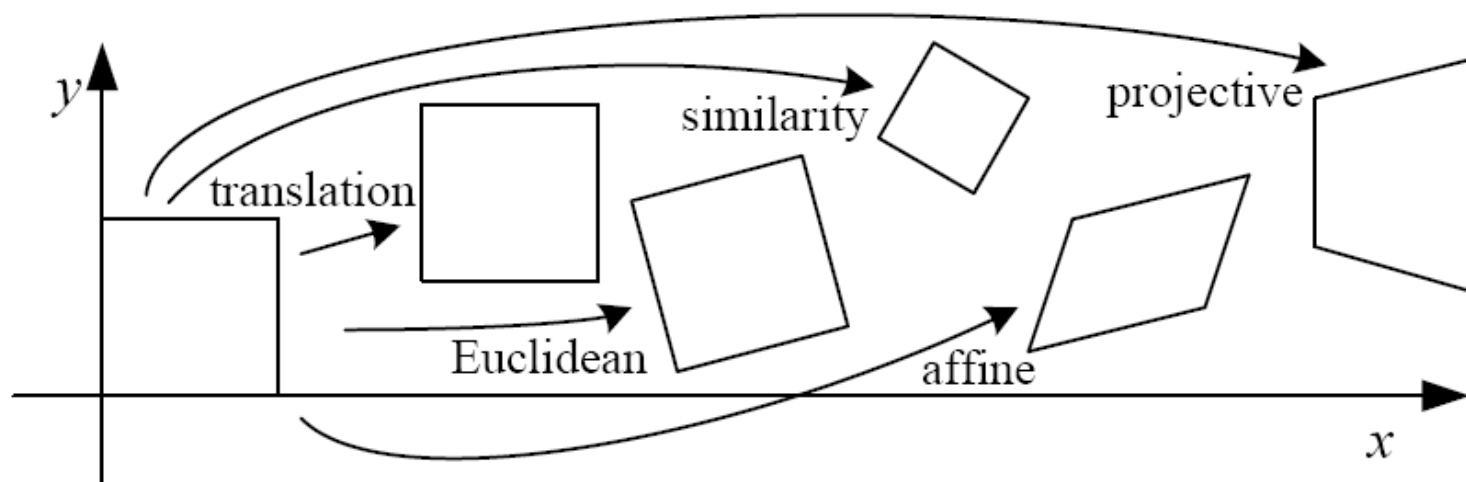


Geometric transformation



References

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

contents

- **2D->2D transformations**
- 3D->3D transformations
- 3D->2D transformations (3D projections)
 - Perspective projection
 - Orthographic projection

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



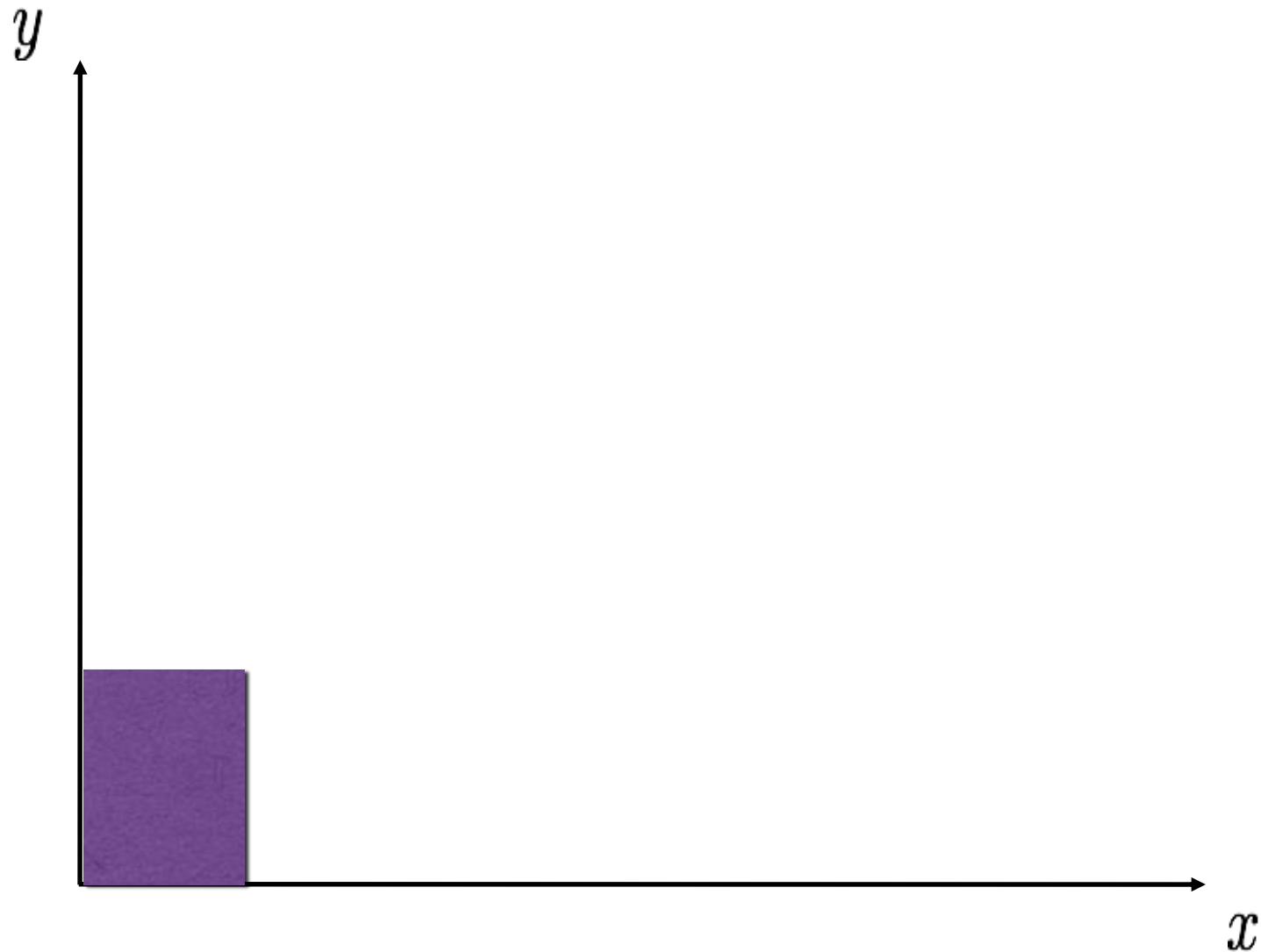
affine



projective

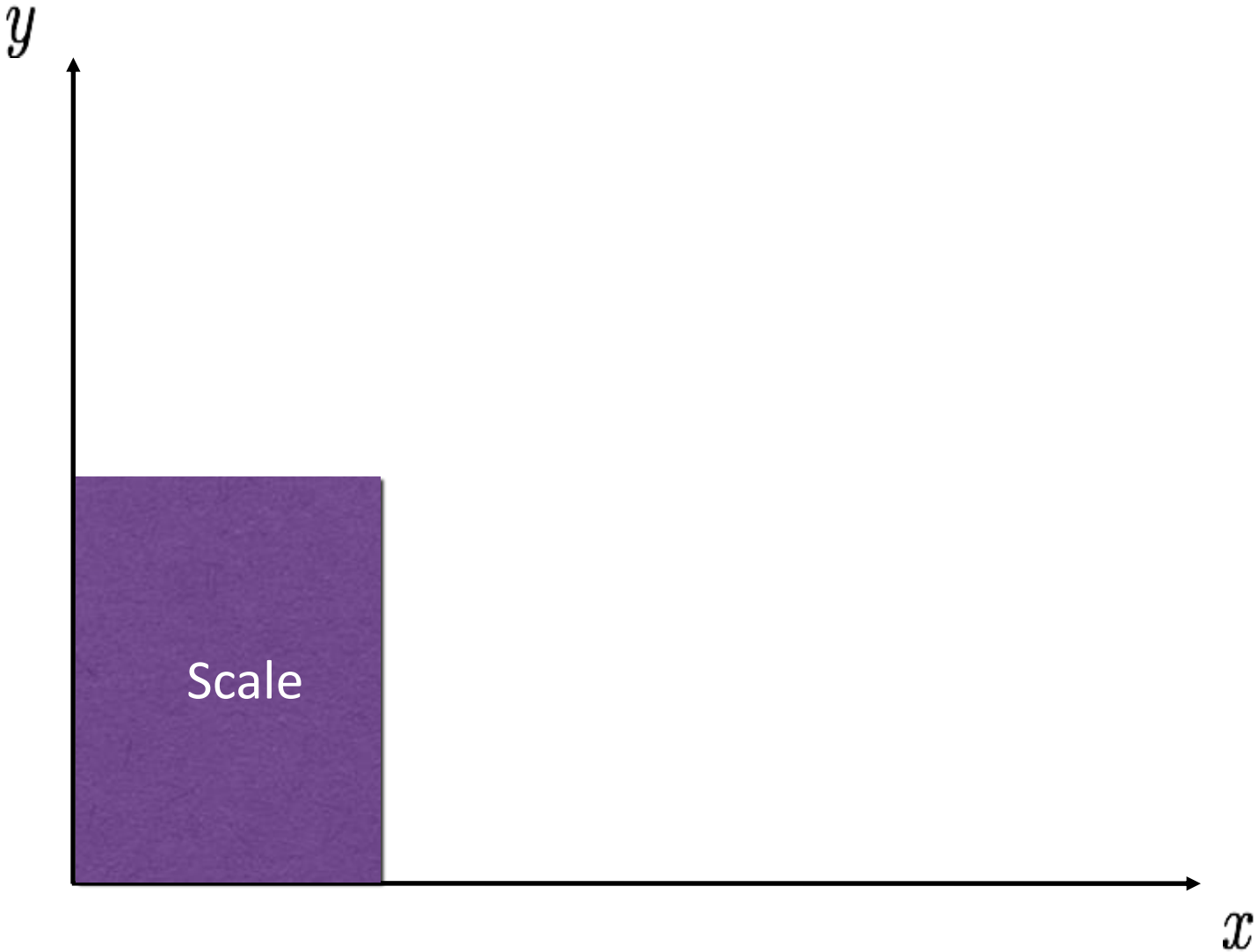
- **Why matrix manipulation?** Because then we can easily concatenate transformations (for example translation and rotation).

