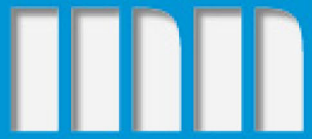


Graphical systems: introduction

Example process

Program

Monitor

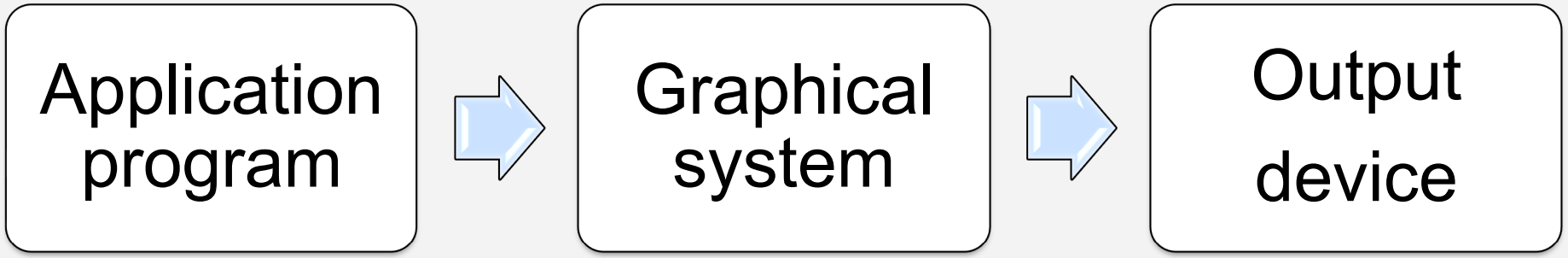
3D model, 2D shape, animation, CT scan….

Printer, projector, plotter, movie file, picture file, stereolitograph..

Platform

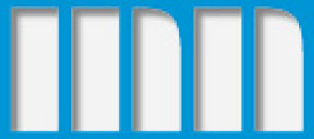
PC Win, PC Lin, Mac, SGI...
PS, XBOX, Wii, ...

CG reference model



- Inside the boxes – standards
- Between the boxes – standard interfaces
- Separate modeling and rendering
- Separate device-dependent and device-independent parts

Reference model – detailed



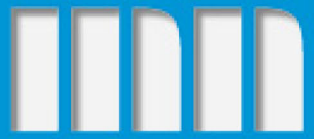
Application program

- Graphical data
  - Models, textures, description, mapping…
- Animation
  - Scripted, procedural (physics), interactive
- Application logic

Data sources

- Modeling, capturing, simulation…

Reference model – detailed



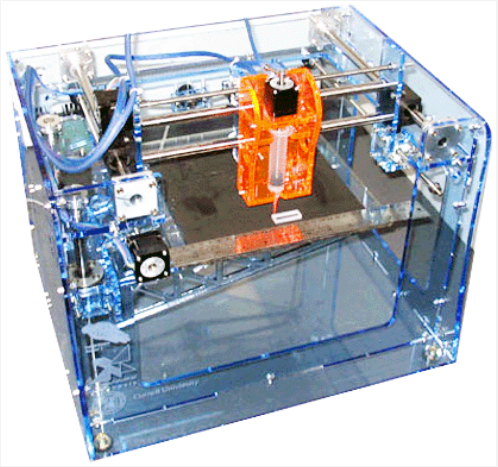
Graphical system

- Data processing (input, conversion)
- Transformations
- Projection
- Clipping, visibility, lighting
- Rasterization

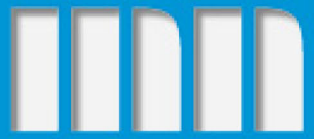
Reference model – detailed

Output device

- Device driver
- Physical device
- Output format

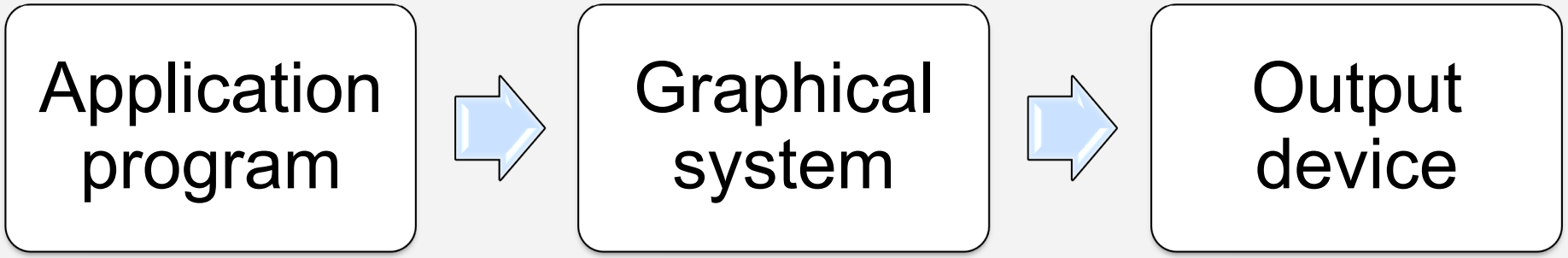


Advantages of CGRM



- Device-independent application development
- Application-independent device development
- Standard interface GS ↔ device
  - Hardware acceleration, optimization
- Standard interface APP ↔ GS
  - Rapid development, transferrable code

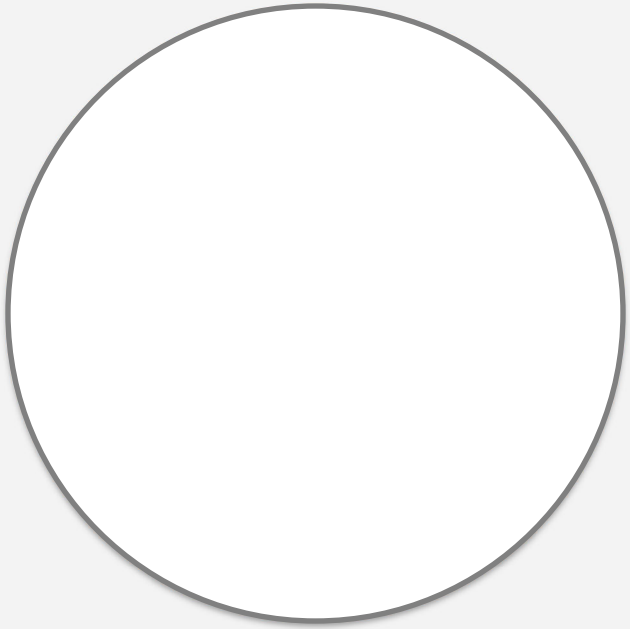
CG reference model



Digital imagery fundamentals

Visual stimuli

- Geometry
- Color
- Motion

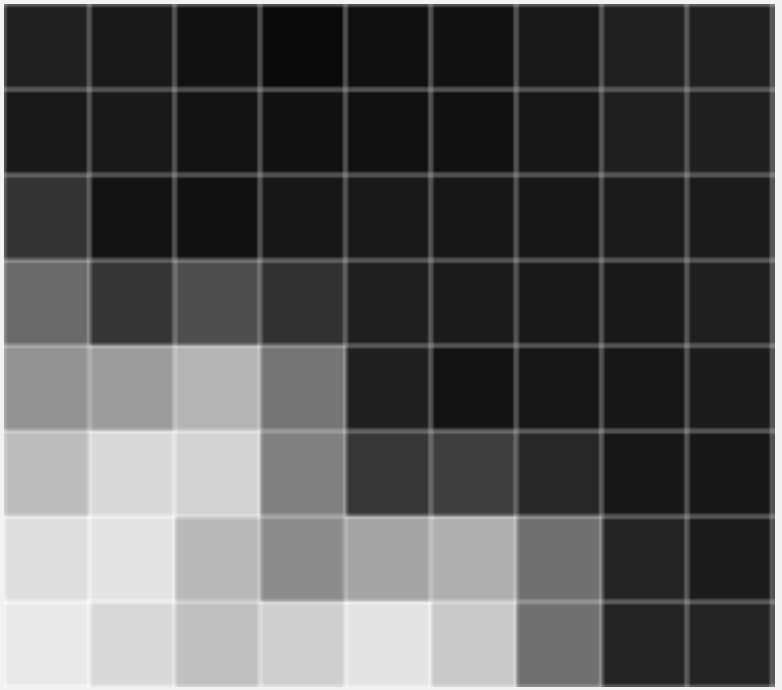


Geometry

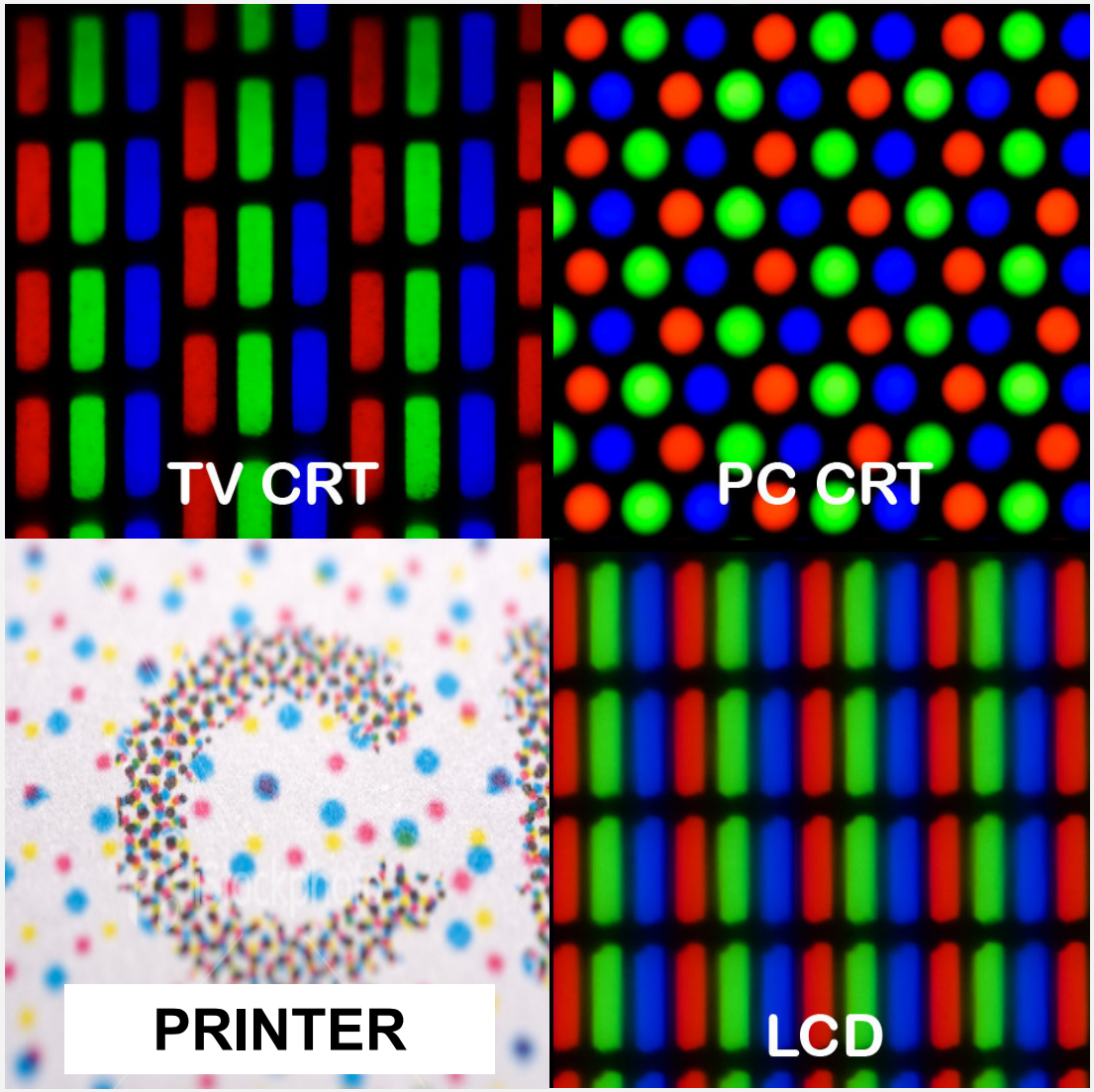
- Continuous (analog) v. Discrete (digital)

