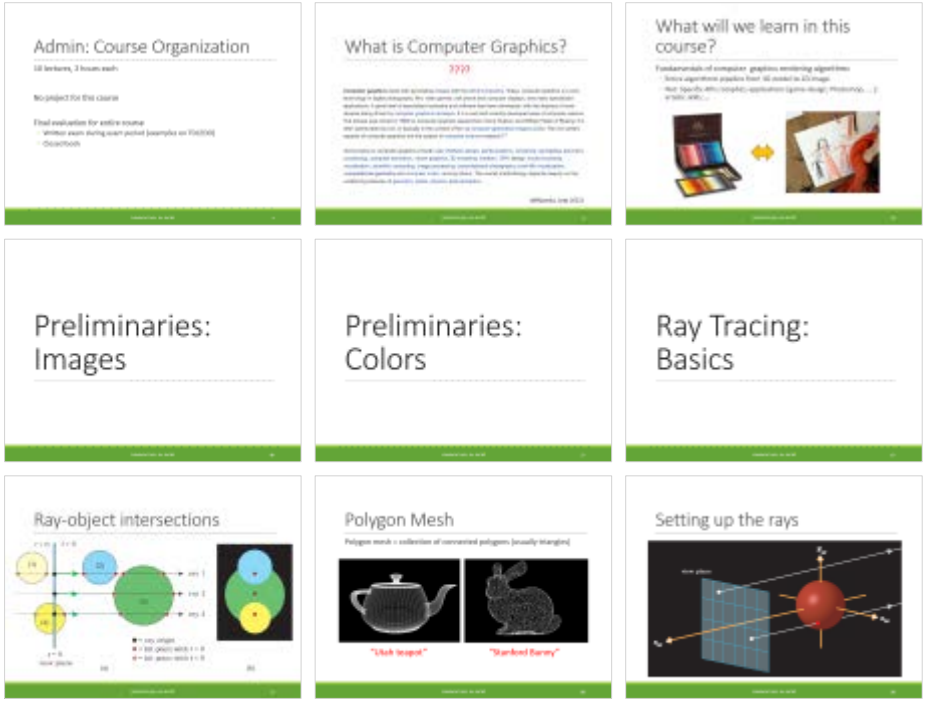
Fundamentals of Computer Graphics

PHILIP DUTRÉ
DEPARTMENT OF COMPUTER SCIENCE

Lecture Overview



Relevant sections in book: Chapters 3, 19

(Illustrations from Ray Tracing From The Ground Up, Physically-Based Rendering, Fundamentals of Computer Graphics)

(page numbering might skip some slides due to 'hidden' slides in my presentation.

Admin: Course Organization

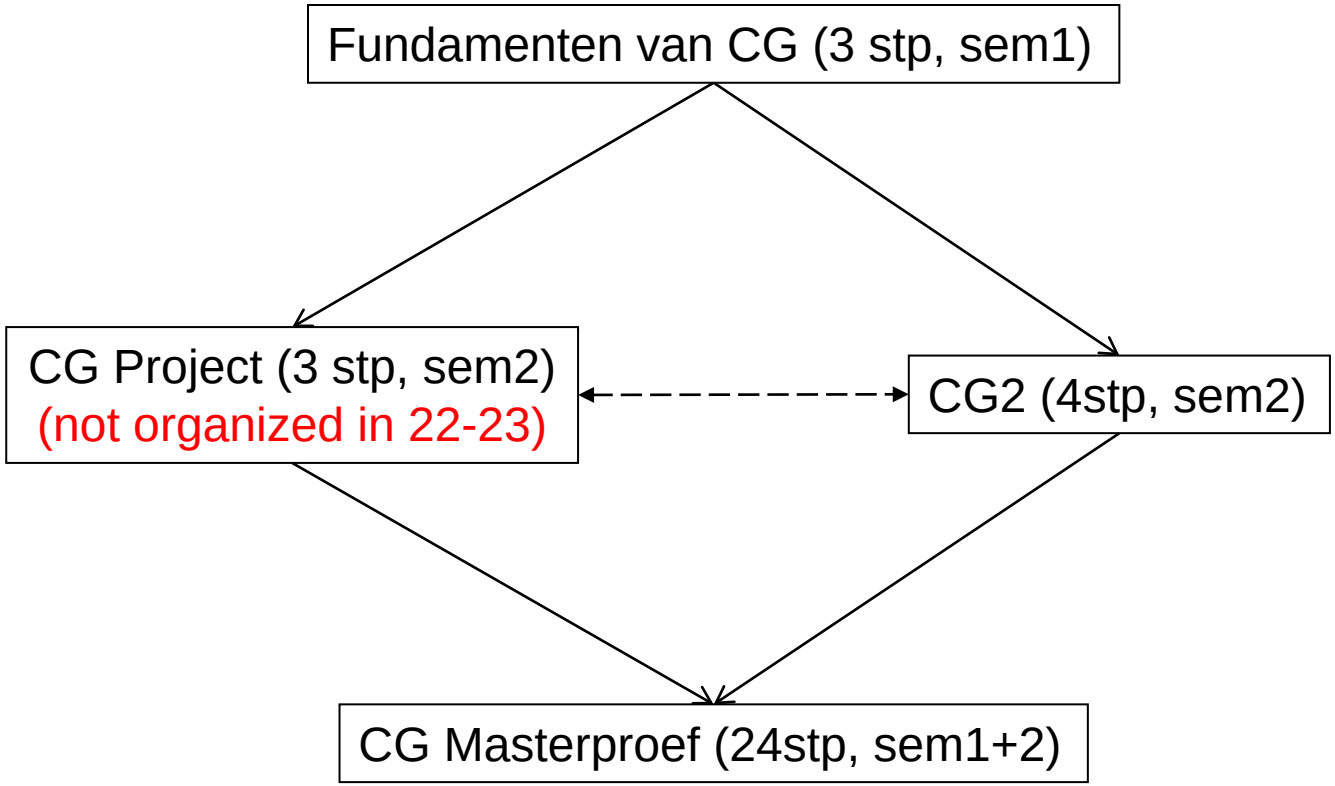
10 lectures, 2 hours each

No project for this course

Final evaluation for entire course

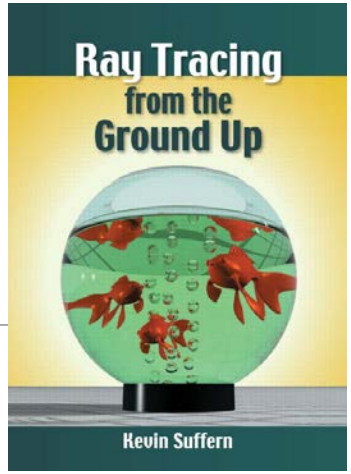
- Written exam during exam period (examples on TOLEDO)
- Closed book

Admin:
Other Graphics Courses



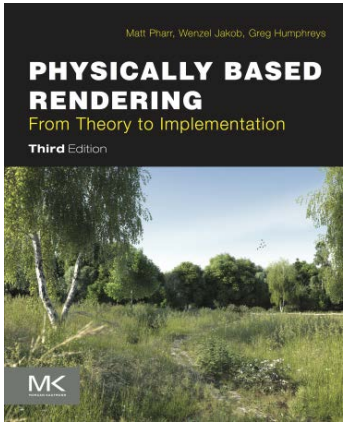
Admin:
Course Materials

Ray Tracing from the Ground Up



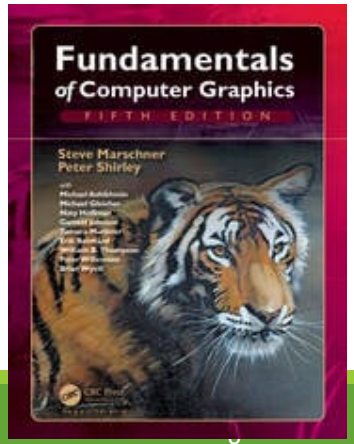
Other books are also usable:

- Physically Based Rendering
- Fundamentals of Computer Graphics
- …
- (some study material is from these books)



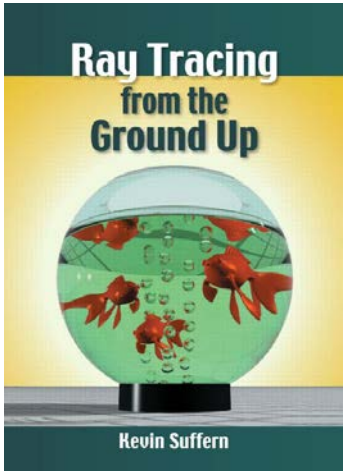
Slides + additional course material available online

Course summary on Google Docs



Admin:
Course Materials

How will we use this book?



Lectures:

- Overall motivation, how things fit together
- Book: detailed explanation + code

What is Computer Graphics?

????

Computer graphics deals with generating images with the aid of computers. Today, computer graphics is a core technology in digital photography, film, video games, cell phone and computer displays, and many specialized applications. A great deal of specialized hardware and software has been developed, with the displays of most devices being driven by computer graphics hardware. It is a vast and recently developed area of computer science. The phrase was coined in 1960 by computer graphics researchers Verne Hudson and William Fetter of Boeing. It is often abbreviated as CG, or typically in the context of film as computer generated imagery (CGI). The non-artistic aspects of computer graphics are the subject of computer science research.[1]

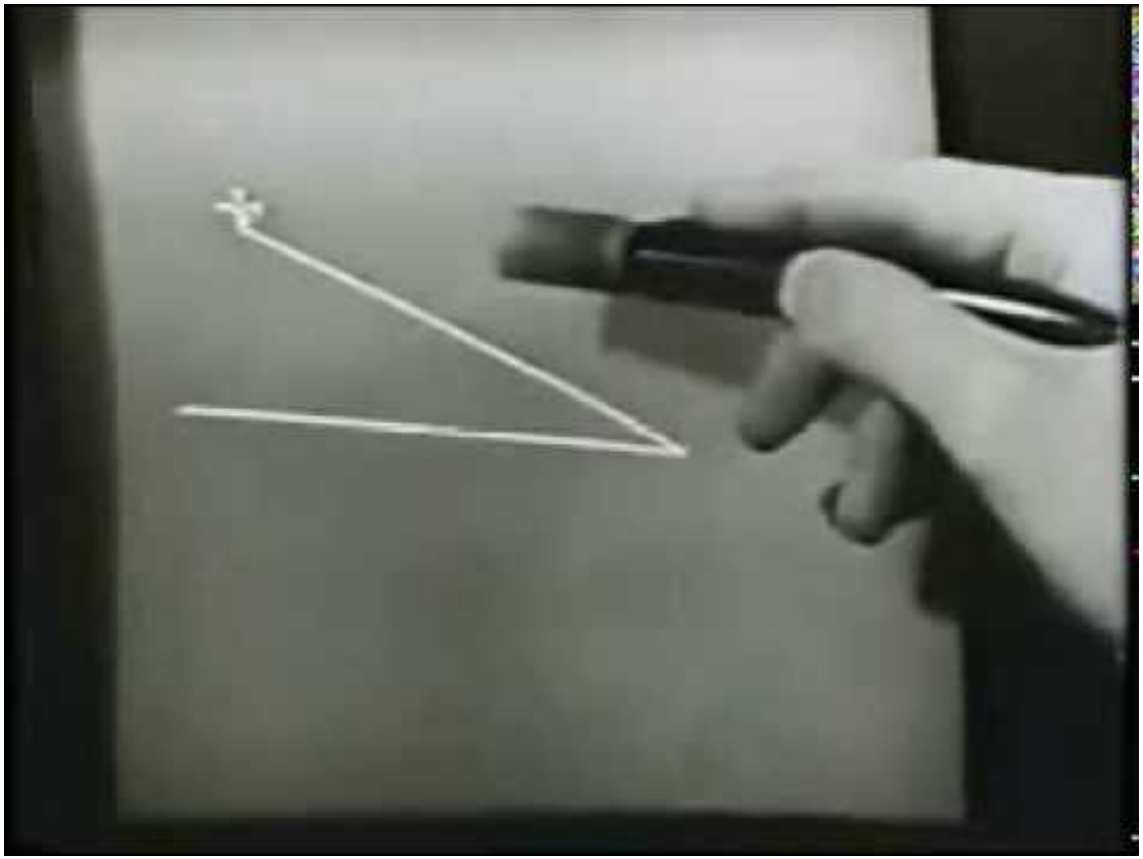
Some topics in computer graphics include user interface design, sprite graphics, rendering, ray tracing, geometry processing, computer animation, vector graphics, 3D modeling, shaders, GPU design, implicit surfaces, visualization, scientific computing, image processing, computational photography, scientific visualization, computational geometry and computer vision, among others. The overall methodology depends heavily on the underlying sciences of geometry, optics, physics, and perception.

(Wikipedia, Sept 2022)

What is Computer Graphics?

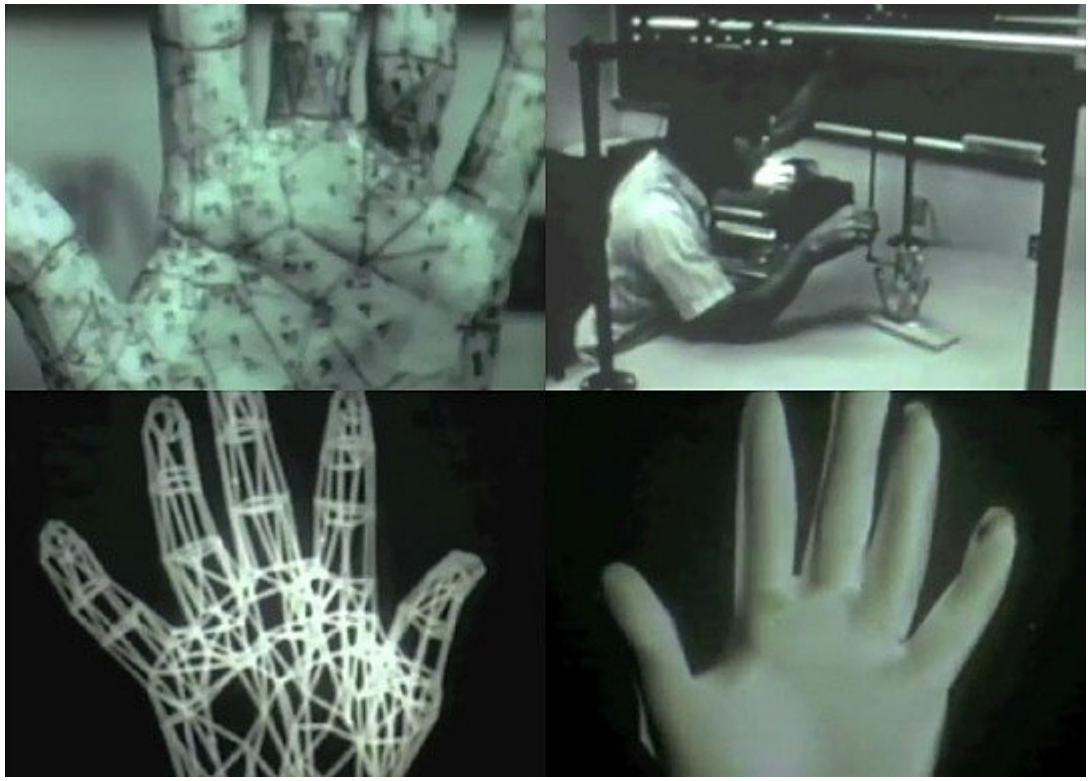
“Computer graphics enhances the human imagination.”

“Computer graphics enhances the human imagination”



(Sketchpad, 1963)

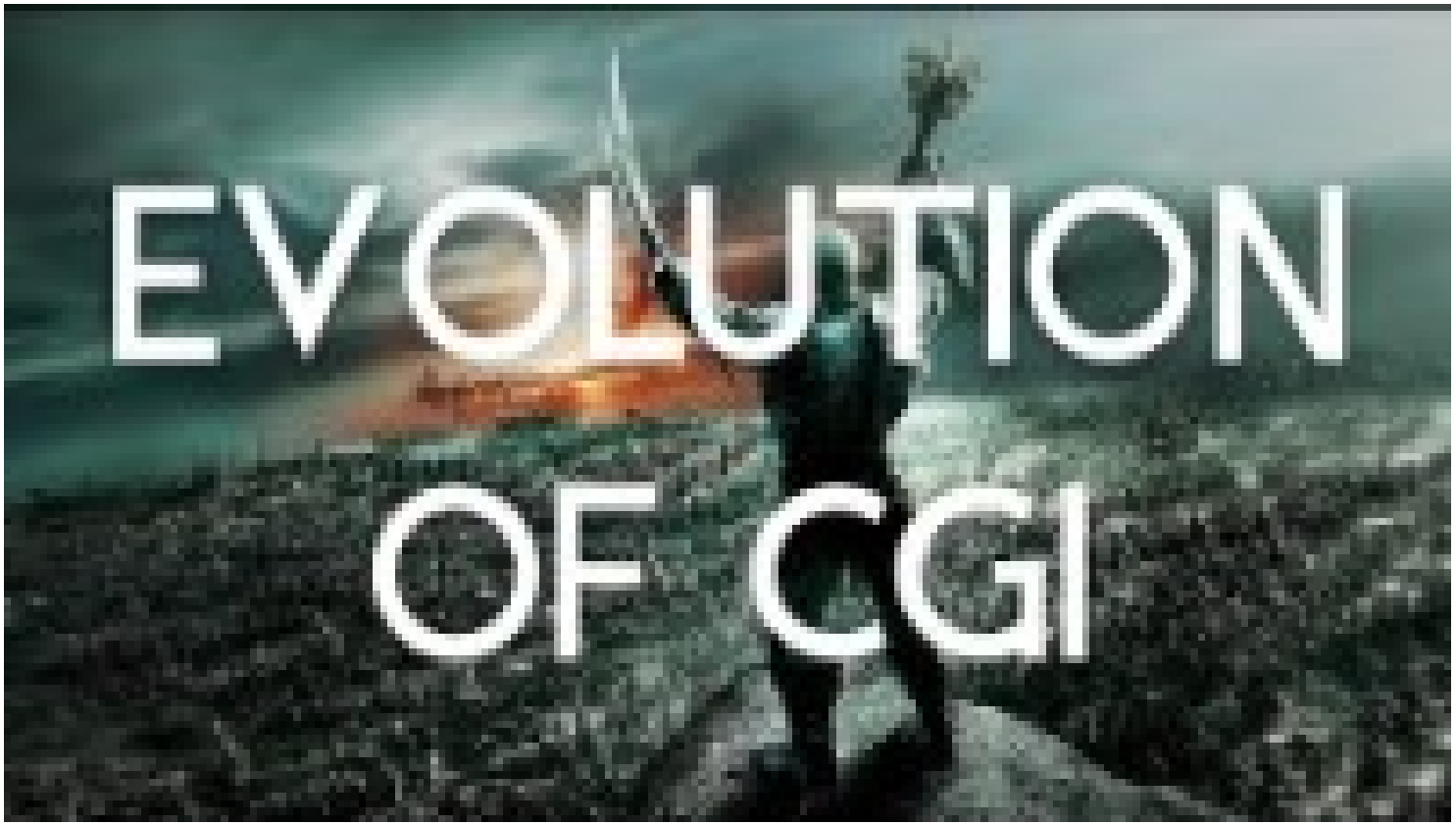
“Computer graphics enhances the human imagination”