2D planar transformations



scale

- How?



scale

y

$$x' = ax$$

$$y' = by$$

Scale

x

scale

y



Scale

$$x' = ax$$

$$y' = by$$

matrix representation of
scaling:

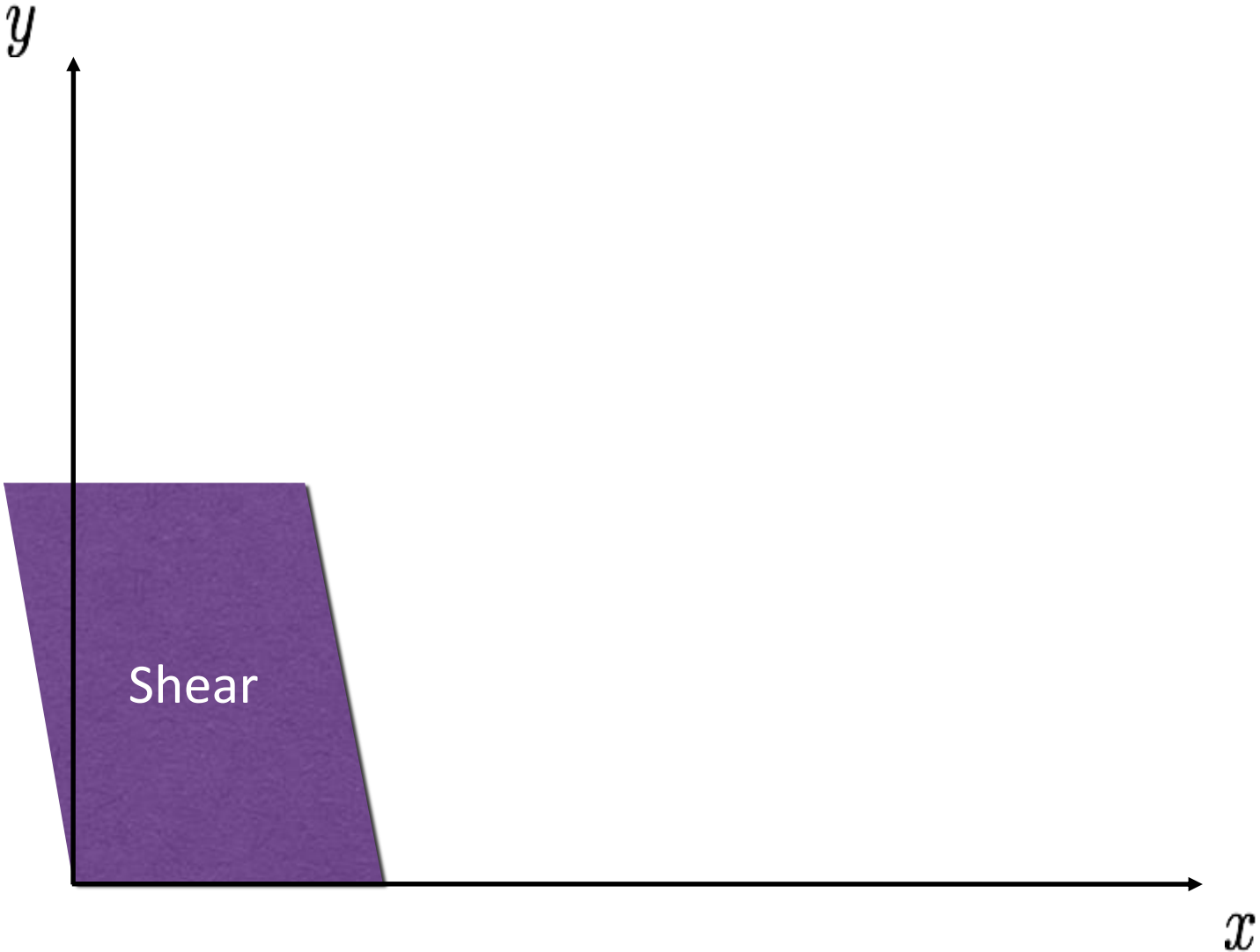
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}}_{\text{scaling matrix } S} \begin{bmatrix} x \\ y \end{bmatrix}$$

scaling matrix S

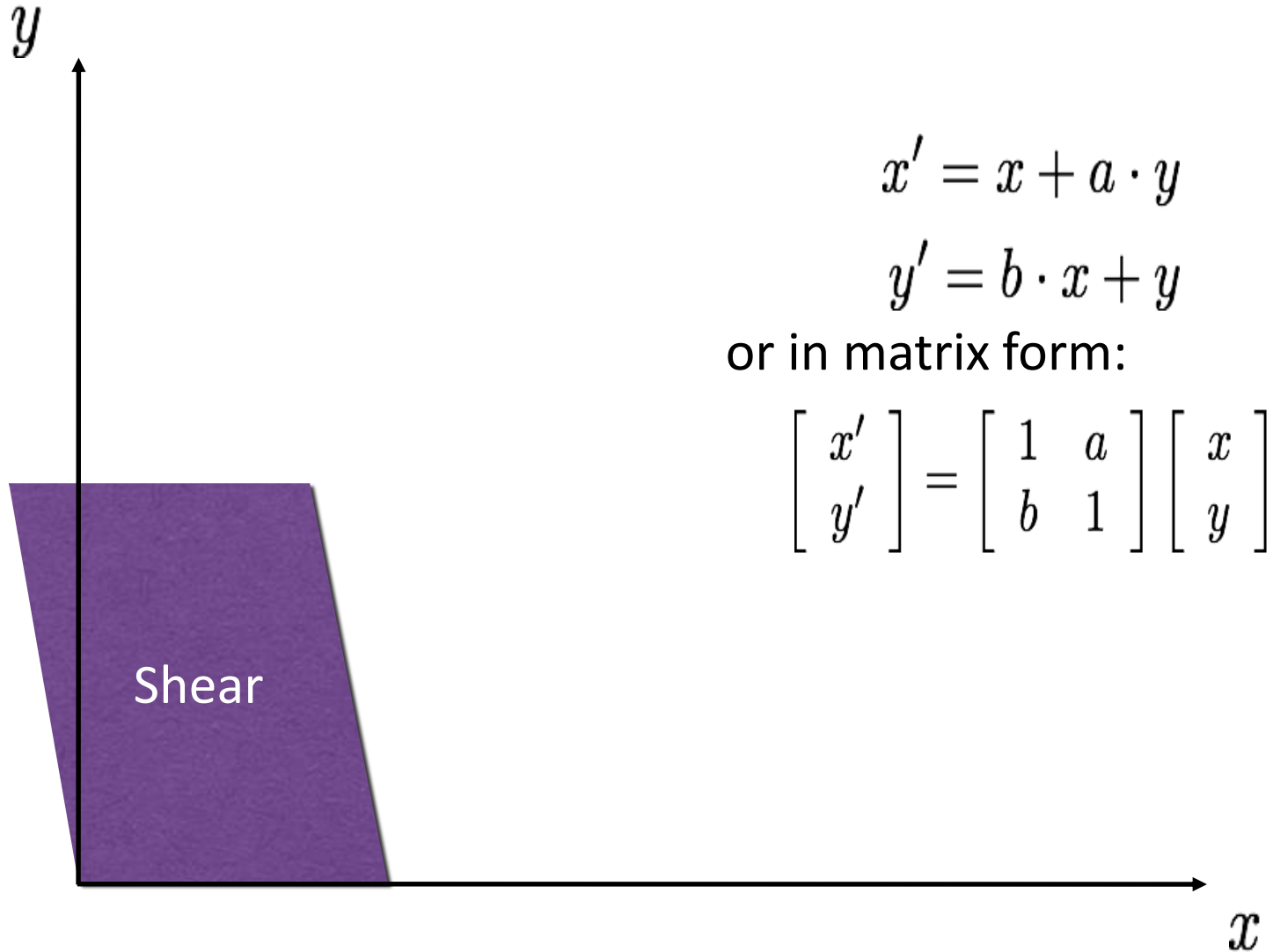
x

Shear

- How?