Discrete representation

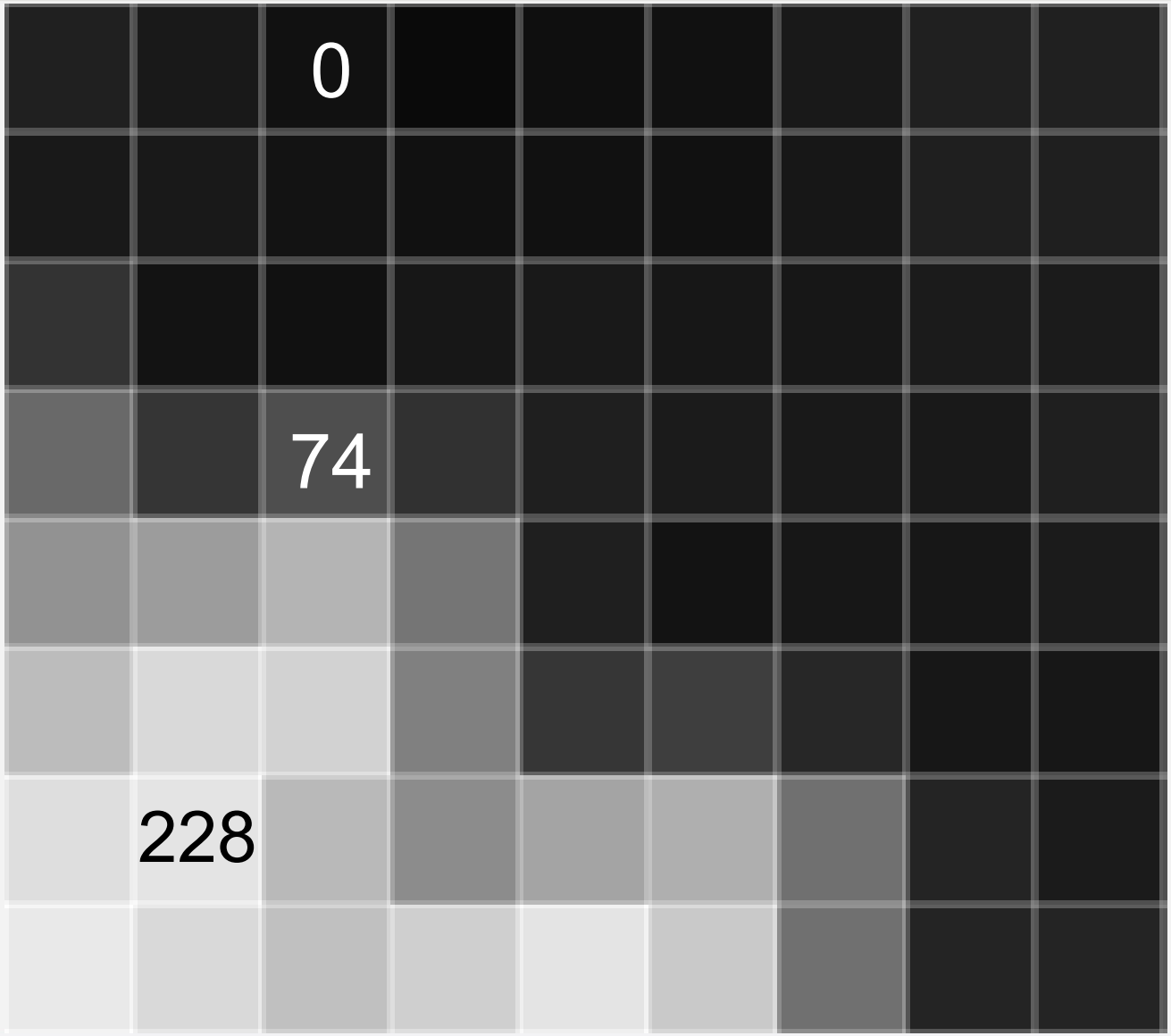
- Pixel = picture element
- Image resolution = digital size : physical size
  - DPI, PPI (dots per inch, points per inch)
  - 72 – 130 dpi (monitors)
  - 150 – 600 dpi (print)
  - 600 – 1200 dpi (scanners)



Devices close-up



Pixel values

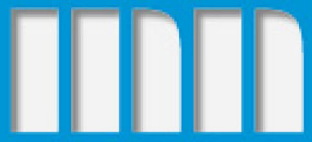


Color

- Color (h 4bit

Lenna Sjööblom, miss November 1972

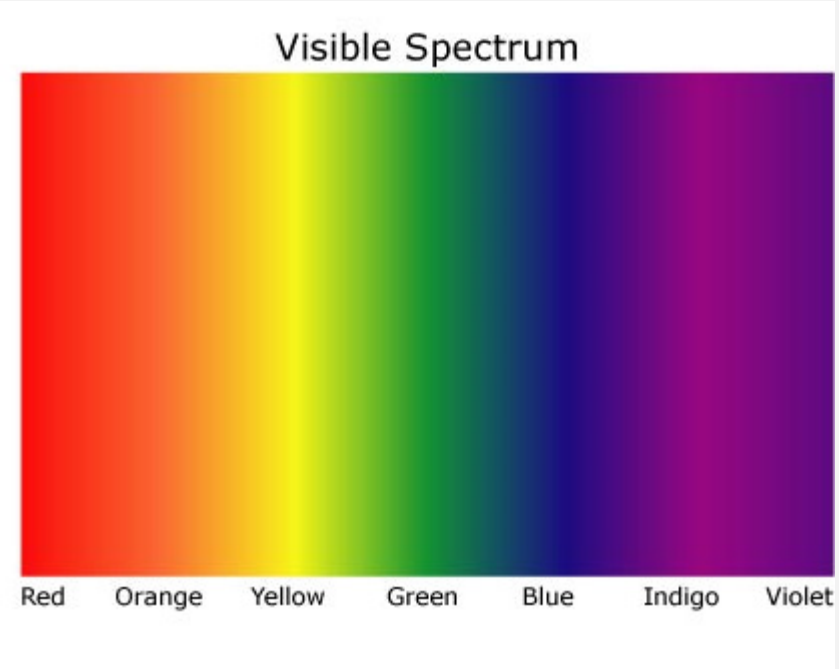
Digital color representation

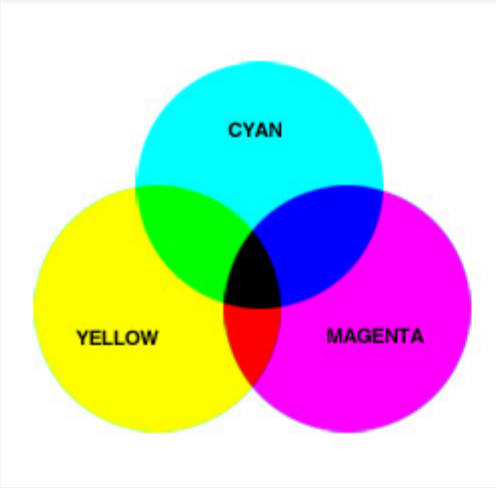


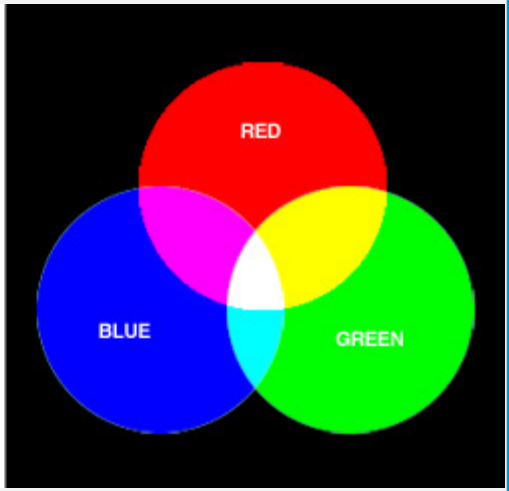
- R-G-B
  - e.g. palette mode (remember GIFs ?)
  - e.g. 24 bit colors, each pixel = 8 x 8 x 8 bits =
    = 0..255 red, 0..255 green, 0..255 blue
- C-M-Y-K
- Other color models: HSV, YUV

Color

- Continuous (analog) v. Discrete (digital)

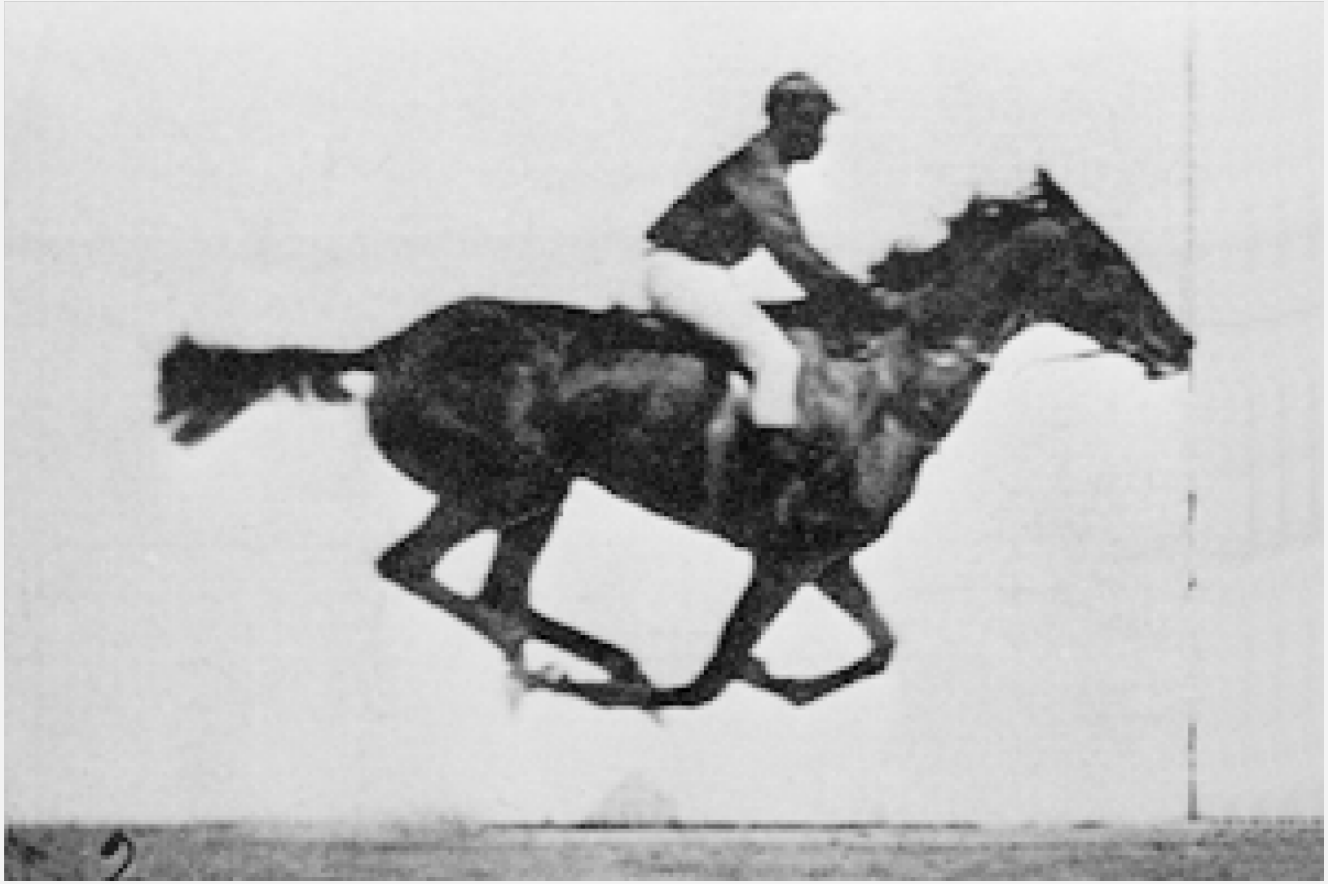






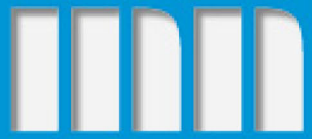
Motion

- Continuous (analog) v. Discrete (digital)



