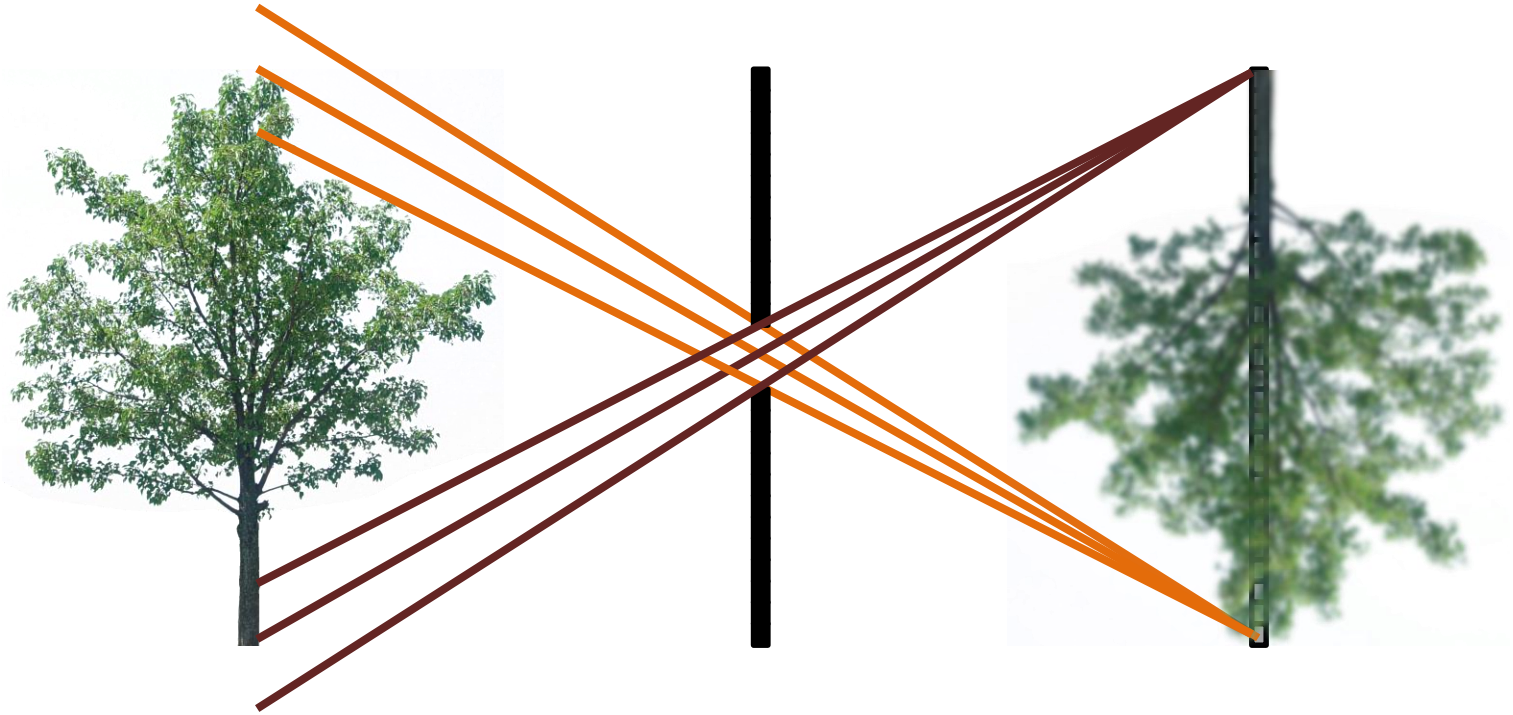
real-world object



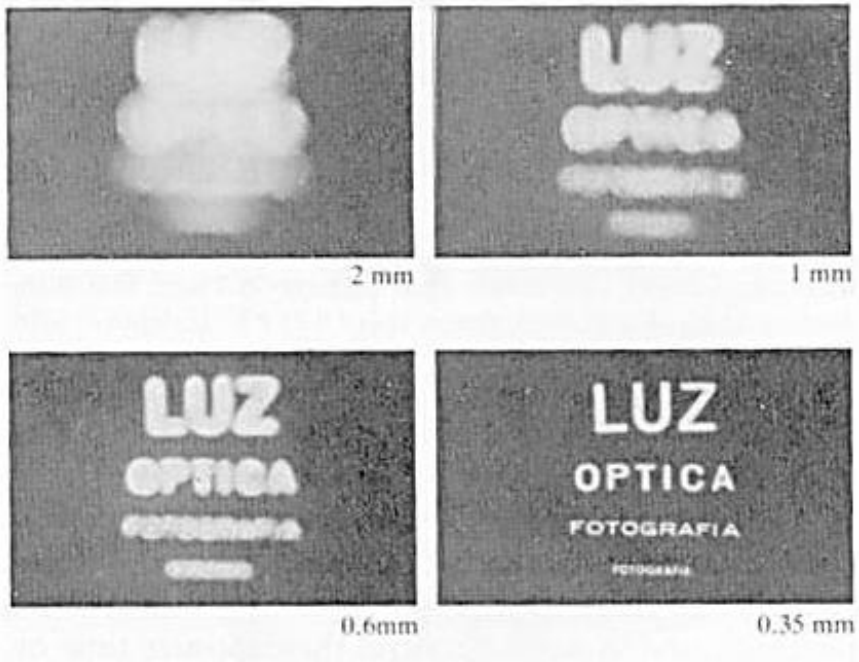
Pinhole size

- More light from different points of the object are integrated in the same pixel (integration->mean-> blur).

real-world object

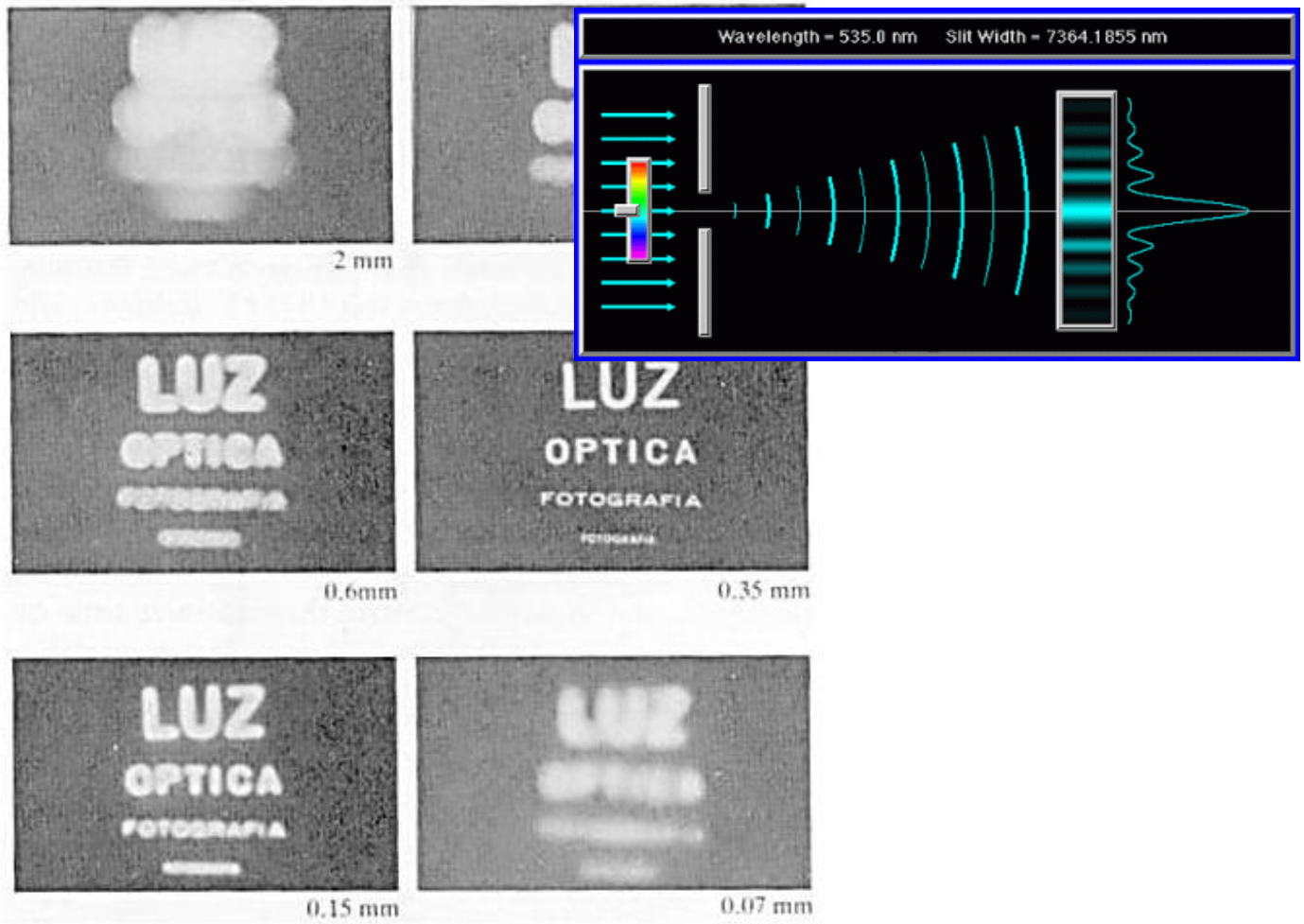


Side note: Pinhole size example



- Is smaller pinhole means sharper image?

Side note: Pinhole size example

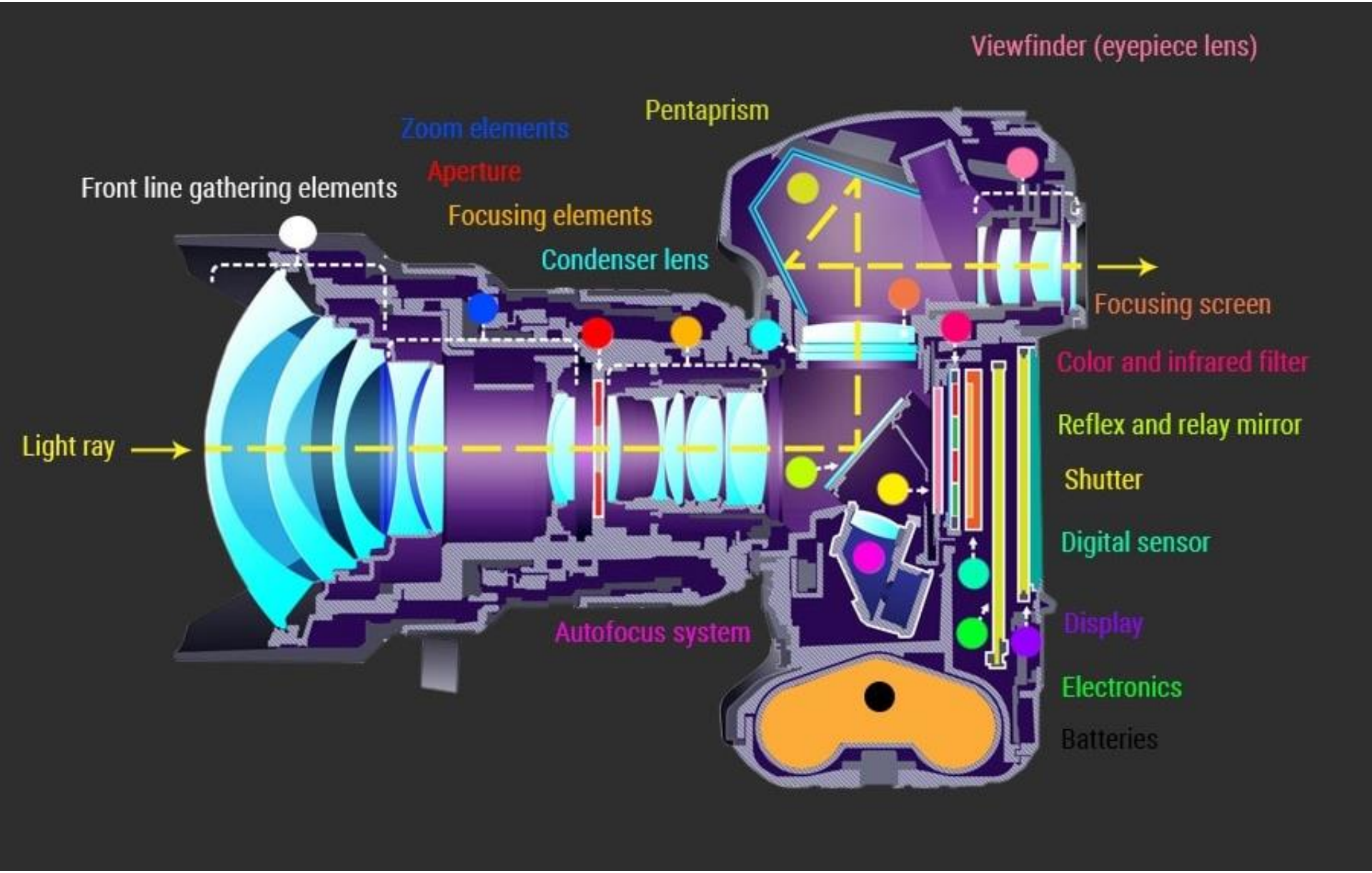


- No! due to diffraction (Wave–particle duality of photons).

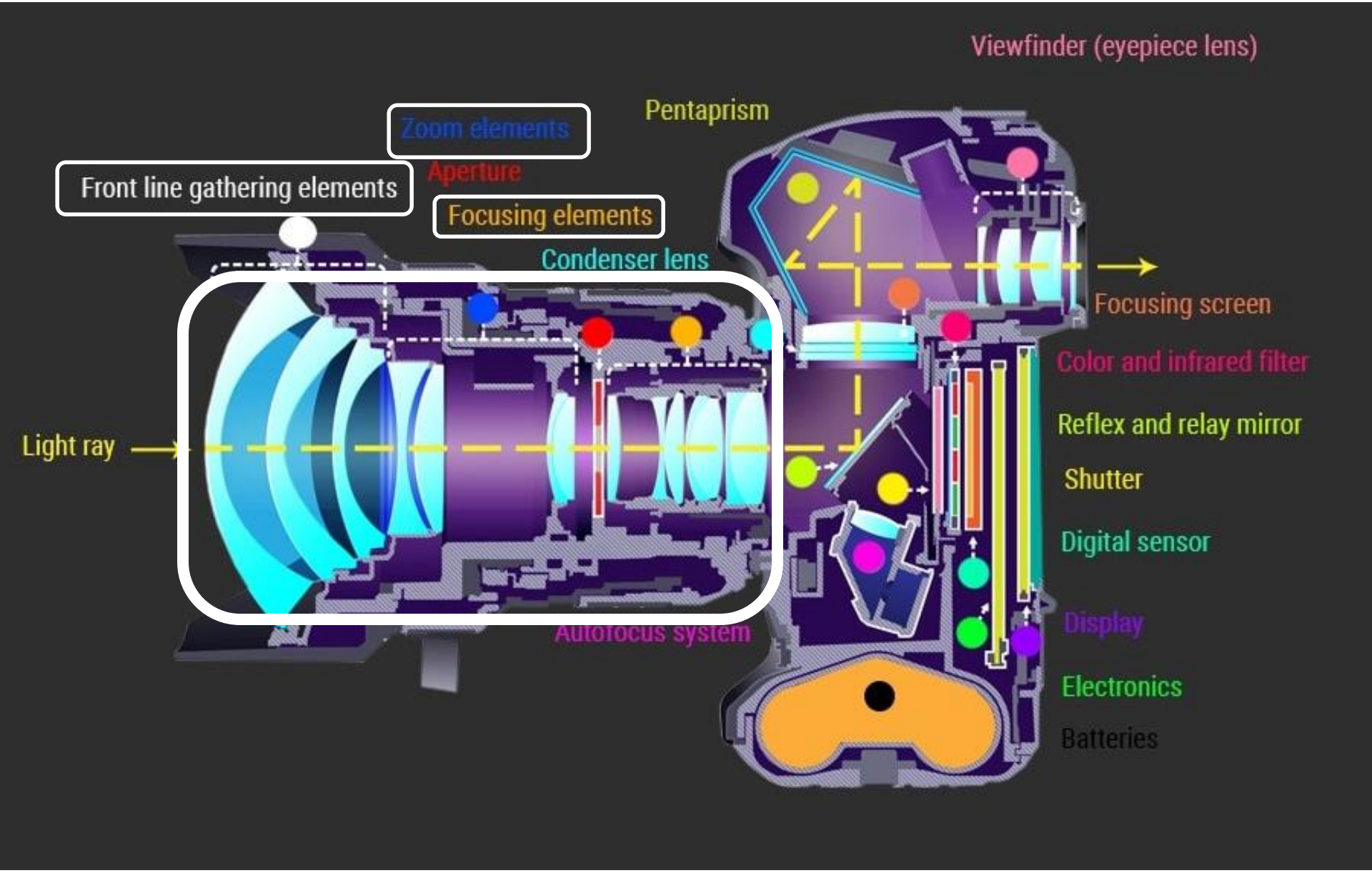
Contents

- BRDF
- Pinhole camera
- **Digital camera**
- The human eye
- Image transformations