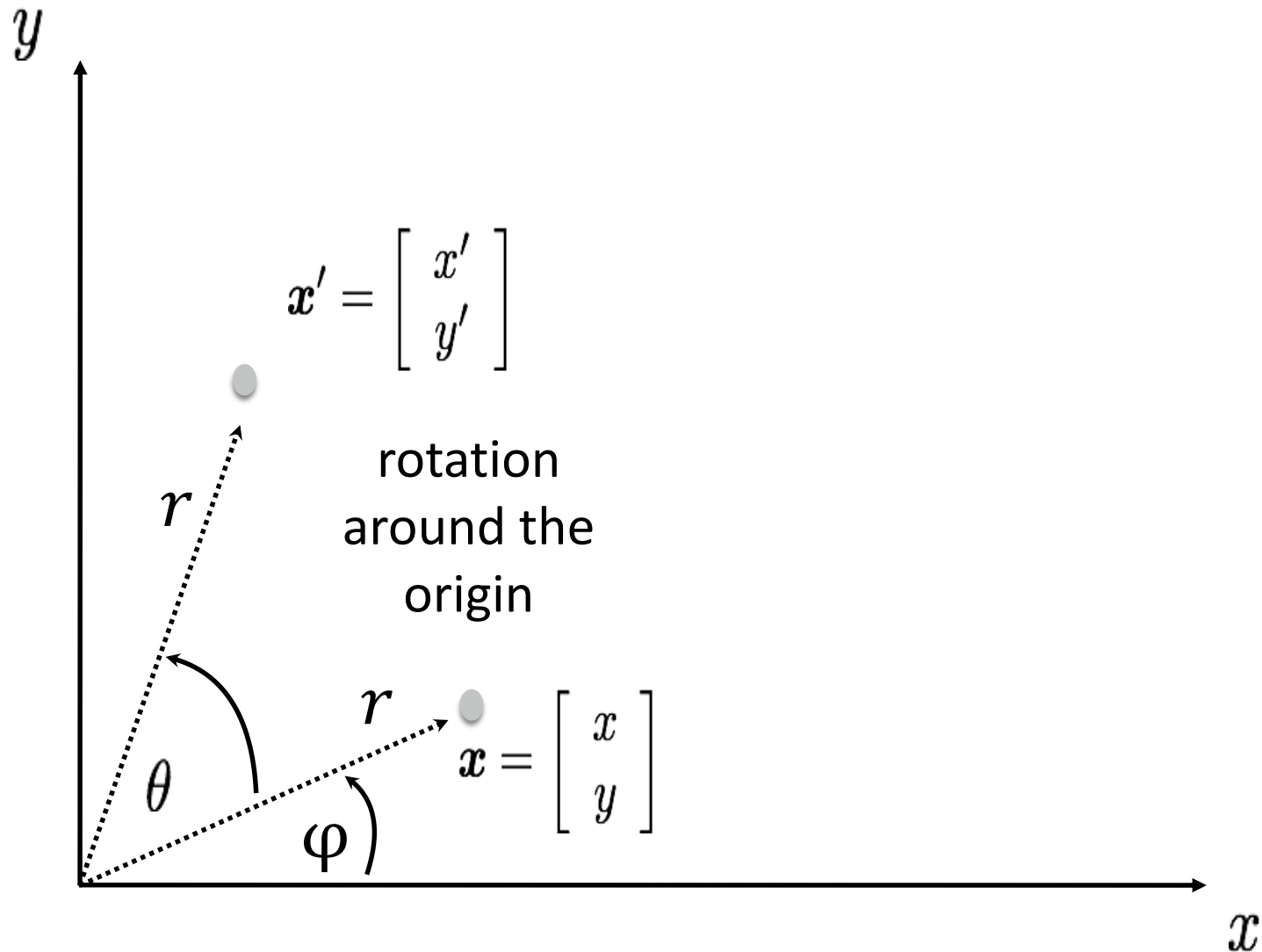


Shear

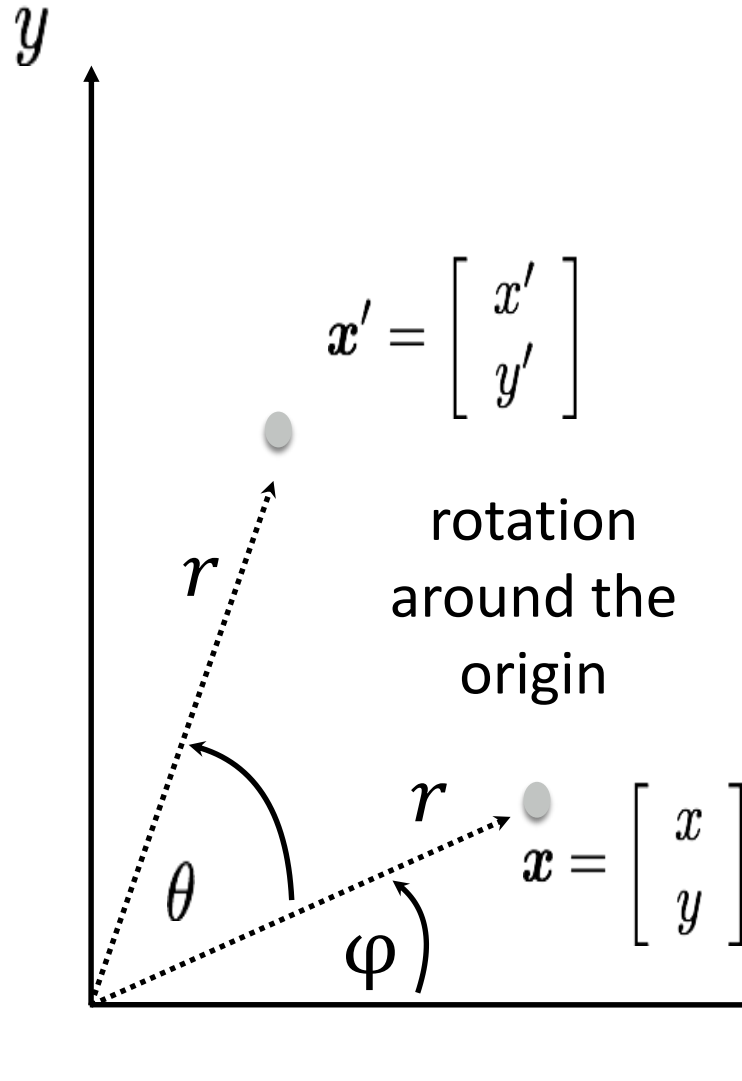


Rotation

- How?



Rotation



Polar coordinates...

$$x = r \cos(\phi)$$

$$y = r \sin(\phi)$$

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

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

Trigonometric Identity...

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

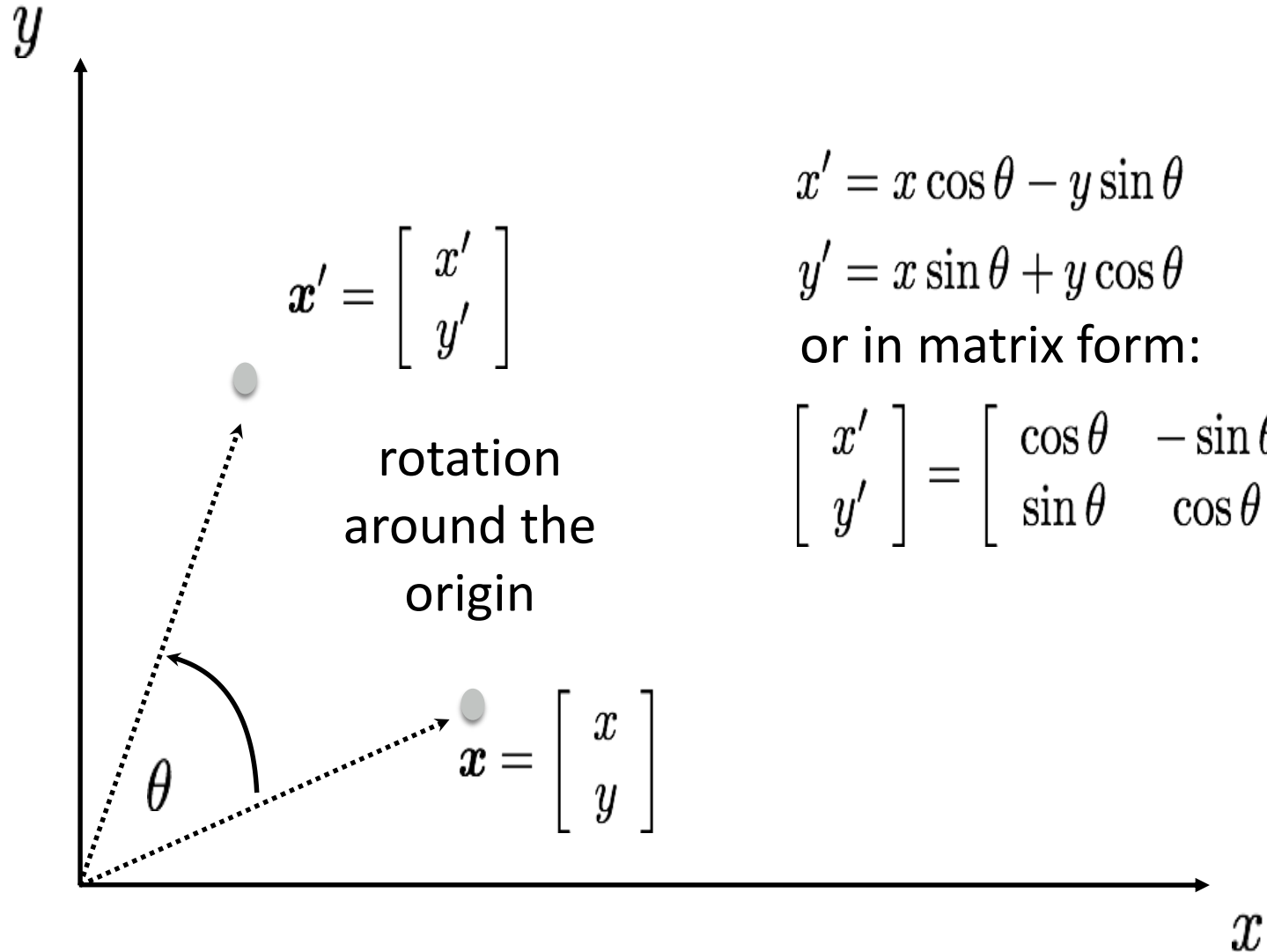
$$y' = r \sin(\phi) \cos(\theta) + r \cos(\phi) \sin(\theta)$$

Substitute...

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

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

Rotation



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

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

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?