Eadweard Muybridge – The Horse in Motion (1878)

Limits in numbers (examples)

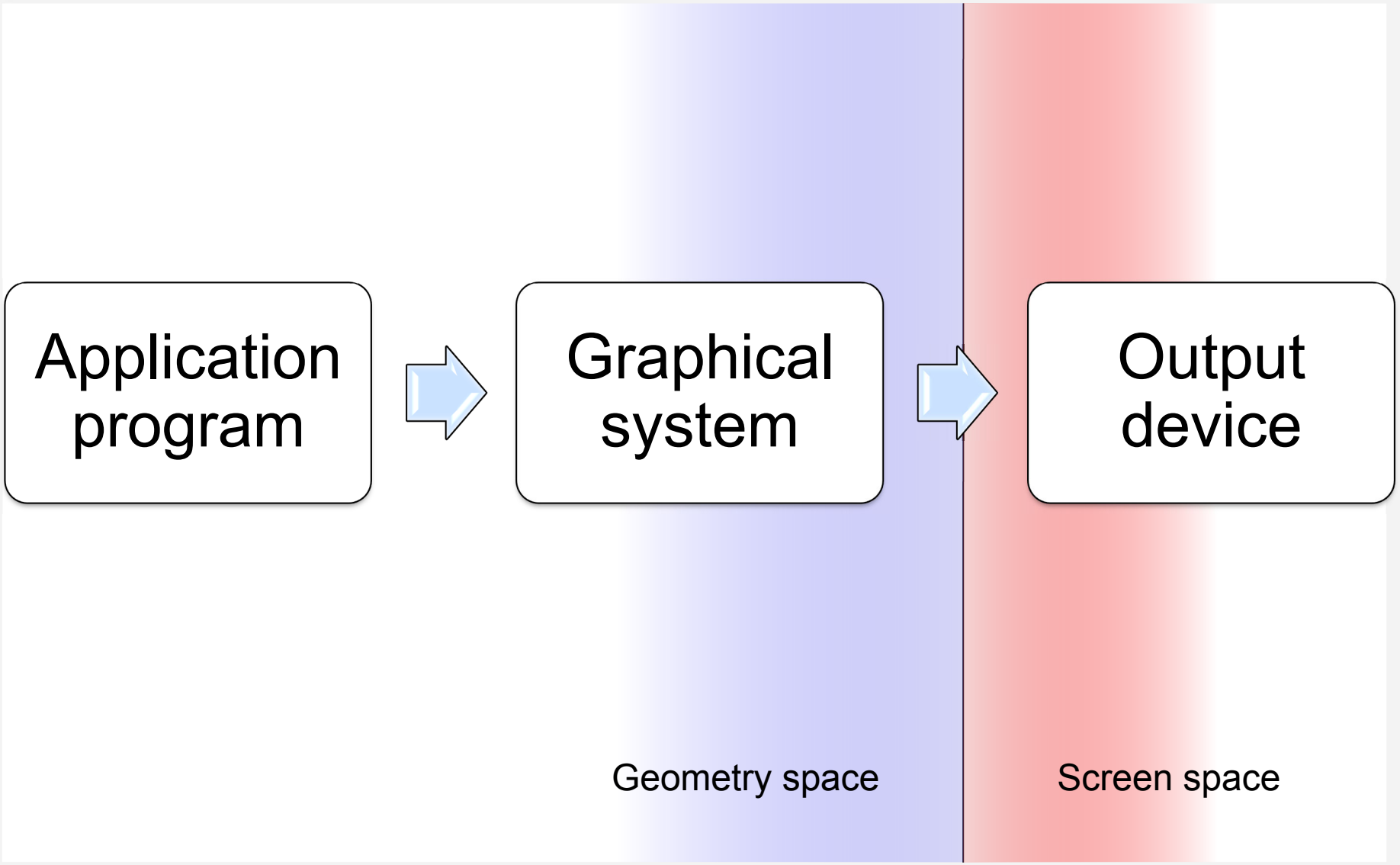


- Image size
  - 128x176 (mobiles) – 1920x1080 fullHD
  - 1600x1200 (2mpix), 2480x3508 (A4@300dpi)
- Color depth
  - 1 bit (black/white), 8bit (256 colors)
    16bit (65536), 24bit (16.7 milion)
- Framerate, refresh rate
  - 15fps, 24fps, 30fps
  - 50hz, 60hz, 100hz, 120hz

Limits in numbers (examples)

- 29 595 009 024 000 possible pictures
  - 1680 × 1050 × 24bit
  - 37 years of movies @ 25 fps
- 640 × 480 × 256 (grayscale) = 78 643 200
  - 874 hours ~ 550 movies
- flickr.com = ~ 4 000 000 000 photos
  - 5 years of movies

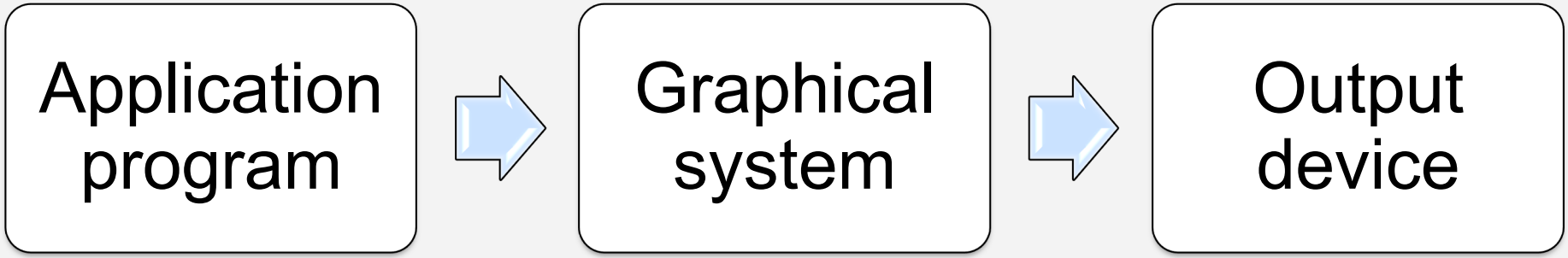
CG reference model



Graphical information and rendering

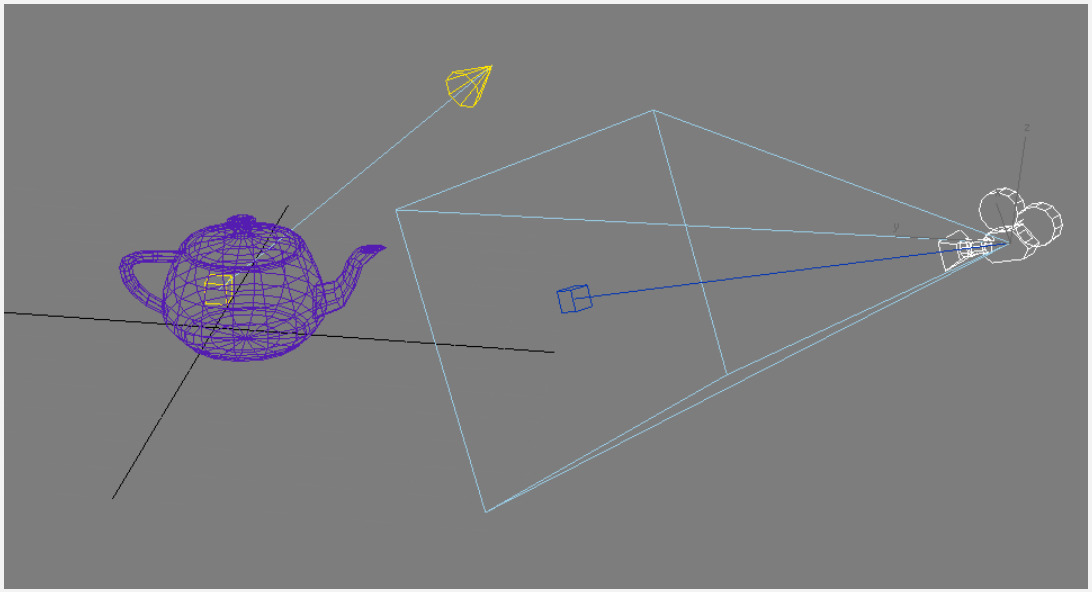
Our focus

- 3D objects in geometry space
  - some concepts explained in 2D, then extended
- Object representation (inside APP, GS)
- Object rendering (GS → Output device)



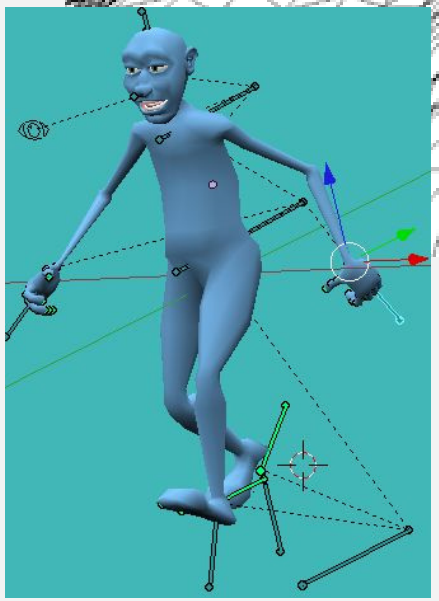
Geometry space

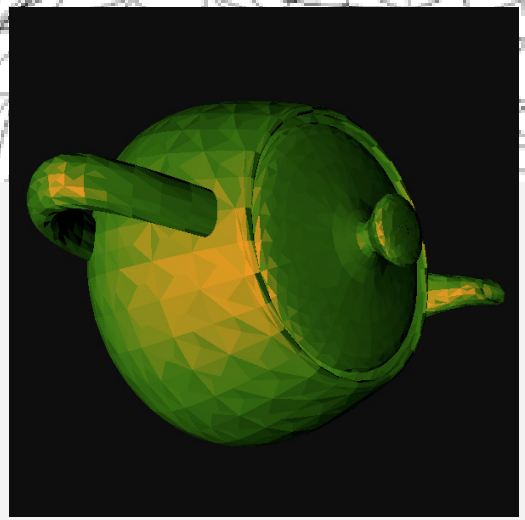
- Scene
  - Virtual representation of world
- Objects
  - Visible objects (“real world”)
  - Invisible objects (e.g. lights, cameras, etc.)