Digital single-lens reflex camera (DSLR)

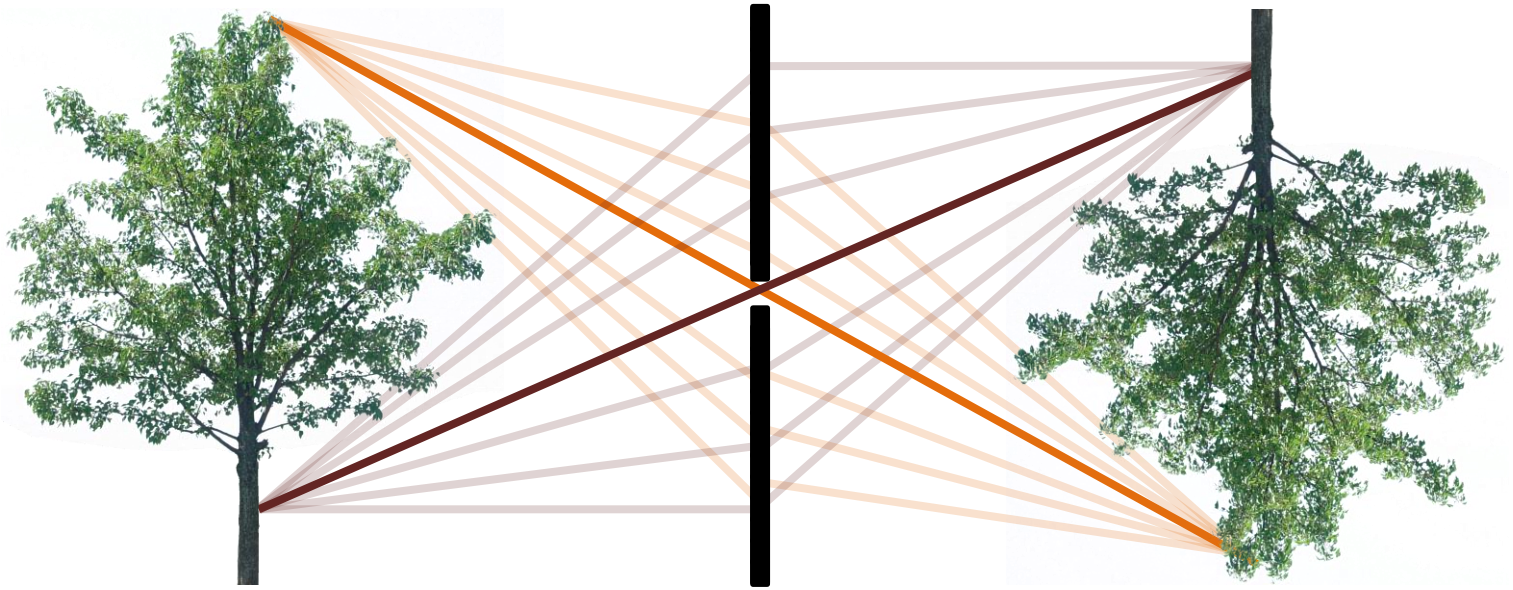


Digital single-lens reflex camera (DSLR)



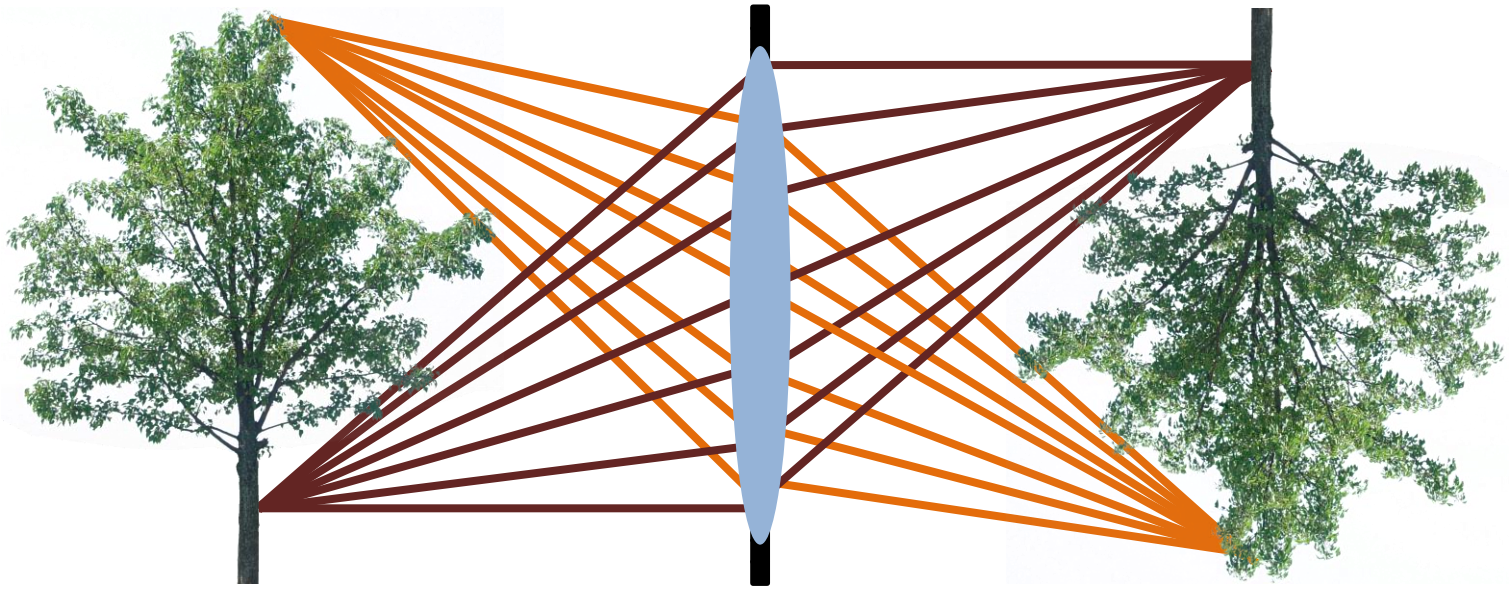
Pinhole to lens

- All new cameras has lenses in them, what about it?



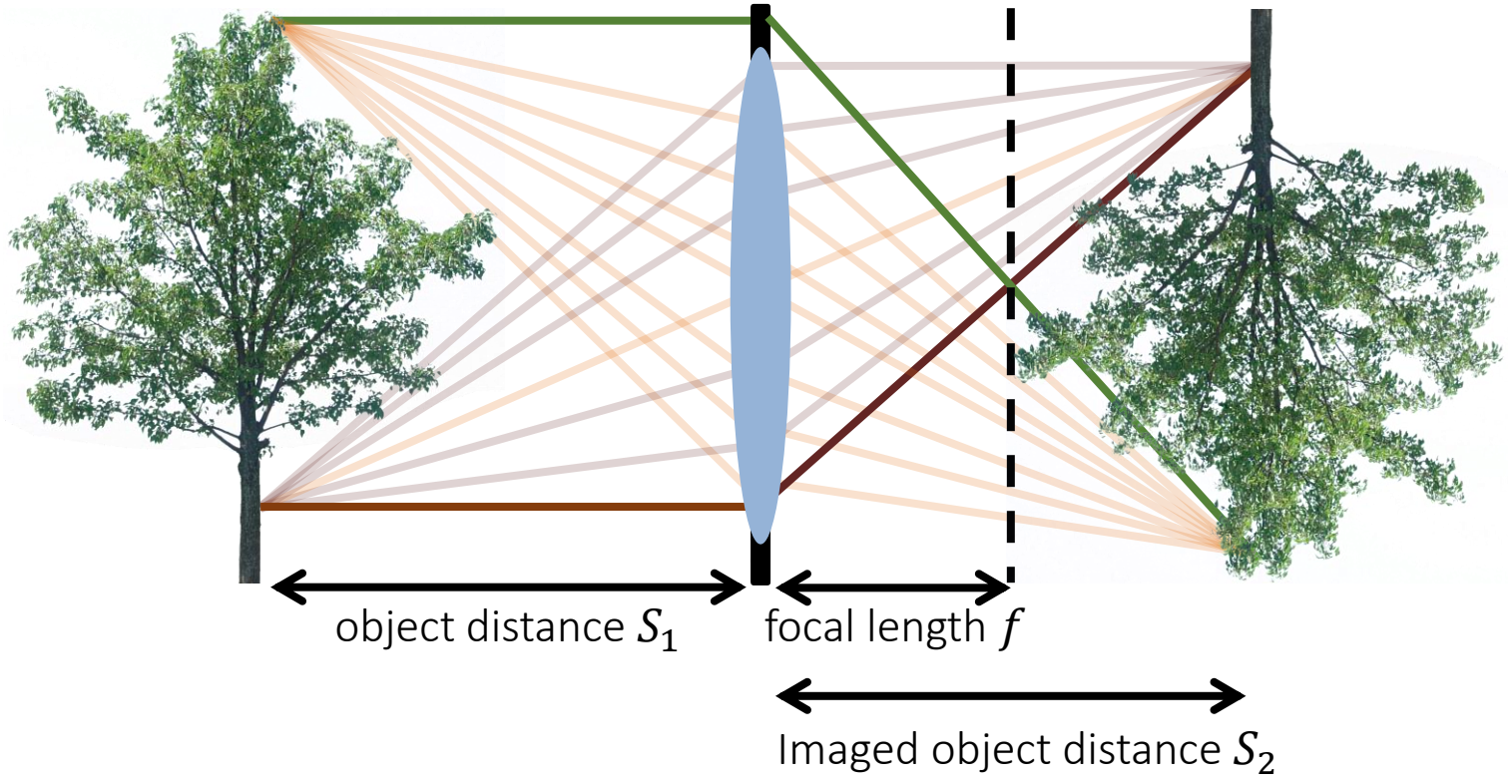
The lens camera

- The lens can replace the pinhole, while giving the advantage of **more input light**.



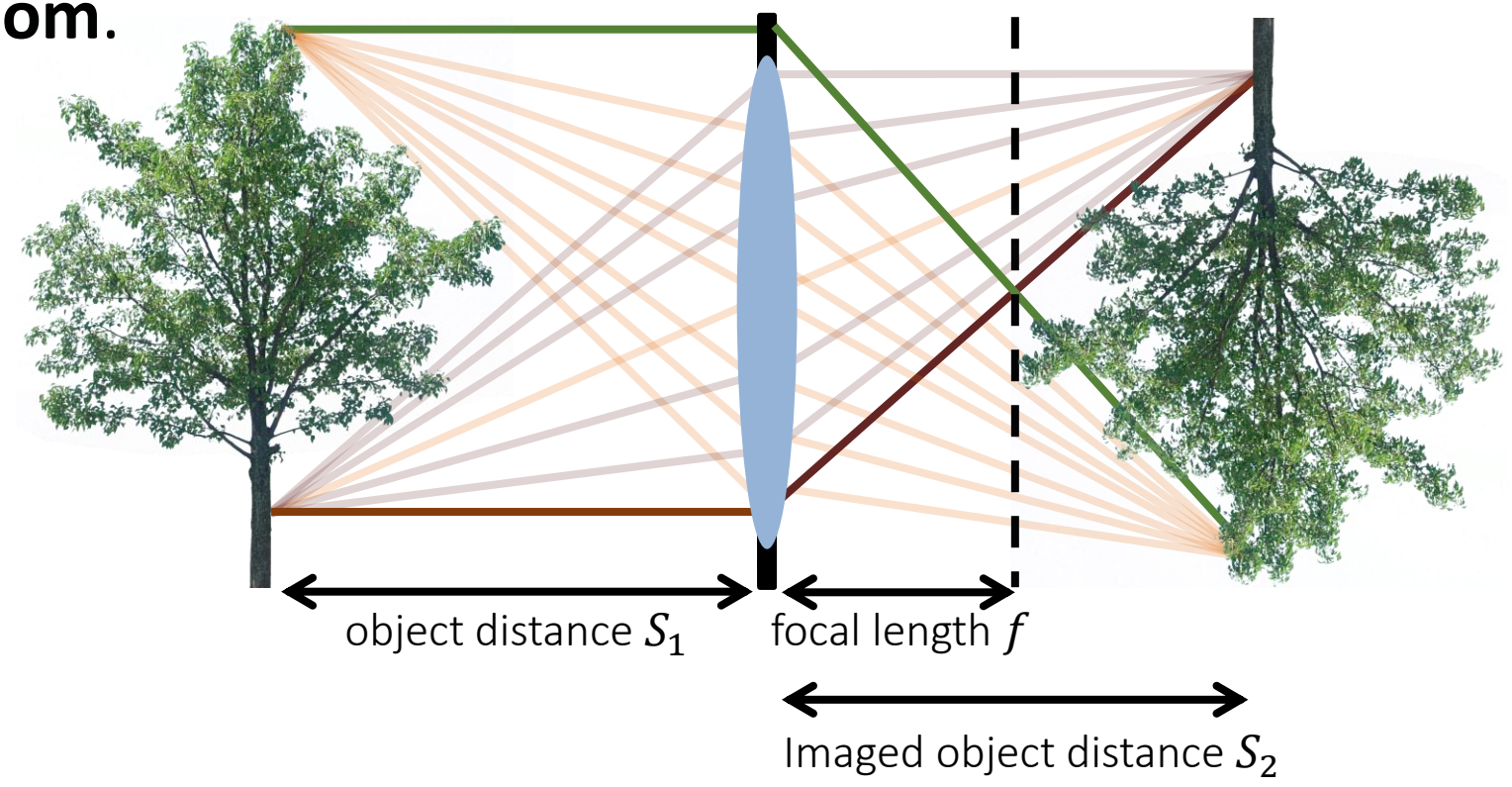
Focal length

- In a lens camera, **focal length f** is distance where parallel rays intersect.



Focal length

- Thin lens equation: $\frac{1}{S_1} + \frac{1}{S_2} = \frac{1}{f}$
- As seen above, the focal length also responsible for **zoom**.