(Ed Catmull, 1972)

(image sequence inspired by Nvidia launch event August 2018)

“Computer graphics enhances the human imagination”

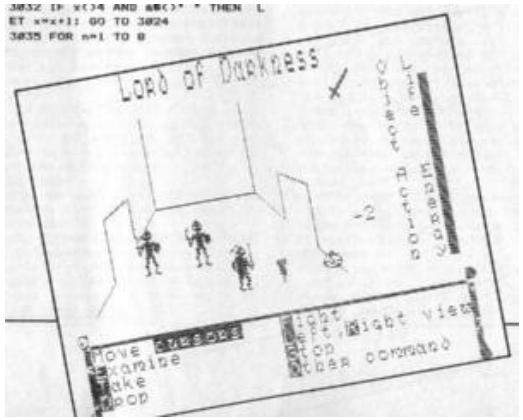


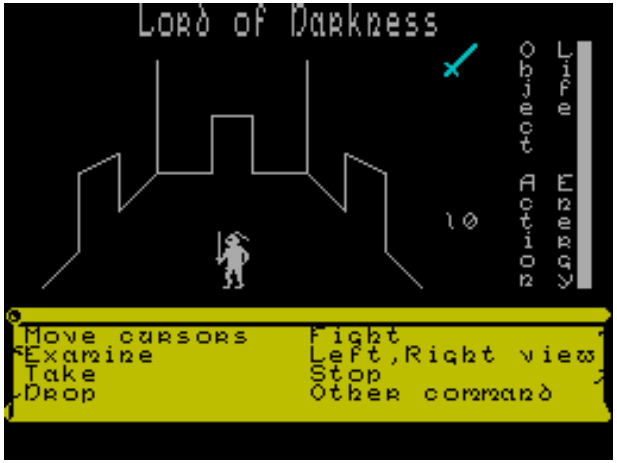
https://youtu.be/pWqT8i4GaMo

Games

https://youtu.be/vUpv-qyxr3A

Personal History …





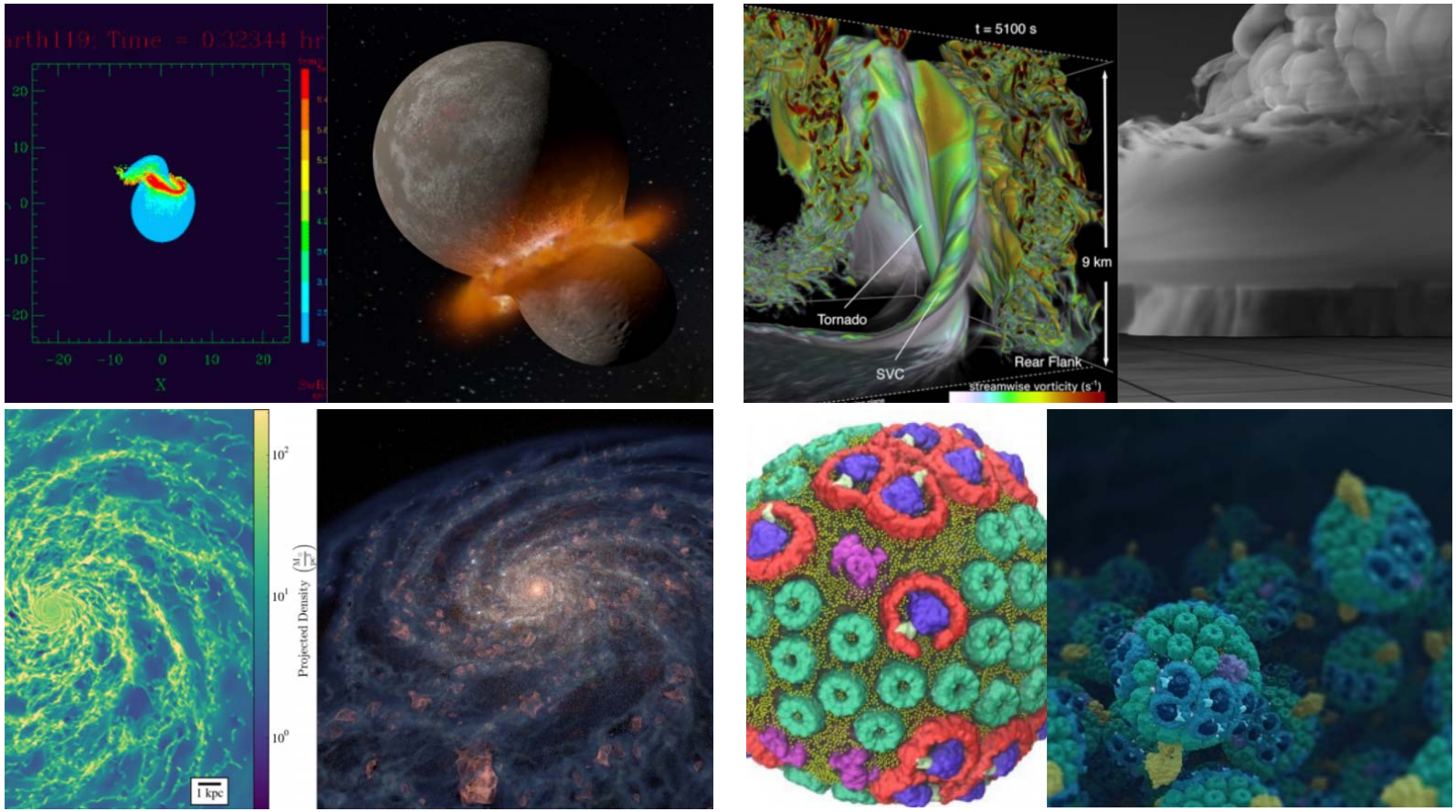
(ZX Computing, 1985)

Personal History …

Medical Imaging

https://youtu.be/QoWSgxzrtjU

Scientific Visualization



https://blog.siggraph.org/2019/07/why-cinematic-scientific-visualization-is-more-important-than-ever.html/

Architecture

Interior Design

When You Flip Through an IKEA Catalog, 75% of the 'Photography' You See is CGI

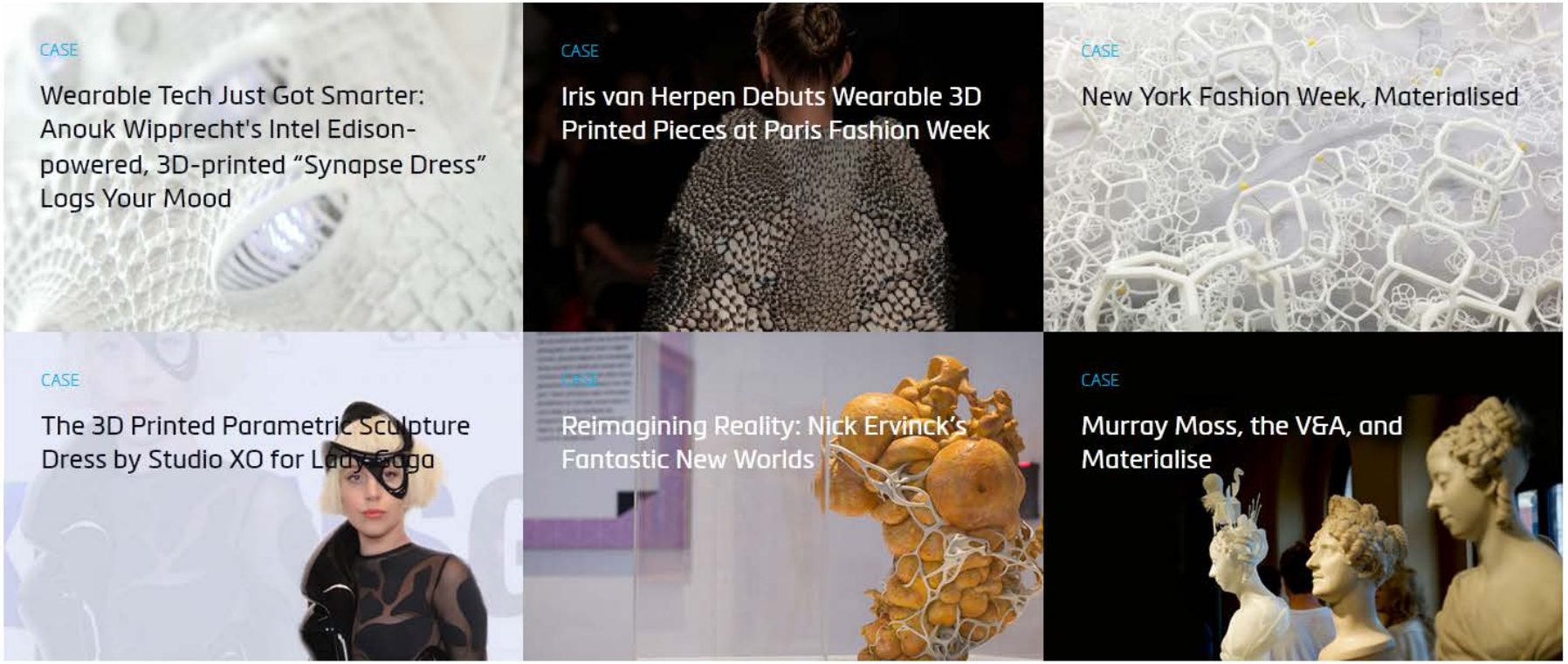
AUG 28, 2014 DL CADE

Share 255 Tweet 19 COMMENTS

Digital Art

(SIGGRAPH 2019 Art Gallery Preview)

Art & 3D printing



(www.materialise.com)

Online Design

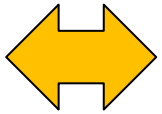
https://www.heroforge.com/

Some Recent Research

What will we learn in this course?

Fundamentals of computer graphics rendering algorithms

- Entire algorithmic pipeline from 3D model to 2D image
- Not: Specific APIs; Graphics applications (game design, Photoshop, …); artistic skills; …



What will we learn in this course?

Fundamentals of computer graphics rendering algorithms

- Entire algorithmic pipeline from 3D model to 2D image
- Focus on mathematics and physics
- Strong focus on modeling & generating images
- Less (or no) focus on animation

- Main algorithmic approach: ray tracing

3D Computer Graphics

Modeling

Rendering

Animation

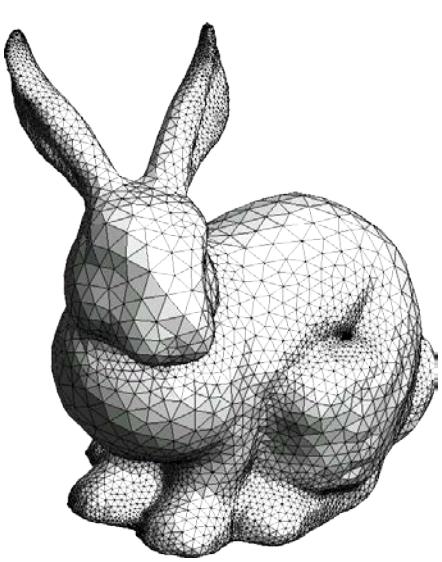
3D Computer Graphics

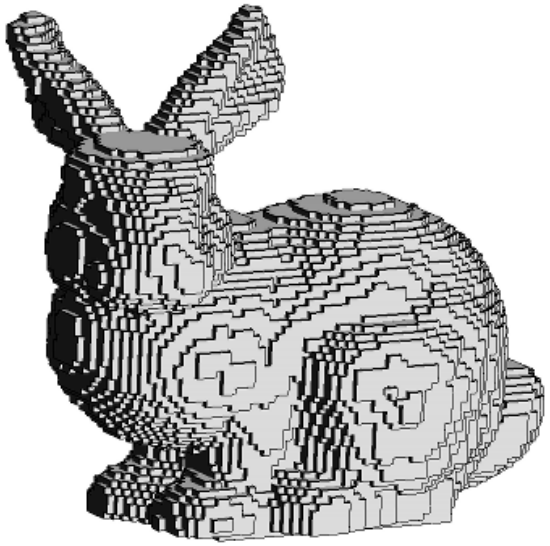
Modeling

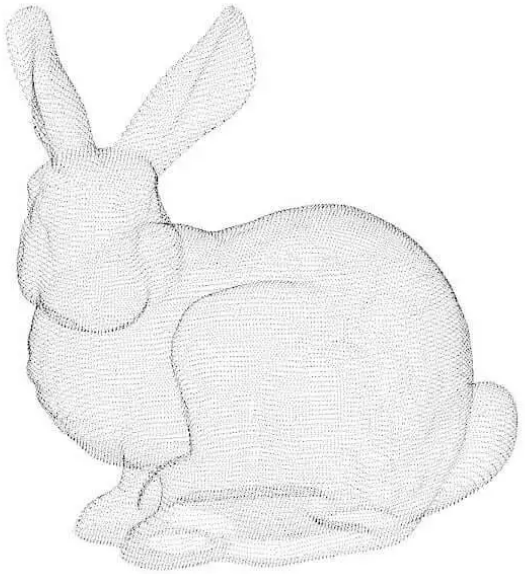
- Representation of 3D models, capturing, manipulation, scanning, …