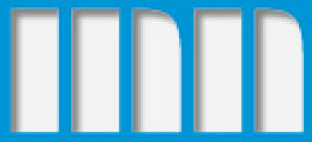
Dimensionality

- 2D
  - Shapes, images
- 2.5D
  - Surfaces, terrains
- **3D**
  - **Objects, scenes**
- 4D
  - Animation





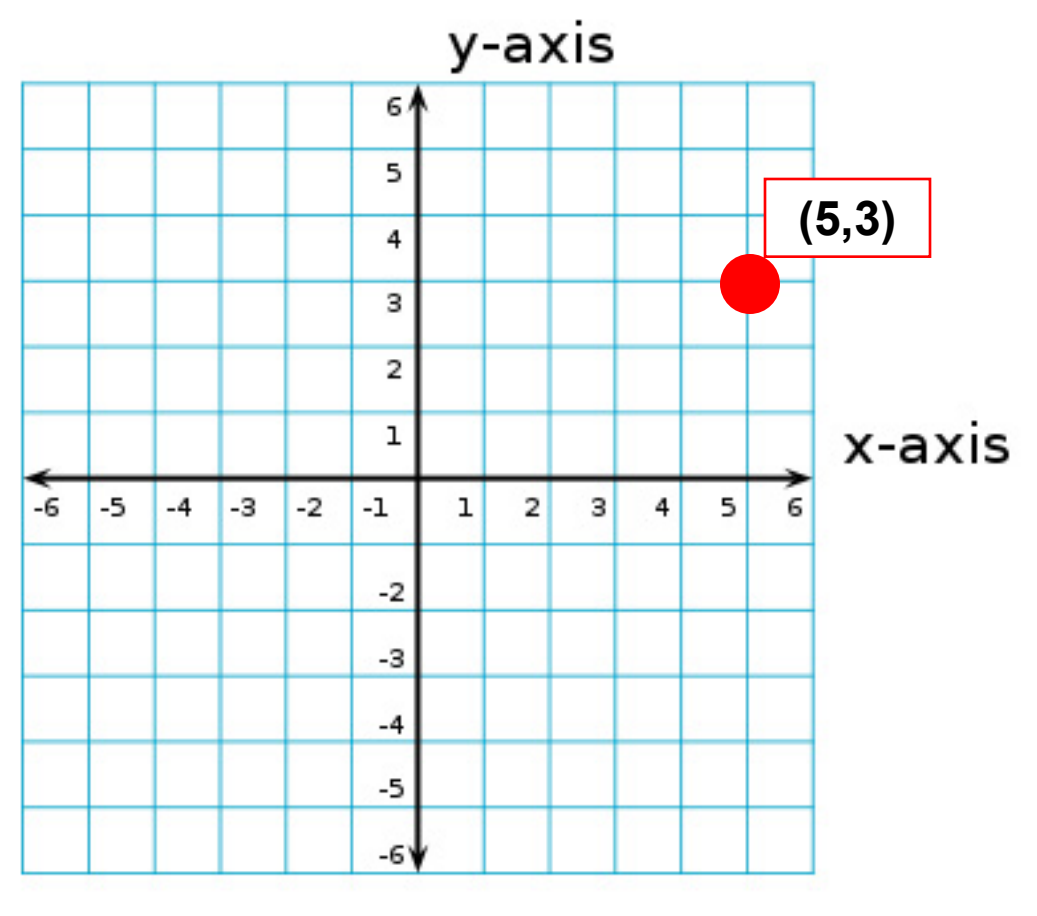
Full scene definition



- Objects
  - What objects, where, how transformed
    - To be discussed early during course
  - How they look – color, material, texture...
    - To be discussed later during course
- Camera
  - Position, target, camera parameters

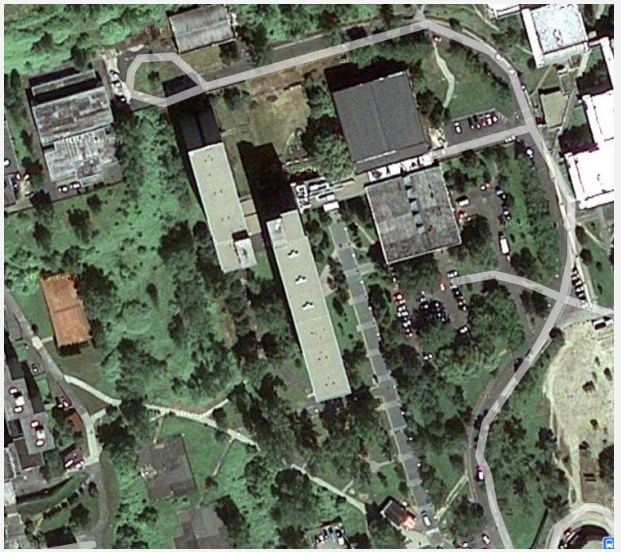
Coordinate system

- Cartesian coordinates in 2D
  - Origin
  - x axis
  - y axis

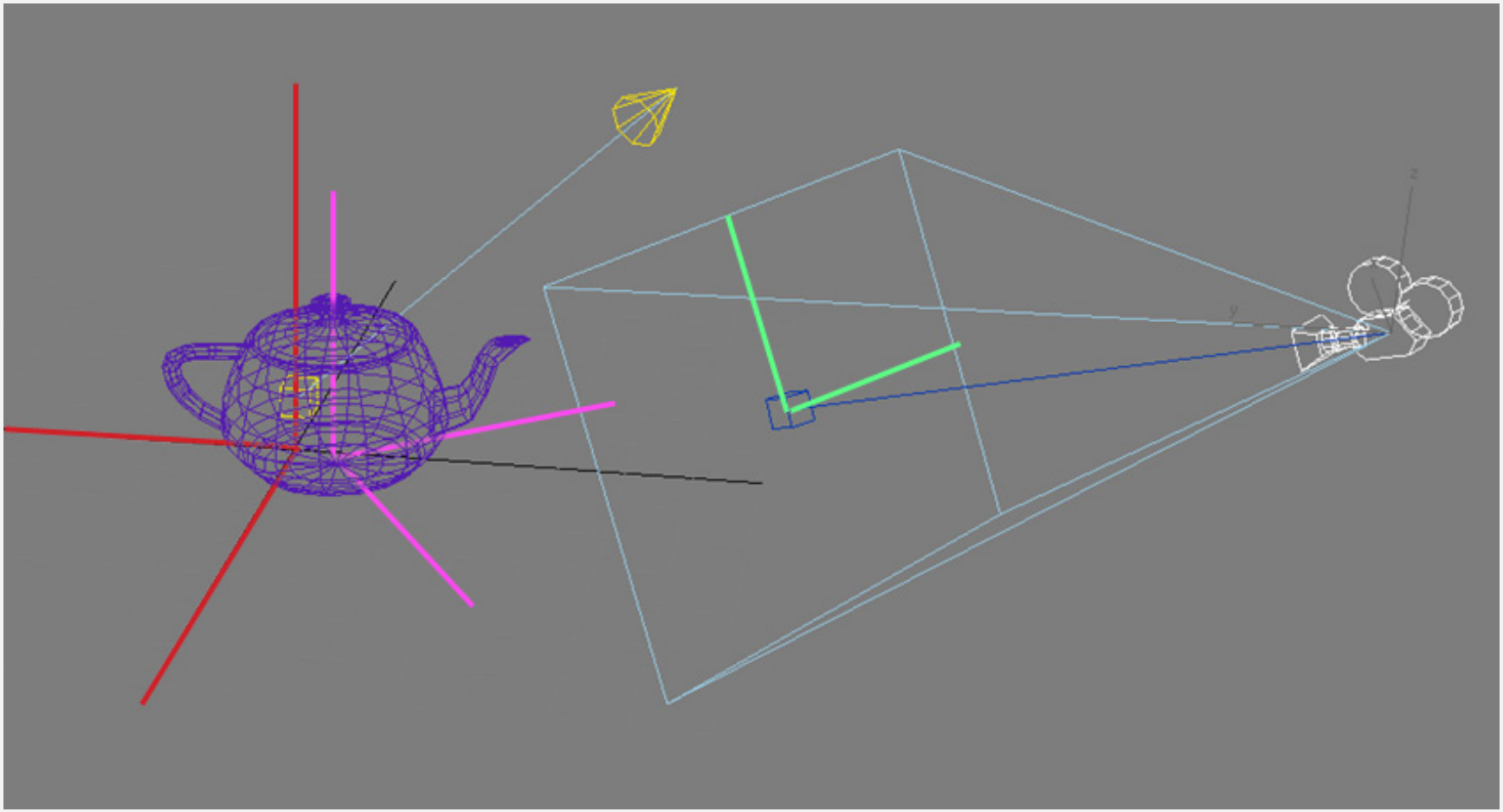


Coordinate systems

- Global
  - e.g. 48.160038,17.065397
- Local
  - e.g. 9th floor, room #7
- Camera
- (Window)

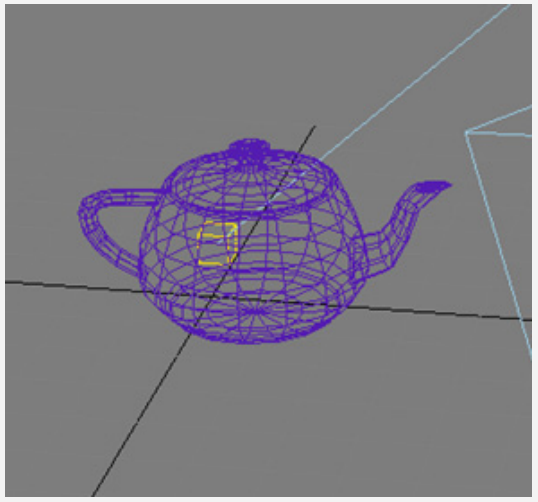


Global/local/camera coords.



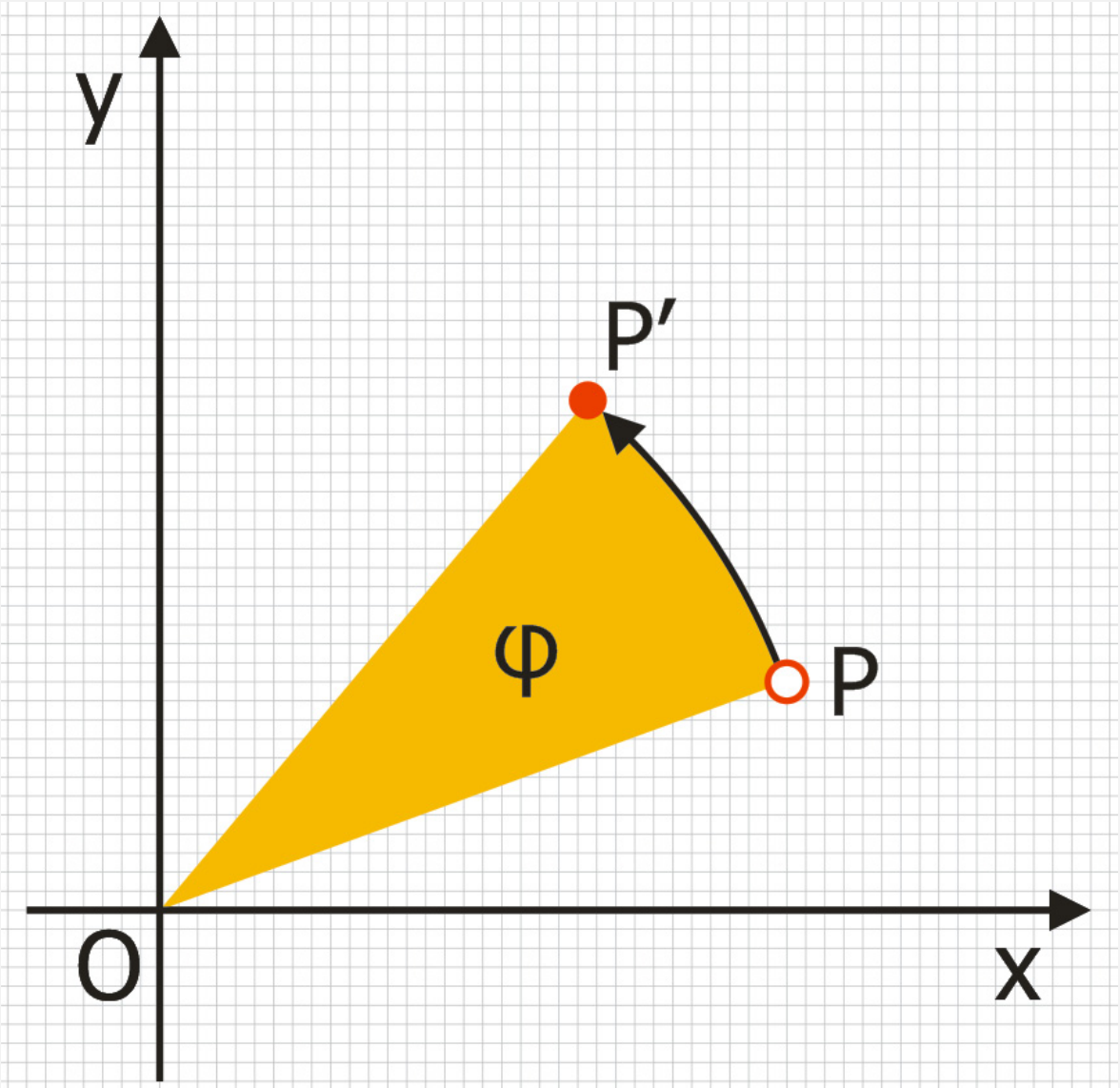
Rendering pipeline

- Model transformation
  - local → global coordinates
- Viewport transformation
  - global → camera
- Clipping
- Rasterization
- Texturing