Side note: thin lens approximation

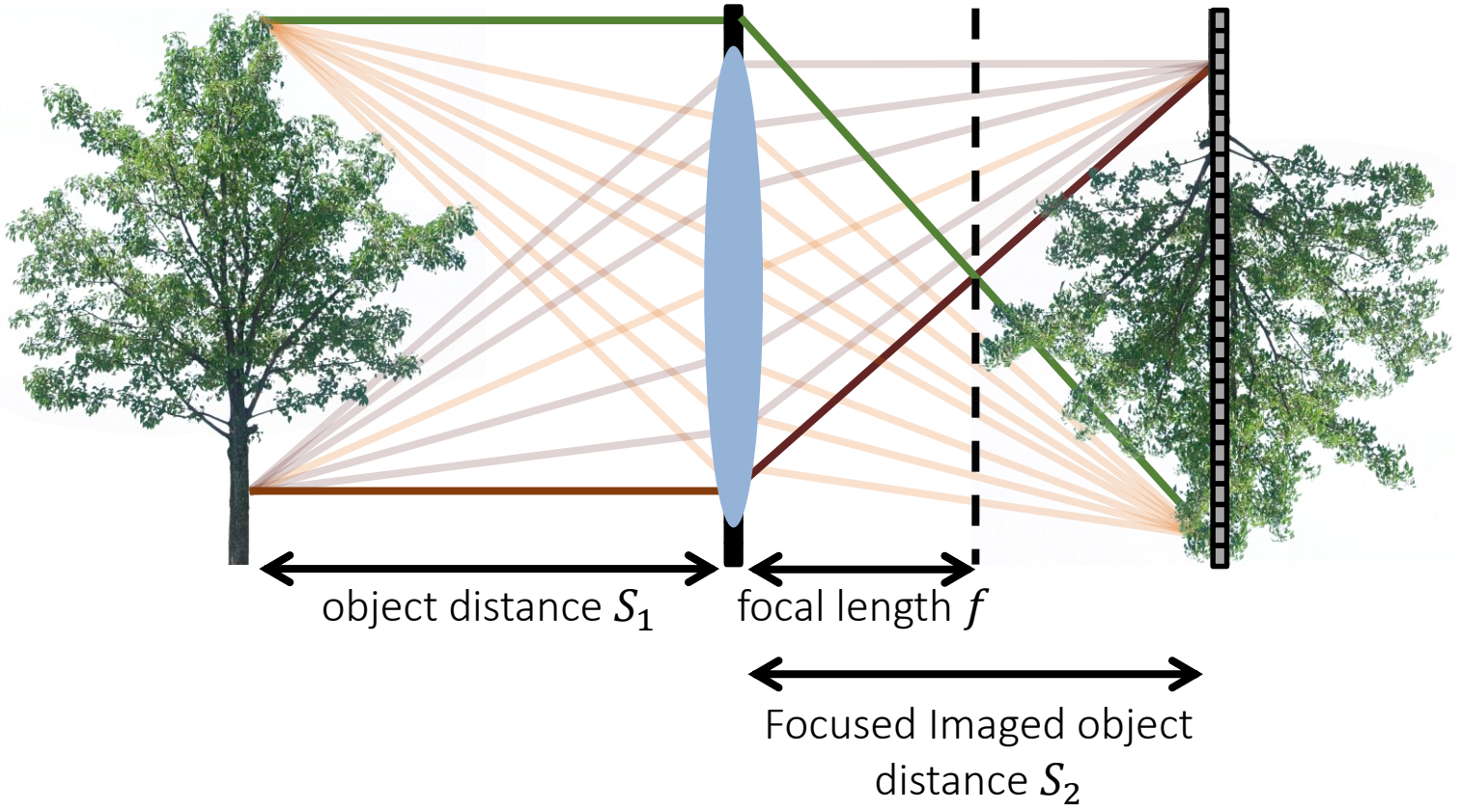
- Thin lens equation: $\frac{1}{S_1} + \frac{1}{S_2} = \frac{1}{f}$
- The thin lens equation assumes thin lens, which is not common for most cameras. Apparently, this is still relatively correct for complex lenses.
- From here on we will assume we are dealing with thin lenses- known as the **thin lens approximation.**

Focal length

- Thin lens equation: $\frac{1}{S_1} + \frac{1}{S_2} = \frac{1}{f}$
- The focal length f is determined by the curvature of the lens itself (**Lensmaker's equation-** out of scope).
- What happens when a sensor is placed at S_2? What happens when it isn't?

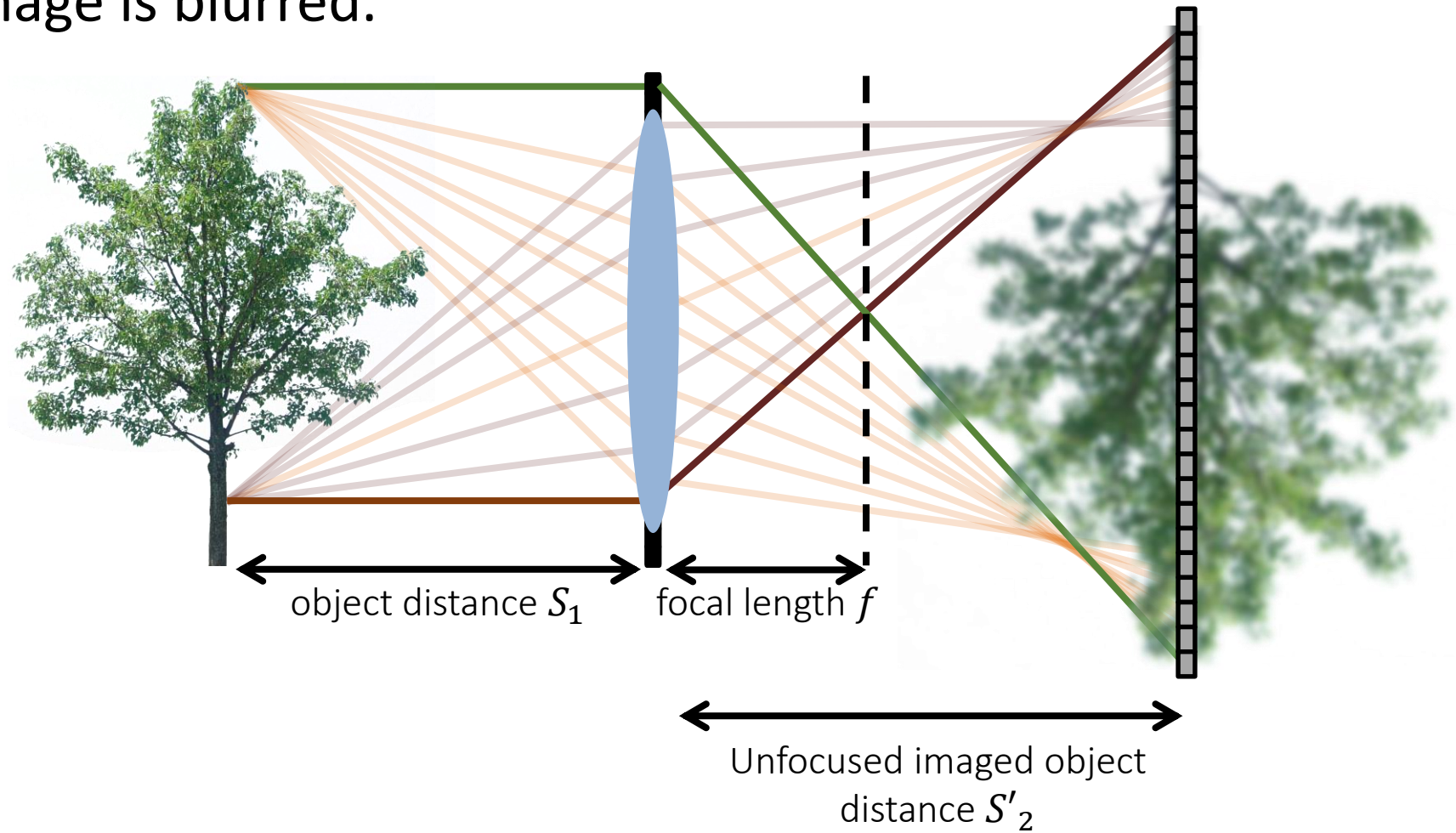
Focal length

- Thin lens equation: $\frac{1}{S_1} + \frac{1}{S_2} = \frac{1}{f}$
- As already seen, when a sensor is placed at S_2 the resulted image is focused.



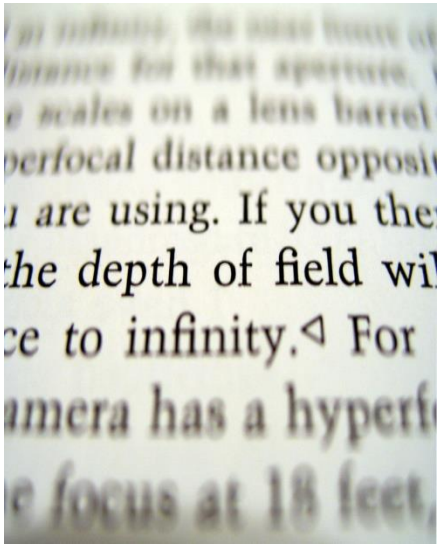
Focal length

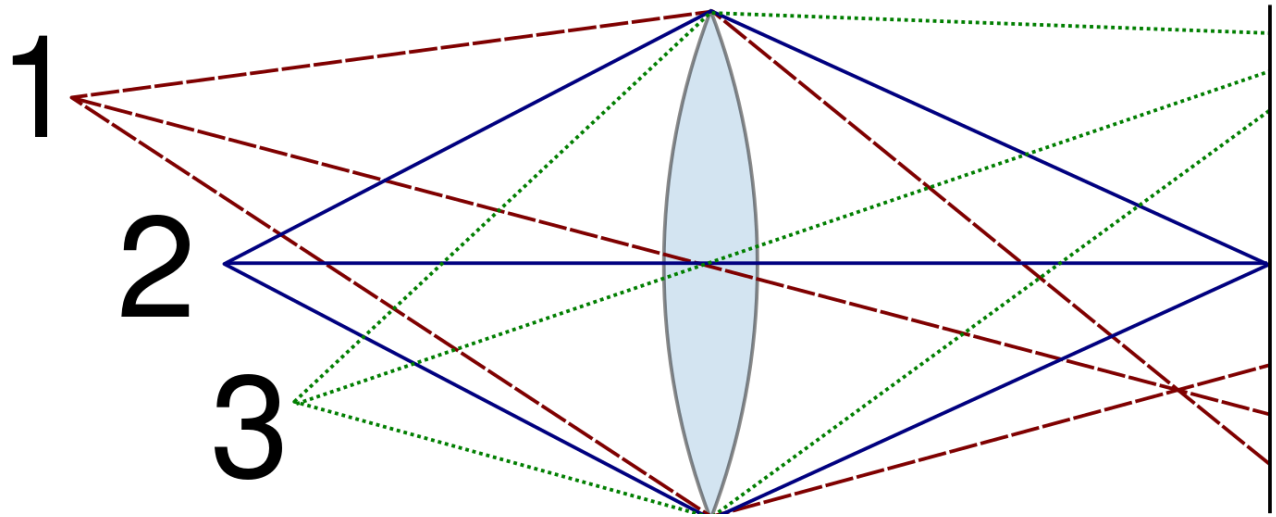
- Thin lens equation: $\frac{1}{S_1} + \frac{1}{S_2} = \frac{1}{f}$
- when the sensor doesn't respect the lens equation, the image is blurred.

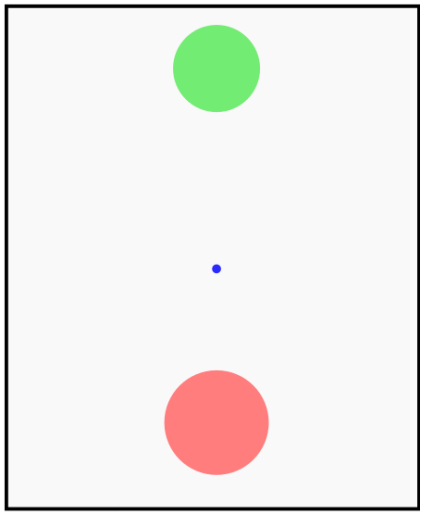


Depth of field (DOF)

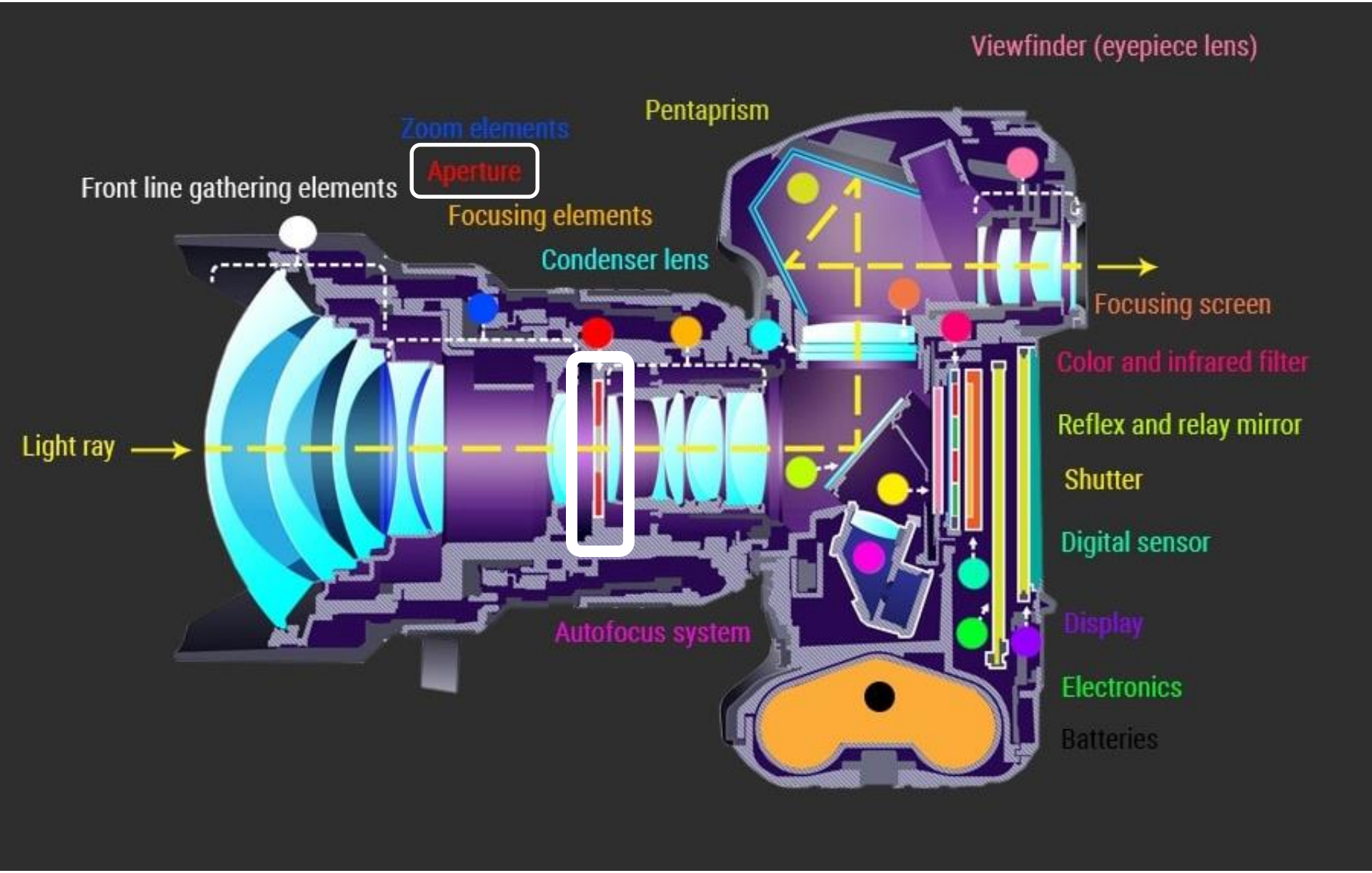
- **Depth of field** is the distance between the nearest and the farthest objects that are in acceptably sharp focus in an image.
  - "acceptably sharp" is human determined and is measured by the spot size that a point light is out of focus (called **circle of confusion**).