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

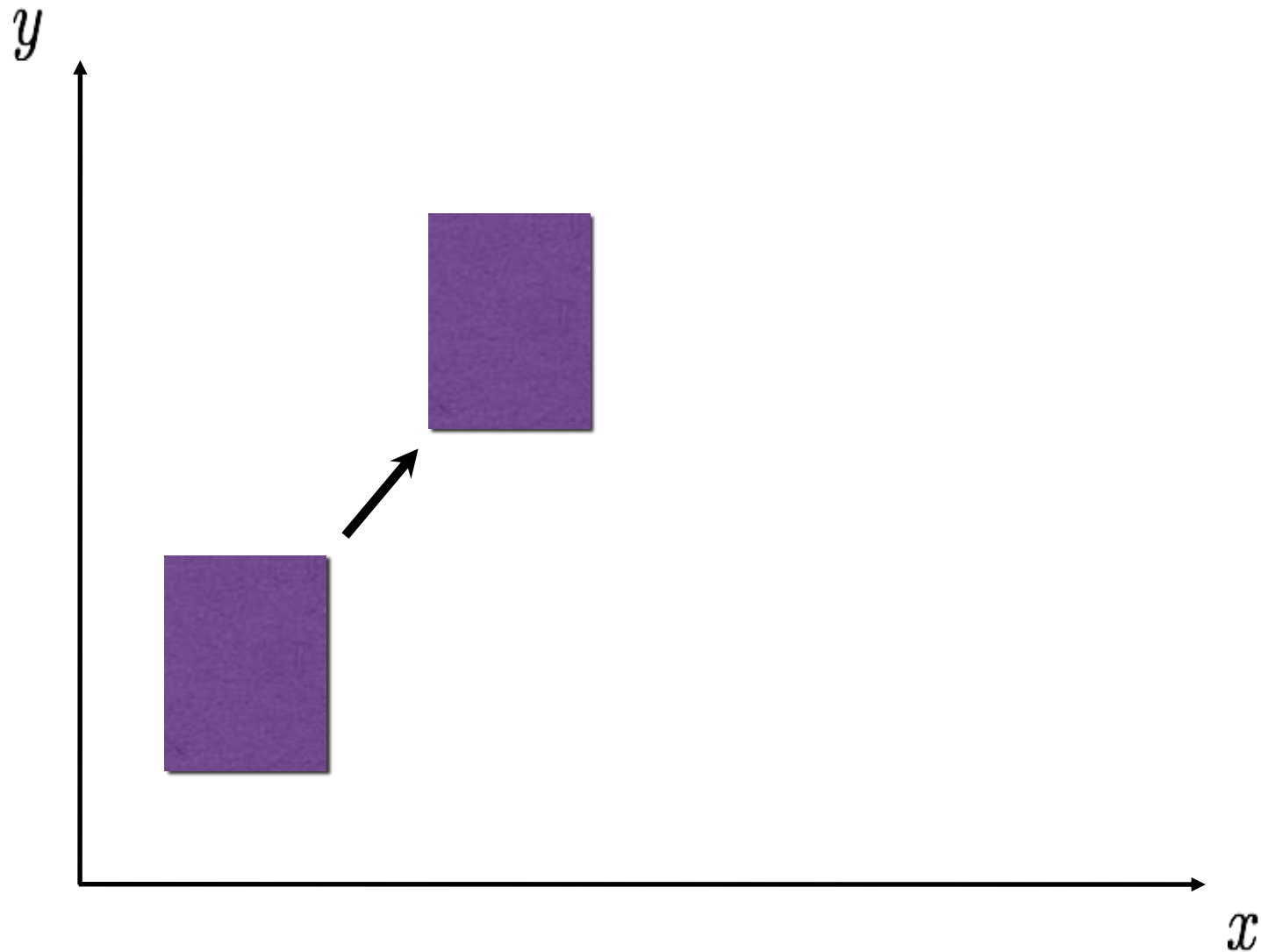
\mapsto

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

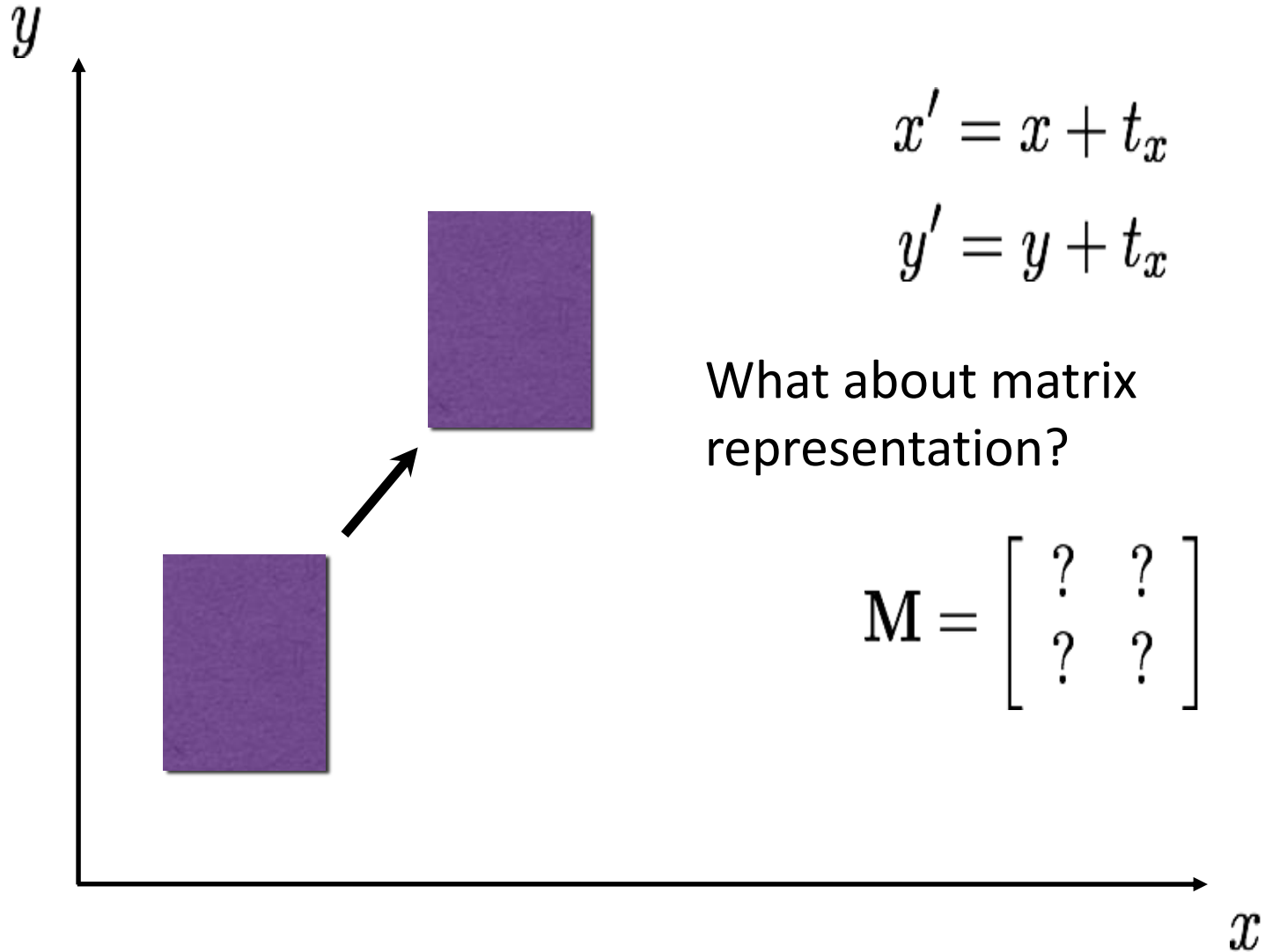
- Easy with matrix multiplication!

Translation

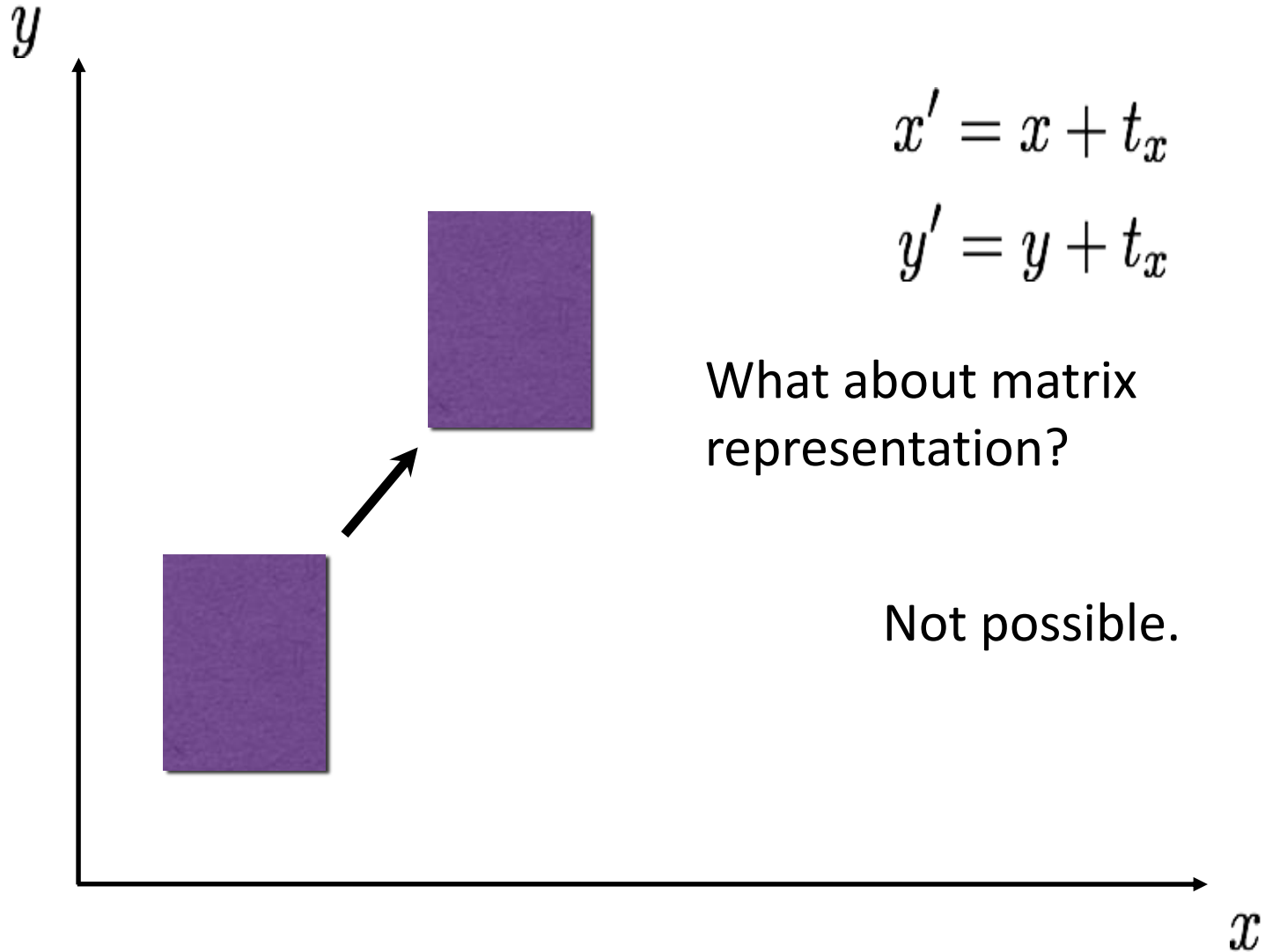
- How?



Translation



Translation



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

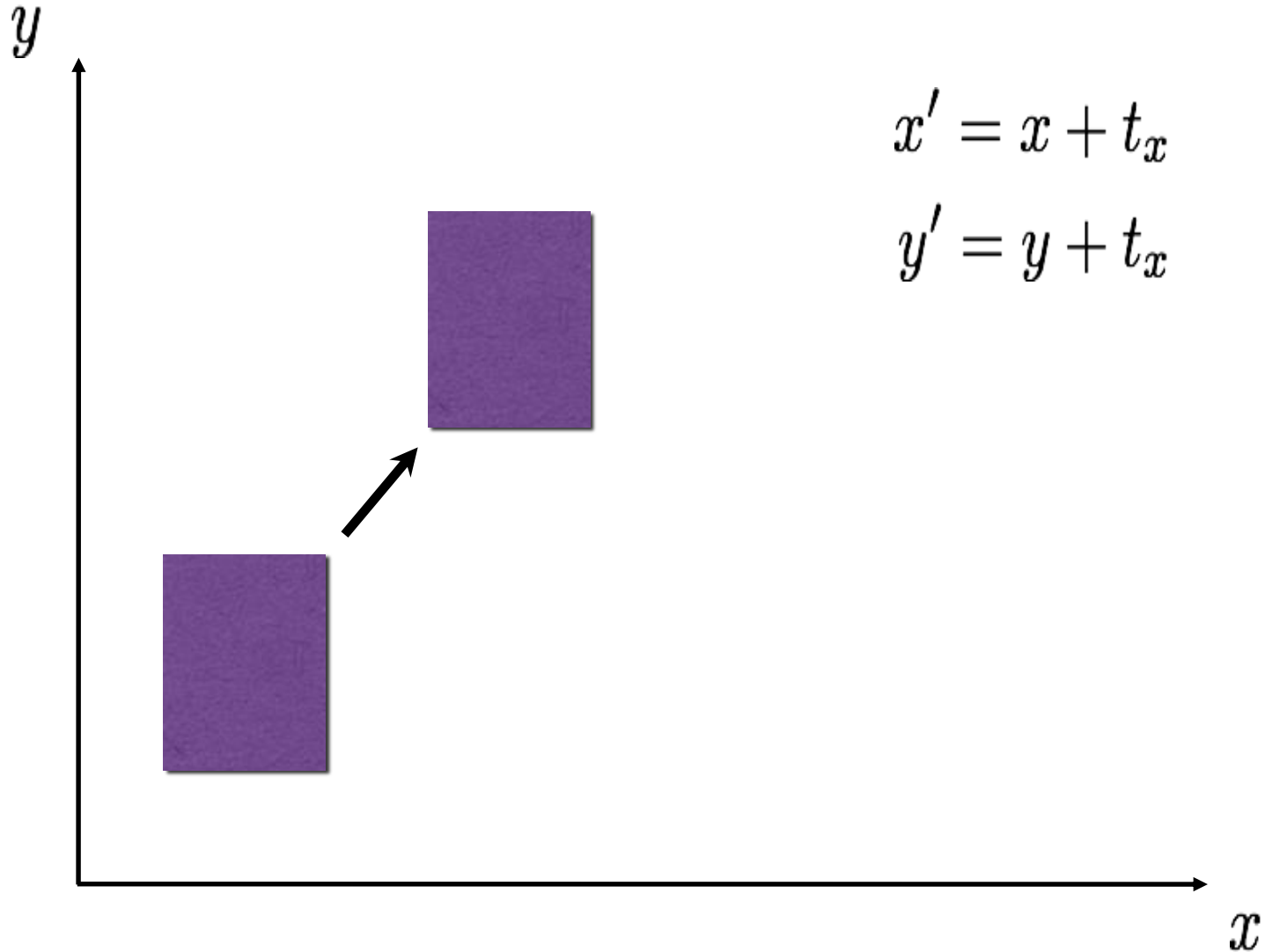
- **Homogeneous coordinates** represent 2D point with a 3D vector:
- Homogeneous coordinates are only defined up to scale.