Transformations

Transformations – rotate



angle φ
‹0..360°›
‹0..2π›

Transformations – rotate

P(x,y) → P′(x′,y′)

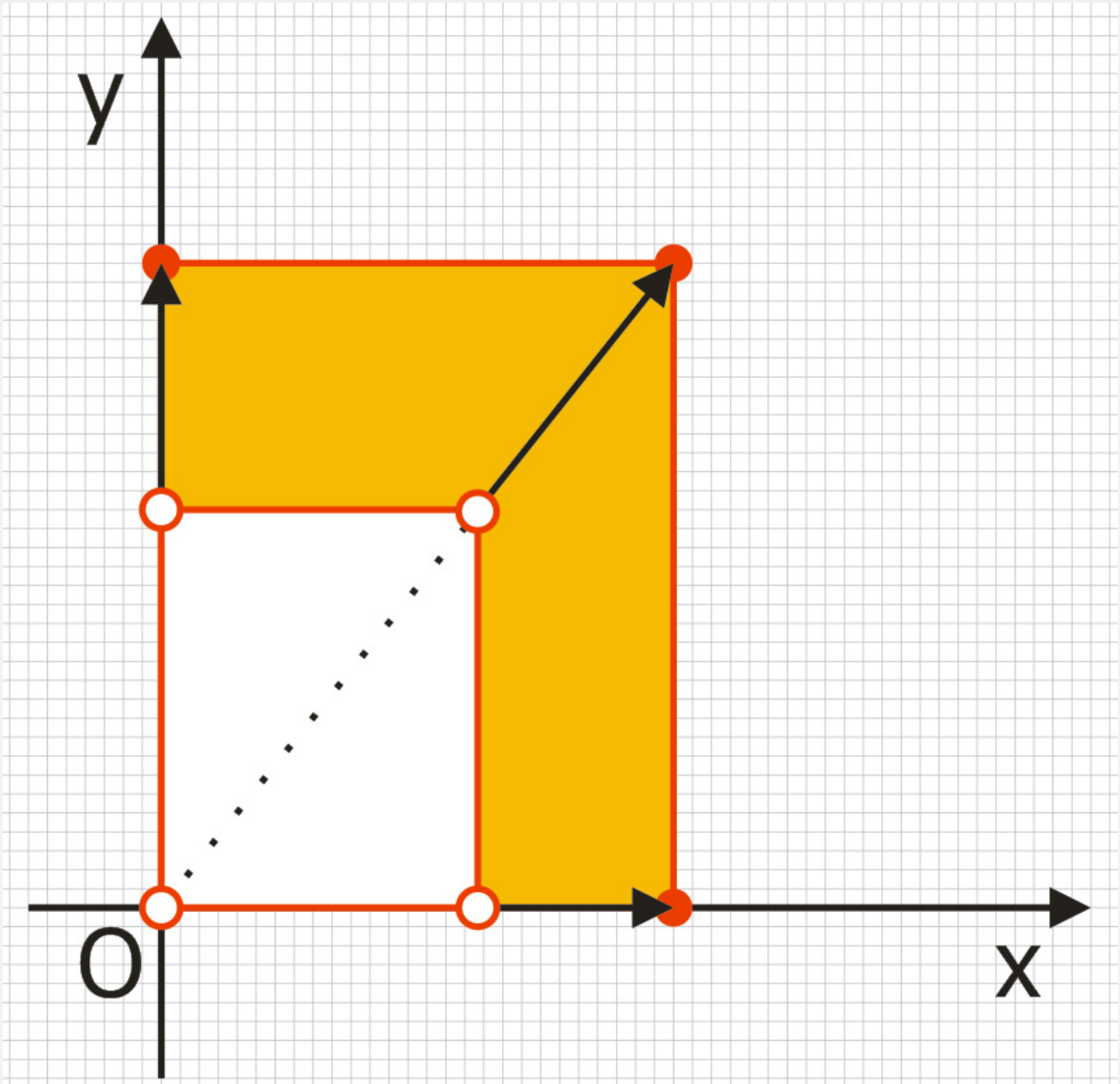
x′ = x.cos φ − y.sin φ

y′ = y.cos φ + x.sin φ

Matrix notation:

$$(x', y', 1) = (x, y, 1)\begin{pmatrix} \cos\varphi & \sin\varphi & 0 \\ -\sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Transformations – scale



factor s

Transformations – scale

$\mathbf{P(x,y) \rightarrow P'(x',y')}$

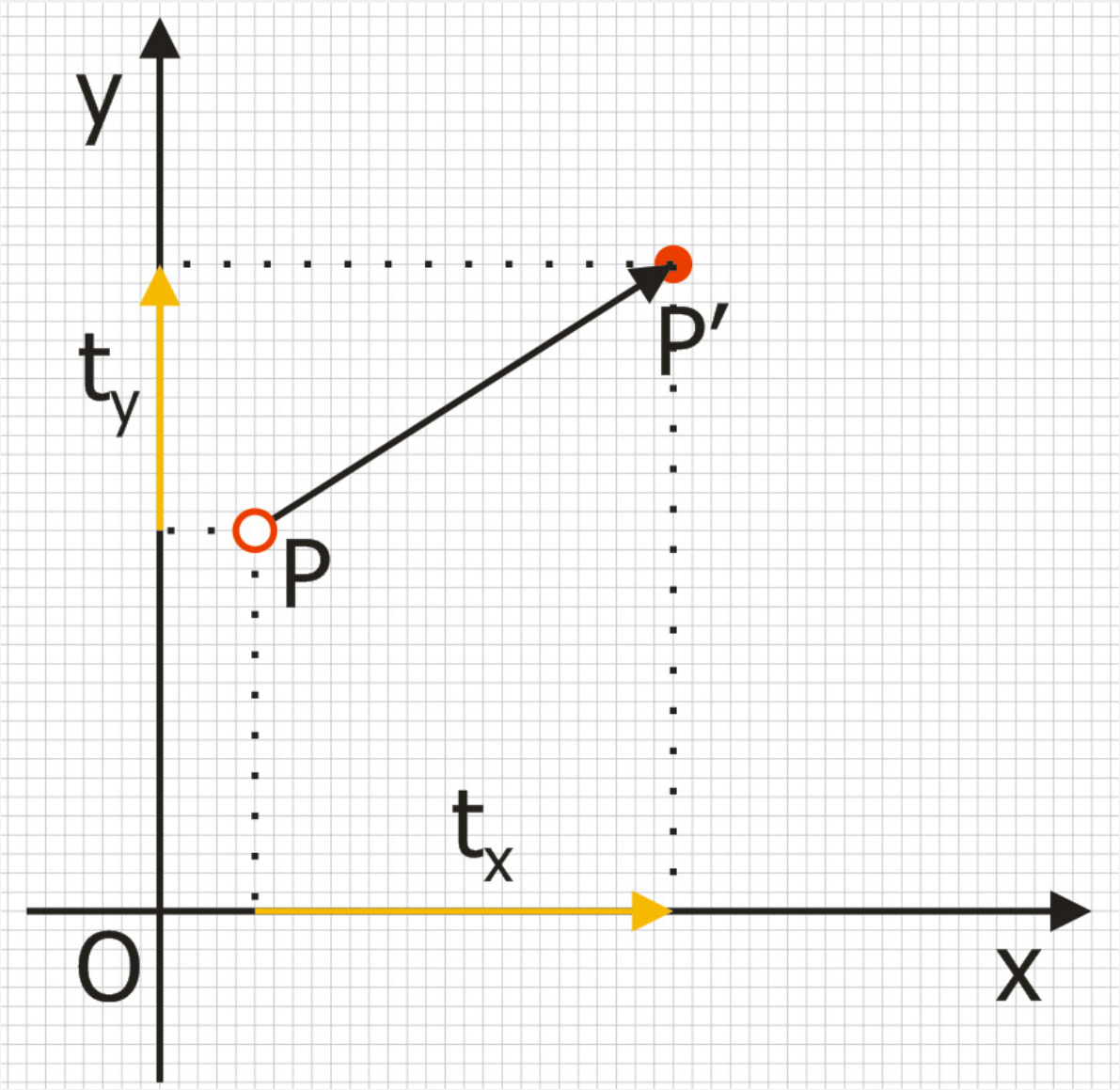
$\mathbf{x' = x.s_x}$

$\mathbf{y' = y.s_y}$

Matrix notation:

$$(x', y', 1) = (x, y, 1)\begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Transformations – translate