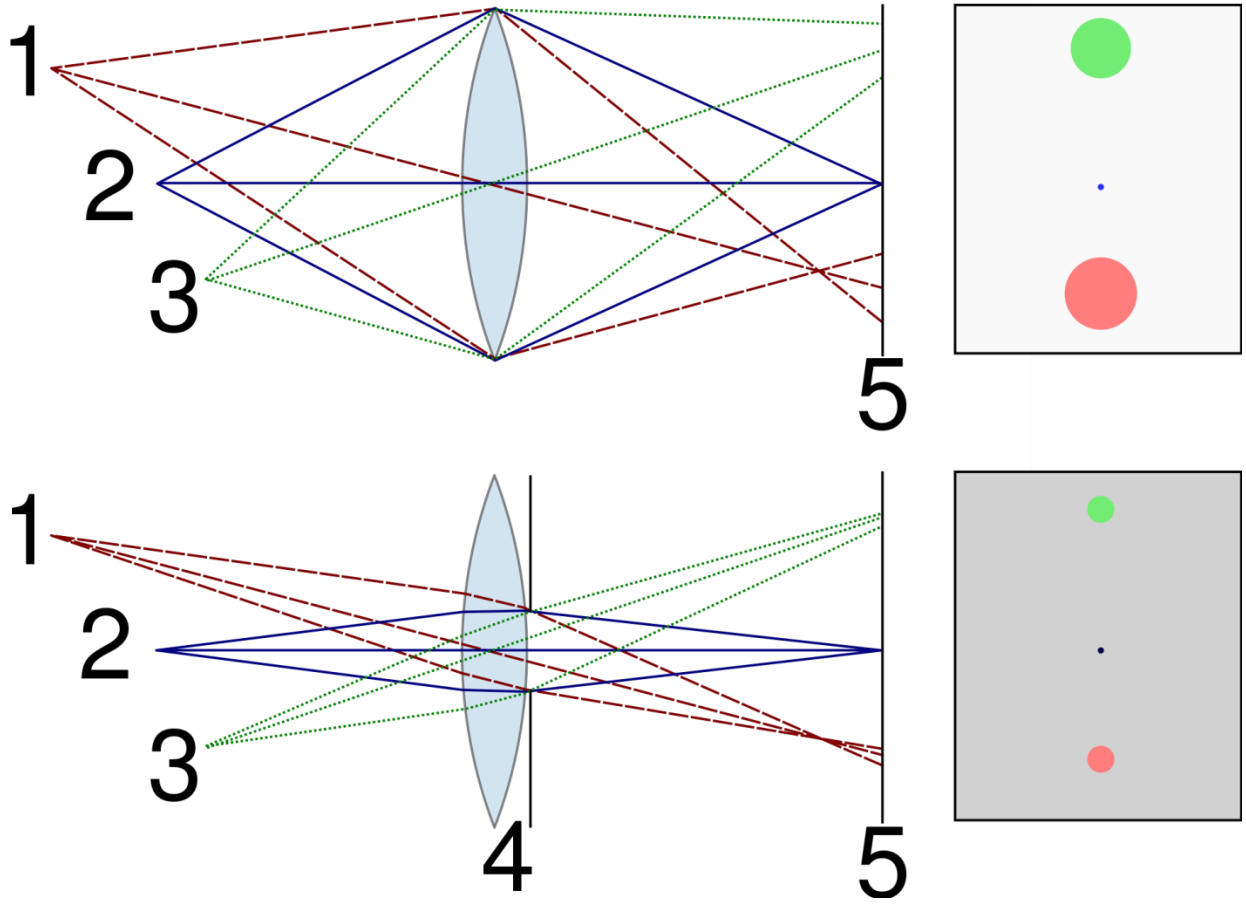


Digital single-lens reflex camera (DSLR)



Aperture

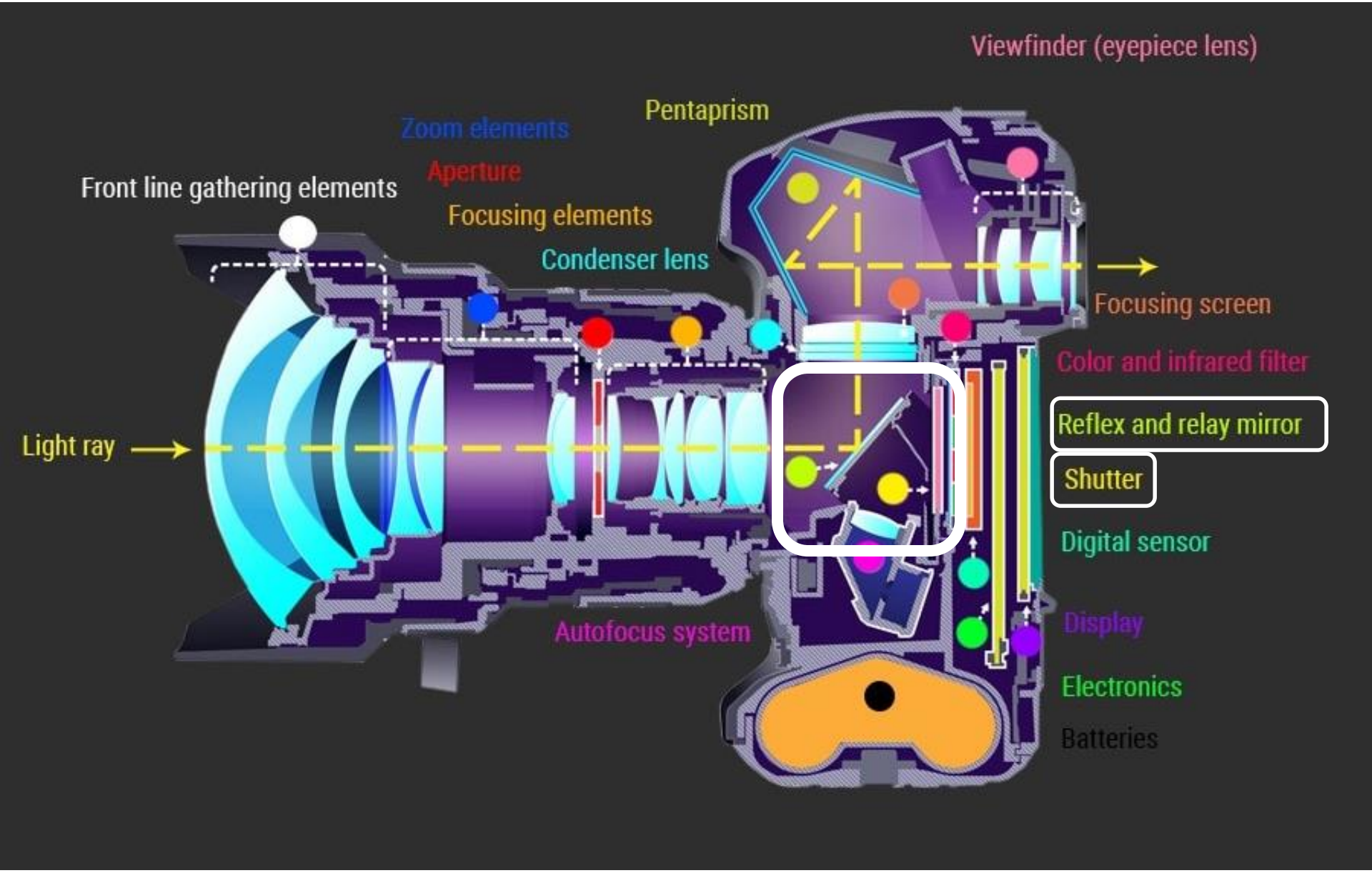
- An **aperture** is a hole or an opening through which light travels. The aperture helps enlarging the DOF, but also inputs less light, making the image darker.



Side note: aperture blur

- Changing the default shape of the aperture will give different shapes to “circle of confusion”.

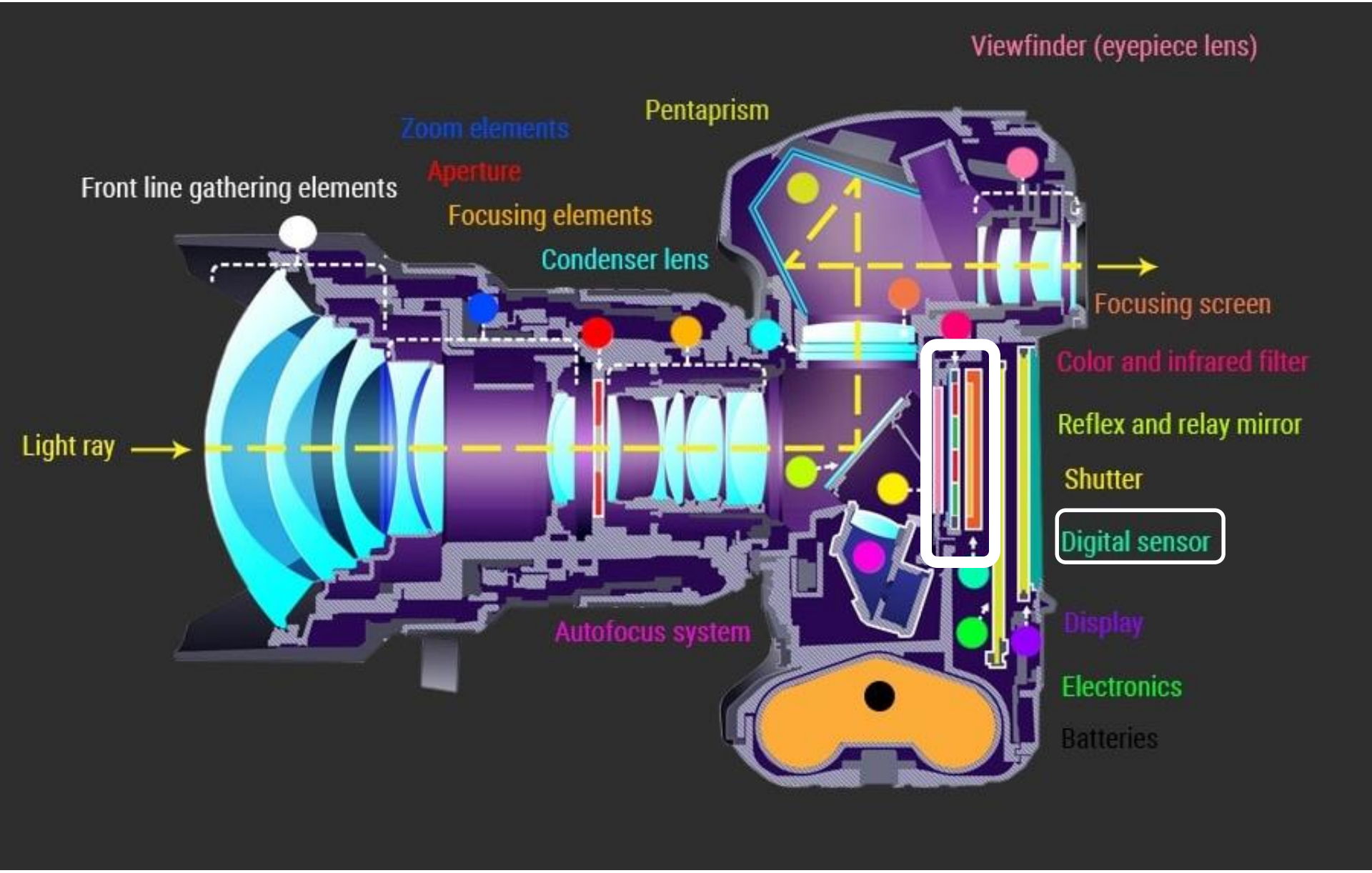
Digital single-lens reflex camera (DSLR)



shutter

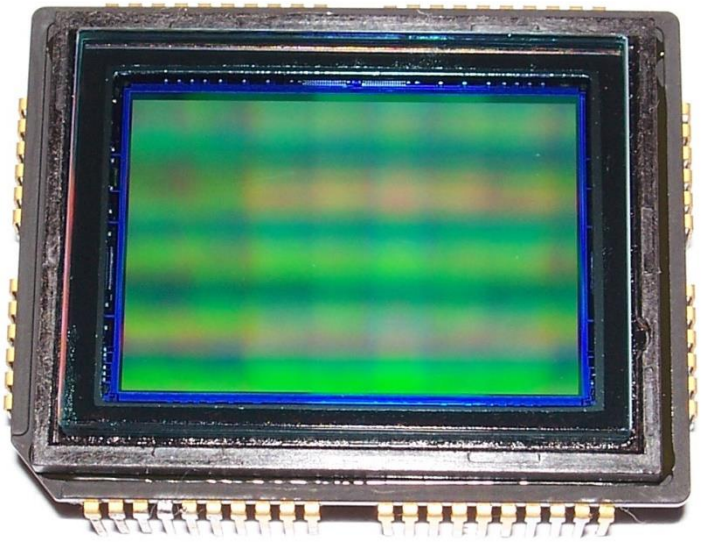
- A common why to make the image lighter is to increase **exposure time** (also called "**shutter speed**", although measurements units are seconds).
- Larger exposure time can also lead to image blur if camera is not stable during capture
- https://www.youtube.com/watch?v=ptfSW4eW25g (shutter and reflex mirror operation slow-mo)

Digital single-lens reflex camera (DSLR)



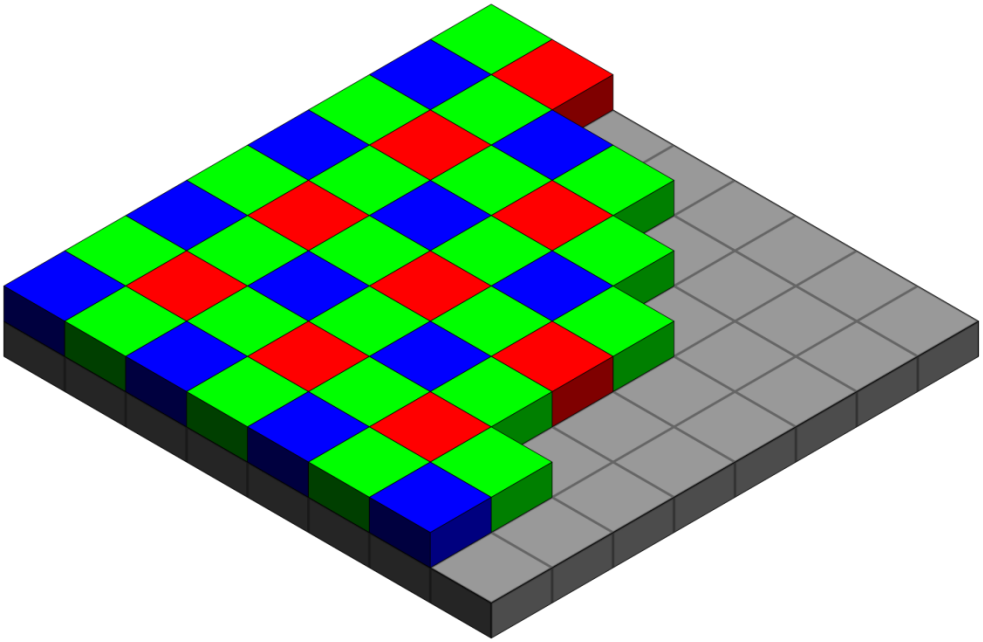
Photosensors

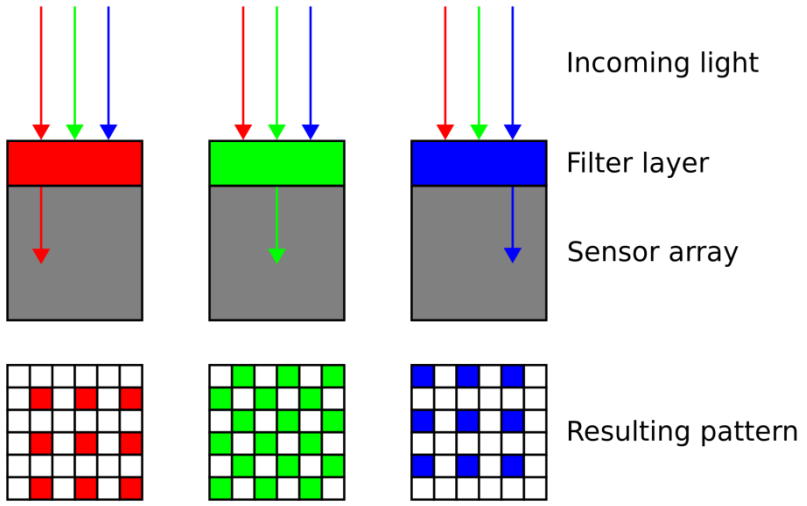
- Sensor that transforms different input light intensity to different valued electrical charges.
  - Each pixel in resulting raw image comes from one capacitor in a 2D array which makes the sensor.
- Two types of such sensors are called **CCD** (charge-coupled device) or **CMOC** (complementary metal-oxide-semiconductors).