heterogeneous homogeneous
coordinates coordinates

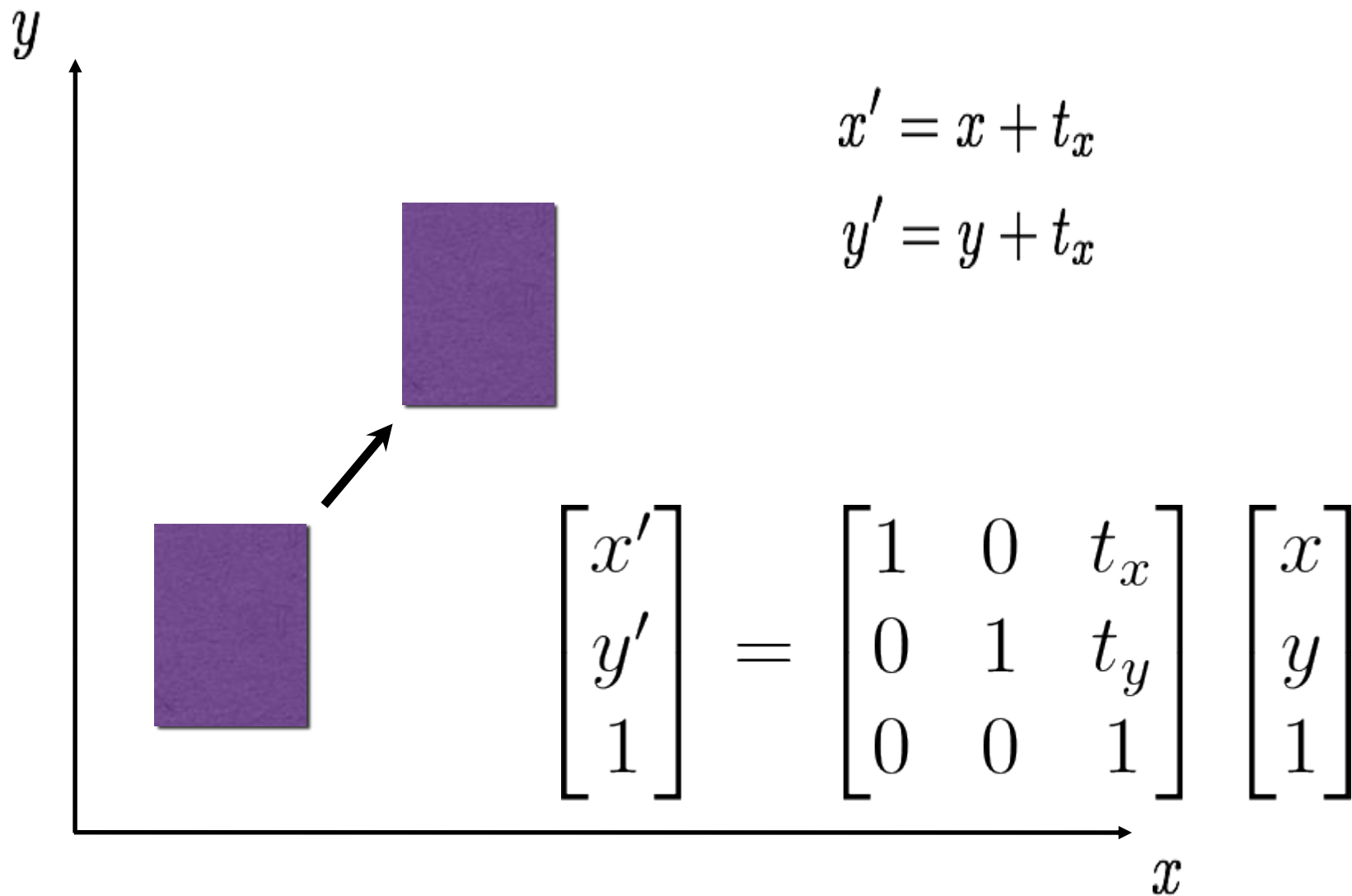
$$\begin{bmatrix} x \\ y \end{bmatrix} \rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \stackrel{\text{def}}{=} \begin{bmatrix} ax \\ ay \\ a \end{bmatrix}$$

Translation

- How do we do it now?



Translation



Side note: linear transformation

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

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

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

- 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

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

- 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

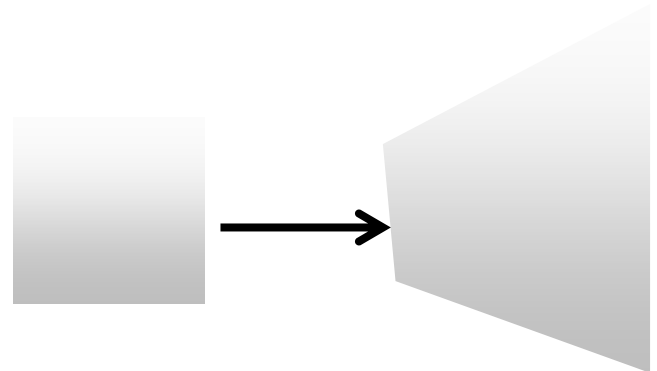
$$\begin{bmatrix} x'' \\ y'' \\ w \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} a & 0 & t_x \\ 0 & b & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix}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

- 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~~

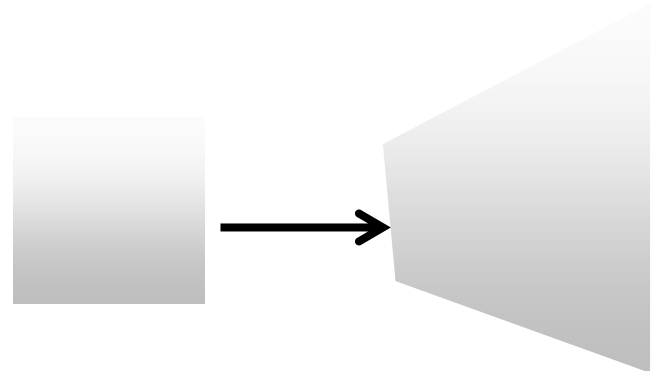
$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$



Projective transformation

- How many DOFs do we have here?

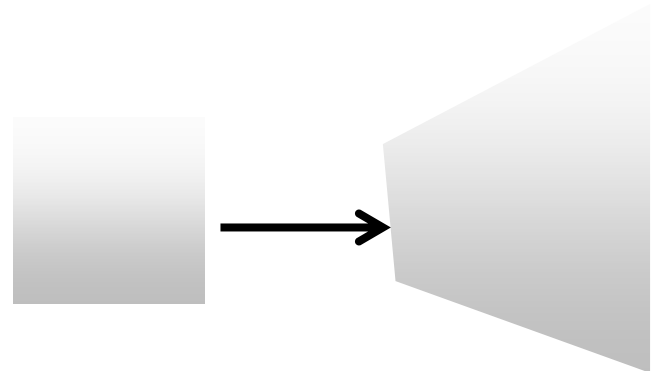
$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$



Projective transformation

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

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



contents

- 2D->2D transformations
- **3D->3D transformations**
- 3D->2D transformations (3D projections)
 - Perspective projection
 - Orthographic projection

3D->3D transformations

- Exactly the same as 2D->2D transformations from earlier, just with 3D.
- What do we see here?

$$\begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & \cos\theta & -\sin\theta & t_y \\ 0 & \sin\theta & \cos\theta & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix}$$

3D->3D transformations

- Exactly the same as 2D->2D transformations from earlier, just with 3D.
- What do we see here?

$$\begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & \cos\theta & -\sin\theta & t_y \\ 0 & \sin\theta & \cos\theta & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix}$$

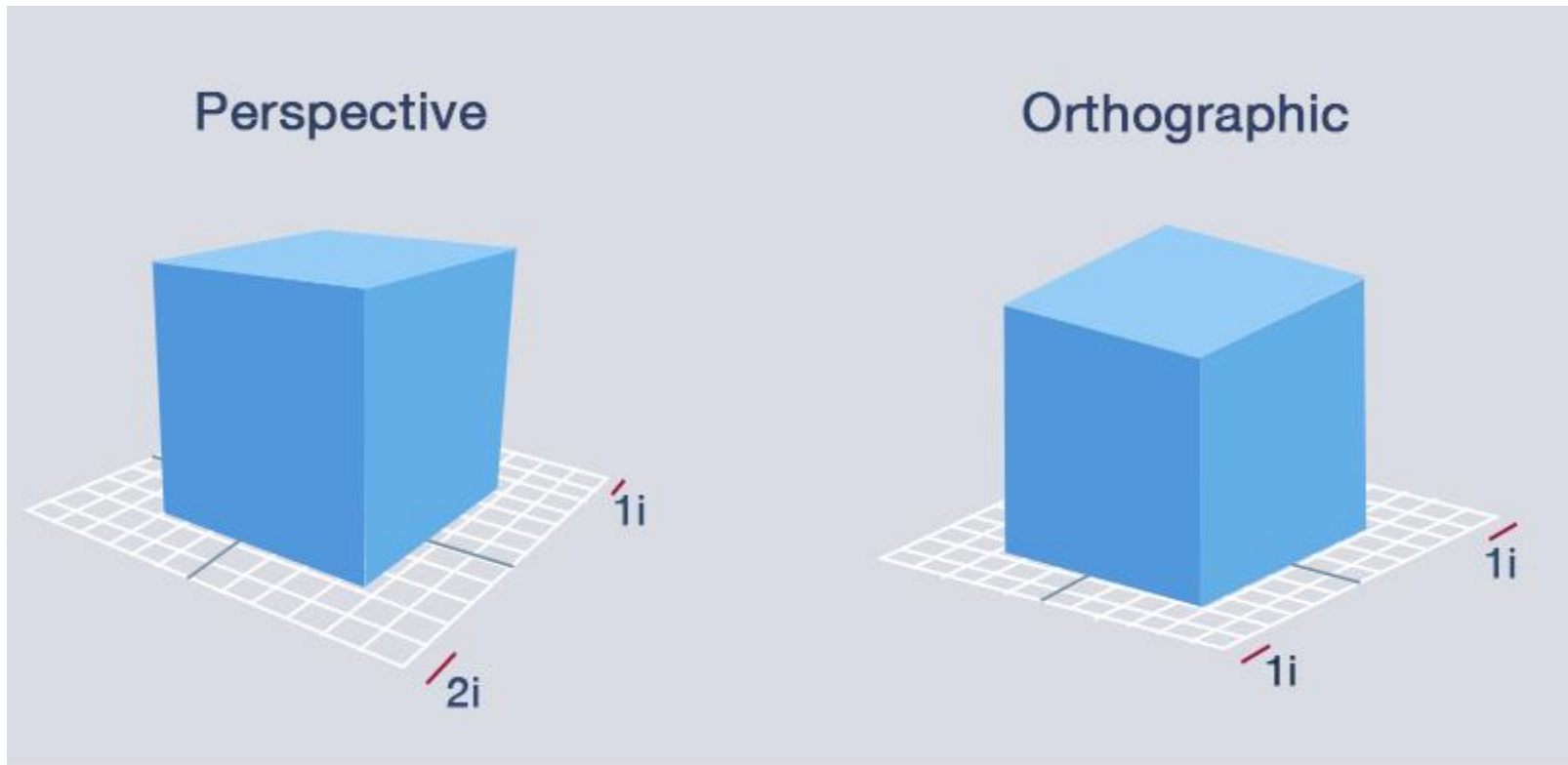
– Rotation around x axis and then translation

contents

- 2D->2D transformations
- 3D->3D transformations
- 3D->2D transformations (3D projections)
 - **Perspective projection**
 - Orthographic projection

3D projection

- **3D projection** is any method of mapping three-dimensional points to a two-dimensional plane.
- Two types of projections are **orthographic** and **perspective**.