Rendering

Animation







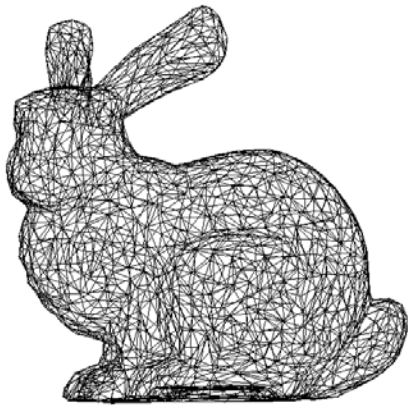
3D Computer Graphics

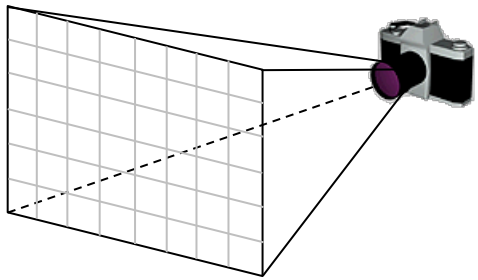
Modeling

Rendering

- 3D-to-image, shading, textures, materials, …

Animation





3D Computer Graphics

Modeling

Rendering

- 3D-to-image, shading, textures, materials, …

Animation

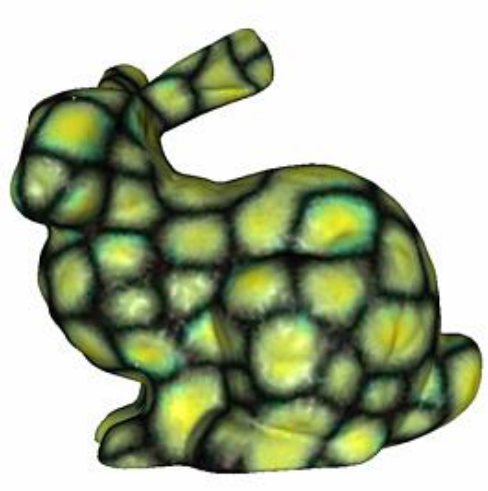
3D Computer Graphics

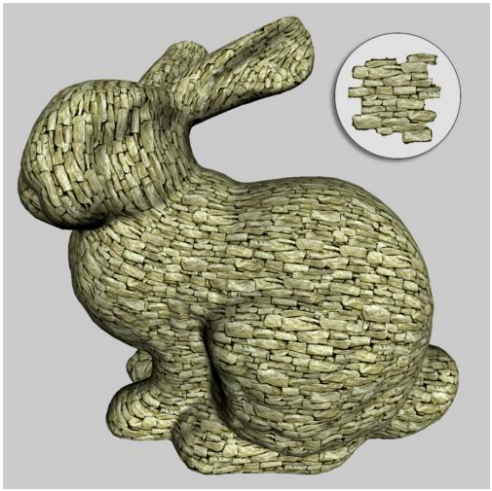
Modeling

Rendering

- 3D-to-image, shading, textures, materials, …

Animation





3D Computer Graphics

https://blenderartists.org/t/bunny-stanford-stress-test/698581

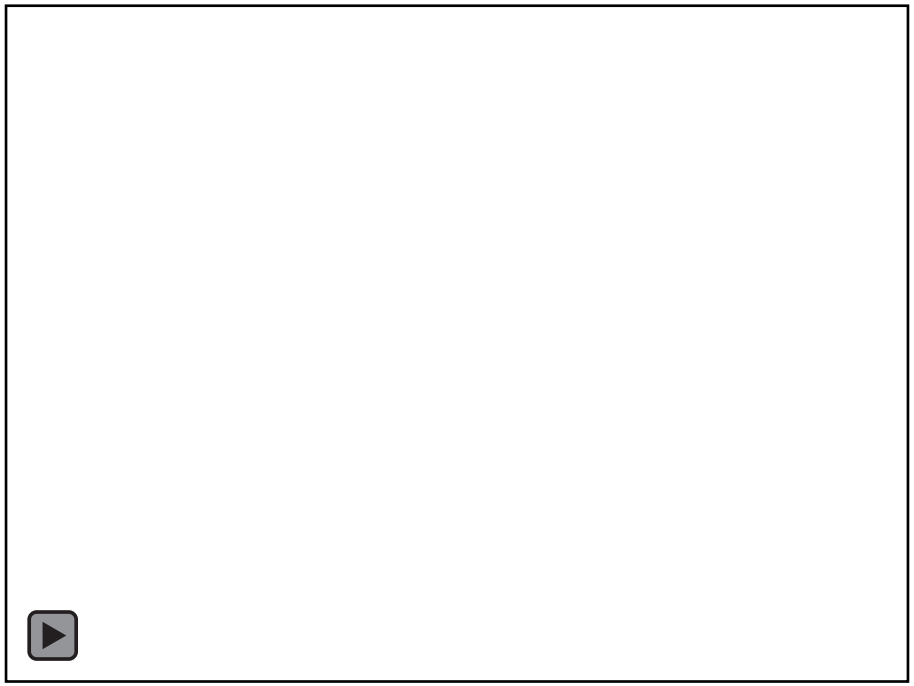
3D Computer Graphics

Modeling

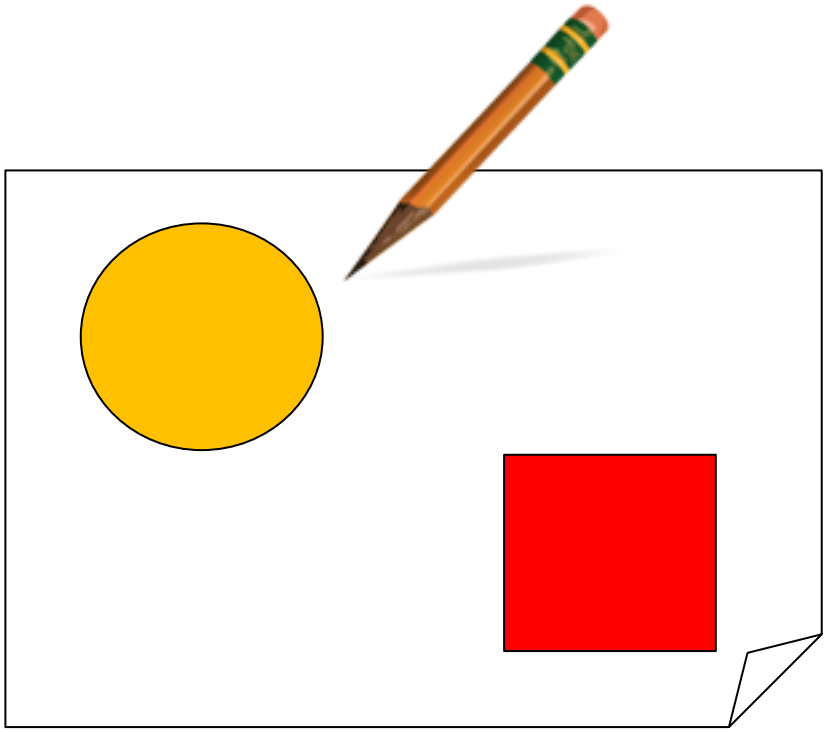
Rendering

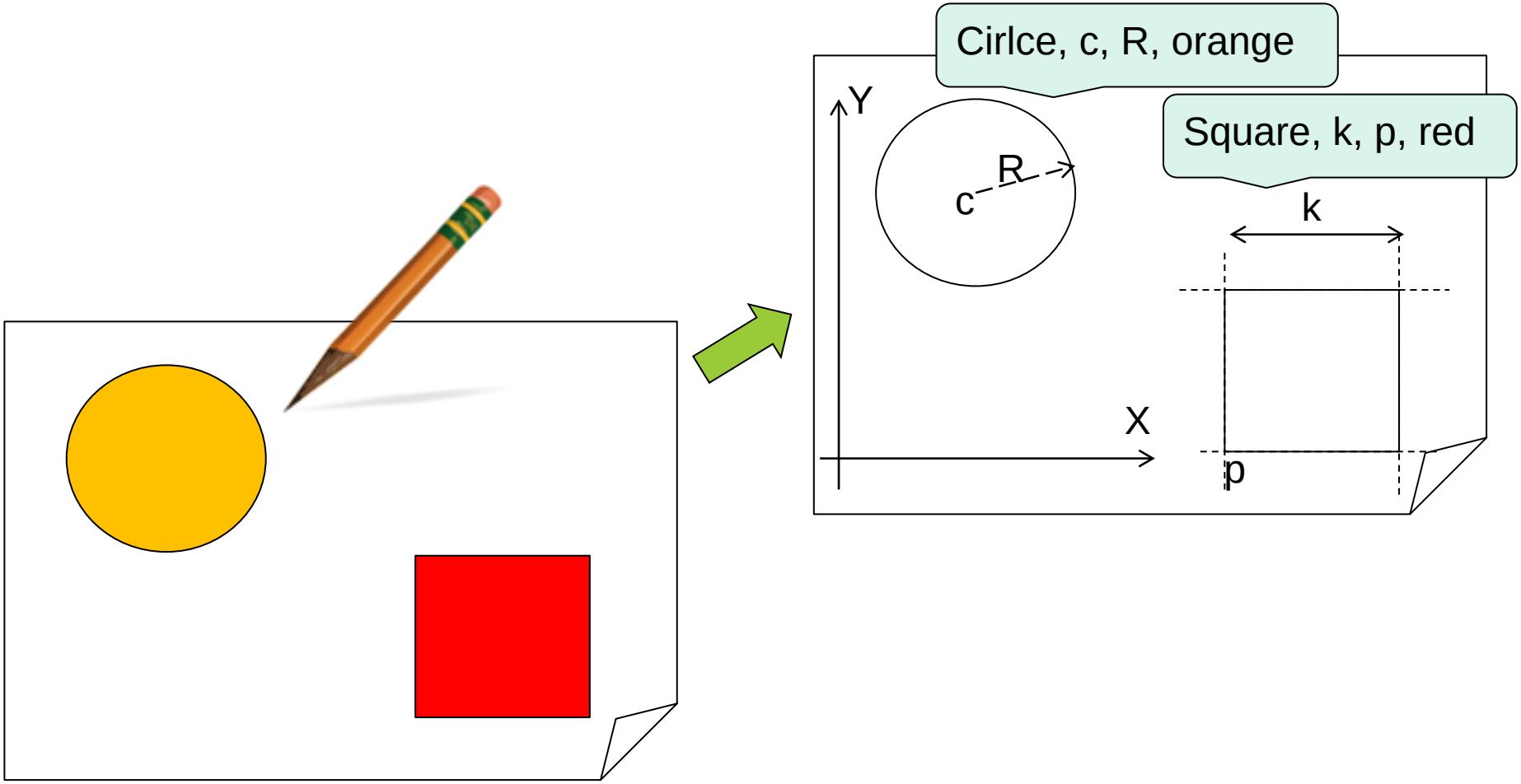
- 3D-to-image, shading, **textures**, materials, …