Bayer filter

- On top of each photosensor is a Bayer filter which filter the input light to one of the 3 primary colors R,G & B.
- This filter has twice as much green elements compared to the others, to mimic the human eye physiology.



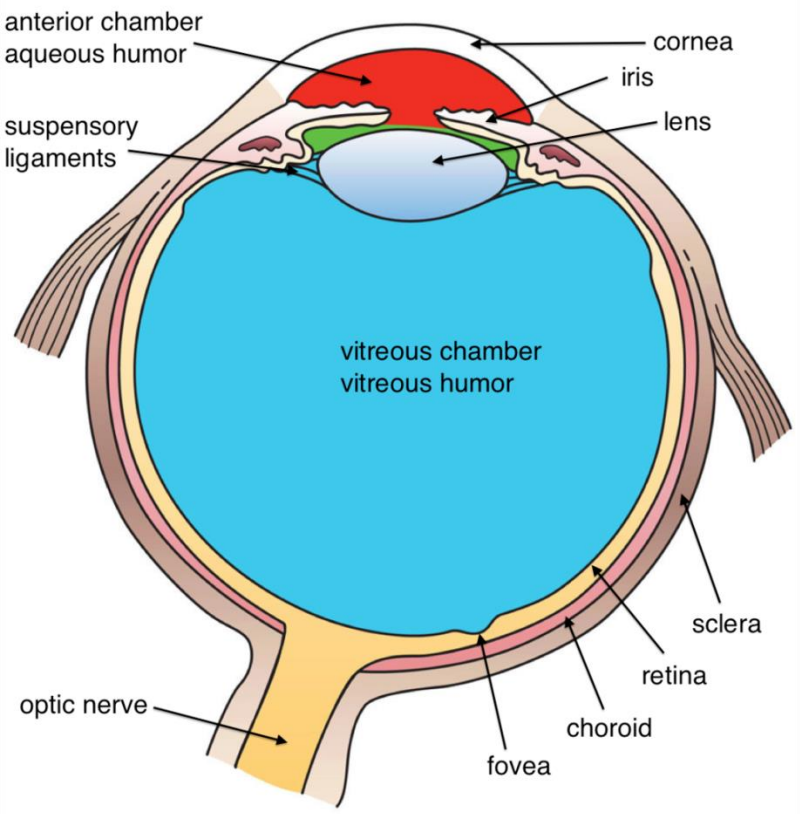


Contents

- BRDF
- Pinhole camera
- Digital camera
- **The human eye**
- Image transformations

Human color perception

- The modern camera tries to mimic the human eye:
  - Iris = aperture.
  - Lens.
  - Retina = photosensor.



Human color perception

- In the retina there are two types of photoreceptors:
  - Rods: responsible for low-light vision – maximum sensitive in green region of 500nm. Rods are not responsible for color vision though.
  - Cones: responsible for color vision. 3 types (short, medium & long), each has maximum sensitive for different colors (roughly RGB…). this is way the RGB were chosen as primary colors.

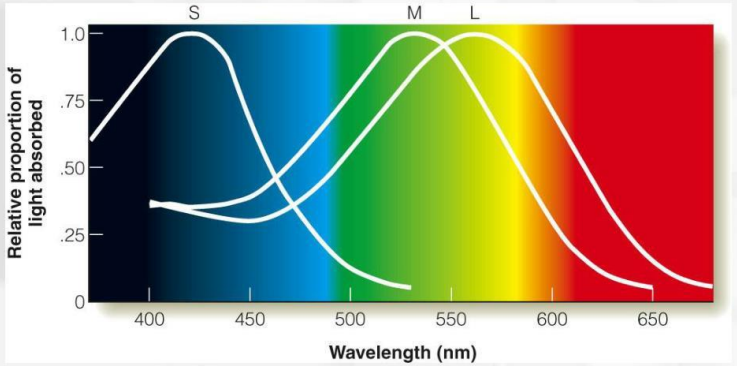
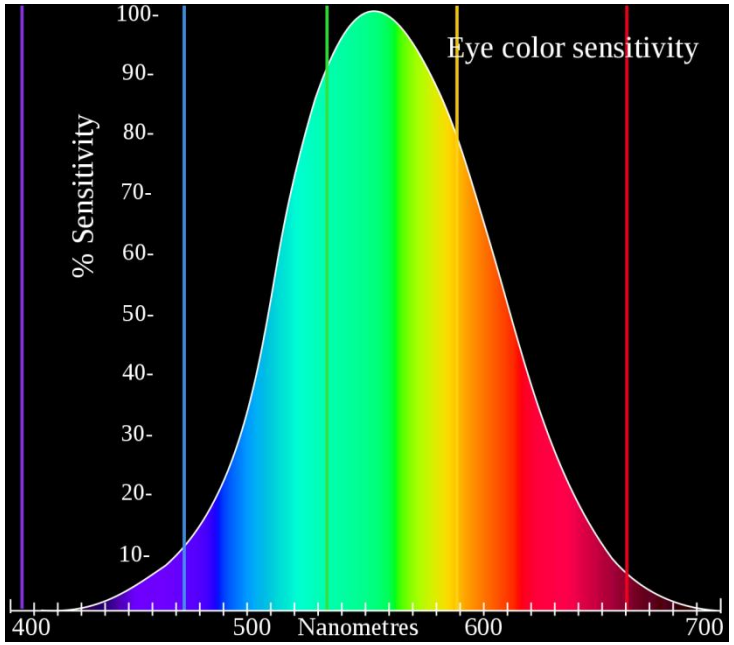


Figure 9.9 Absorption spectra of the three cone pigments.

Human color perception

- The eye color sensitivity (Luminosity function) was defined by experiments- one result from this experiments is that green color is more sensitive than others.
  - This is way Bayer filter has double the greens!



Contents

- BRDF
- Pinhole camera
- Digital camera
- The human eye
- **Image transformations**

objective

- Being able to do all of the below transformations with matrix manipulation:

translation

rotation

scale

shear

affine

projective

- **Why matrix manipulation?**

objective

- Being able to do all of the below transformations with matrix manipulation:

translation

rotation

scale

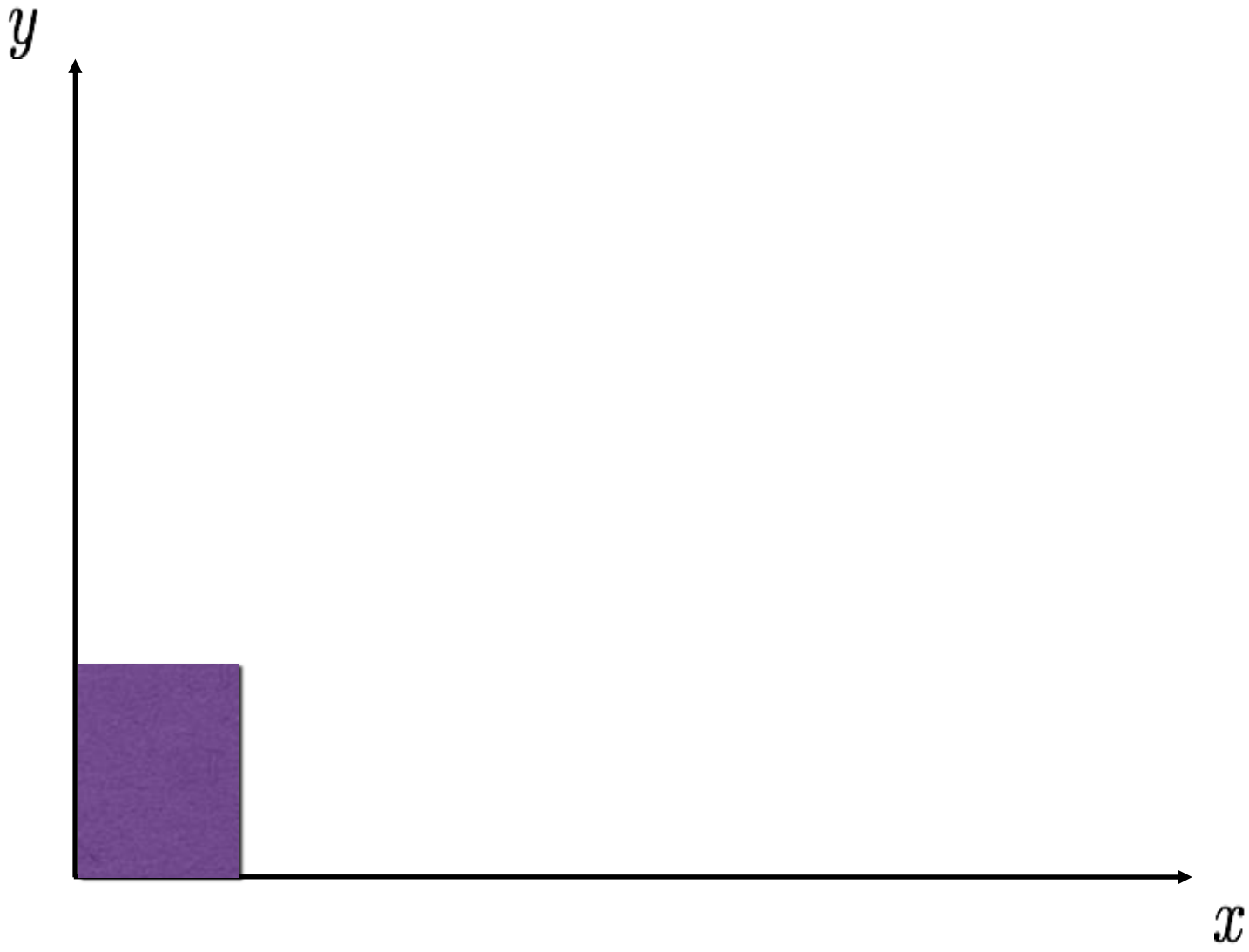
shear

affine

projective

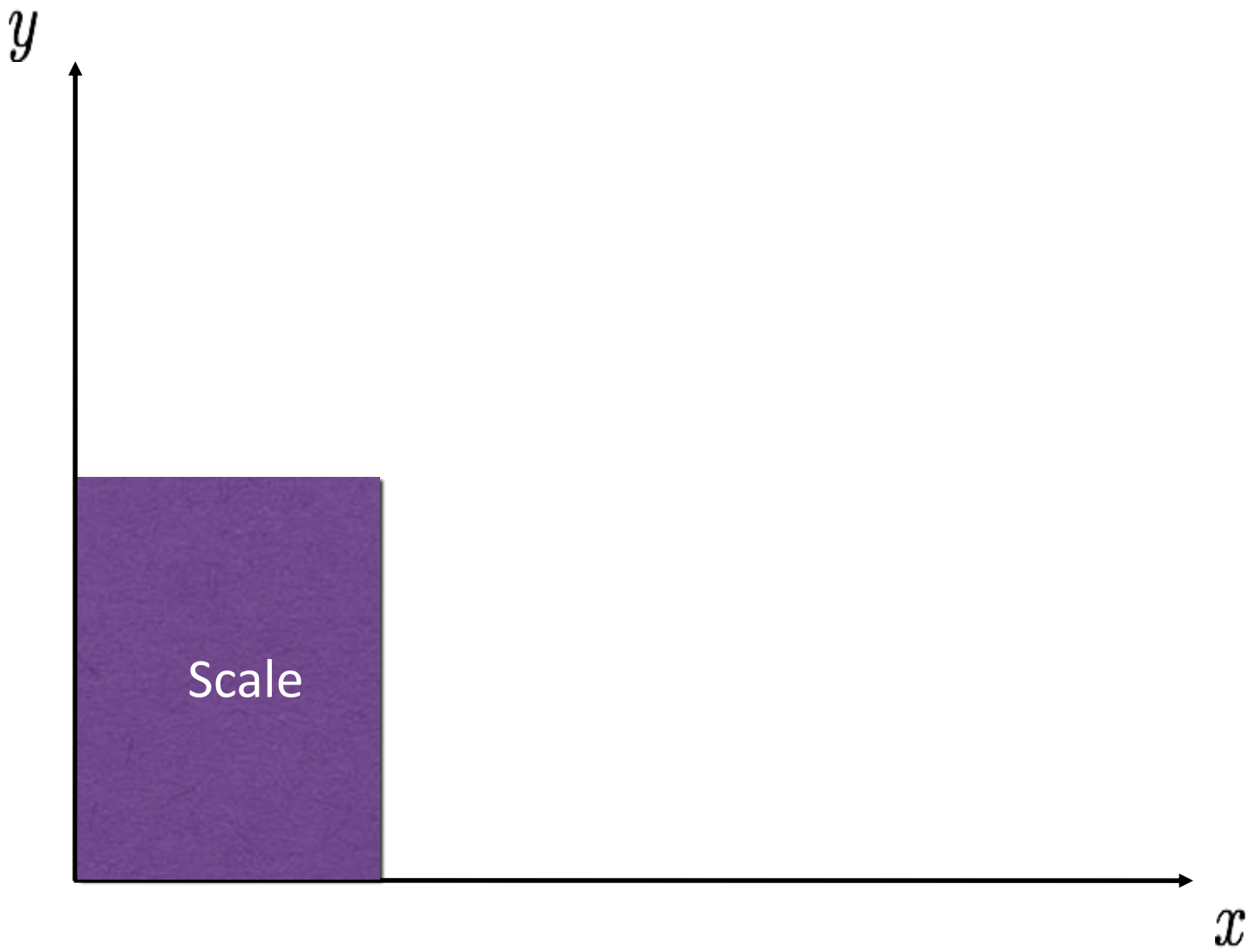
- **Why matrix manipulation?** Because then we can easily concatenate transformations (for example translation and rotation).