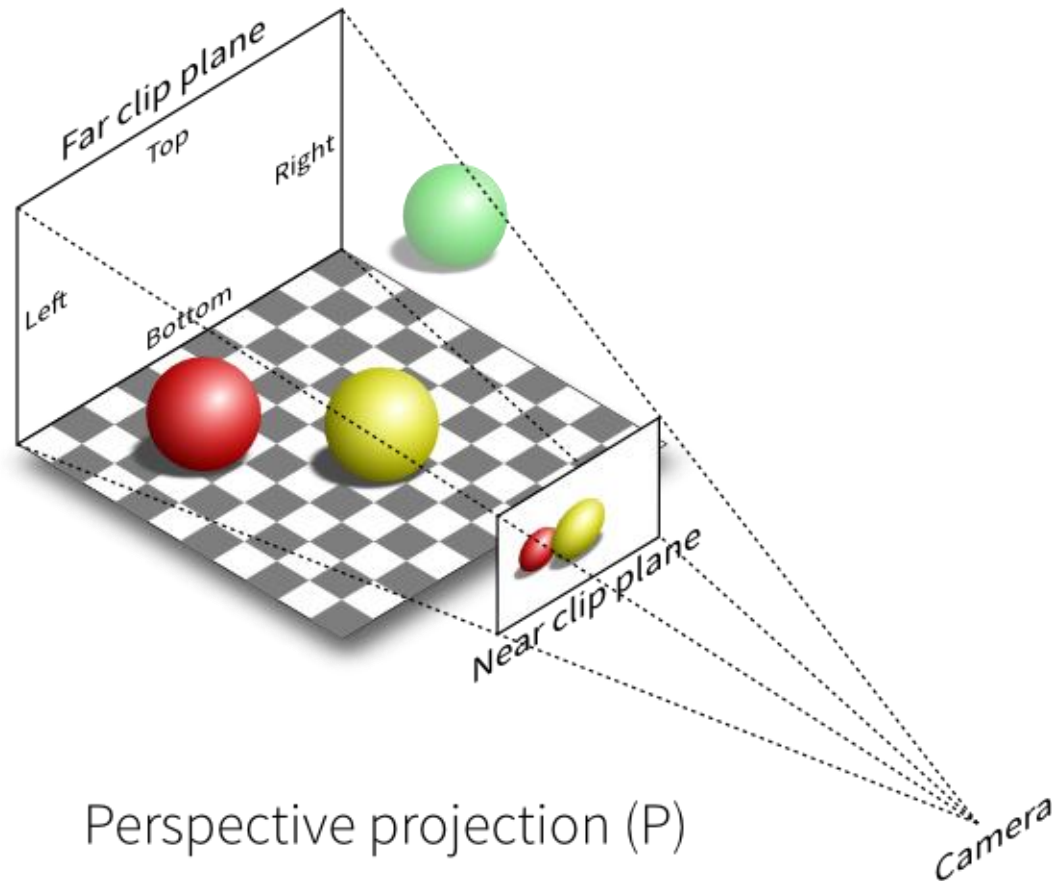


Perspective- definition

1. the art of drawing solid objects on a two-dimensional surface so as to give the right impression of their height, width, depth, and position in relation to each other when viewed from a particular point.
2. a particular attitude toward or way of regarding something; a point of view.

Perspective projection

- Perspective projection is the kind of projection we get from a regular image of a regular (pinhole) camera.



perspective manipulation



Street art- perspective manipulation



perspective manipulation- Ames Room

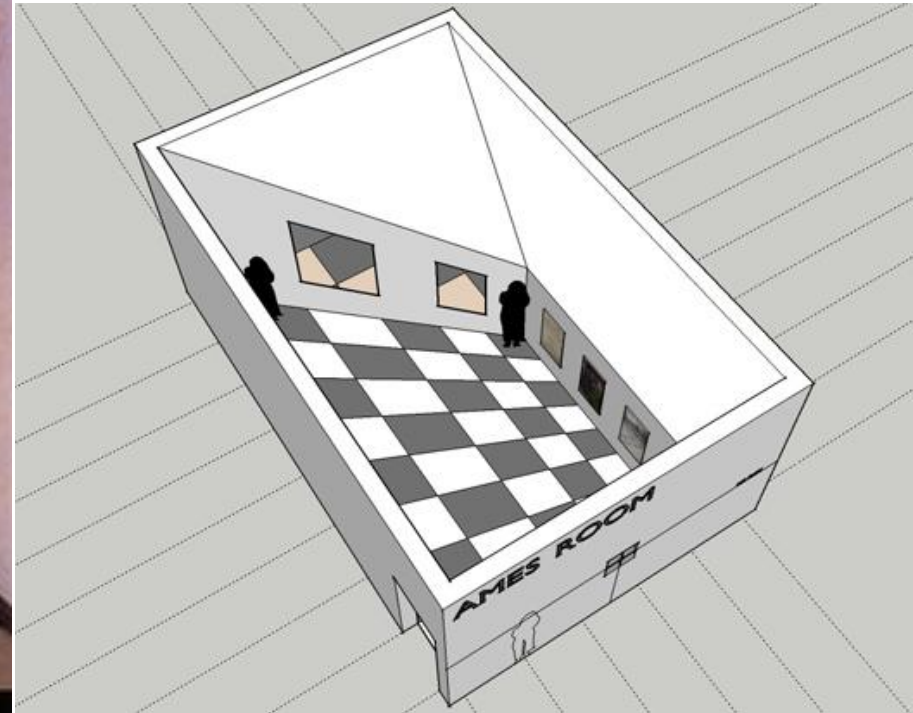
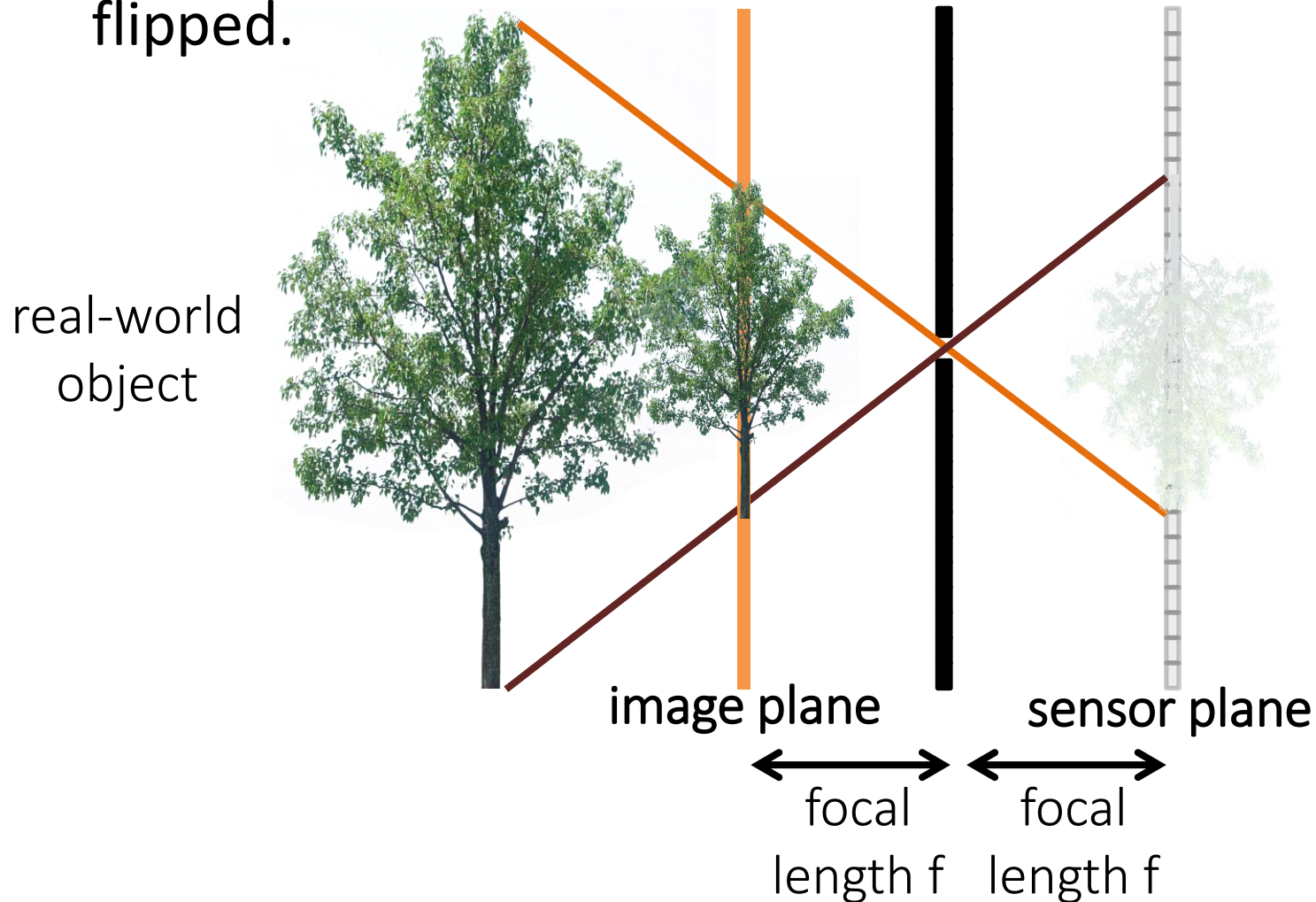


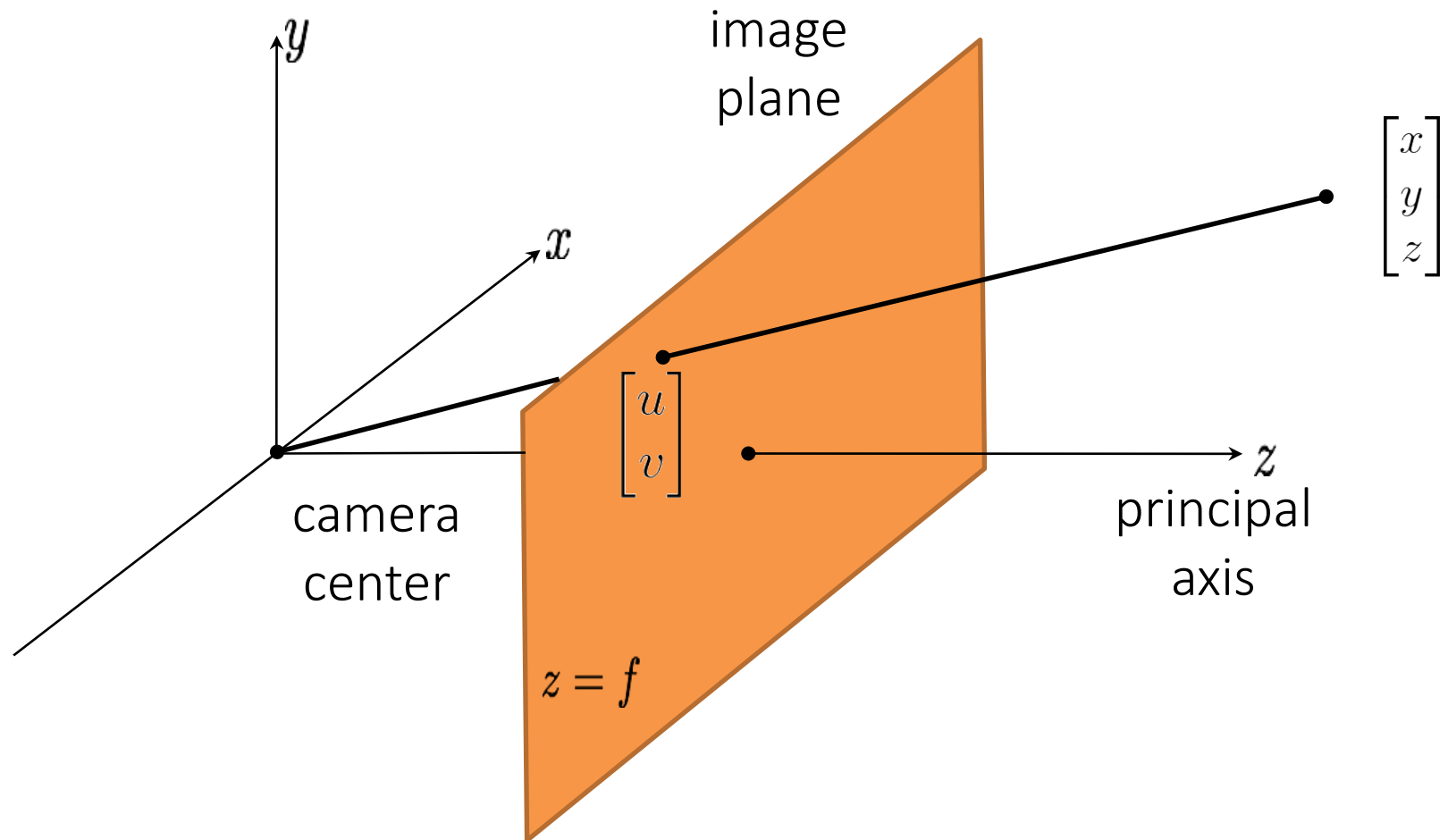
Image plane

- When dealing with imaged 3D scenes, we tend to use the **image plane** rather than the sensor plane which is flipped.