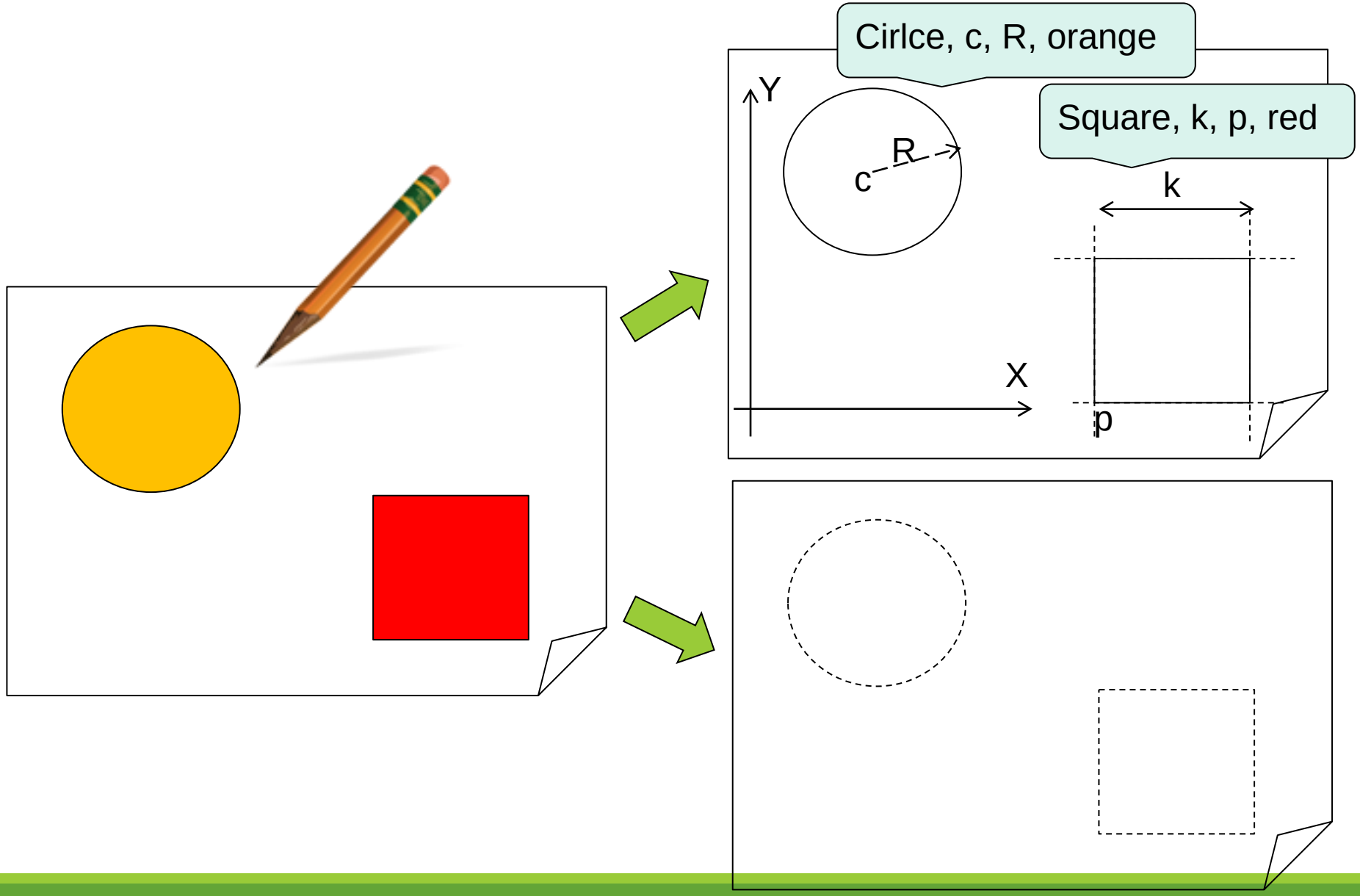
Animation

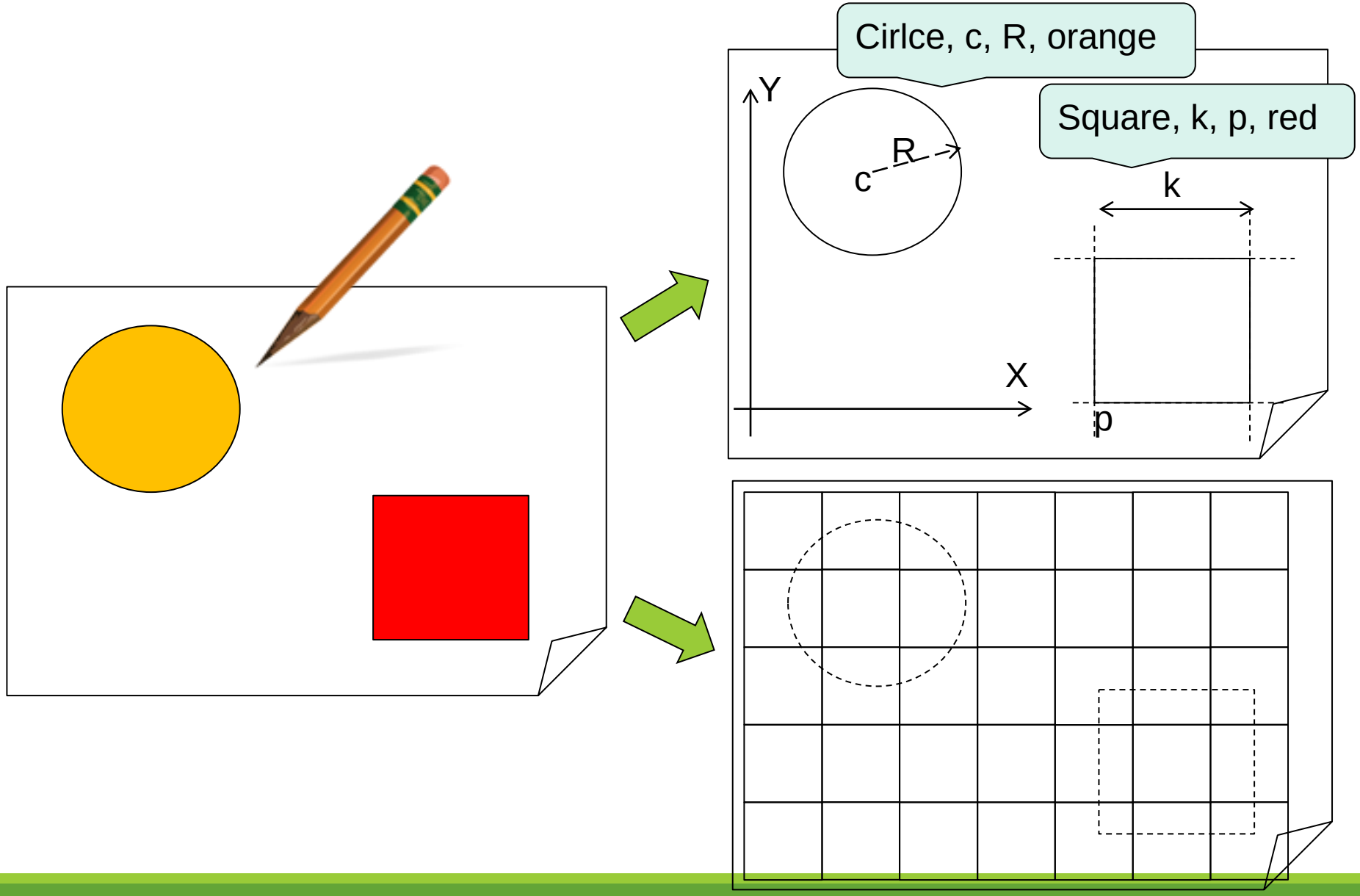


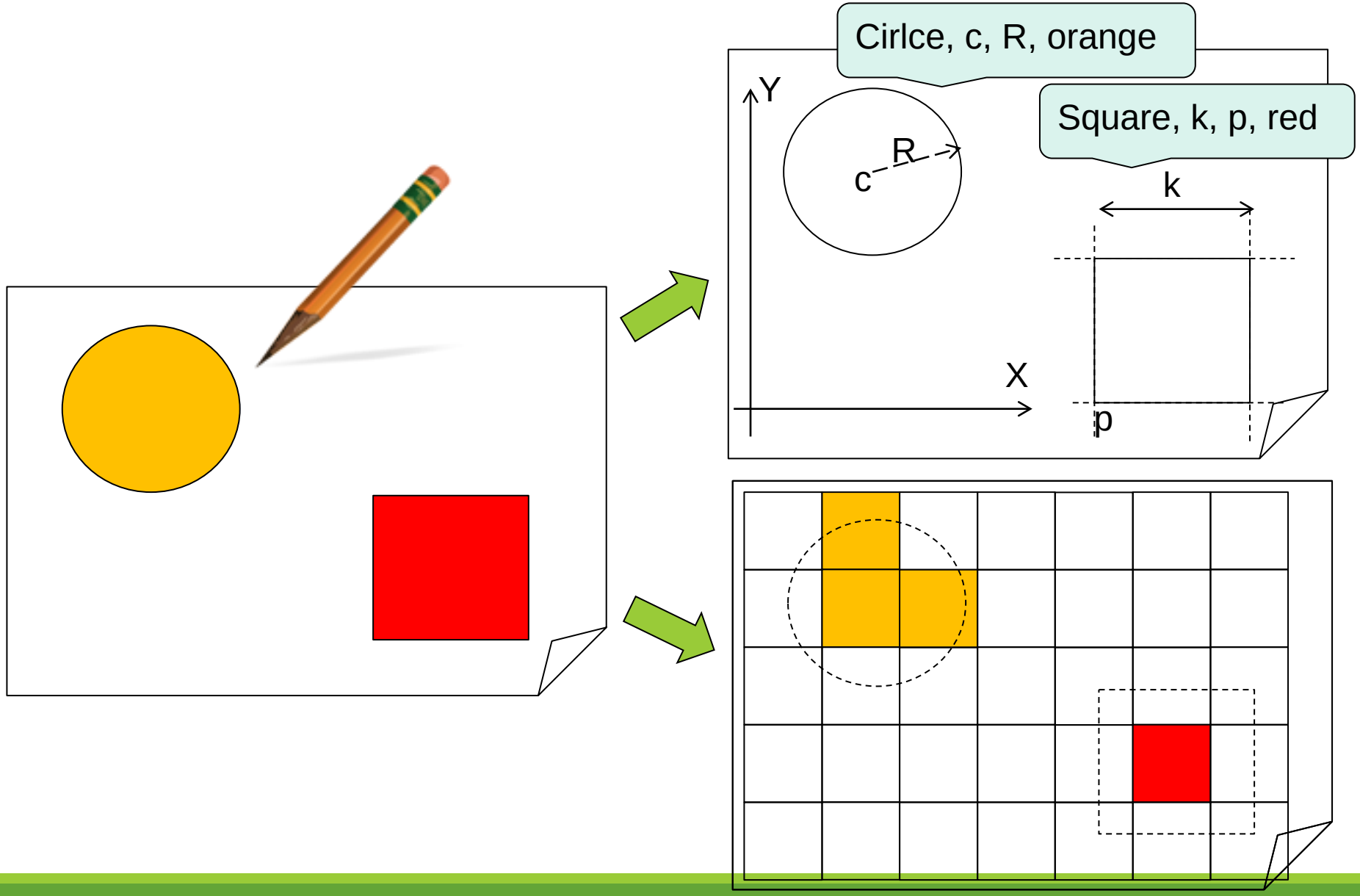
Preliminaries: Images

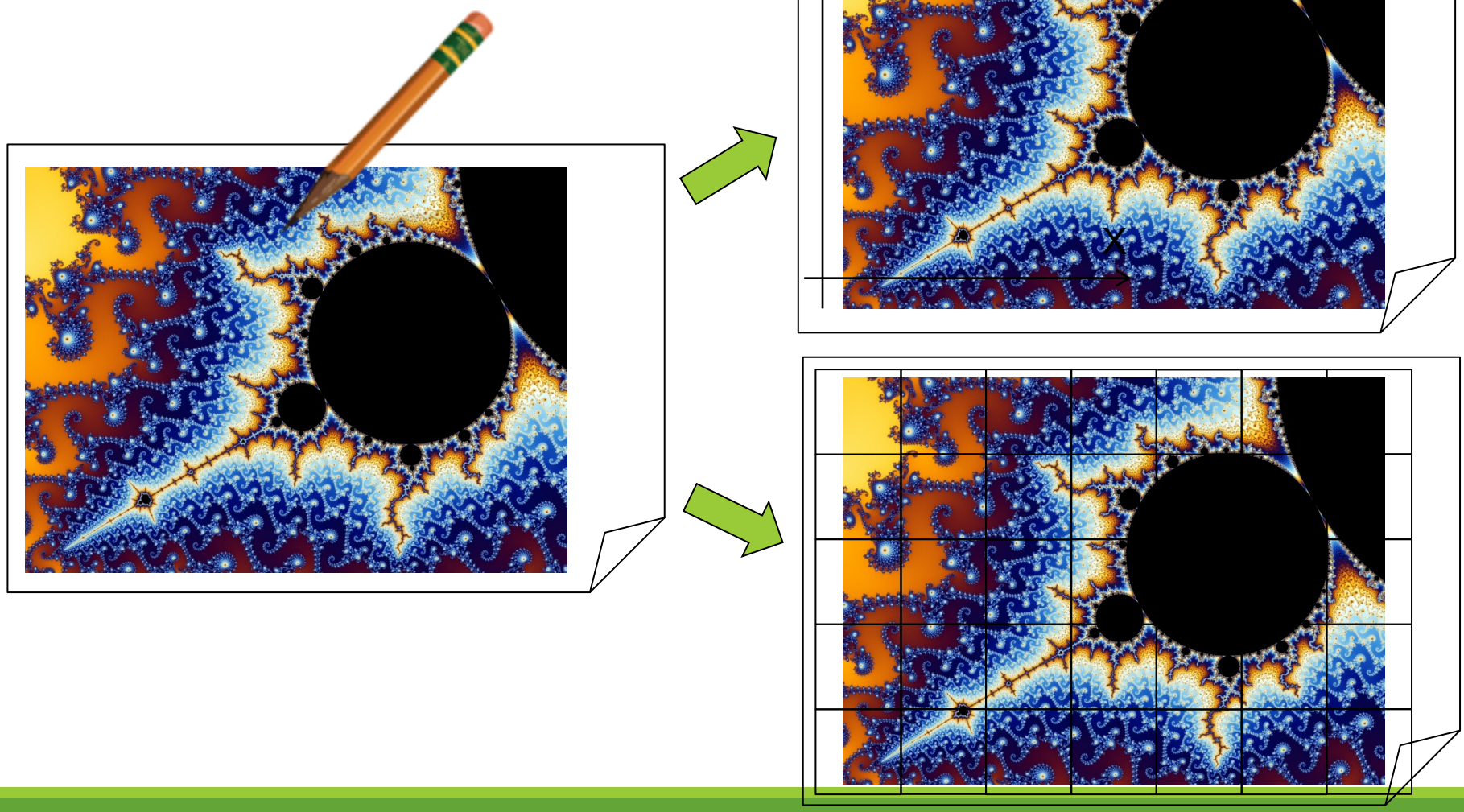


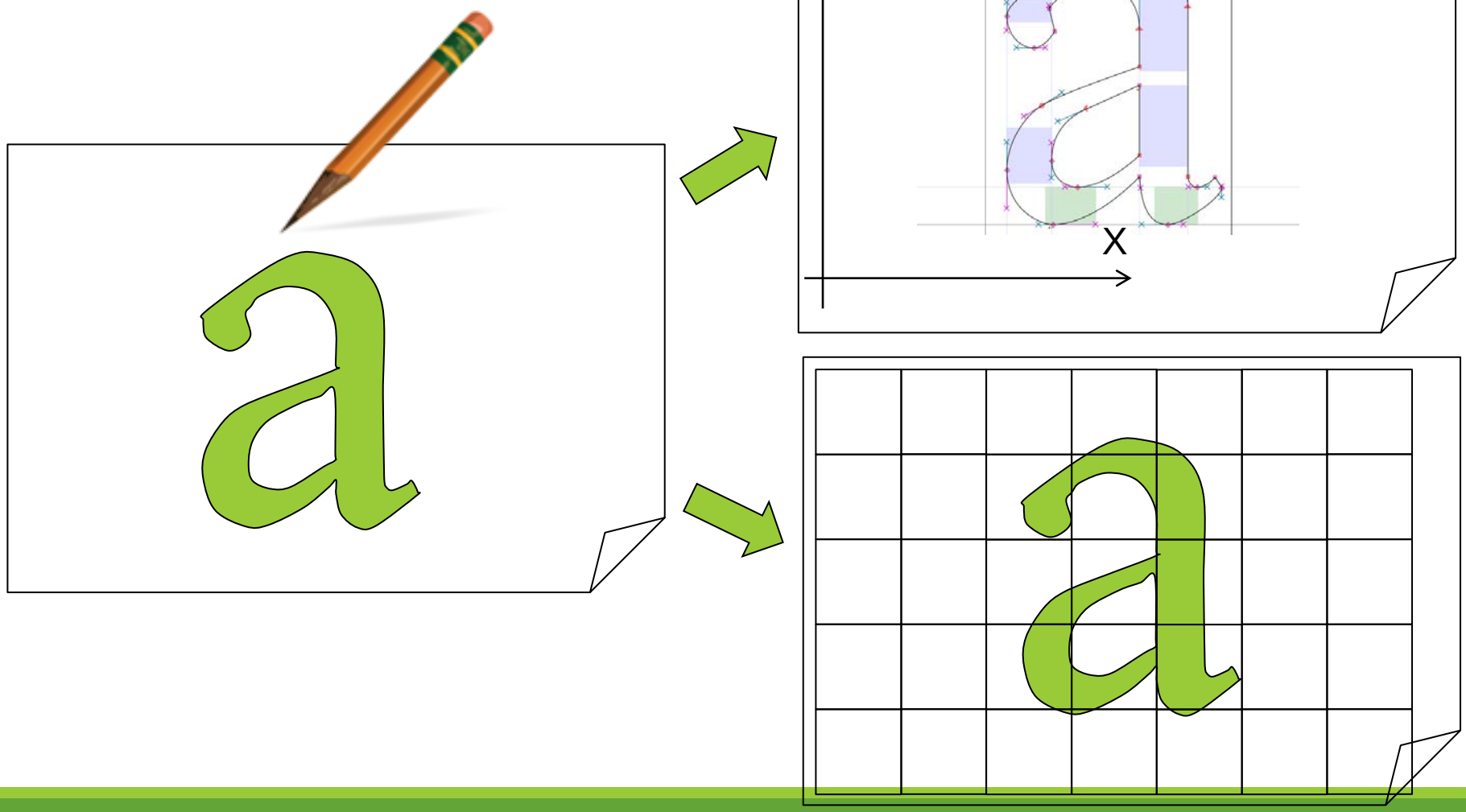

Cirlce, c, R, orange
Square, k, p, red
Y
X
c
R
k
p


Cirlce, c, R, orange
Square, k, p, red
Y
X
c
R
k
p


Cirlce, c, R, orange
Square, k, p, red
Y
X
c
R
k
p


Cirlce, c, R, orange
Square, k, p, red
Y
X
c
R
k
p


F(x,y) = 0
Y
X


F(x,y) = 0
Y
X

Raster image representation

2D array of numbers

Major advantage: arbitrary content

- approximate arbitrary functions with increasing resolution
- works because memory is cheap (brute force approach!)

Disadvantage: some operations are difficult

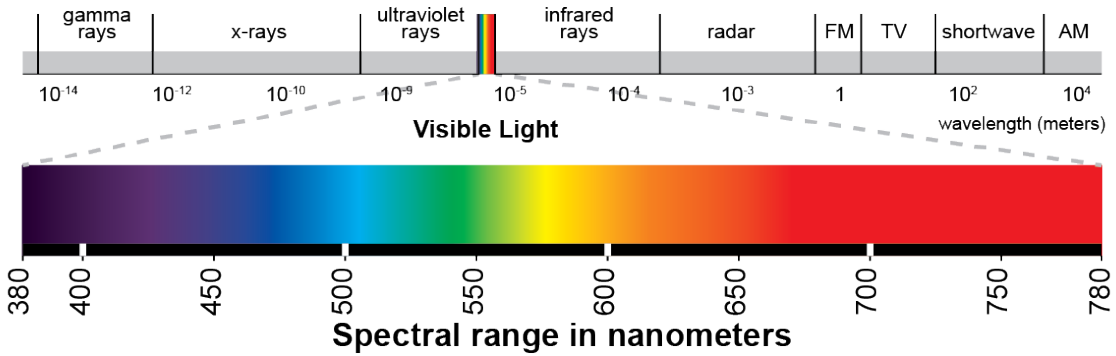
- E.g. scaling / zooming

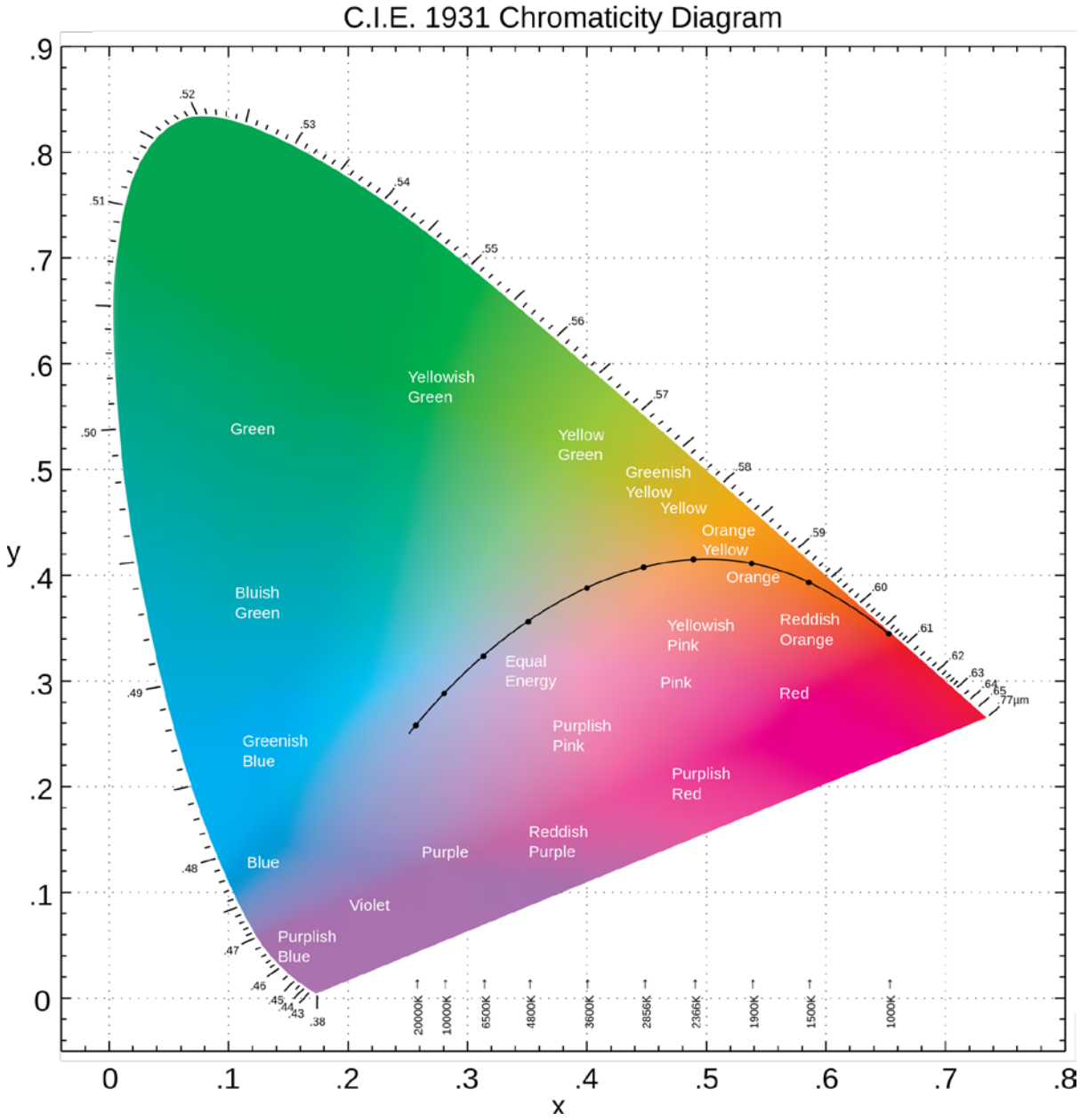
Preliminaries: Colors

Colors

Color science

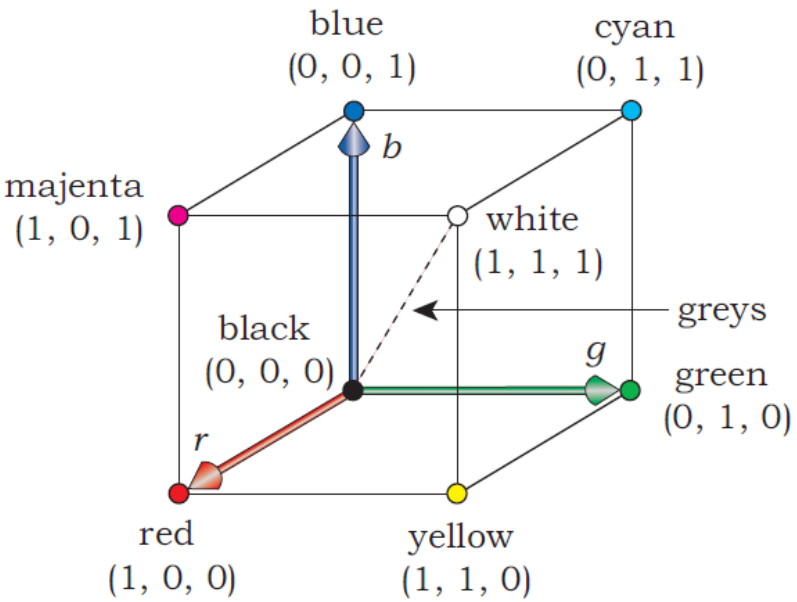
- Physics
  - Radiometry
  - Light reflection
- Perception
  - (Human) Eye
- Psychology
  - Meaning, mood, …
- …

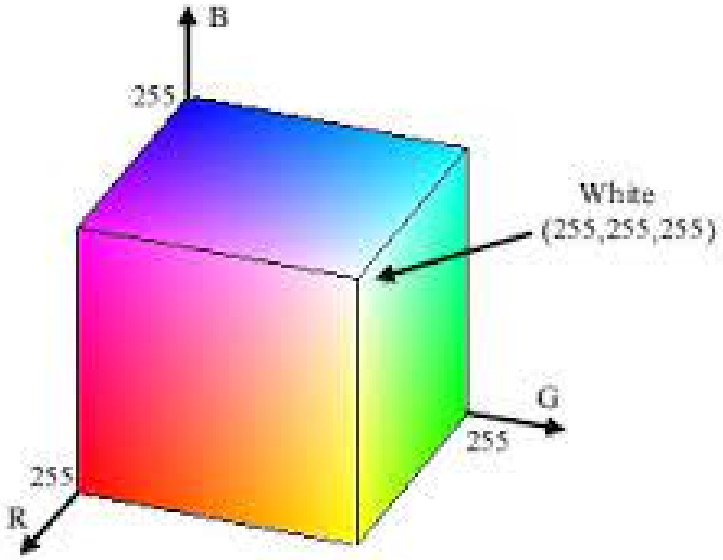



C.I.E. 1931 Chromaticity Diagram
y
x
Green
Yellowish Green
Yellow Green
Greenish Yellow
Yellow
Orange Yellow
Orange
Reddish Orange
Bluish Green
Yellowish Pink
Equal Energy
Pink
Red
Greenish Blue
Purplish Pink
Purplish Red
Reddish Purple
Purple
Blue
Violet
Purplish Blue
20000K
10000K
6500K
4800K
3600K
2856K
2366K
1900K
1500K
1000K

Representing colors

Red-green-blue additive color model



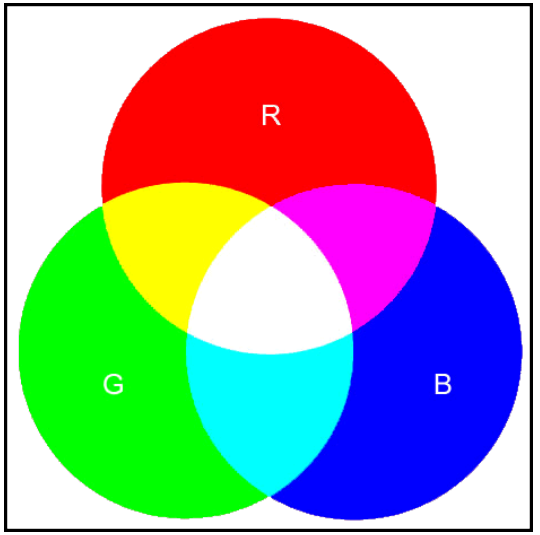


Not all visible colors can be represented by (positive) r,g,b triplets

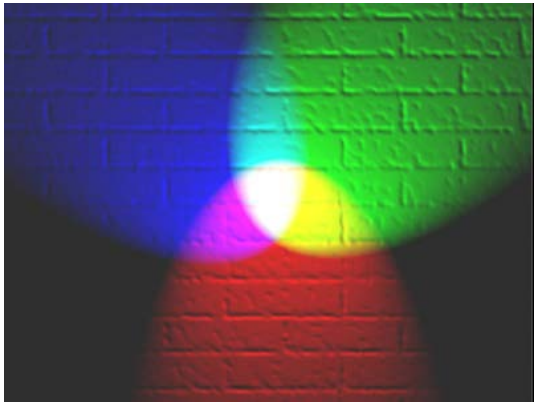
Representing colors

Additive RGB model:

- red + green = yellow
- green + blue = cyan
- blue + red = magenta
- red + green + blue = white



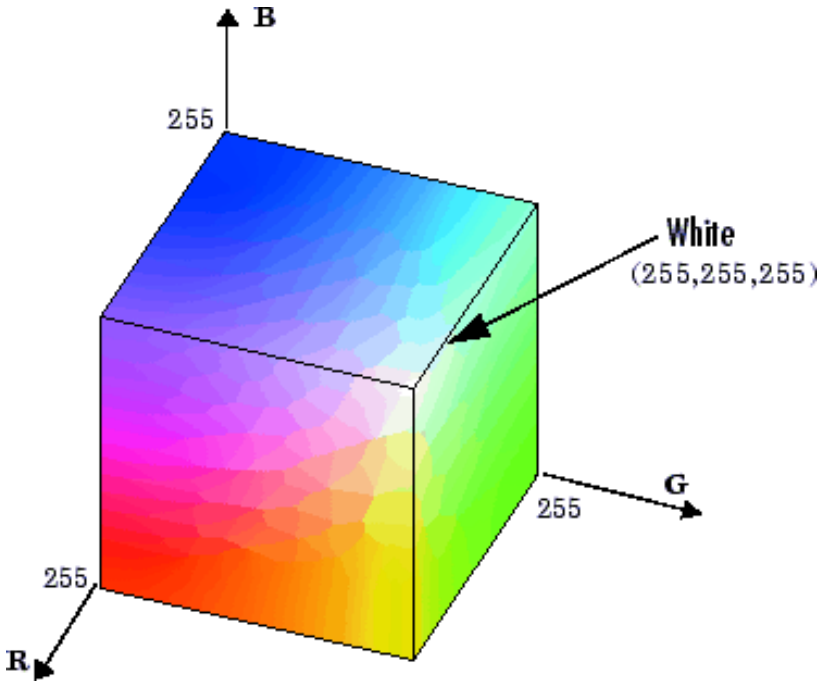
Approximation of true wavelength model of colors



Representing colors

Per pixel in an image:

- 1 byte for each color channel:
  - 256 different values each for red, green, blue
  - 16,777,216 different colors
  - ➔ “24 bit color”
- More bits / color channel are possible
  - Some file formats use floats …

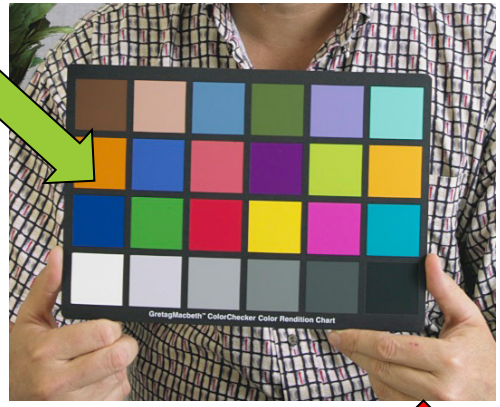


Representing colors

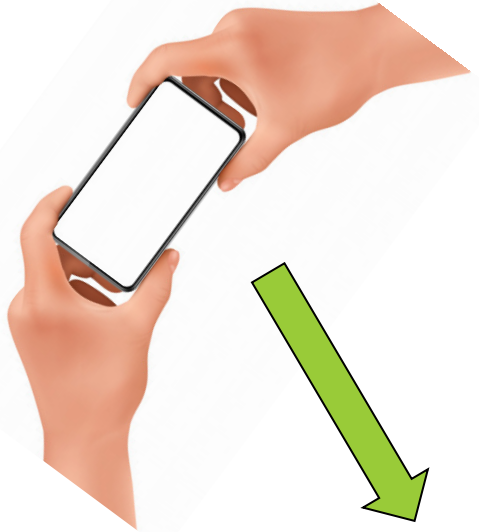
Be careful!