vector
$t(t_x,t_y)$

Transformations – translate

$P(x,y) \rightarrow P'(x',y')$

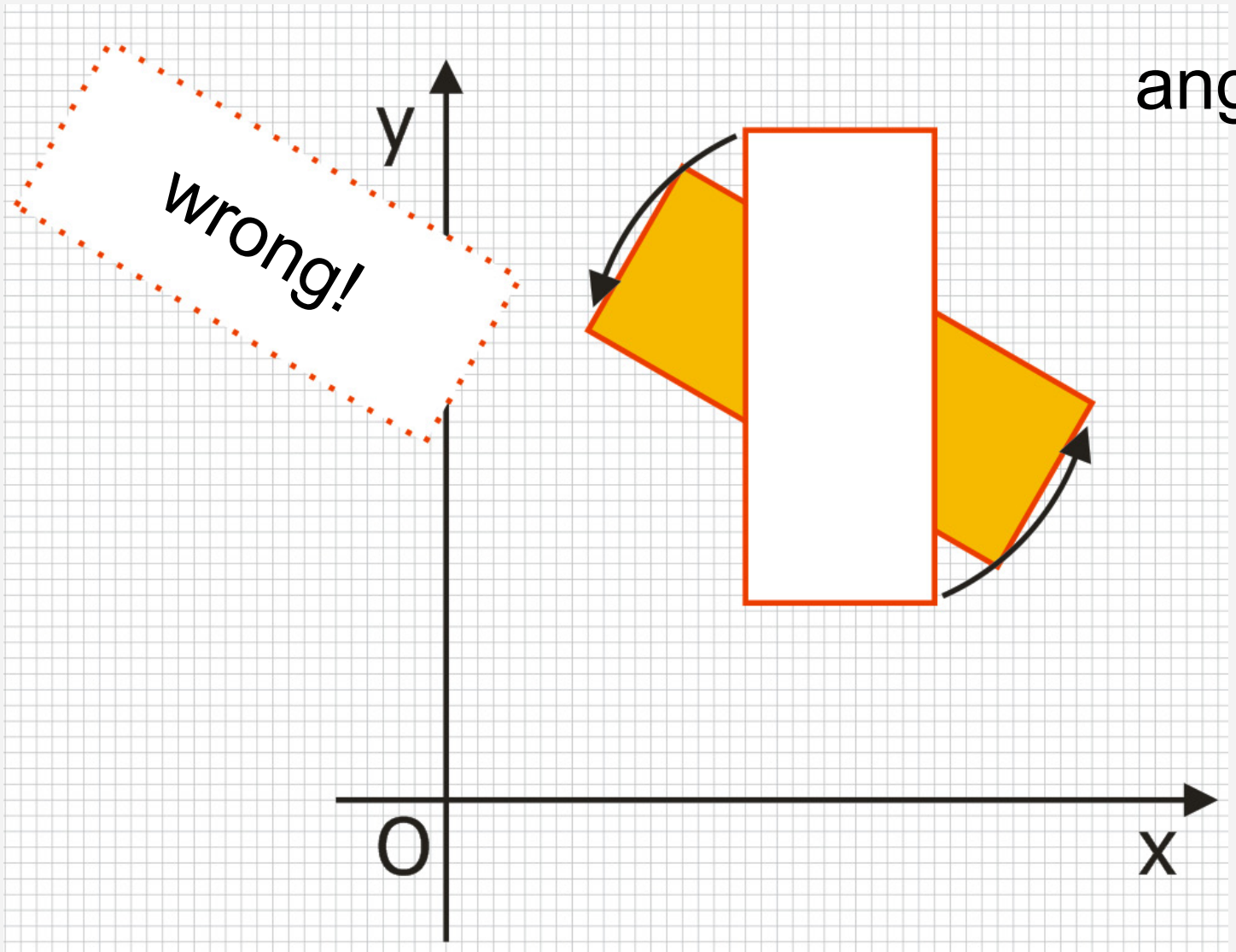
$x' = x + t_x$

$y' = y + t_y$

Matrix notation:

$$(x', y', 1) = (x, y, 1)\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{pmatrix}$$

Problem: local rotation



Transformation composition

1. translate rotation center to origin: $t(t_x,t_y)$
2. rotate by φ
3. inverse translate by $t'(-t_x,-t_y)$

Matrix notation:

$$(x',y',1) = (x,y,1)\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{pmatrix}\begin{pmatrix} \cos\varphi & \sin\varphi & 0 \\ -\sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -t_x & -t_y & 1 \end{pmatrix}$$

3D transformations

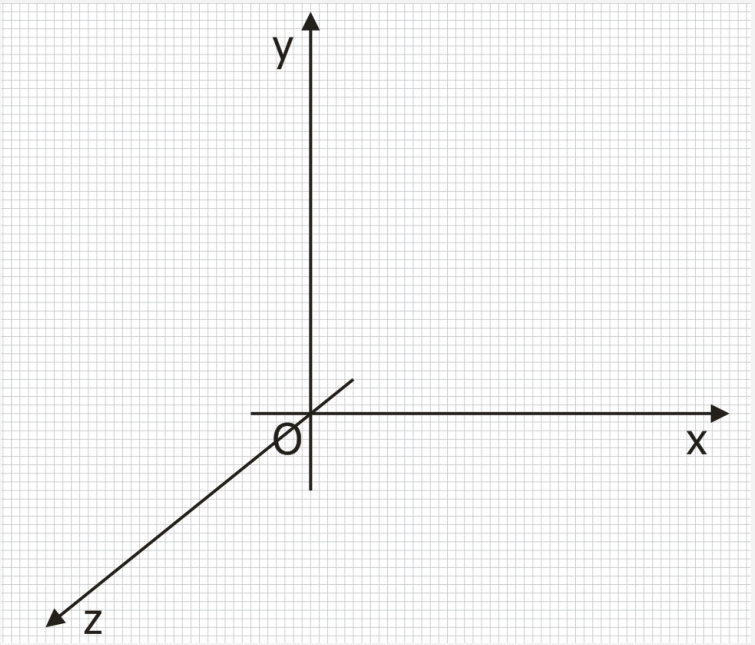
- scale $\begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$ translate $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ t_x & t_y & t_z & 1 \end{pmatrix}$

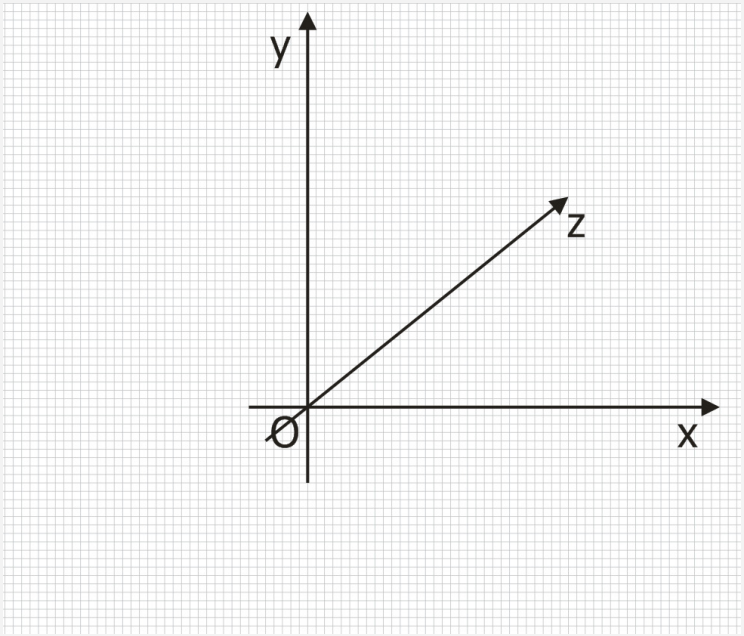
- rotate

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\varphi_x & -\sin\varphi_x & 0 \\ 0 & \sin\varphi_x & \cos\varphi_x & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\varphi_y & 0 & \sin\varphi_y & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\varphi_y & 0 & \cos\varphi_y & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\varphi_z & \sin\varphi_z & 0 & 0 \\ -\sin\varphi_z & \cos\varphi_z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