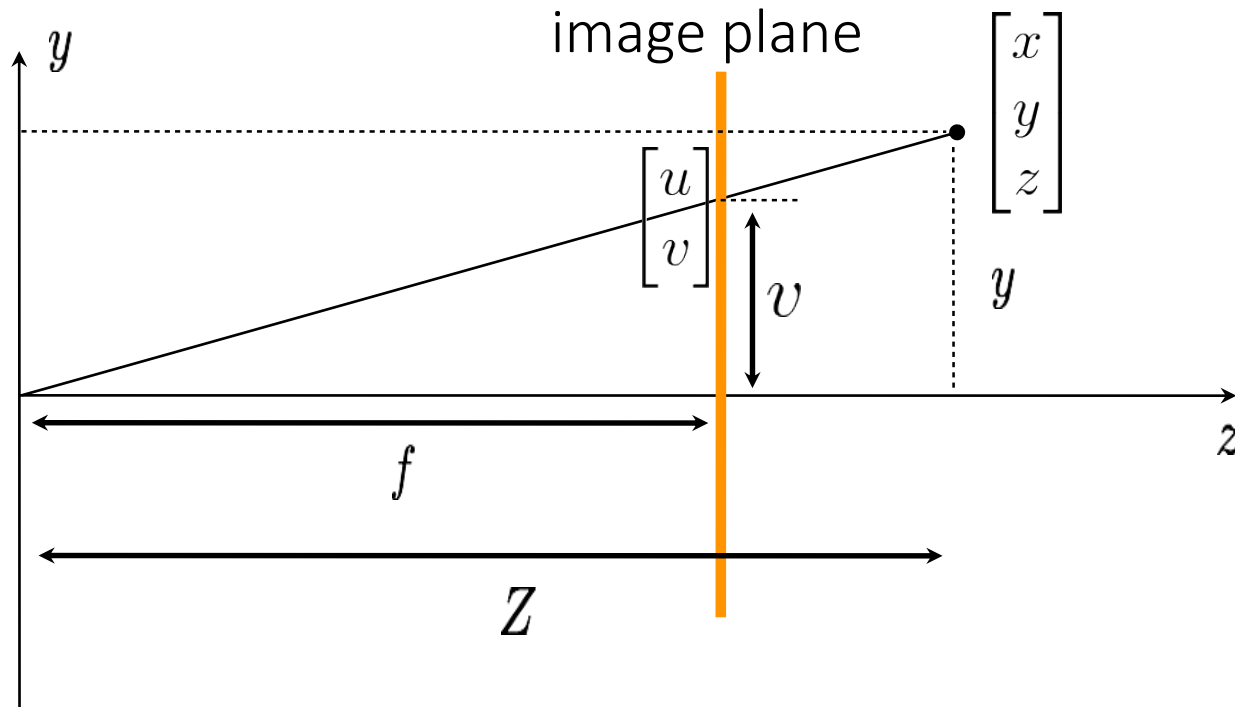
Perspective projection

- **Perspective projection** (also known as **perspective transformation**) is a linear projection where three dimensional objects are projected on the image plane.



Perspective projection

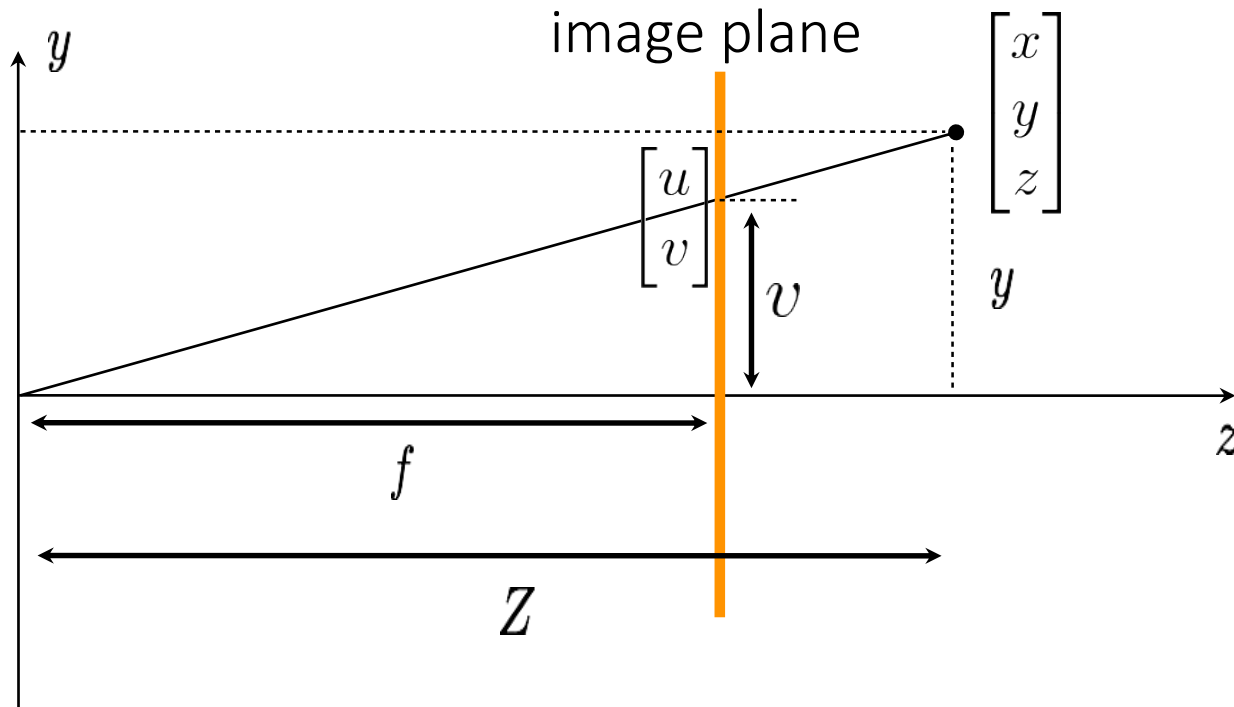
- What is the relationship between y & v ?



Perspective projection

- Using triangle proportions (Thales' theorem) we can easily conclude that:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} \mapsto \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} f \frac{x}{z} \\ f \frac{y}{z} \end{bmatrix}$$



Perspective projection

- Let's use the homogeneous coordinates:

$$\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} fx \\ fy \\ z \end{bmatrix} \mapsto \begin{bmatrix} f \frac{x}{z} \\ f \frac{y}{z} \\ z \end{bmatrix}$$

– Units of $[m]$

Perspective projection

- Let's split into 2 matrices and use 3D->2D homogenous coordinates:

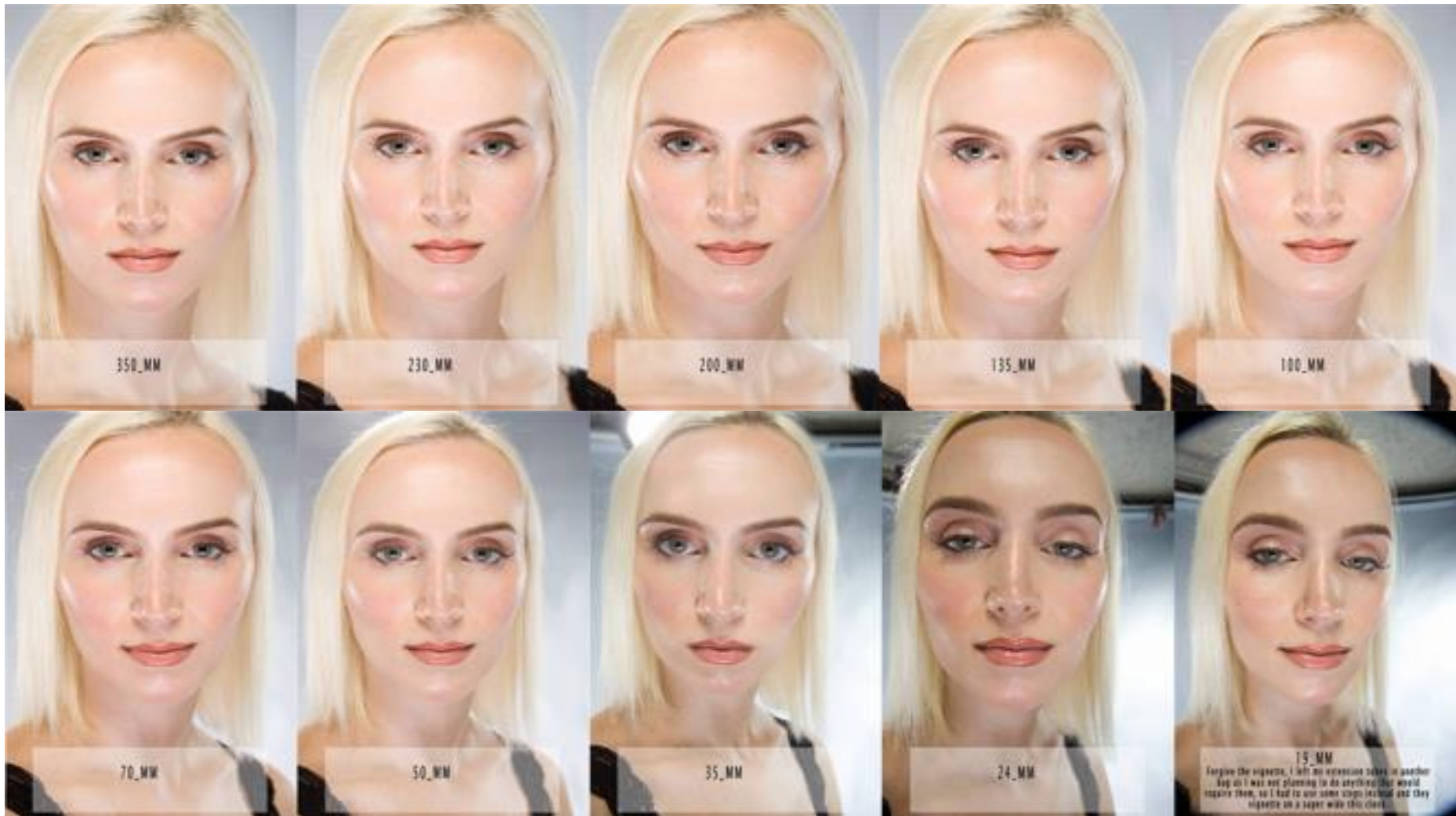
$$\underbrace{\begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Intrinsic camera matrix}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{Projection matrix}} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} fx \\ fy \\ z \end{bmatrix} \mapsto \begin{bmatrix} f \frac{x}{z} \\ f \frac{y}{z} \end{bmatrix}$$