2D planar transformations

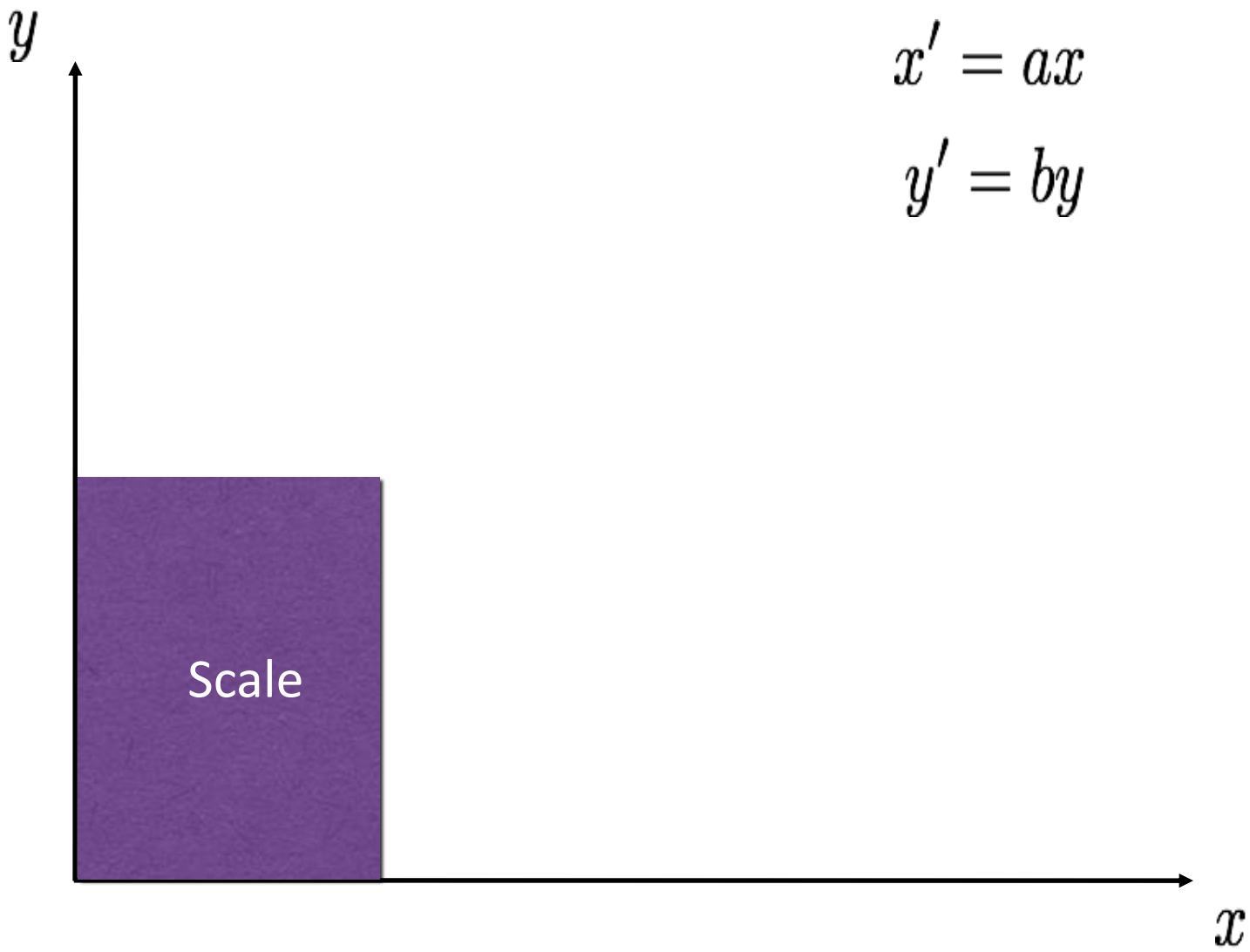


scale

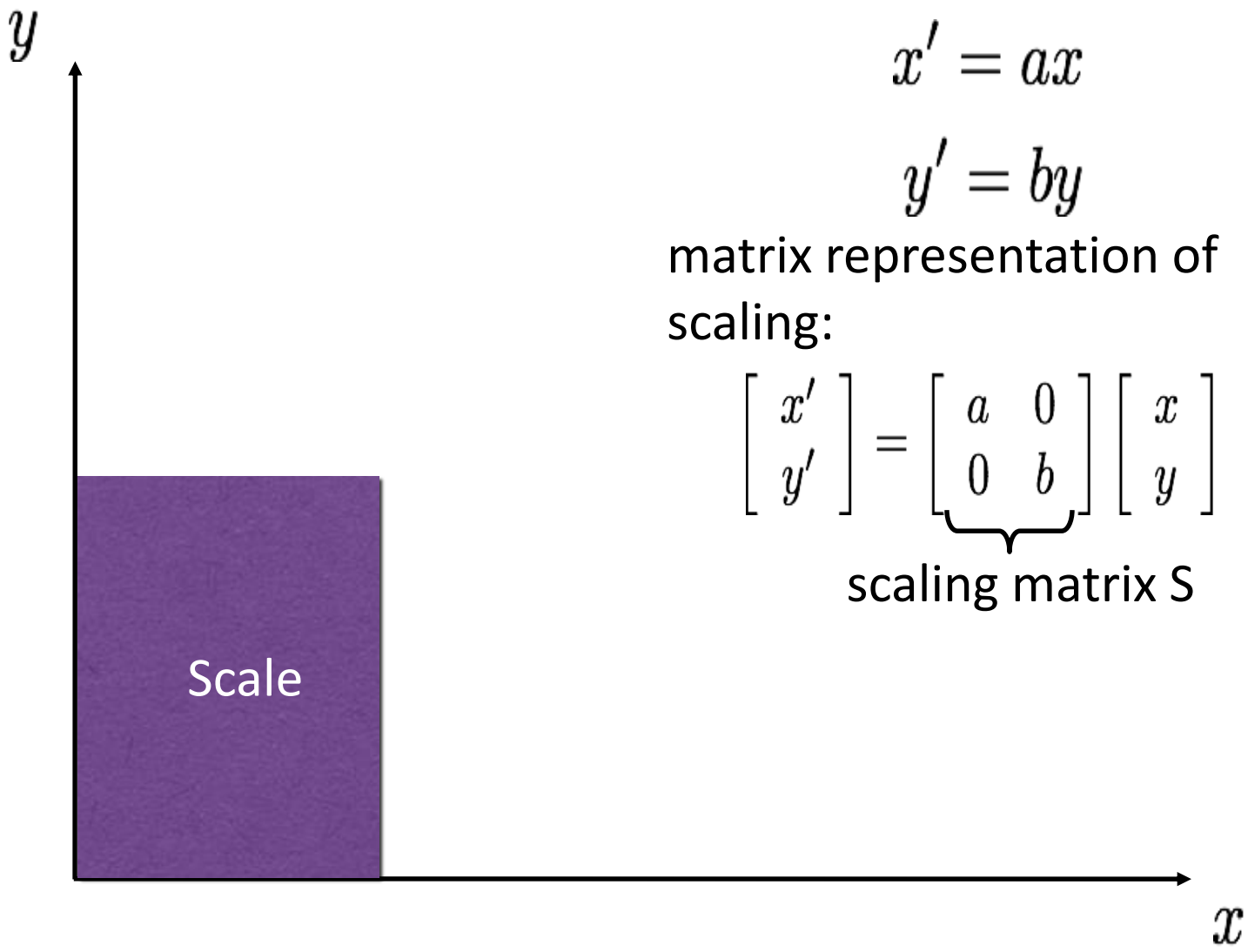
- How?



scale



scale



$$x' = ax$$

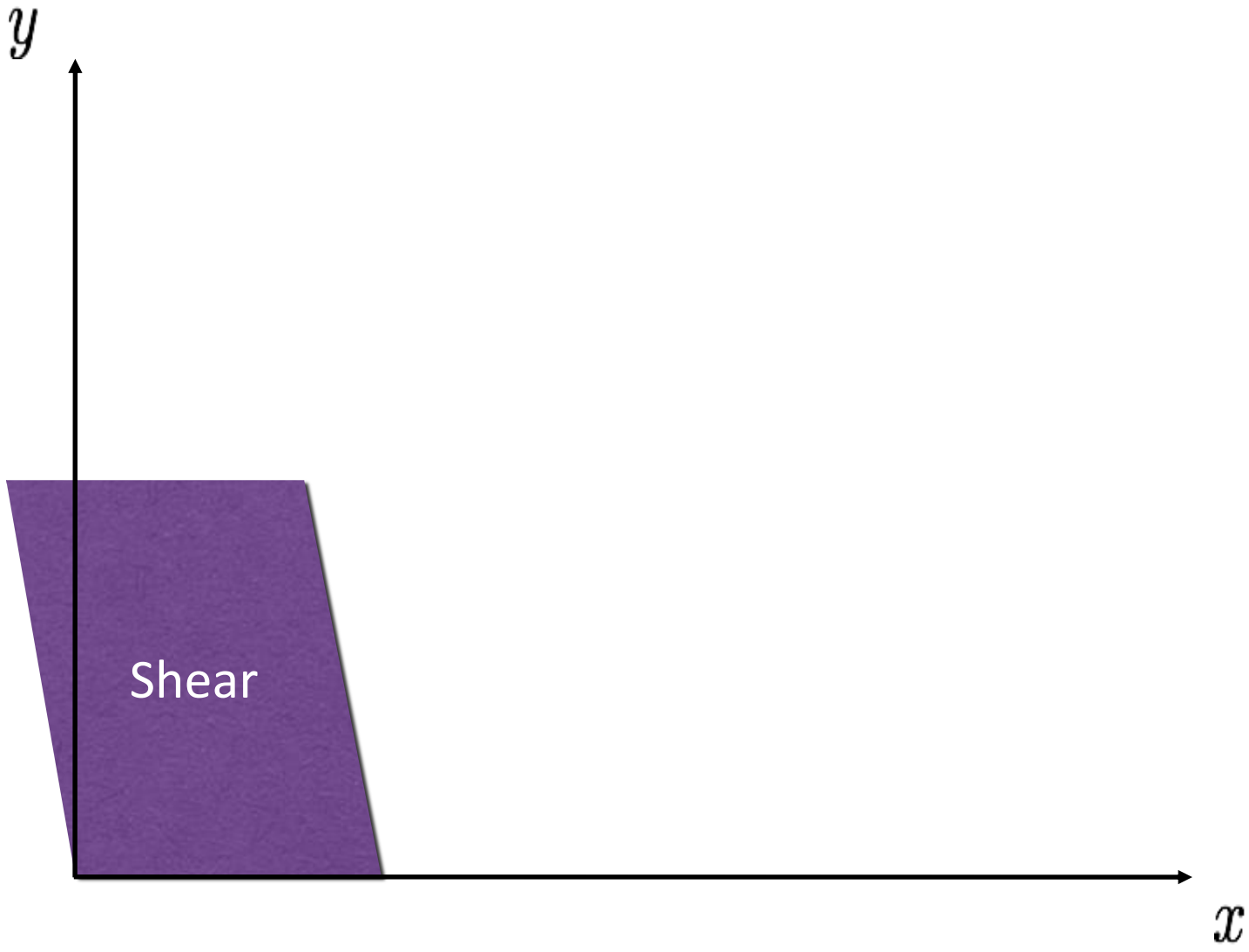
$$y' = by$$

matrix representation of scaling:

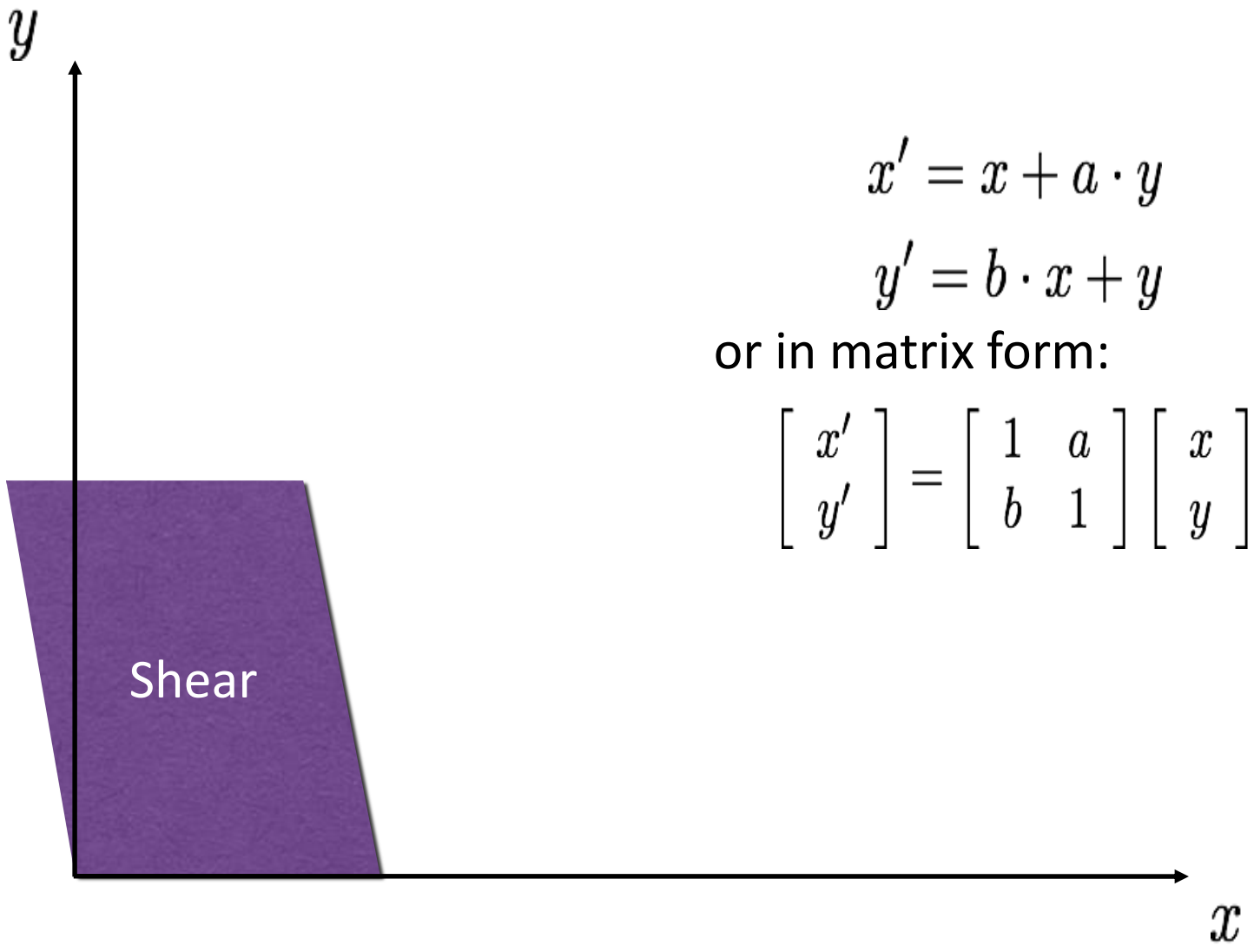
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}}_{\text{scaling matrix S}} \begin{bmatrix} x \\ y \end{bmatrix}$$

Shear

- How?



Shear



$$x' = x + a \cdot y$$

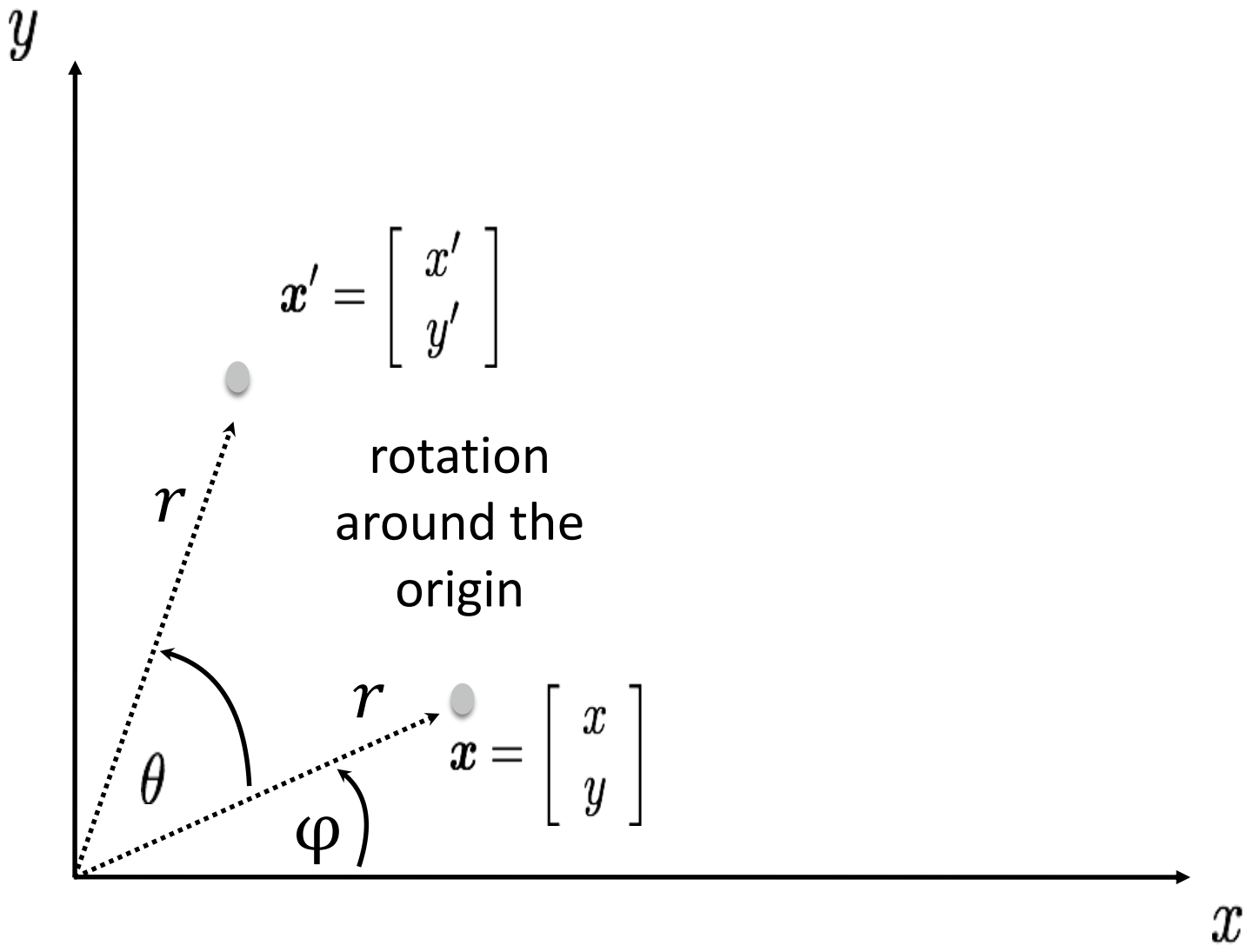
$$y' = b \cdot x + y$$

or in matrix form:

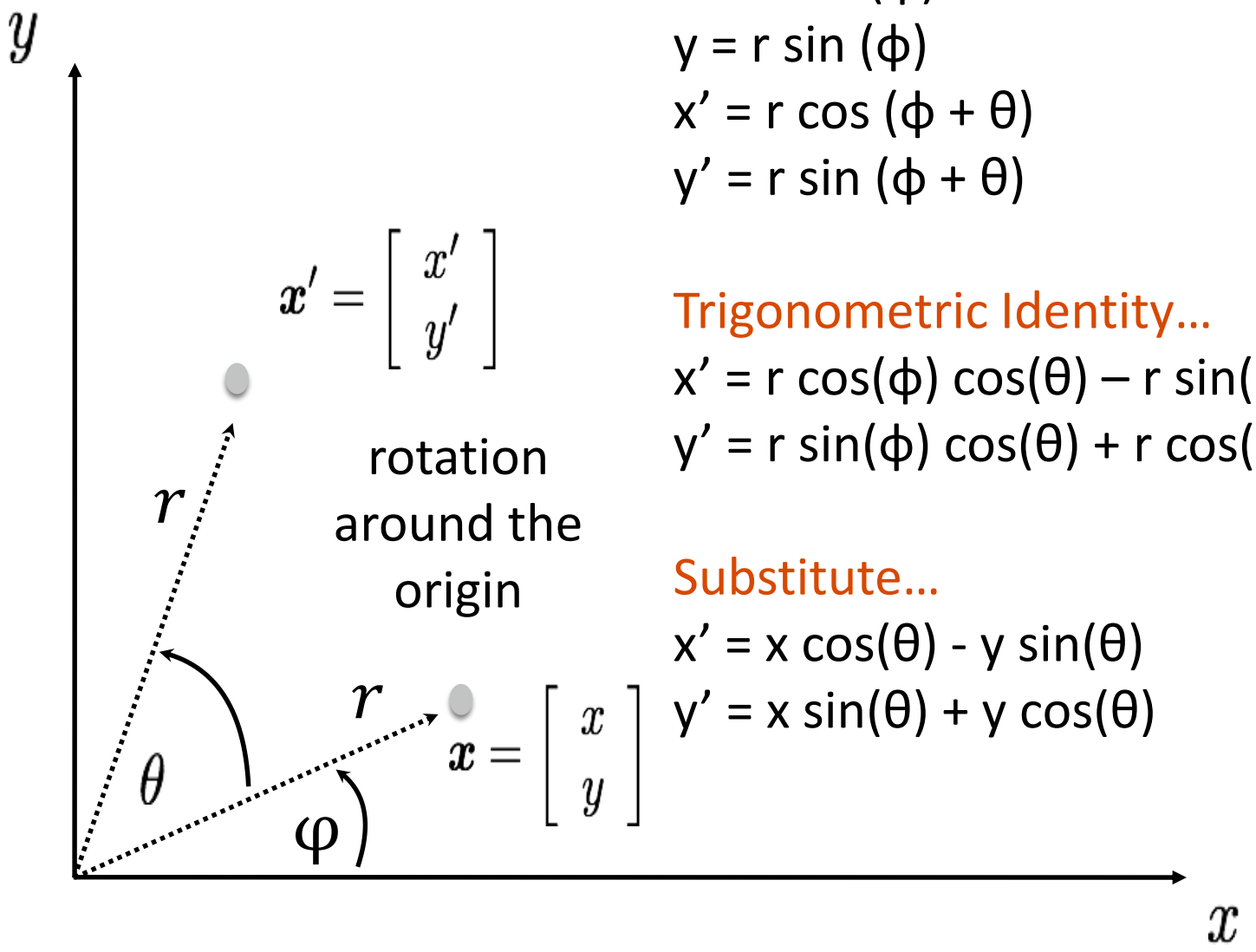
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & a \\ b & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Rotation

- How?



Rotation



Polar coordinates…

$x = r \cos(\phi)$

$y = r \sin(\phi)$

$x' = r \cos(\phi + \theta)$

$y' = r \sin(\phi + \theta)$

Trigonometric Identity…

$x' = r \cos(\phi) \cos(\theta) - r \sin(\phi) \sin(\theta)$

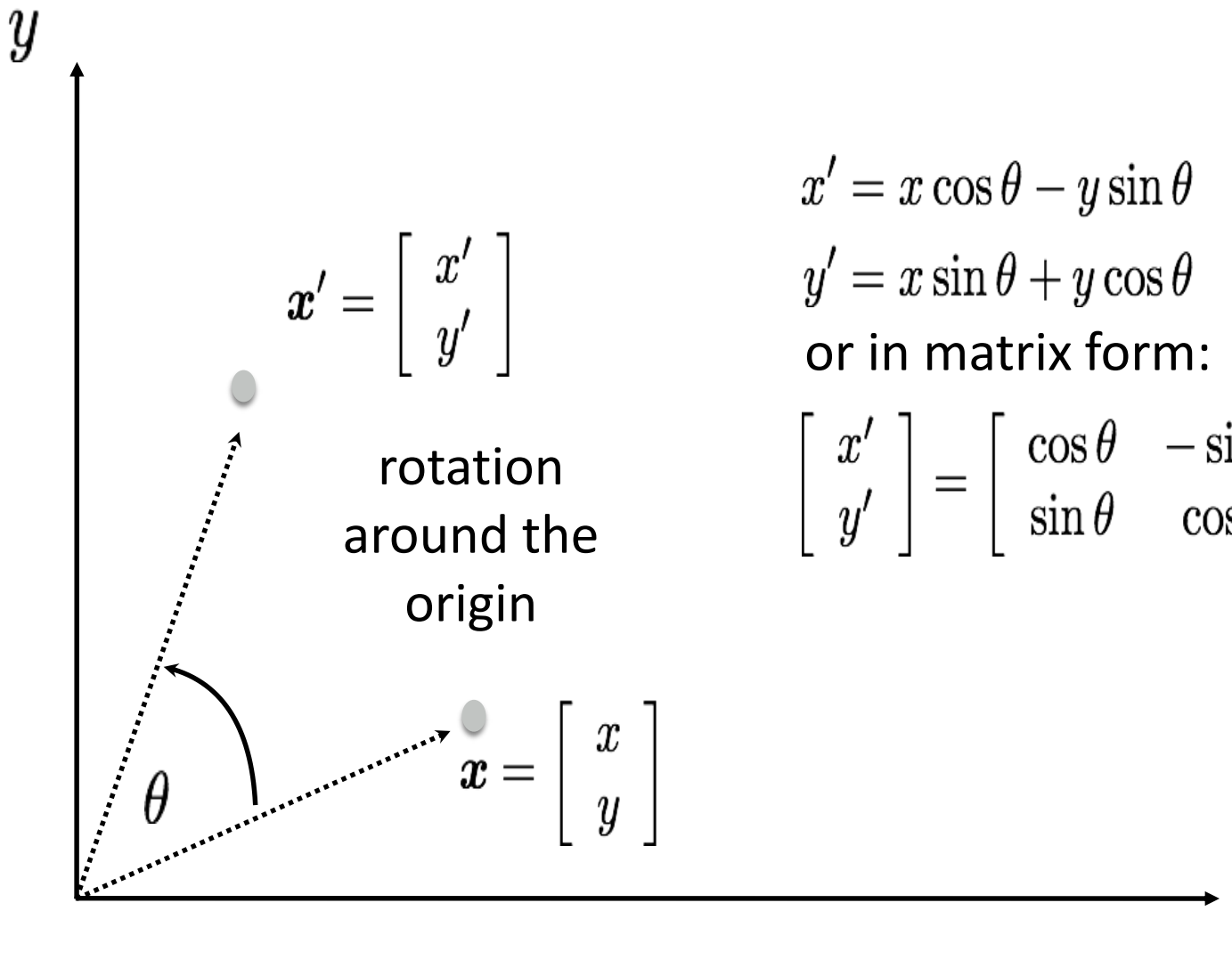
$y' = r \sin(\phi) \cos(\theta) + r \cos(\phi) \sin(\theta)$

Substitute…

$x' = x \cos(\theta) - y \sin(\theta)$

$y' = x \sin(\theta) + y \cos(\theta)$

Rotation



$$\boldsymbol{x}' = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

$$\boldsymbol{x} = \begin{bmatrix} x \\ y \end{bmatrix}$$

$$x' = x\cos\theta - y\sin\theta$$

$$y' = x\sin\theta + y\cos\theta$$

or in matrix form:

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

Concatenation

- How do we do concatenation of two or more transformations?

Concatenation

- How do we do concatenation of two or more transformations?

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \text{ and then } \begin{bmatrix} x'' \\ y'' \end{bmatrix} = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix}$$

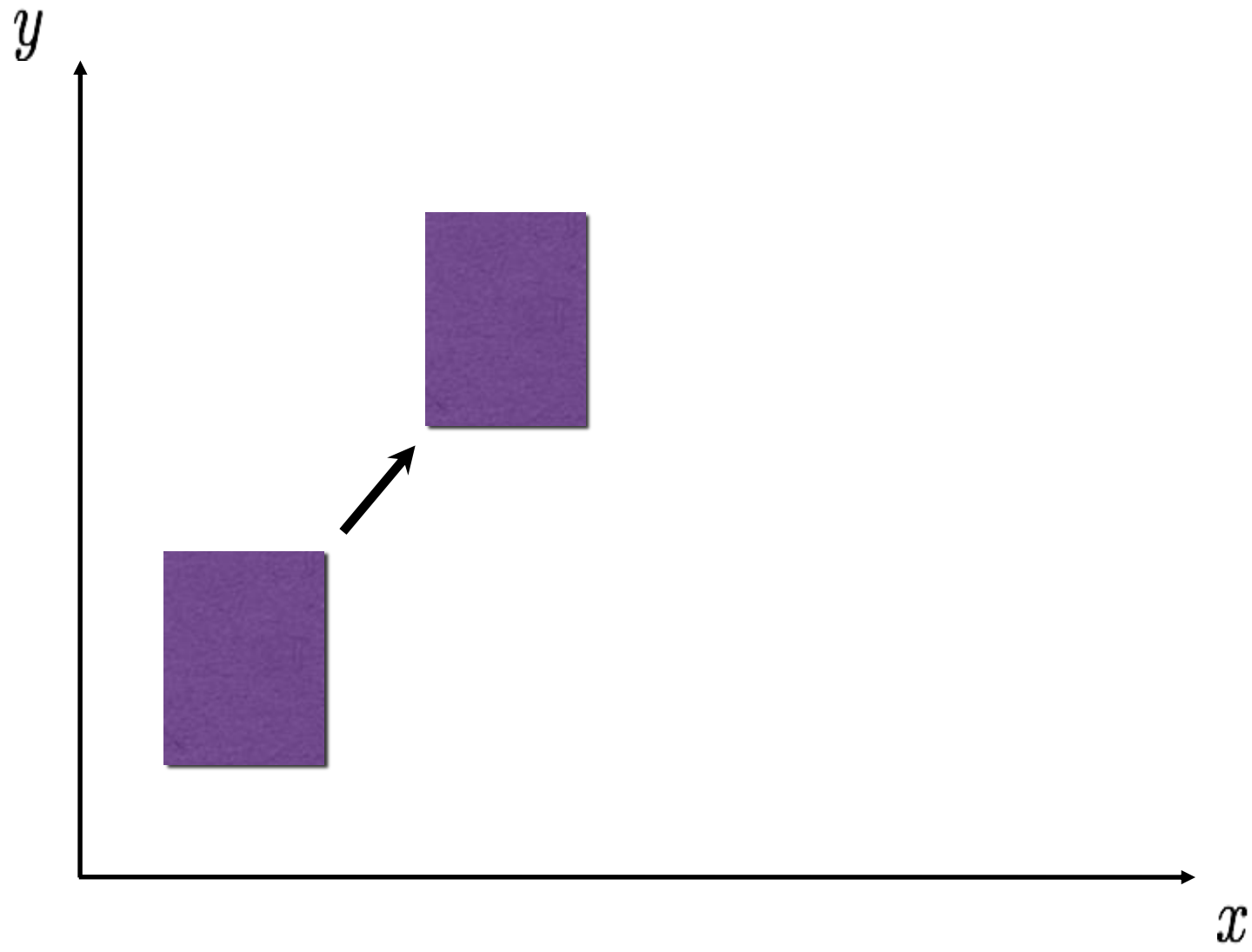
$$\mapsto$$

$$\begin{bmatrix} x'' \\ y'' \end{bmatrix} = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} acos\theta & -bsin\theta \\ asin\theta & bcos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

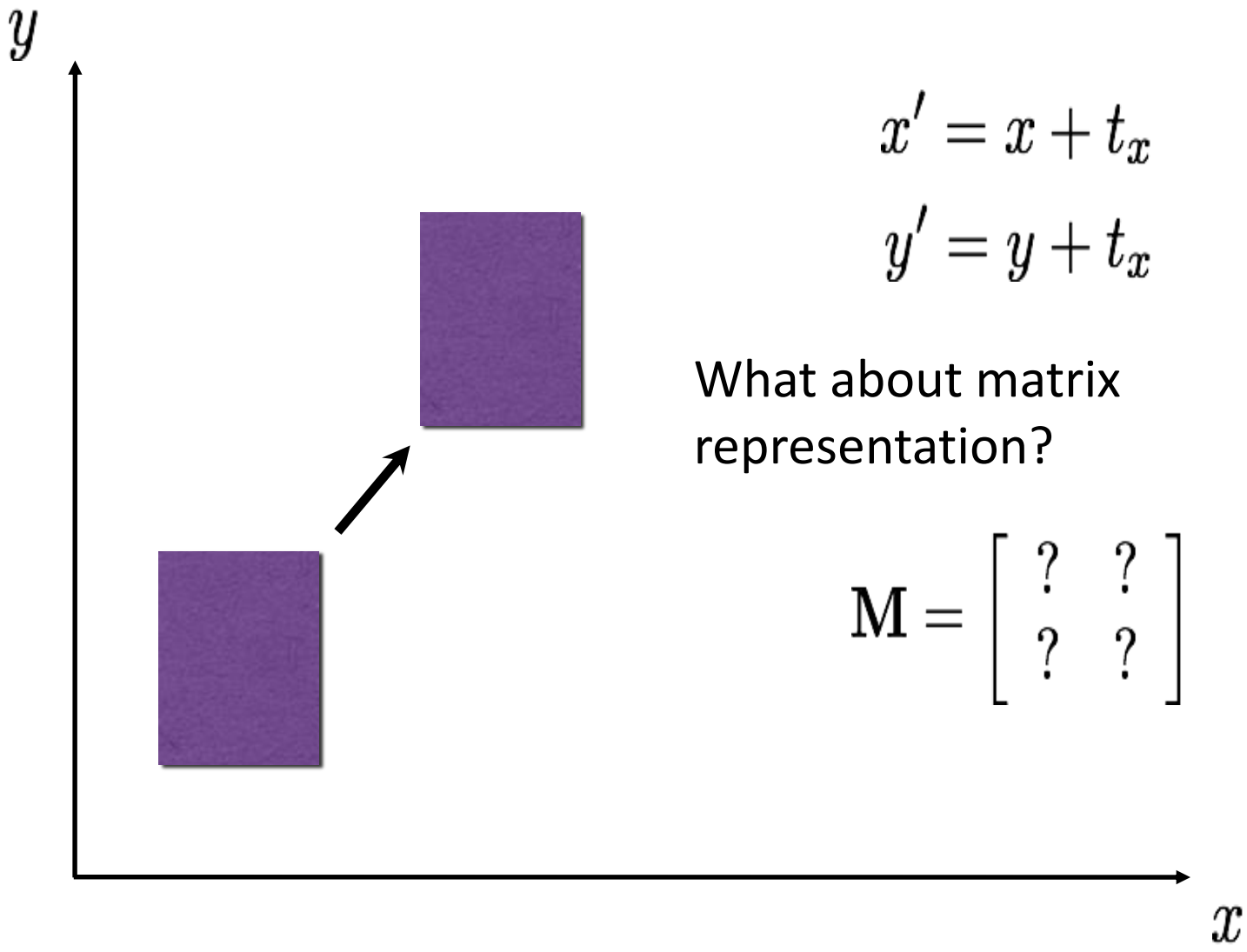
- Easy with matrix multiplication!