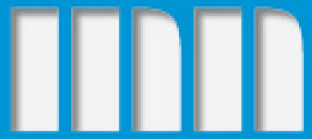
3D coordinate systems

- Right-handed coordinate system
- Left-handed coordinate system





- rotation direction

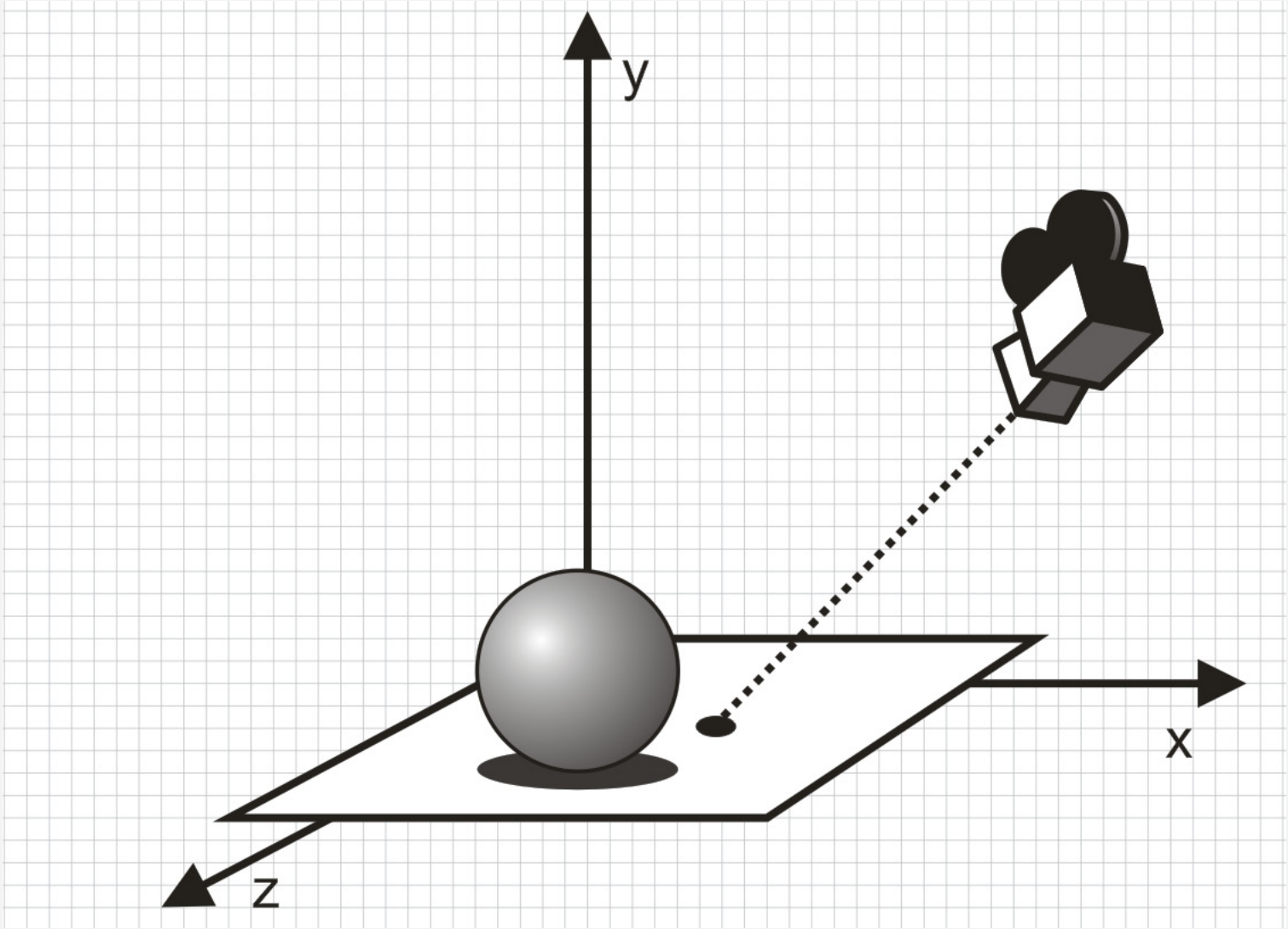


Projection

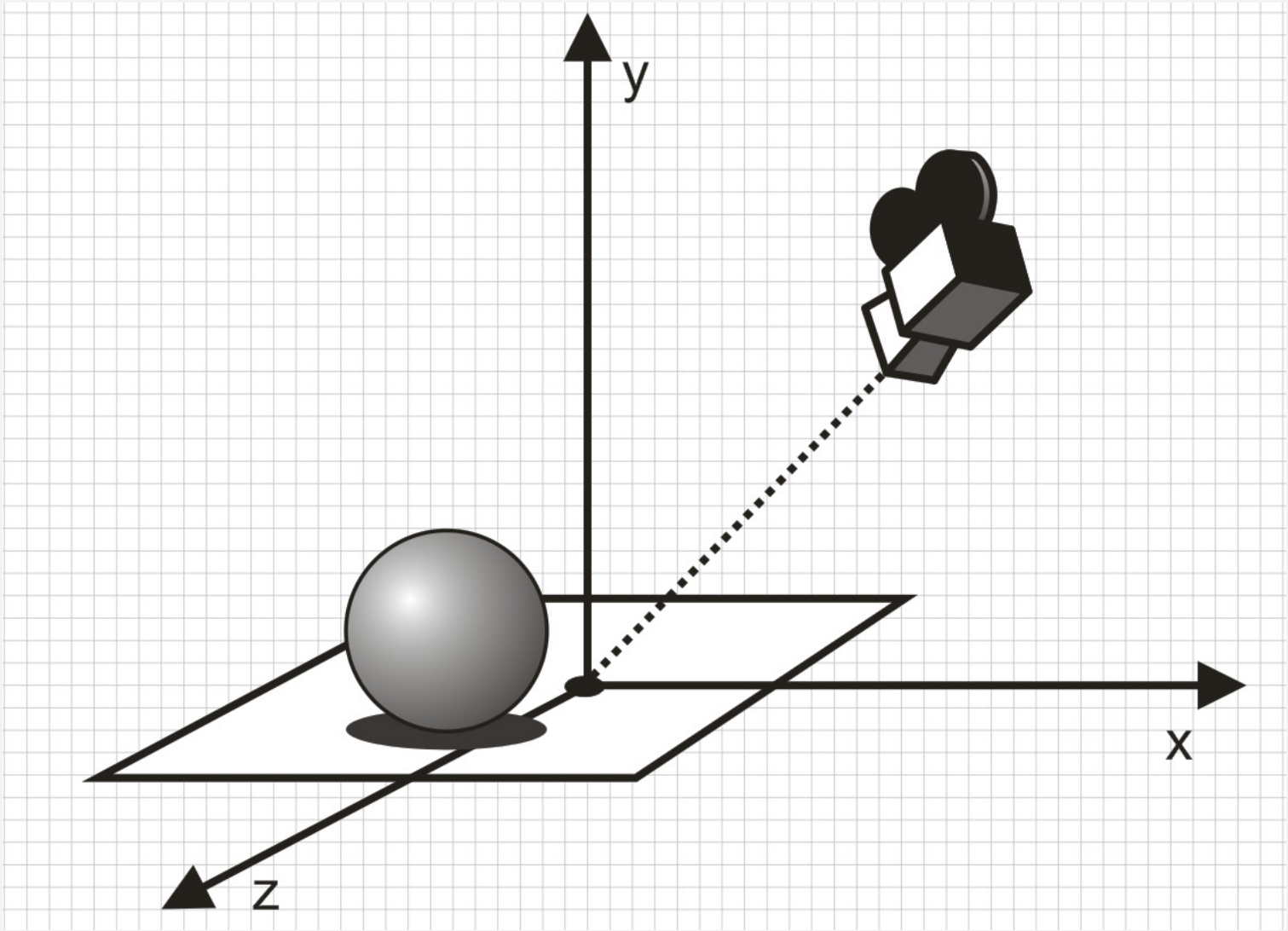
Viewing transformation

1. rotate scene so that camera lies in z-axis
2. projection transformation
3. viewport transformation

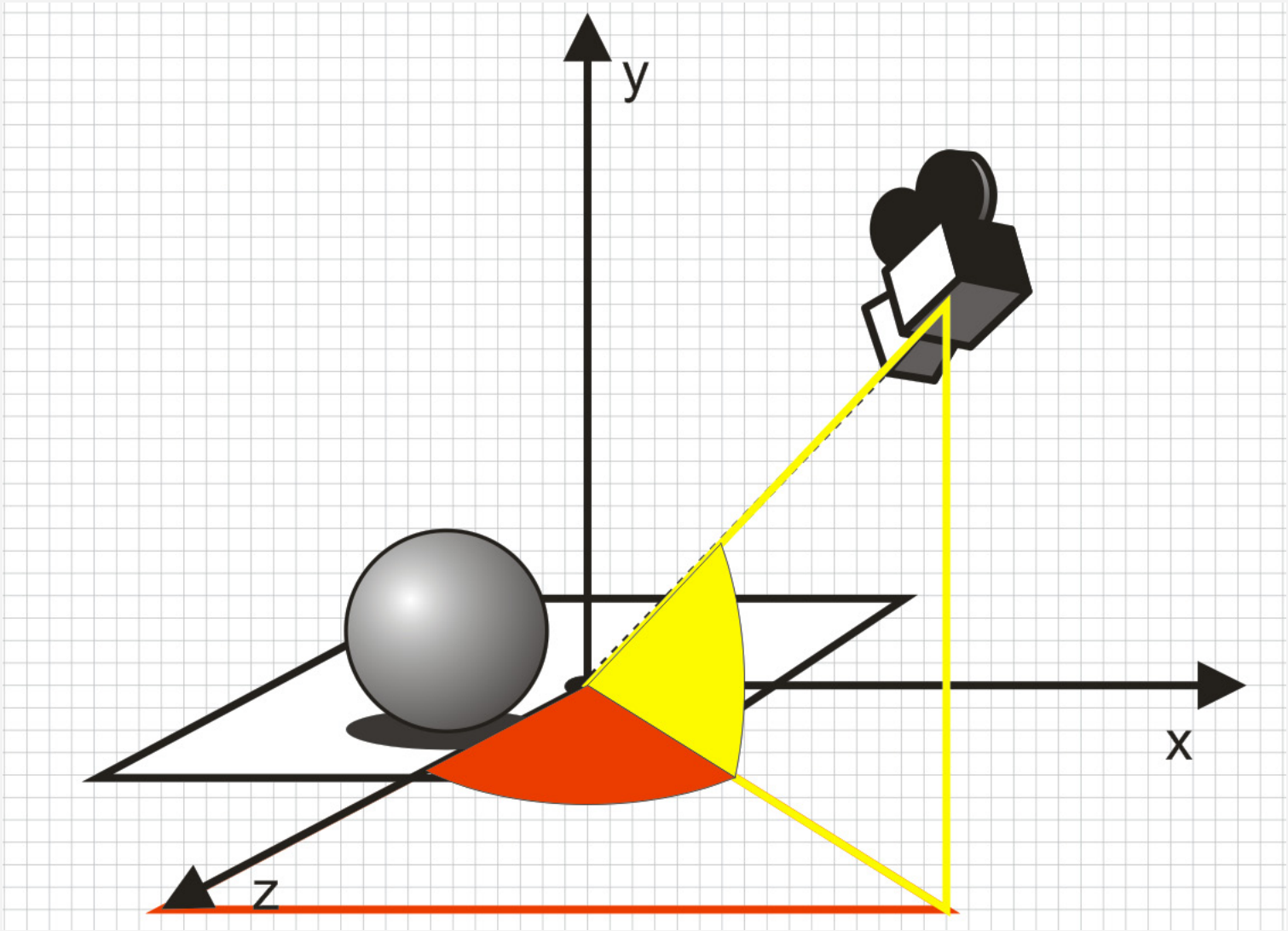
Stage 0



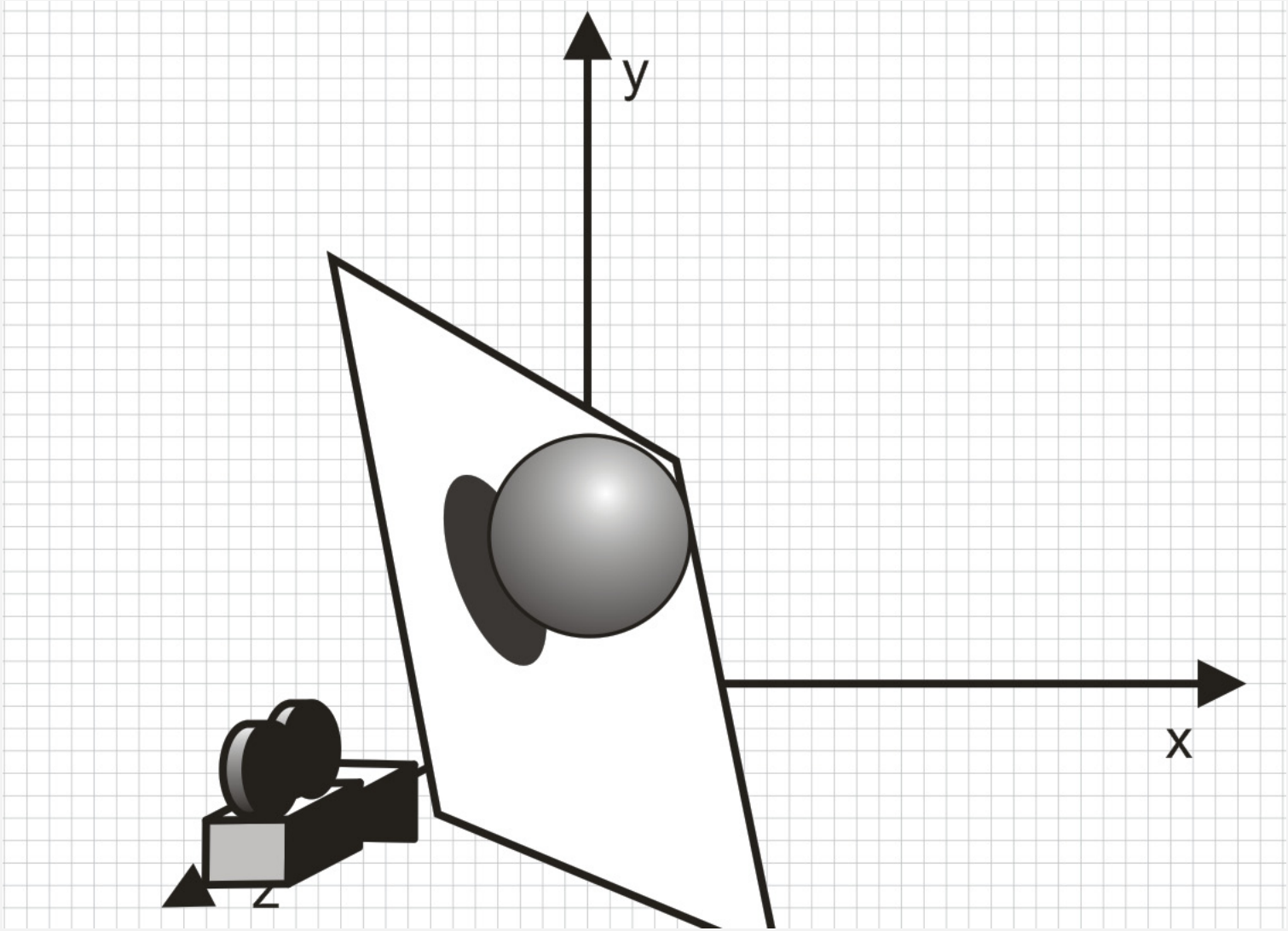
Stage 1 – translate P→P’