Vertigo effect

- Has several different names (**vertigo effect, dolly zoom, lens compression, perspective distortion**) that all mean the same thing.
- intro:
 - <https://www.youtube.com/watch?v=UrhtKvBMZ3g> (until 01:50)

Vertigo effect

https://www.youtube.com/watch?time_continue=168&v=TXY1Se0eg (until 02:55)



Vertigo effect

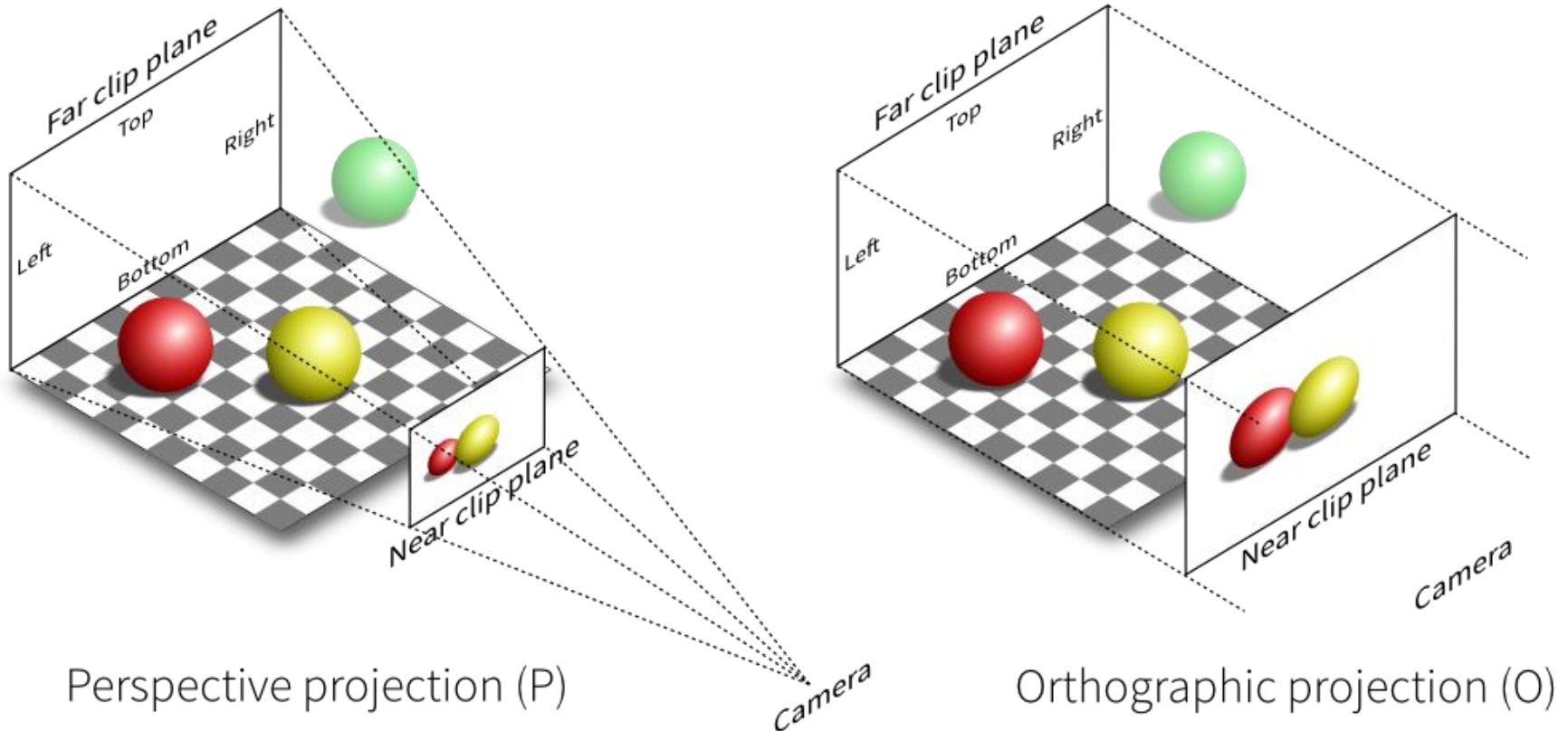


contents

- 2D->2D transformations
- 3D->3D transformations
- 3D->2D transformations (3D projections)
 - Perspective projection
 - **Orthographic projection**

Orthographic projection

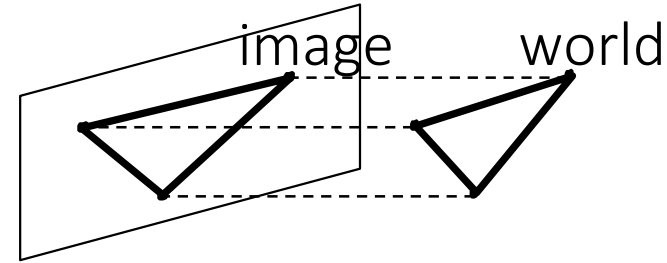
- A different kind of camera model that can be used is **orthographic projection** or **orthographic camera**.
- This kind of projection is invariant to the distance from the camera, and only depends on the object's size.



Orthographic projection

- Orthographic matrix

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \mapsto \begin{bmatrix} x \\ y \end{bmatrix}$$



- Weak perspective matrix (with scale coefficient)

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & z_0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z_0 \end{bmatrix} \mapsto \begin{bmatrix} x/z_0 \\ y/z_0 \end{bmatrix}$$

Orthographic projection

- When can we assume a weak perspective camera?

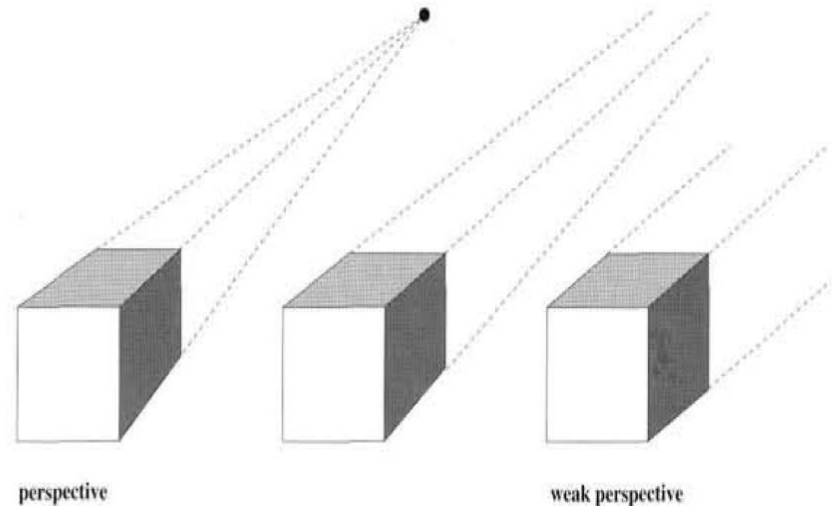
Orthographic projection

- When can we assume a weak perspective camera?
- When dealing with a plane parallel to image plane- z_0 is the distance to this plane.
- When far away objects- we can assume the average distance to the objects as z_0 .



Weak perspective camera

- One way to transform a regular perspective image to an orthographic view is simply taking the picture from a distance with zoom (large focal length).



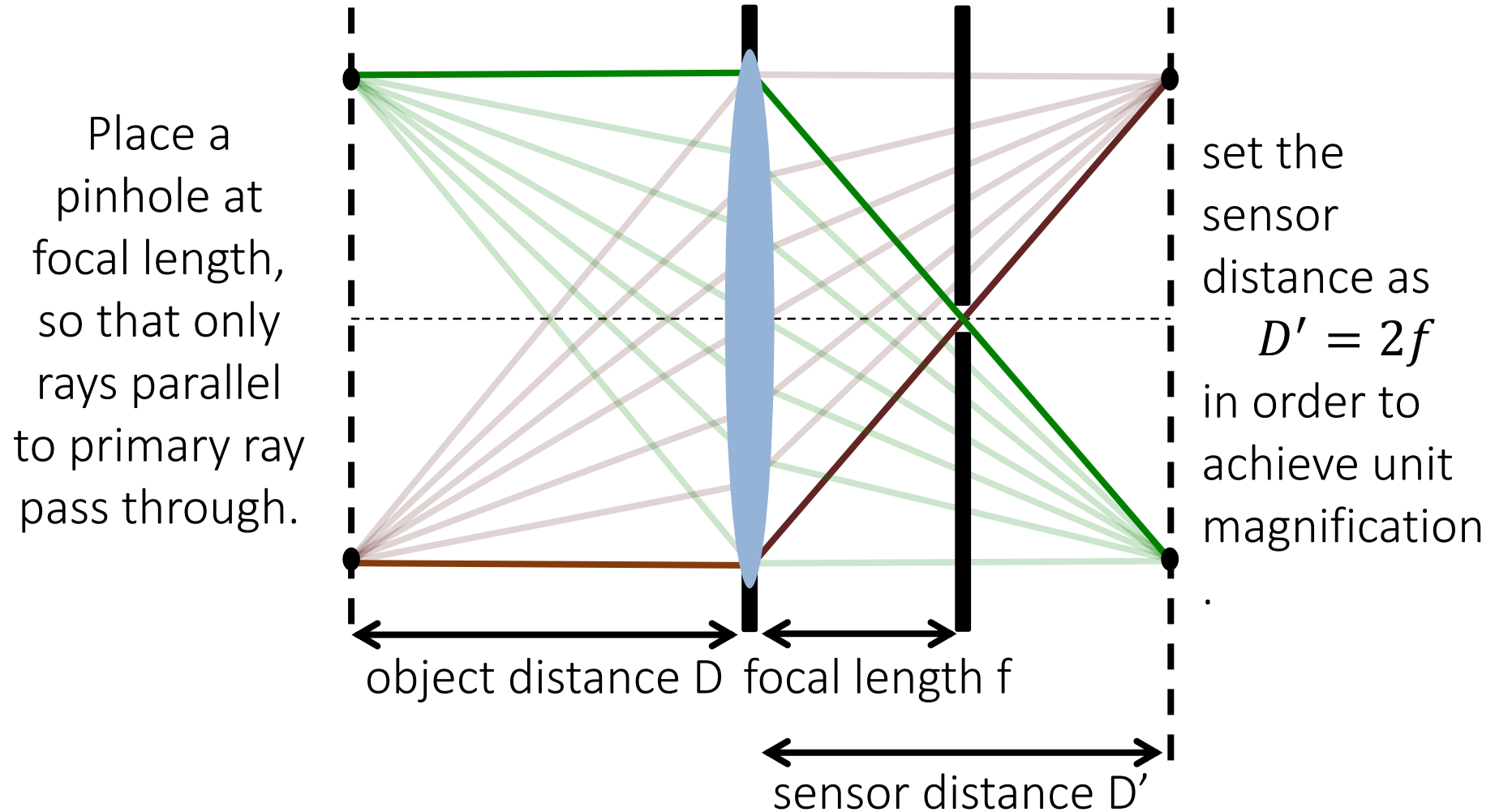
————— increasing focal length —————→

————— increasing distance from camera —————→



Weak perspective camera

- Real orthographic camera:



Weak perspective camera



perspective camera



weak perspective camera

Orthographic projection

- Why we want to assume a weak perspective camera?

Orthographic projection

- Why we want to assume a weak perspective camera?
- Easier to do a lot of image manipulation. For example: image stitching (no projective transformation, just affine), called panorama.