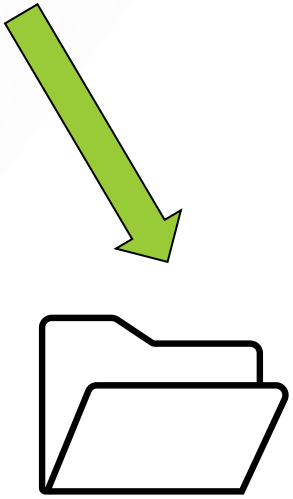
- Storing colors <> displaying colors <> capturing colors …
- “What color do I see when I look at a pixel with color (255,0,0) on my laptop? On my smartphone? On a printer? On a projector screen?”
- “What light spectrum did my camera detect when it attributes (255,0,0) to a pixel? Is this the same color when displaying (255,0,0) on a screen?”

1. Light reflects on colored object



2. Capture colors with camera



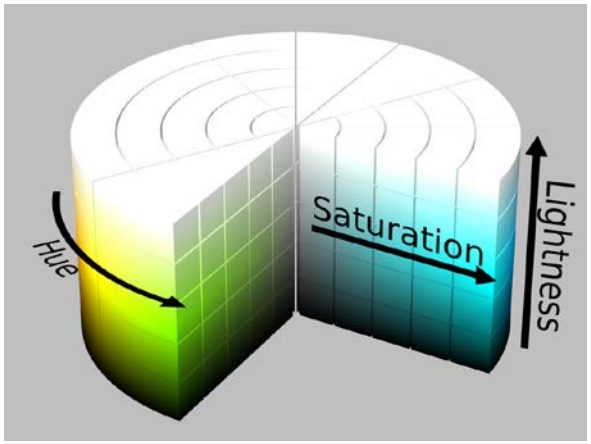


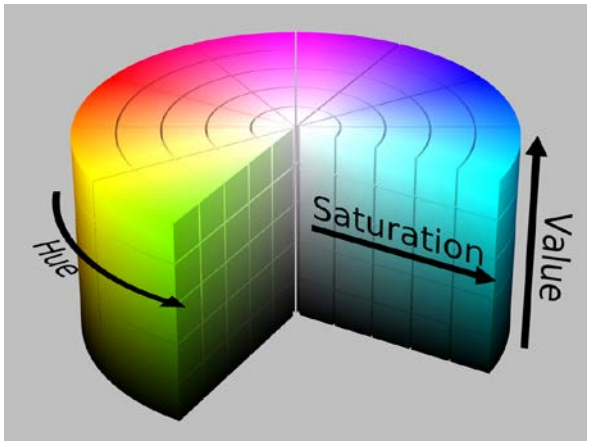
3. Store colors as numbers on file

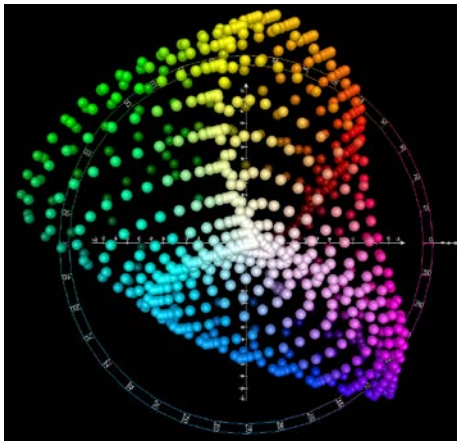
Do I see the same colors?

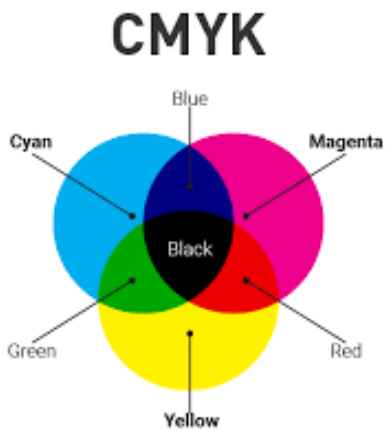
4. Show colors on screen

Many, many color models …



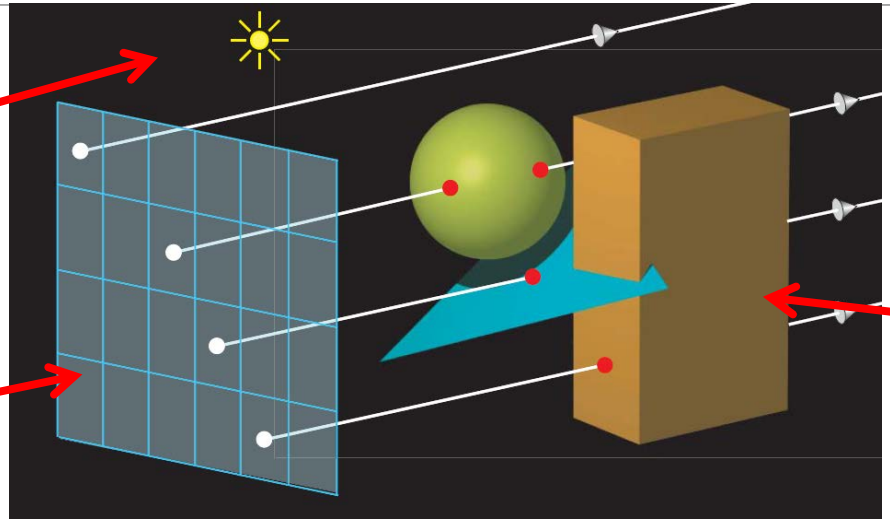






Ray Tracing: Basics

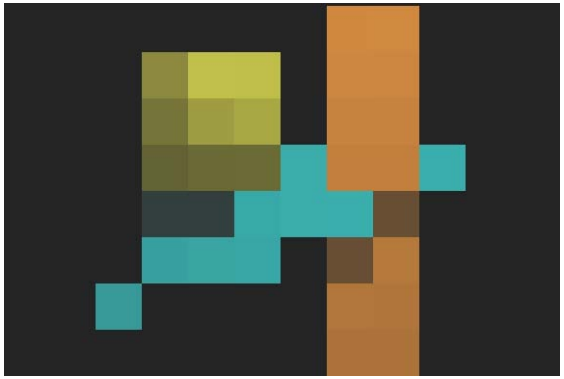
How Ray Tracing Works



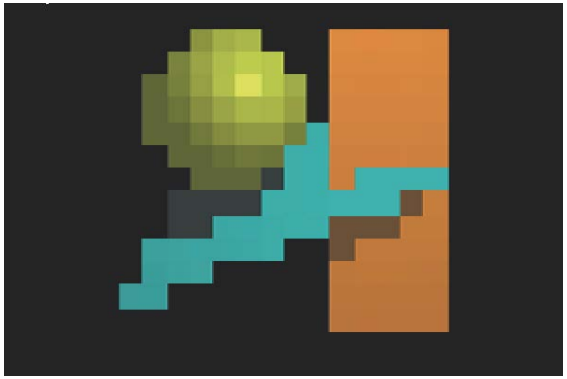

```
for each pixel
    shoot ray through pixel
    compute nearest hitpoint
    compute color of hitpoint
    pixel ← color
```

How Ray Tracing Works

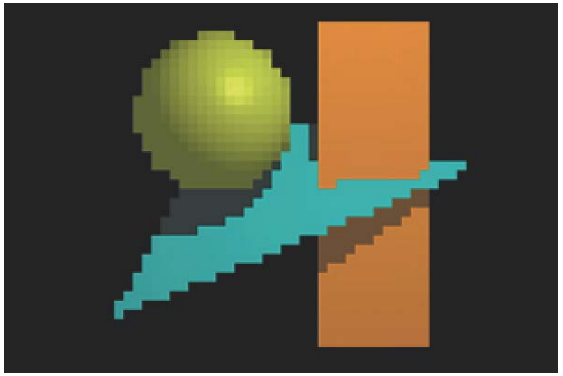
6x4



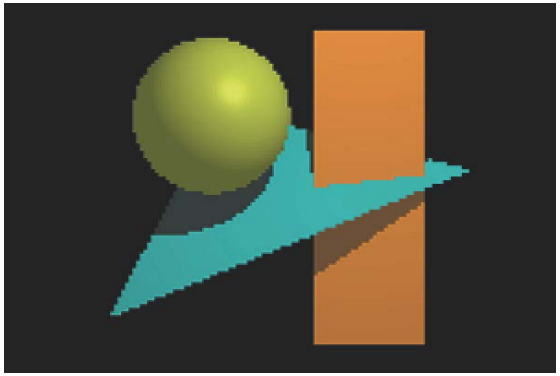
12x8



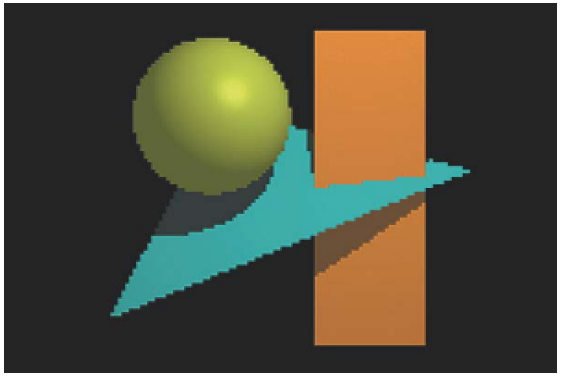
24x16



60x40



120x80



150x100

Ray Tracing by Dűrer

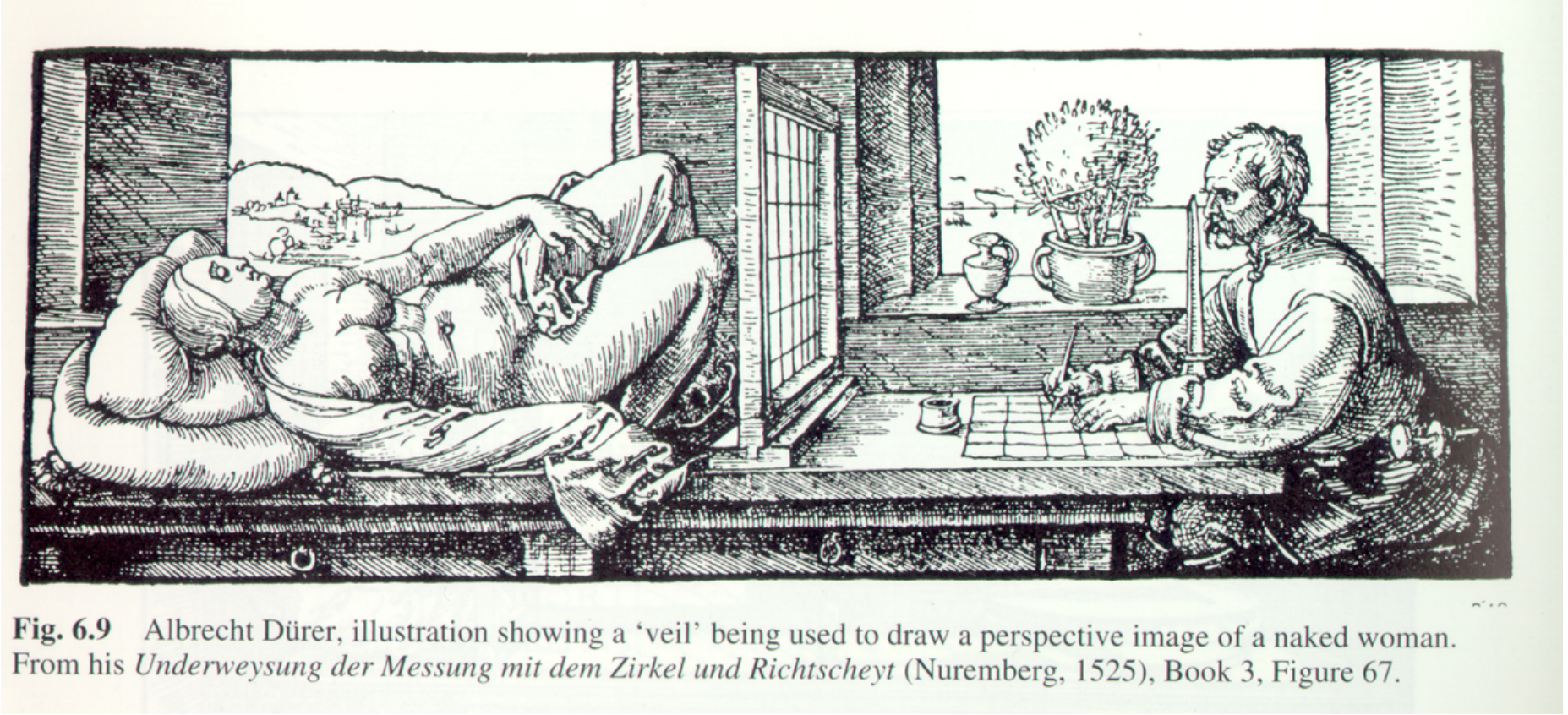
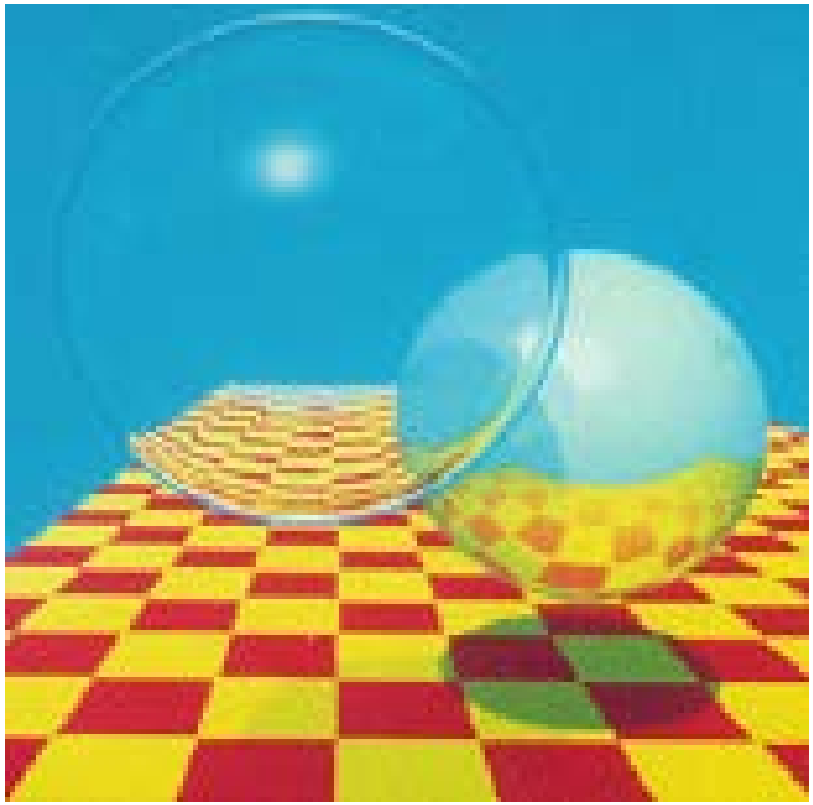
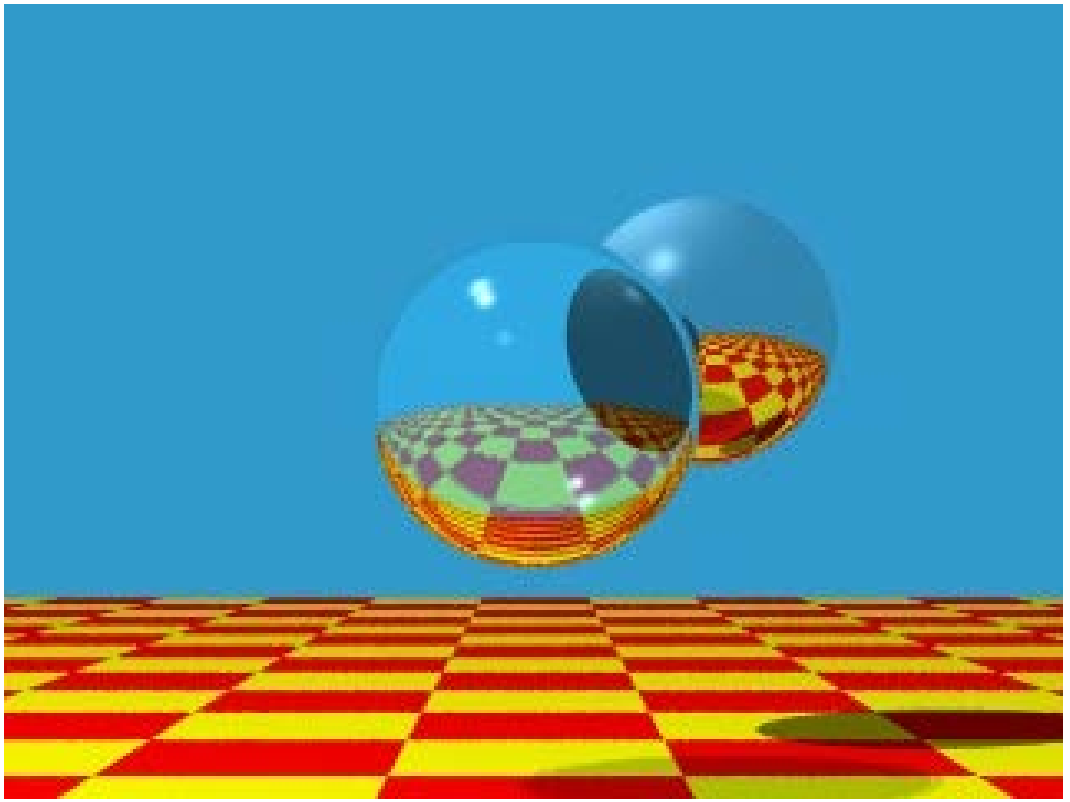


Fig. 6.9 Albrecht Dürer, illustration showing a 'veil' being used to draw a perspective image of a naked woman. From his *Underweysung der Messung mit dem Zirkel und Richtscheyt* (Nuremberg, 1525), Book 3, Figure 67.