Translation

- How?



Translation



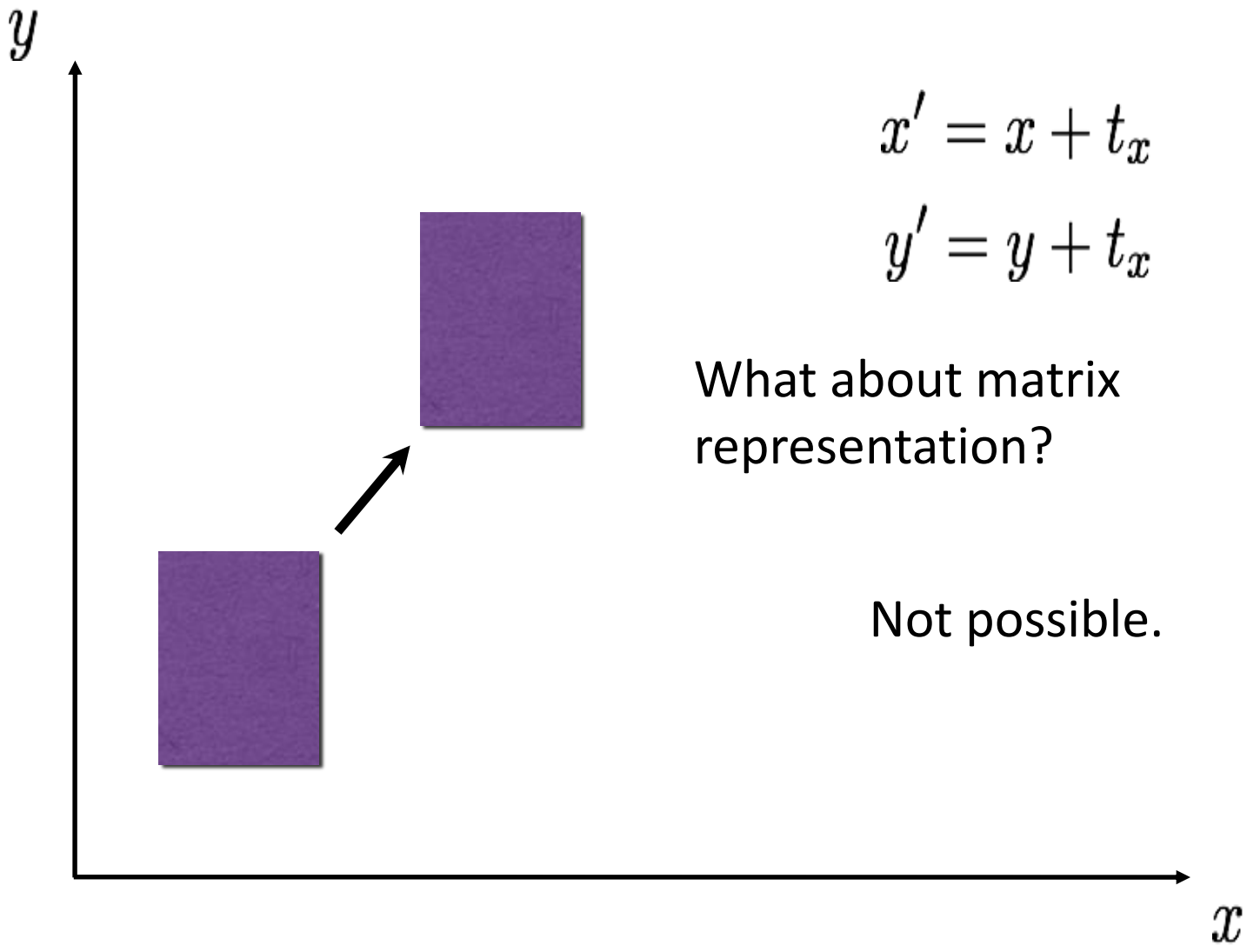
$$x' = x + t_x$$

$$y' = y + t_x$$

What about matrix representation?

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

Translation



$$x' = x + t_x$$

$$y' = y + t_x$$

What about matrix representation?

Not possible.

Homogeneous coordinates

- **Homogeneous coordinates** represent 2D point with a 3D vector.

heterogeneous coordinates homogeneous coordinates

$$\begin{bmatrix} x \\ y \end{bmatrix} \rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Homogeneous coordinates

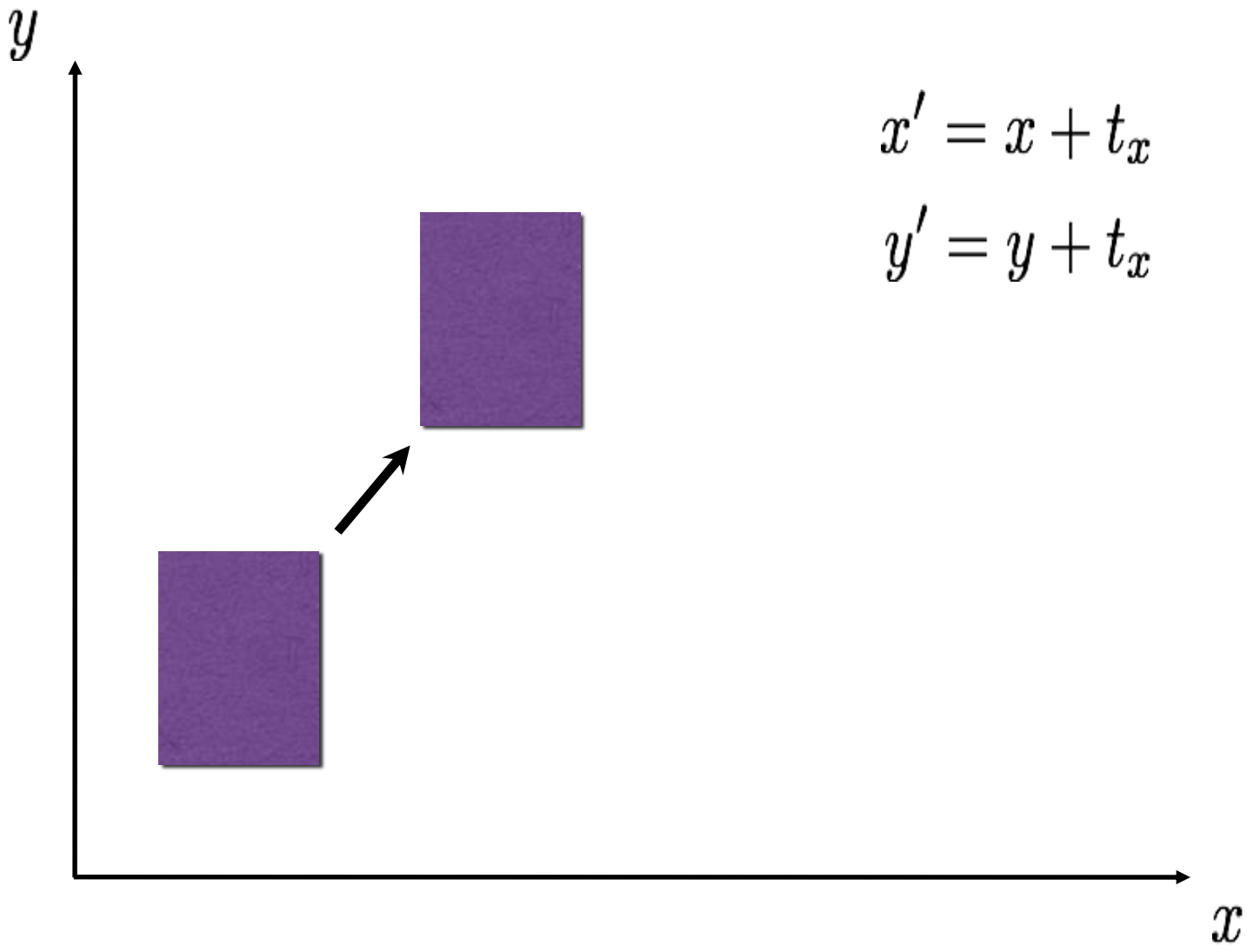
- **Homogeneous coordinates** represent 2D point with a 3D vector:
- Homogeneous coordinates are only defined up to scale.

heterogeneous coordinates homogeneous coordinates

$$\begin{bmatrix} x \\ y \end{bmatrix} \rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \overset{\text{def}}{=} \begin{bmatrix} ax \\ ay \\ a \end{bmatrix}$$

Translation

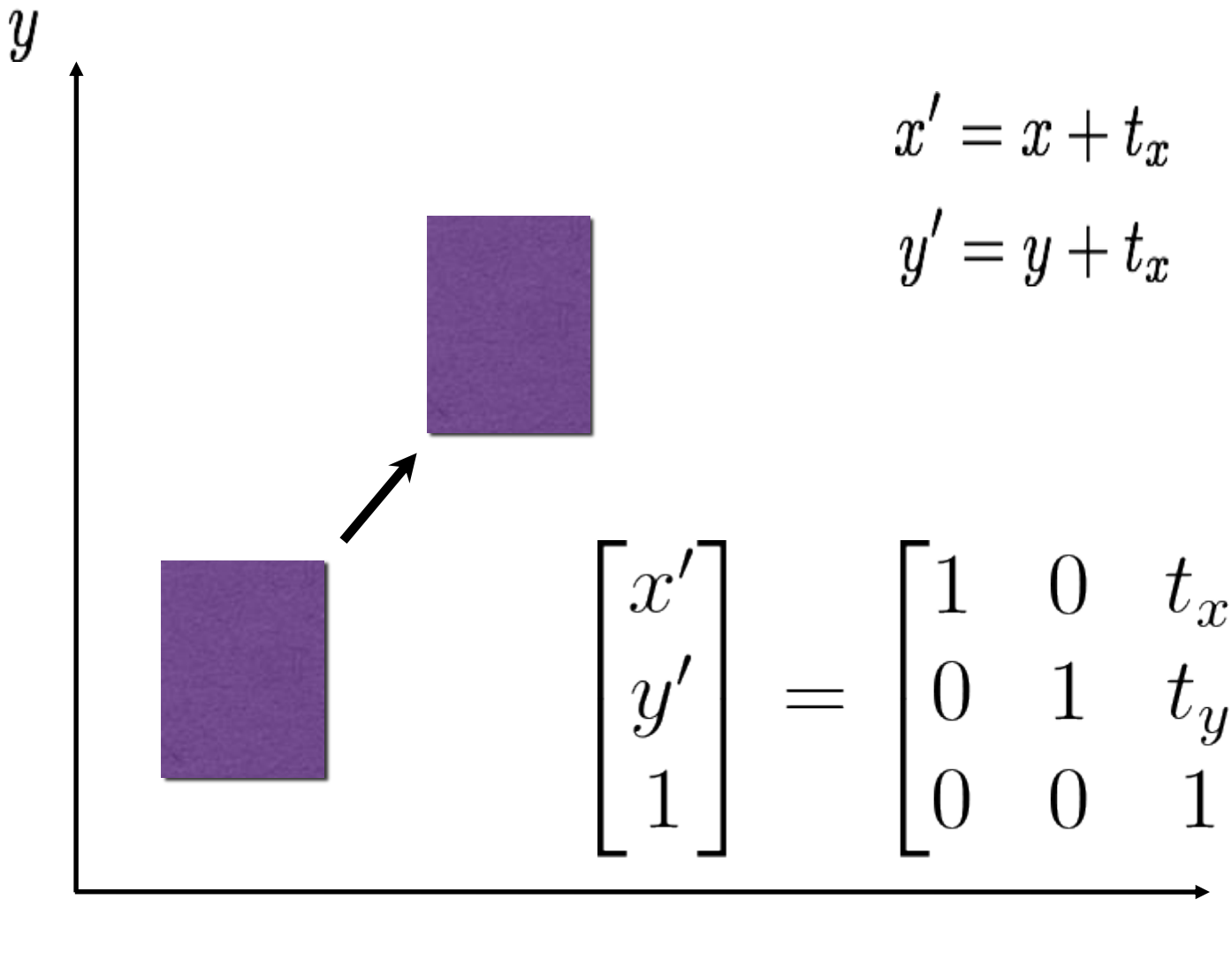
- How do we do it now?



$$x' = x + t_x$$

$$y' = y + t_x$$

Translation



$$x' = x + t_x$$

$$y' = y + t_x$$

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

Side note: linear transformation

- Linear transformation are Transformation that meets additively and scalar multiplication conditions:

$$f(\mathbf{u}+\mathbf{v}) = (\mathbf{u}) + f(\mathbf{v})$$
$$f(c\mathbf{u}) = cf(\mathbf{u})$$

- Translation is **not** a linear transformation since it doesn't meet the scalar multiplication condition.
- Properties of linear transformations:
  - Origin maps to origin
  - Lines map to lines
  - Parallel lines remain parallel
  - Ratios are preserved

Affine Transformations

- Affine transformations are combinations of linear transformations and translations

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

- Properties of affine transformations:
  - ~~Origin maps to origin~~
  - Lines map to lines
  - Parallel lines remain parallel
  - Ratios are preserved

Affine transformation: example

- Translate then scale vs. scale then translate :

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} a & 0 & at_x \\ 0 & b & bt_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

$$\neq$$

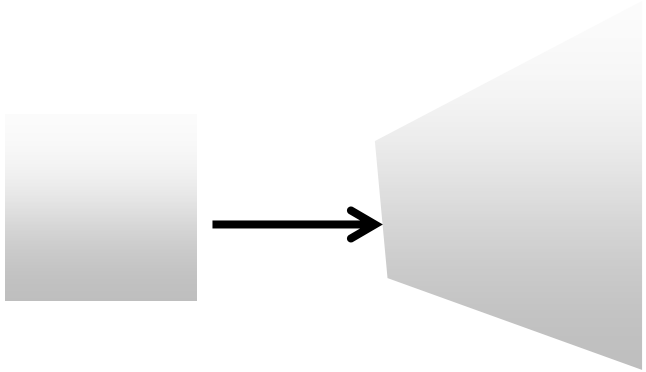
$$\begin{bmatrix} x'' \\ y'' \\ w \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} a & 0 & t_x \\ 0 & b & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix}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

- Order of matrices **DOES** matter ($A \cdot B \neq B \cdot A$)

Projective transformation

- Also known as **homography** or **homographic transformation**.
- A generalization of affine transformation.
- Properties of projective transformations:
  - ~~Origin maps to origin~~
  - Lines map to lines
  - ~~Parallel lines remain parallel~~
  - ~~Ratios are preserved~~

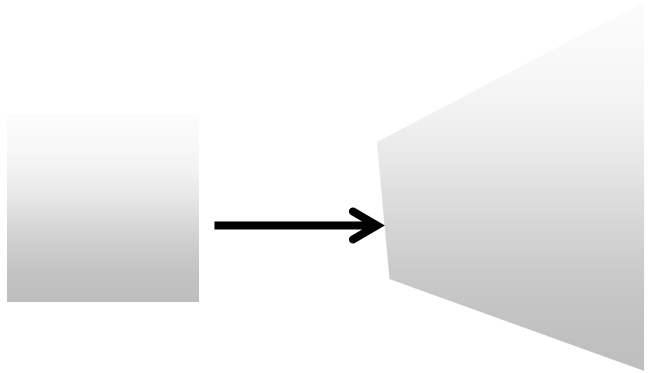
$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$



Projective transformation

- How many DOFs do we have here?

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



Projective transformation

- How many DOFs do we have here?
  - 8, since it is true up to scale (homogenous coordinates)

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