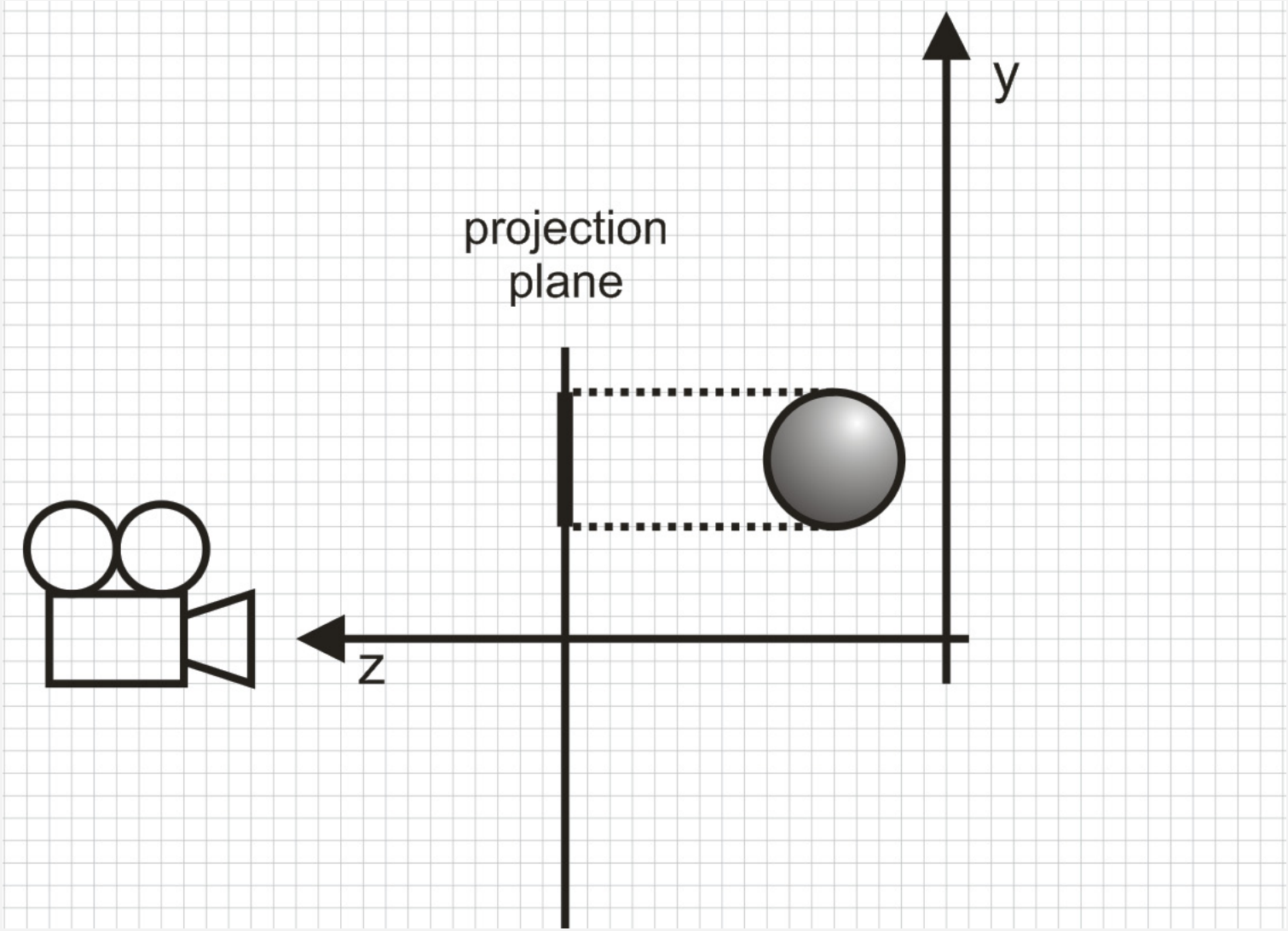
Stage 2 – rotate P’→P’’→P’’’



Rotated scene



Orthogonal projection



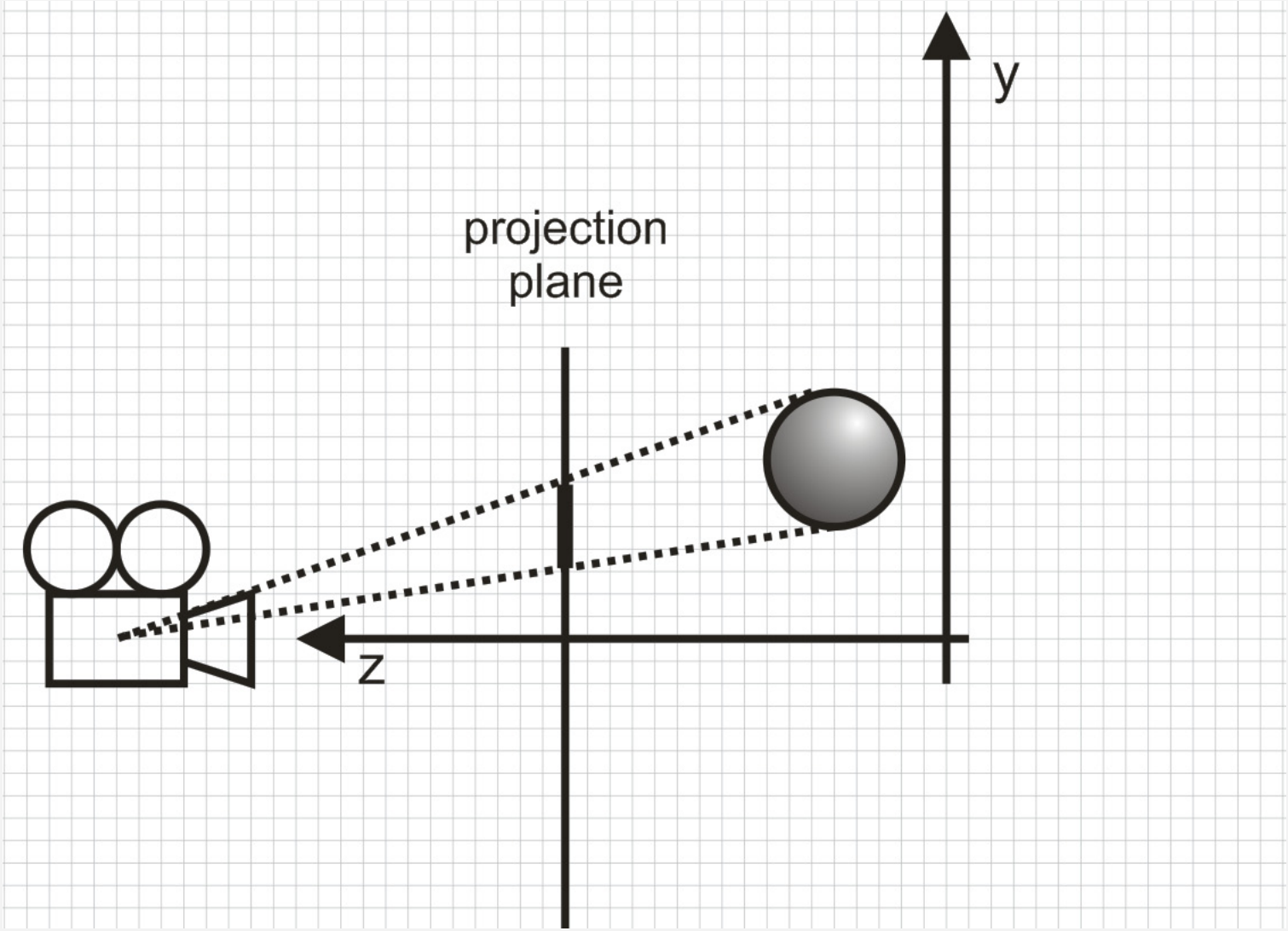
Orthogonal projection

- $x_p = x'''$
- $y_p = y'''$

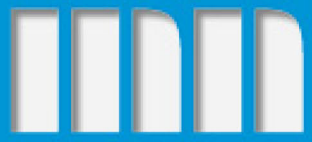
- Matrix notation

$$(x_P, y_P, z_p, 1) = (x''', y''', z''', 1)\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Perspective projection



Perspective projection

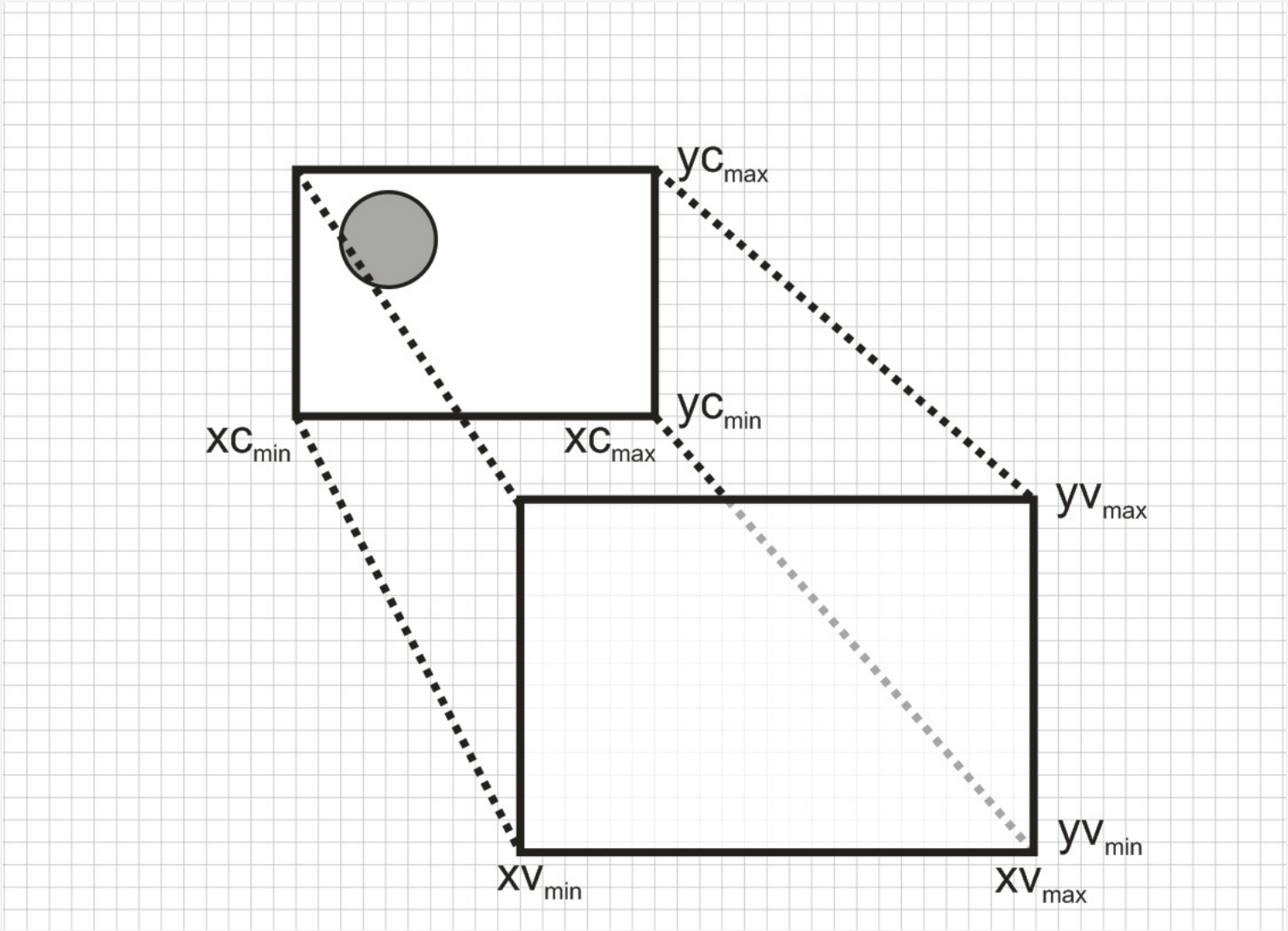


- x_p = ?
- y_p = ?

- Matrix notation

$$(x_P, y_P, z_p, 1) = (x''', y''', z''', 1) \begin{pmatrix} ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{pmatrix}$$

Viewport transformation



Viewport transformation

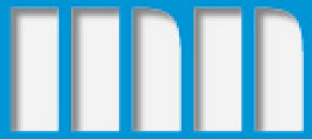
- s_x, s_y – scale factors

$$s_x = \frac{xv_{\max} - xv_{\min}}{xc_{\max} - xc_{\min}} \qquad s_y = \frac{yv_{\max} - yv_{\min}}{yc_{\max} - yc_{\min}}$$

- Matrix notation

$$(x_v, y_v, 1) = (x_p, y_p, 1) \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ -s_x xc_{\min} + xv_{\min} & -s_y yc_{\min} + yv_{\min} & 1 \end{pmatrix}$$

Welcome to the matrix!



1. local → global coordinates
   - translate, rotate, scale, translate
2. global → camera
   - translate, rotate, rotate, project
3. camera → viewport
   - translate, scale, translate

- Transformation combine = matrix multiply

Readings

- Ružický, Ferko – Počítačová grafika a spracovanie obrazu
- Žára a kol. – Moderní počítačová grafika
- http://pg.netgraphics.sk/

Next week:
Rasterization, culling, clipping