Ray Tracing by Whitted

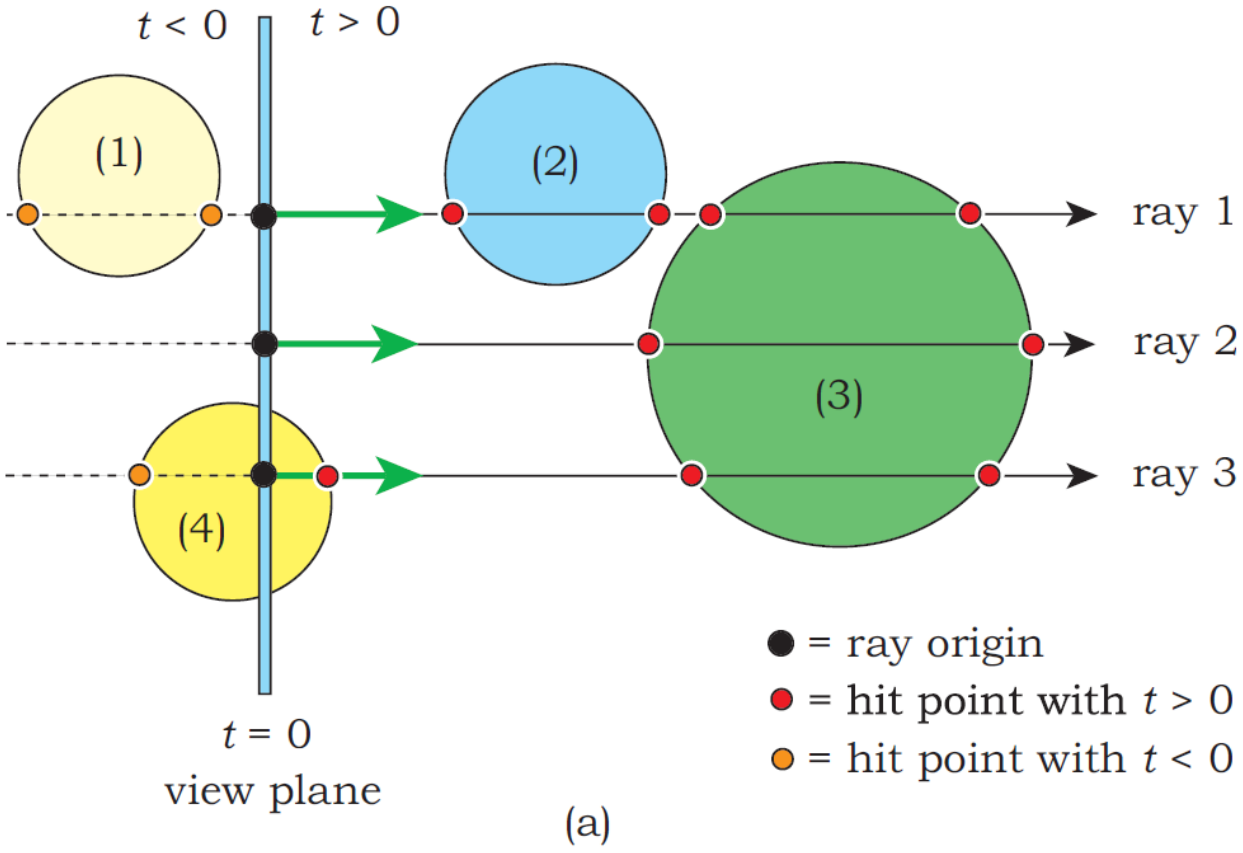




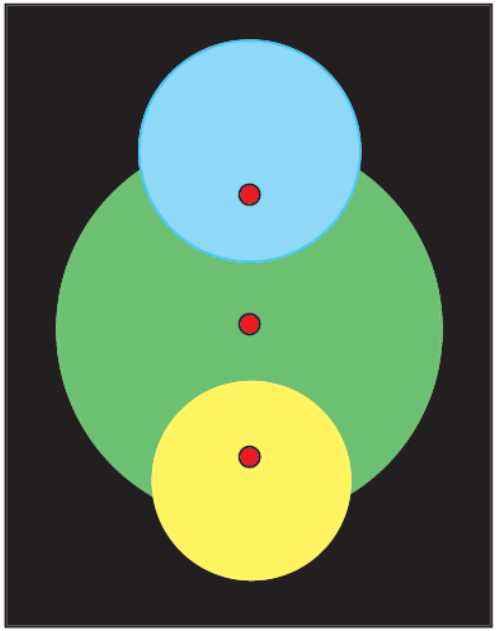
(Whitted 1980)

http://www.youtube.com/watch?v=WV4qXzM641o

Ray-object intersections

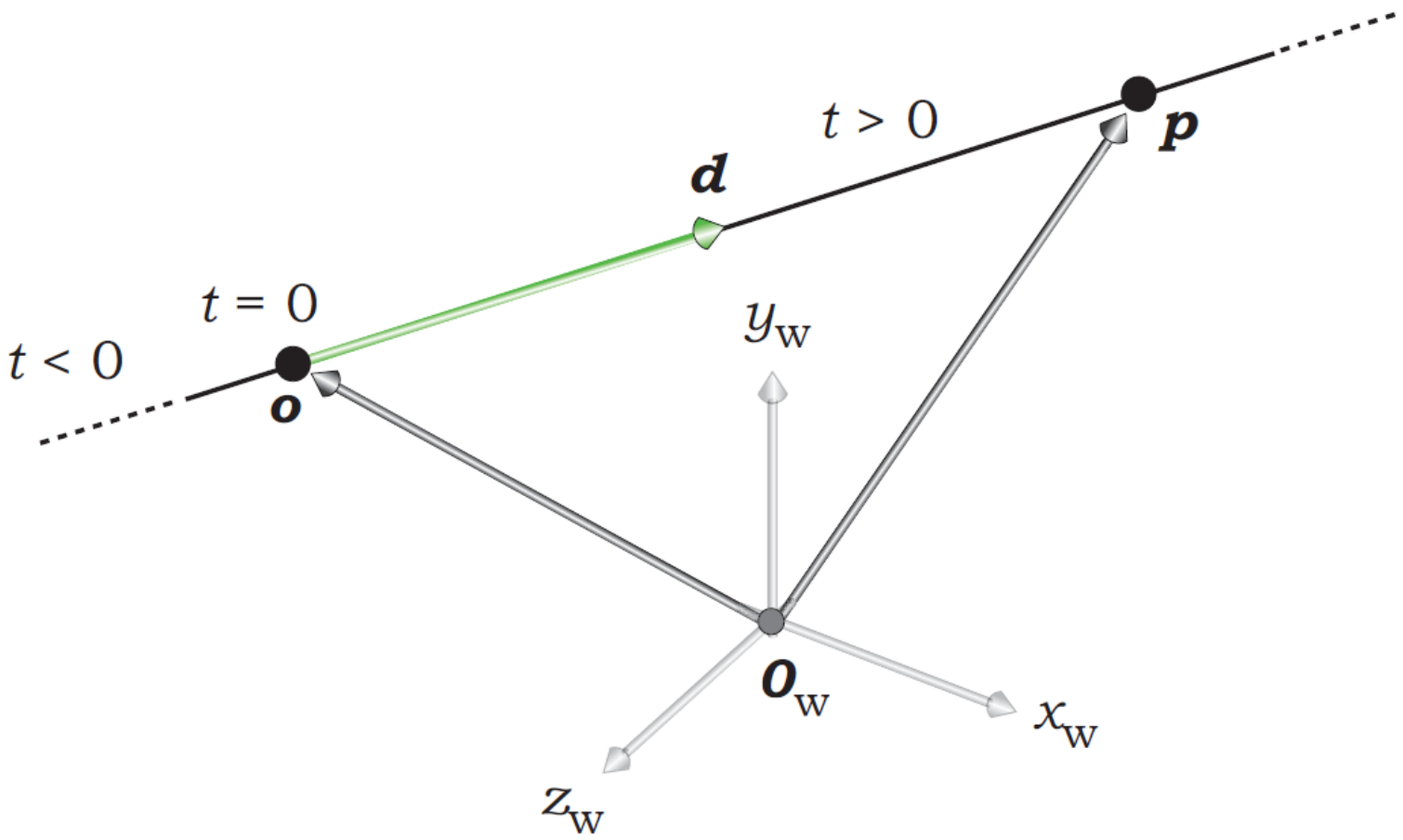


(a)



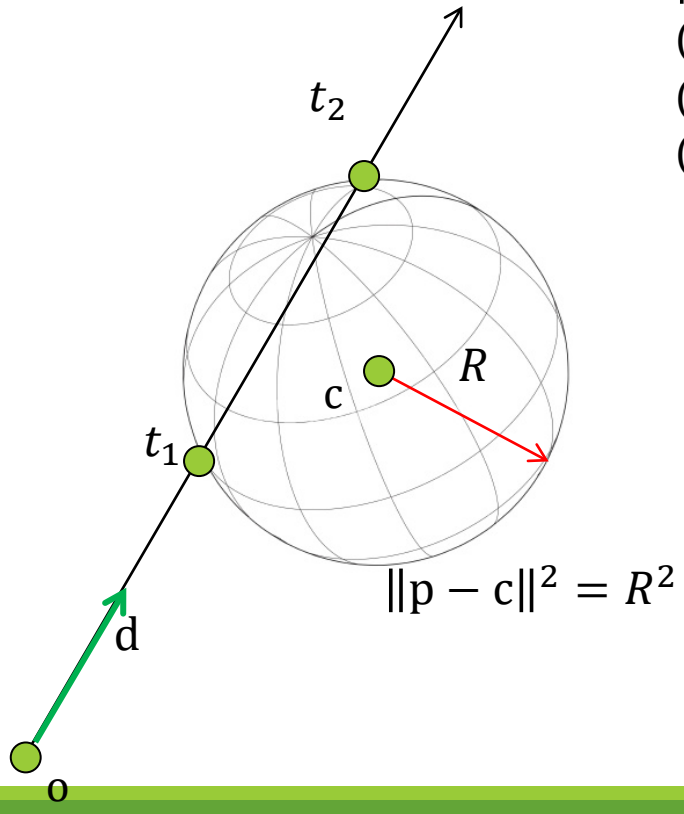
(b)

Ray = origin + direction



Ray-Sphere intersection

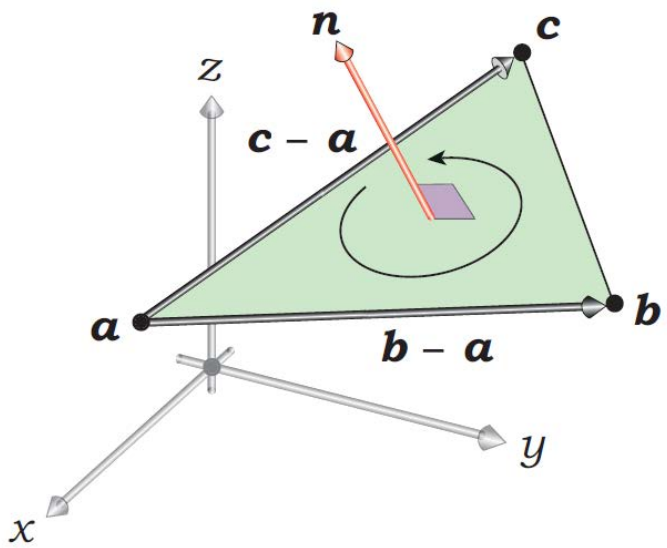
$$\mathrm{p}(t) = \mathrm{o} + t\mathrm{d}$$

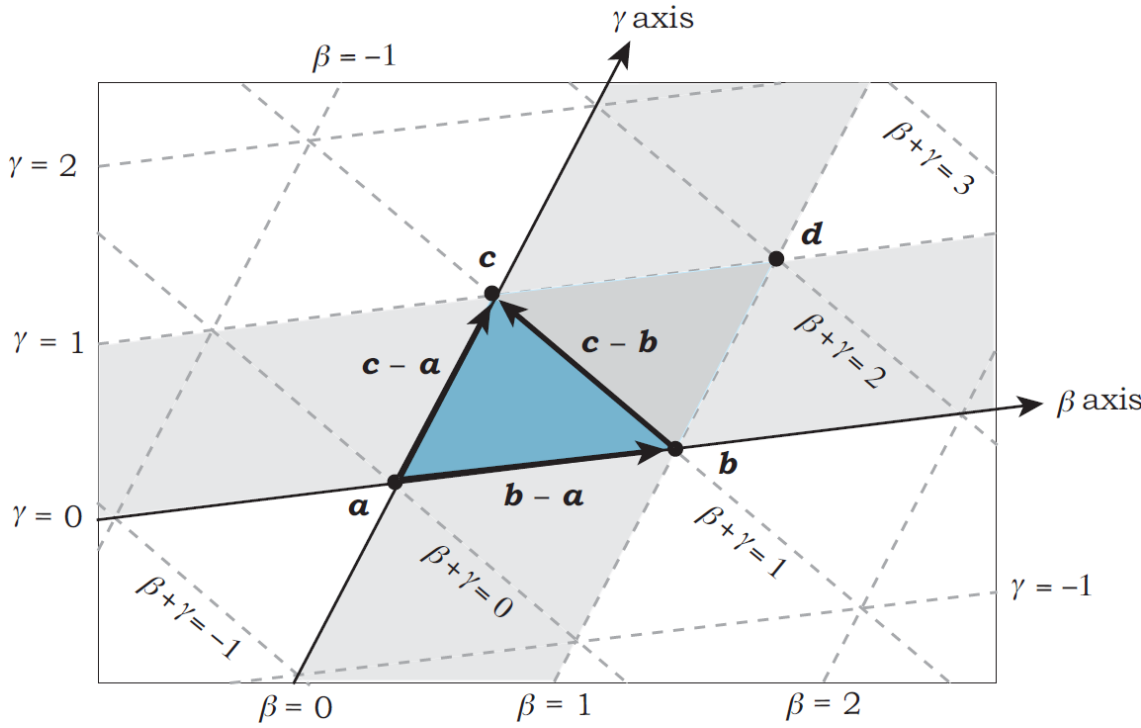


$$\|\mathrm{p} - \mathrm{c}\|^2 = R^2$$
$$(\mathrm{p} - \mathrm{c}) \cdot (\mathrm{p} - \mathrm{c}) - R^2 = 0$$
$$(\mathrm{o} + t\mathrm{d} - c) \cdot (\mathrm{o} + t\mathrm{d} - \mathrm{c}) - R^2 = 0$$
$$(\mathrm{d} \cdot \mathrm{d})t^2 + 2\mathrm{d} \cdot (\mathrm{o} - \mathrm{c})t + (\mathrm{o} - \mathrm{c}) \cdot (\mathrm{o} - \mathrm{c}) - R^2 = 0$$

- 0, 1 or 2 solutions
- Keep smallest t-value > 0
- Ray might start inside sphere!

Ray-Triangle intersection



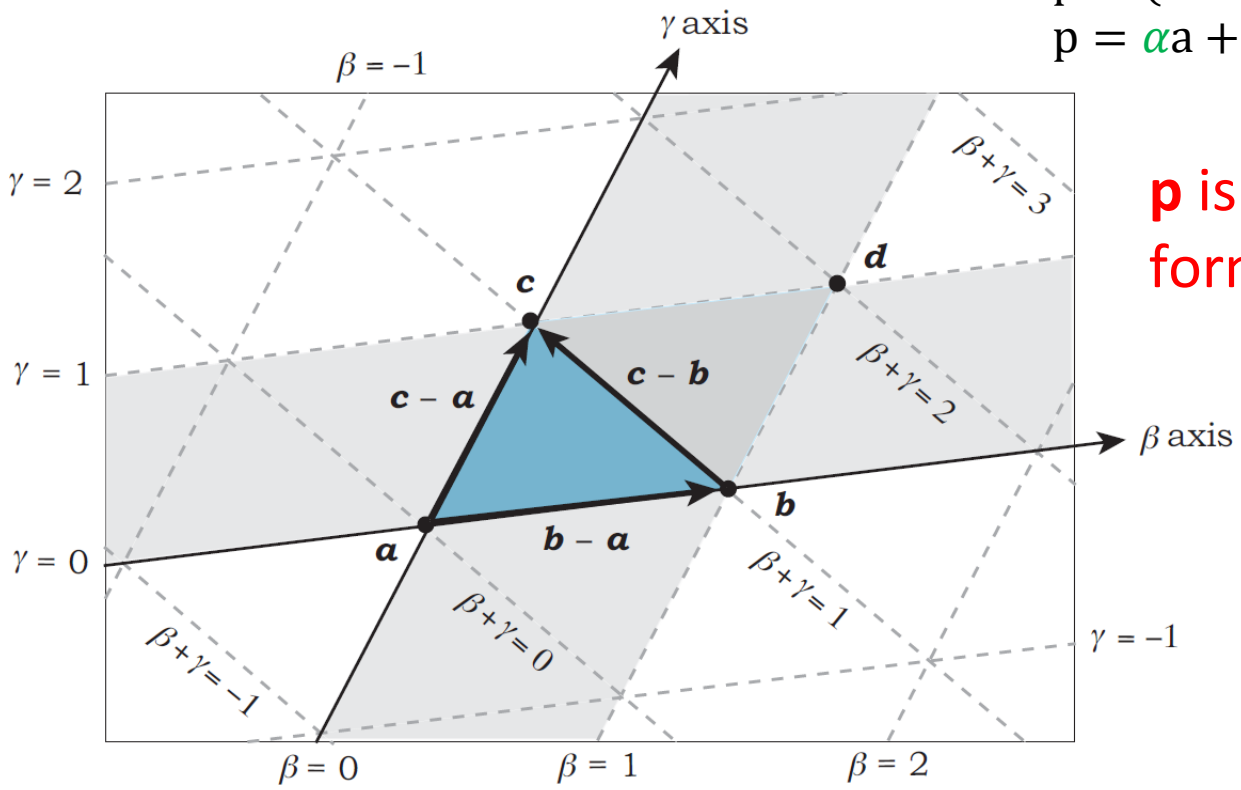


Given 3 points ***a***, ***b***, ***c***

Each point ***p*** in the plane (***a***, ***b***, ***c***): $\mathrm{p} = \mathrm{a} + \beta(\mathrm{b} - \mathrm{a}) + \gamma(\mathrm{c} - \mathrm{a})$

Ray-Triangle intersection

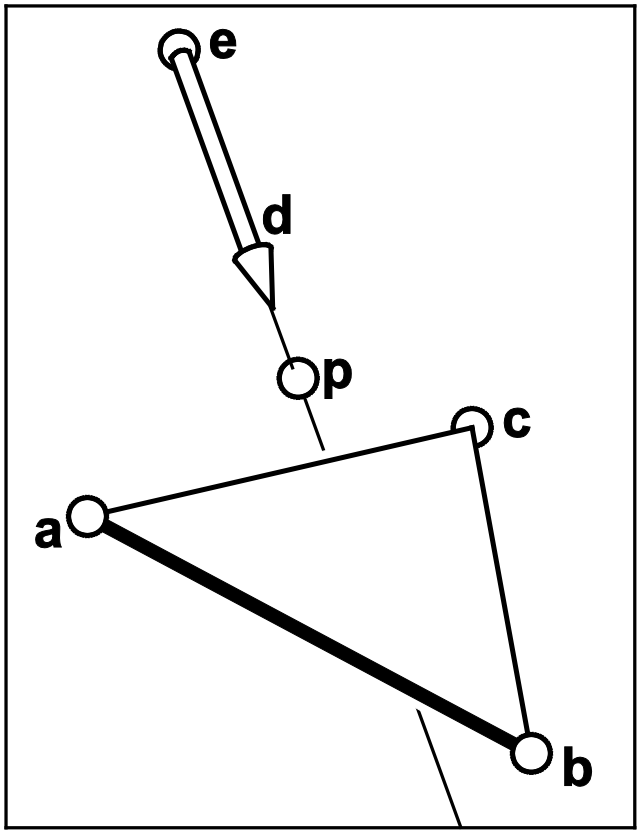
$$p = a + \beta(b - a) + \gamma(c - a)$$
$$p = (1 - \beta - \gamma)a + \beta b + \gamma c$$
$$p = \alpha a + \beta b + \gamma c \quad (\alpha + \beta + \gamma = 1)$$



p is inside the triangle formed by (**a**,**b**,**c**) if:

$$0 < \alpha < 1$$
$$0 < \beta < 1$$
$$0 < \gamma < 1$$

Ray-Triangle intersection



$$\mathrm{p}(t) = \mathrm{e} + t\mathrm{d} = \mathrm{a} + \beta(\mathrm{b}-\mathrm{a}) + \gamma(\mathrm{c}-\mathrm{a})$$

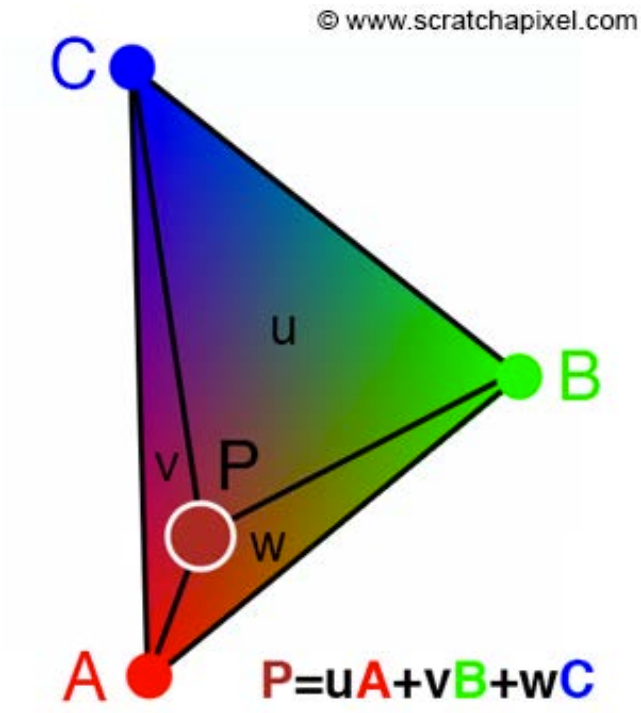
$$\begin{bmatrix} x_a - x_b & x_a - x_c & x_d \\ y_a - y_b & y_a - y_c & y_d \\ z_a - z_b & z_a - z_c & z_d \end{bmatrix} \begin{bmatrix} \beta \\ \gamma \\ t \end{bmatrix} = \begin{bmatrix} x_a - x_e \\ y_a - y_e \\ z_a - z_e \end{bmatrix}$$

Valid intersection if:

t > 0 and point p inside triangle

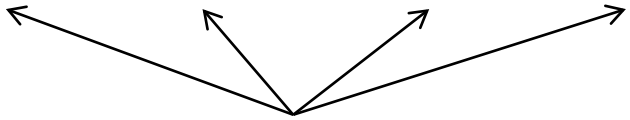
(Other) practical uses of barycentric coordinates

Attributes (e.g. color rgb) linked to triangle vertices can be interpolated



$$p = \alpha a + \beta b + \gamma c \quad (\alpha + \beta + \gamma = 1)$$

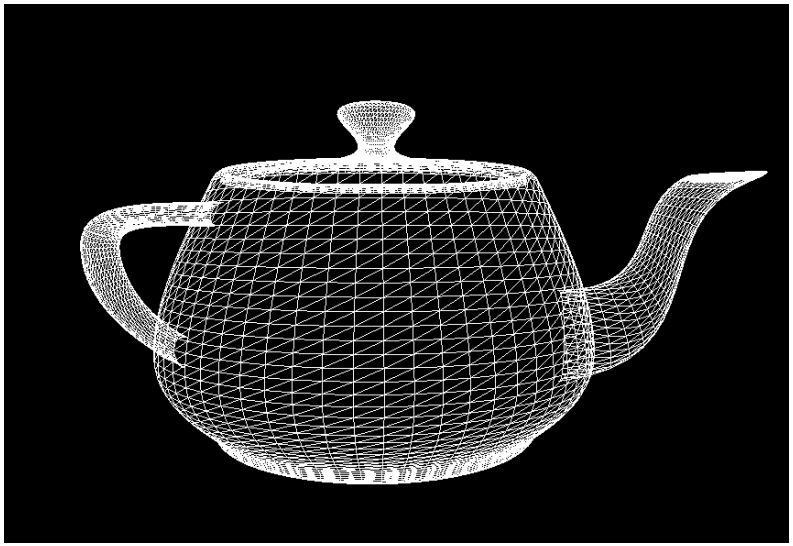
$$p_{attr} = \alpha a_{atrr} + \beta b_{atrr} + \gamma c_{atrr}$$



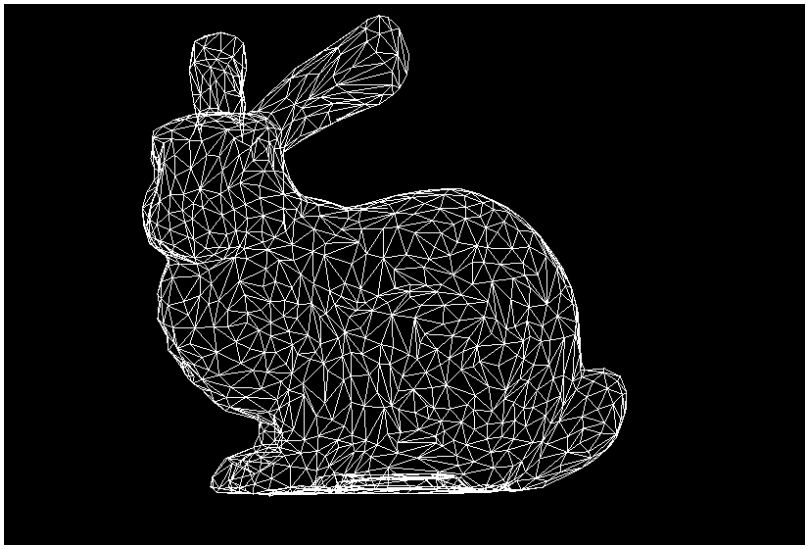
Could be a color, a normal vector, …

Polygon Mesh

Polygon mesh = collection of connected polygons (usually triangles)



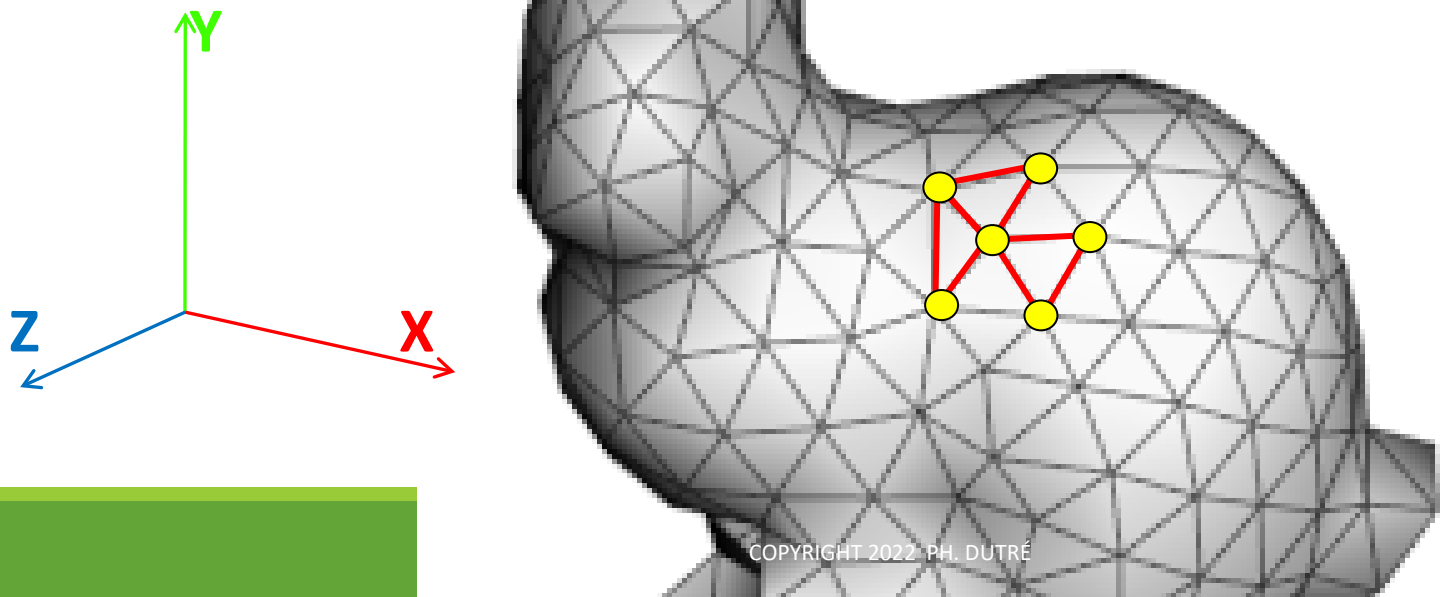
"Utah teapot"



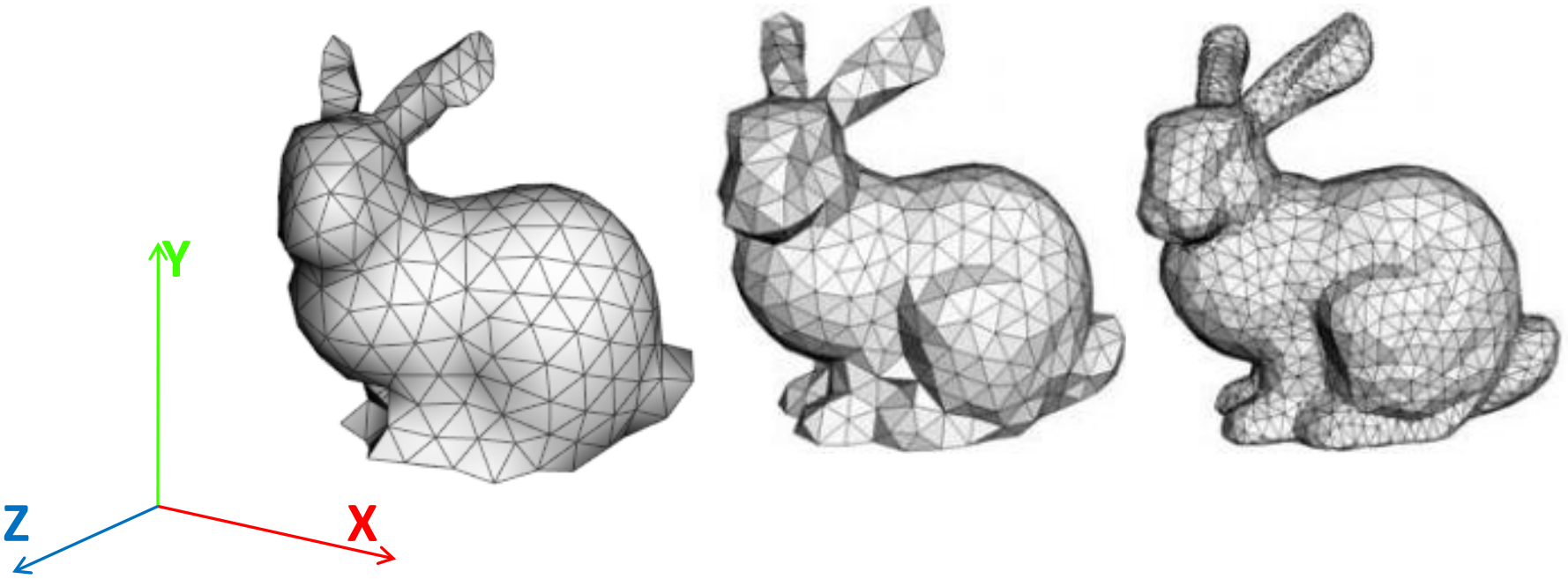
"Stanford Bunny"

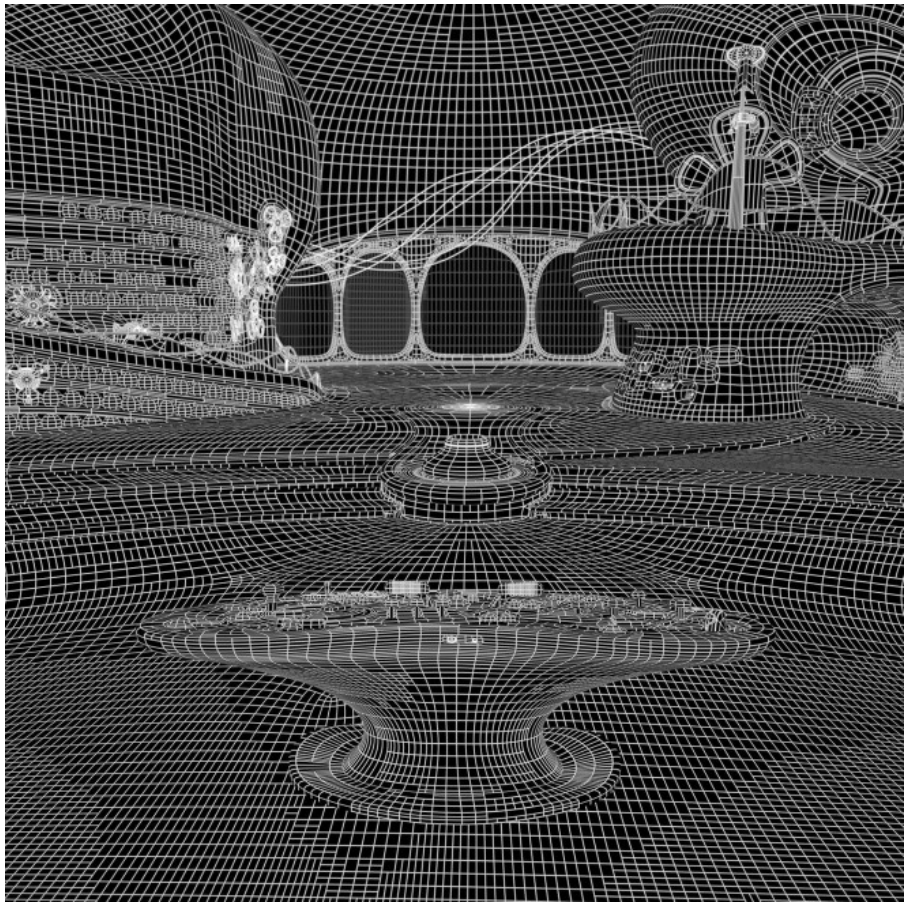
Polygon Mesh

Polygons share edges and vertices, and form a larger 3D surface

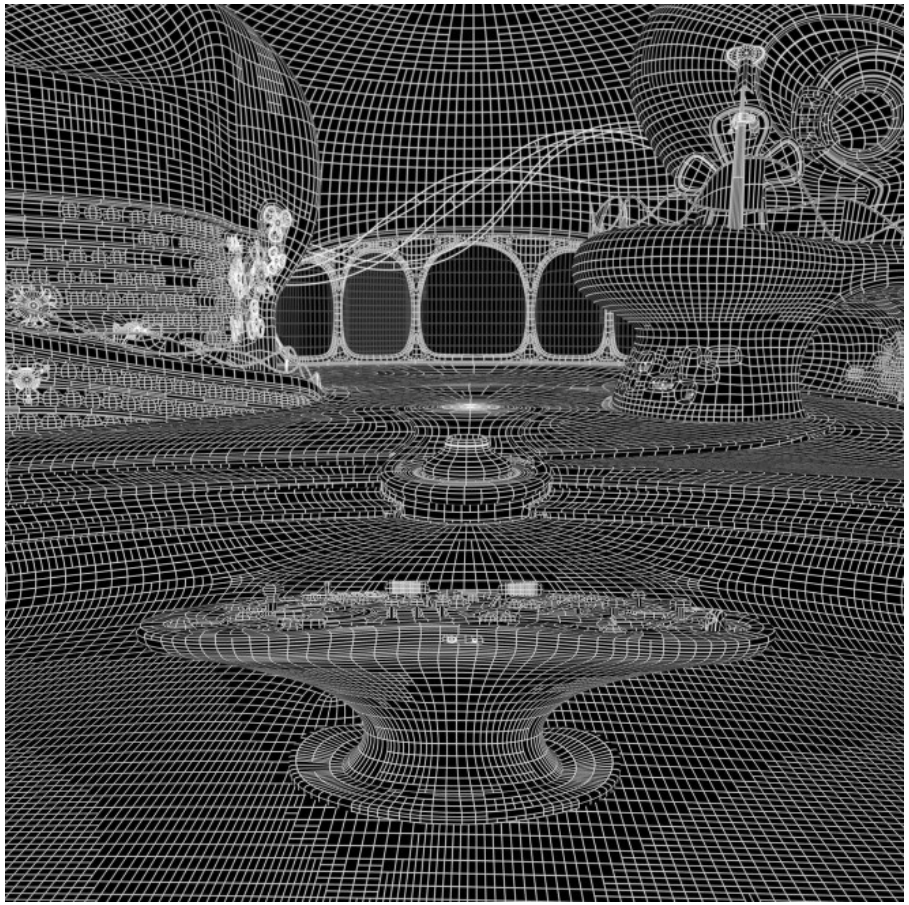


Polygon Mesh



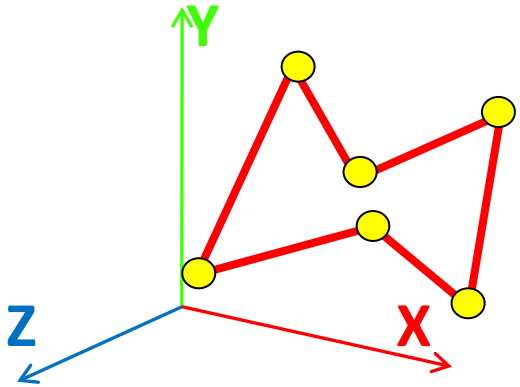


(Pixar, Inside Out, 2015)



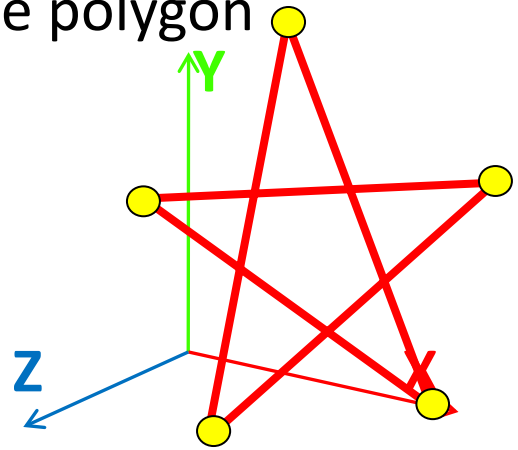
(Pixar, Inside Out, 2015)

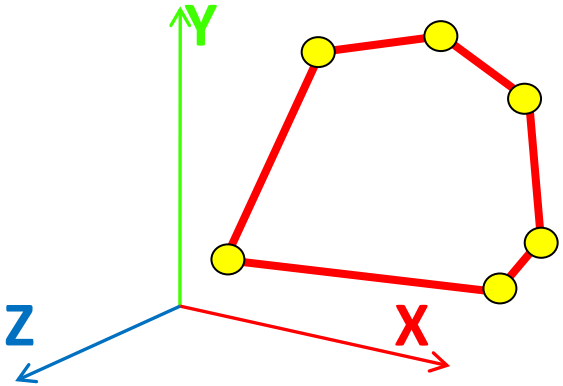
Polygon Mesh



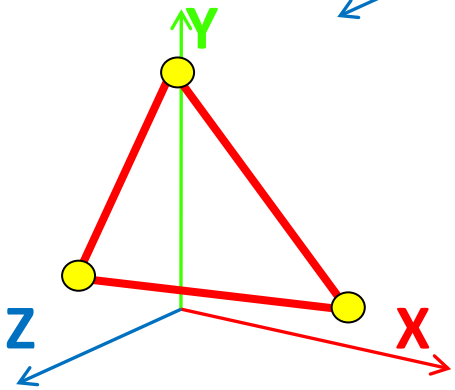
simple polygon
(non-crossing edges)

non-simple polygon





simple convex polygon
(internal angles < π)



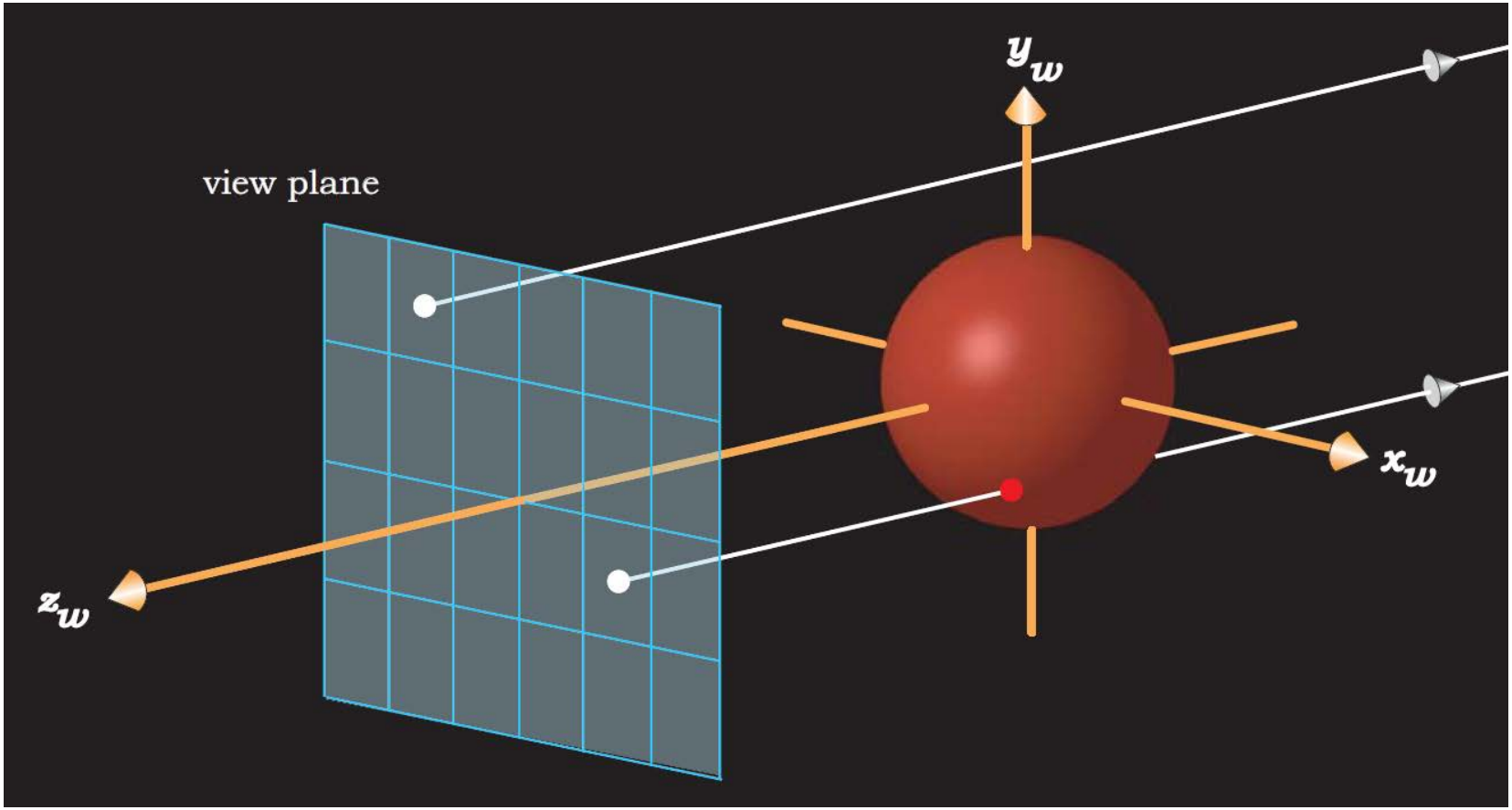
triangle

Polygon Meshes

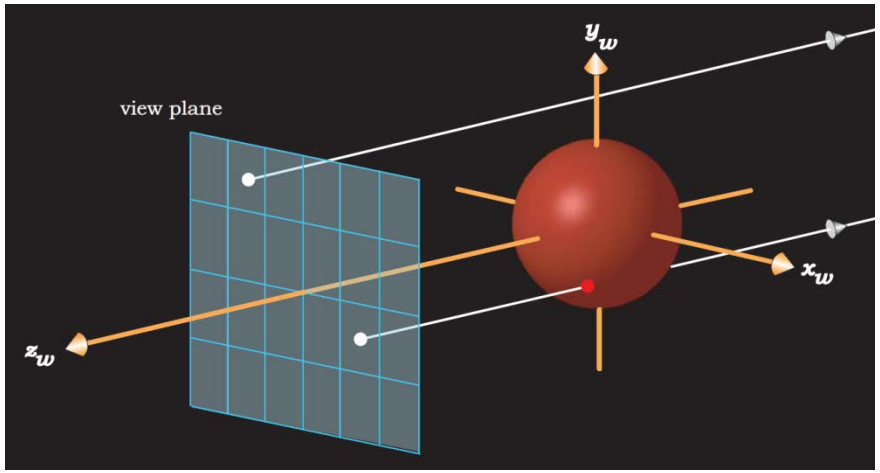
How to intersect a ray with a polygon mesh?

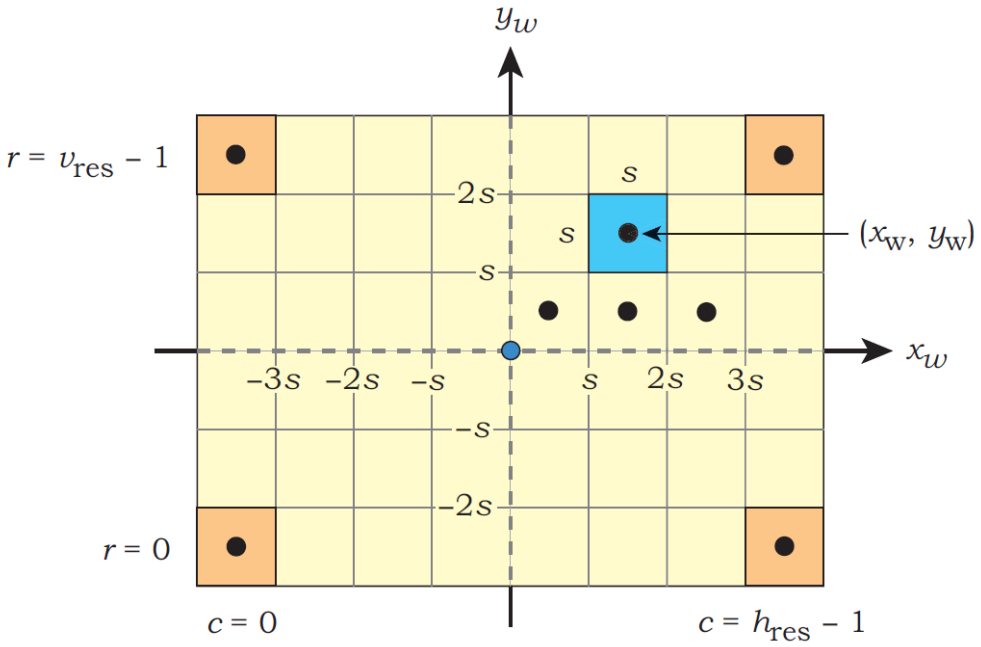
```
init nearest_intersection (distance = infinity)
for each polygon
   compute new_intersection t of ray with polygon
   if t < nearest and t > 0
       nearest_intersection = t
return nearest
```

Setting up the rays



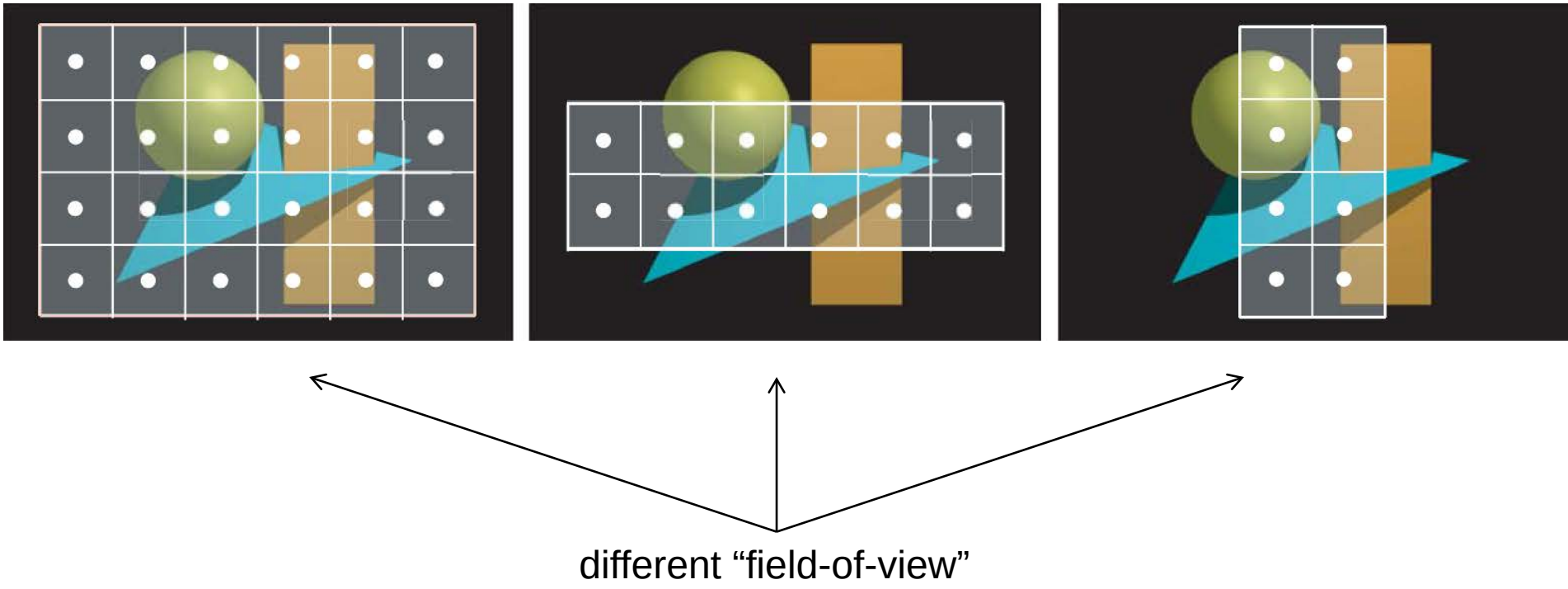
Setting up the rays



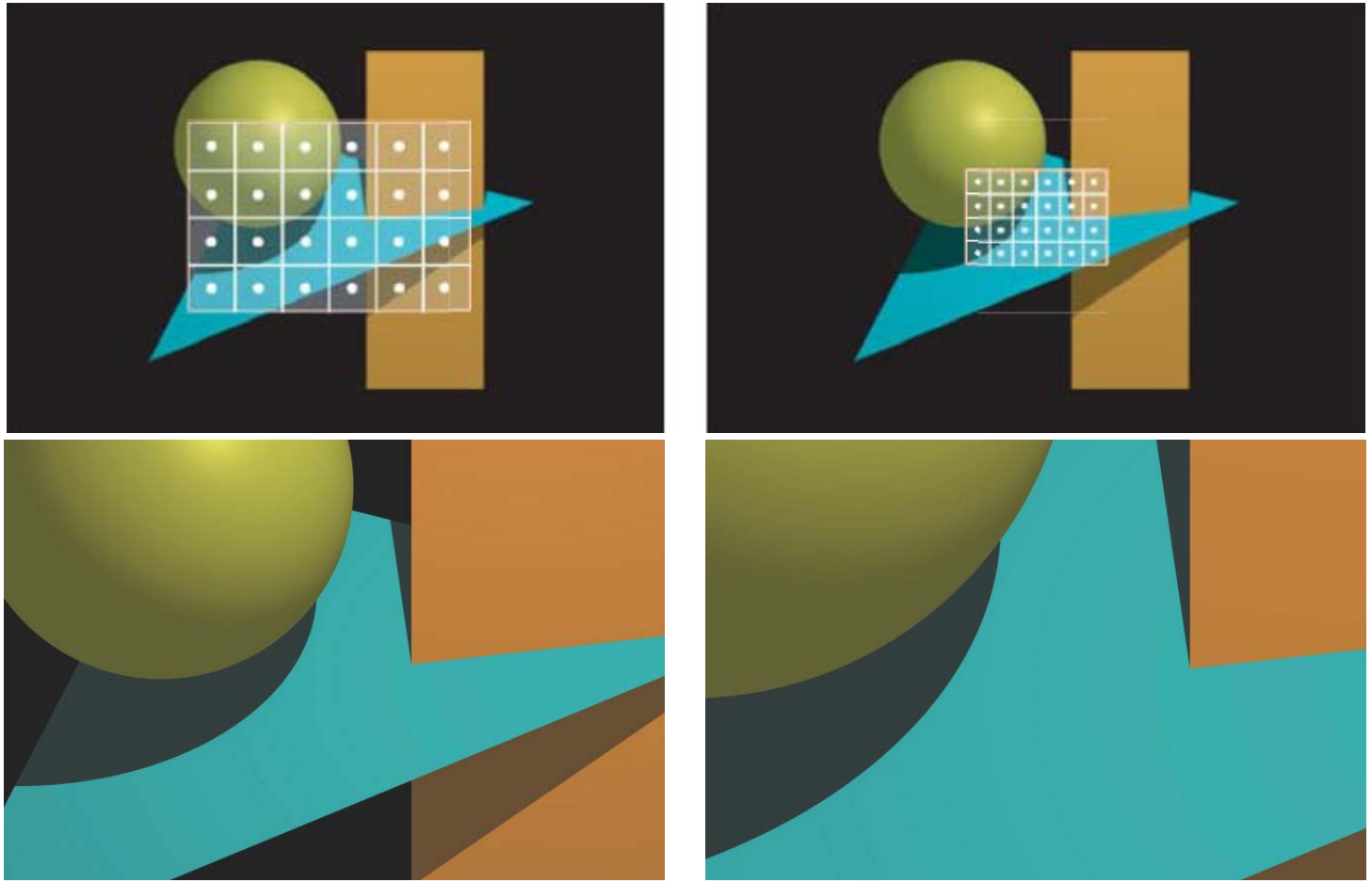


$$x_w = s(c - h_{res}/2 + 0.5)$$
$$y_w = s\left(r - \frac{v_{res}}{2} + 0.5\right)$$
$$d = (0,0,-1)$$

Setting up the rays



Setting up the rays



Setting up the rays