Computer Graphics
Introduction to 3D

EDAF80
Michael Doggett

Slides by Jacob Munkberg 2012-13

2022

Unity Enemies Demo

https://unity.com/demos/enemies

THE RISE OF GAMING REVENUE VISUALIZED

(Inflation Adjusted, 2020)

From the arcade boom to the rise of home consoles and PC gaming, the gaming industry quickly grew to rival and outpace film and television. Once mobile gaming took off, that wave of revenue became an ever-growing tsunami.

Here's how the gaming market's revenue wave has grown over time.

- Cloud
- VR
- Handheld
- Mobile
- PC
- Arcade
- Console

1972
Pong becomes the first commercially successful arcade game

1978
Space Invaders reinvigorates the arcade market

1980
Pac-Man generates more than $1 billion in arcade sales in the U.S. in one year

1983-1985
The video game crash of 1983 hits North America, caused by market oversaturation and high-budget poor-quality ports like *E.T. the Extra-Terrestrial*

1985
Nintendo Entertainment System (NES) and *Super Mario Bros.*

1988
Sega releases the Mega Drive in Japan, released in the U.S. as the Genesis

1989
Nintendo releases the Game Boy handheld system, alongside a port of *Tetris*

1991
Nintendo releases the Super NES (SNES)

1993
Doom popularizes the first-person shooter genre

1994
Sony's PlayStation and Sega's Saturn consoles use CDs instead of cartridges

1997
Nokia releases cell phones with the game *Snake*, the first popular mobile game

2000
The Sims for PC popularizes life simulation games

2001
Microsoft's Xbox console is the first with an integrated ethernet port

2002
Microsoft launches Xbox Live, a subscription service for online multiplayer gaming popularized through its *Halo* series

2004
World of Warcraft launches on the PC, a subscription-based online role-playing game that grows to more than 14 million subscribers

2006
Nintendo's Wii console is targeted at a wide demographic of families and non-gamers

2006
Sony's PlayStation 3 is the first console to use Blu-rays

2007
Apple's iPhone smartphone is released

2009
Angry Birds for mobile becomes the most downloaded freemium game and launches a media franchise

2009
League of Legends, a freemium multiplayer online battle arena, becomes the world's largest esport

2010
Minecraft is released in a public beta

2011
Twitch, a video streaming service primarily for games, is launched

2012
Candy Crush Saga for mobile popularizes limited plays and in-app monetization

2013
Grand Theft Auto V earns $800 million in its first day

2016
Pokémon Go, an Augmented Reality mobile game, reaches 500 million downloads in one year

2017
Nintendo releases the Switch, a hybrid home/portable console

2018
Fortnite, a free-to-play battle royale for PC and consoles, is ported to mobile

2020
Epic Games enters legal battle with Apple, *Fortnite* is removed from App Store

2020
Microsoft, NVidia, Google, and Amazon launch cloud gaming services

2020
PlayStation 5, Xbox Series X and Series S are launched

Arcade $39B

Console $20B

PC $40B

Mobile $85B

2020 total $165B

Cloud

VR

Handheld

Mobile

PC

SOURCE Pelham Smithers

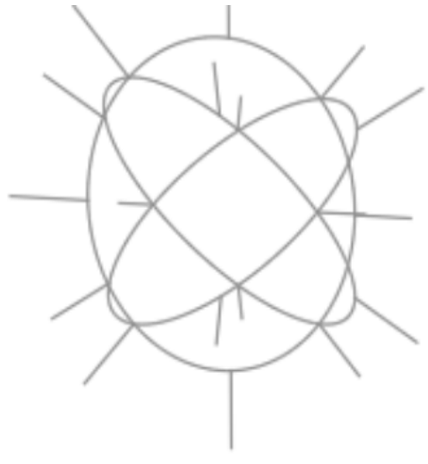
COLLABORATORS RESEARCH + WRITING Omri Wallach | DESIGN + ART DIRECTION Clayton Wadsworth

VISUAL CAPITALIST

/visualcapitalist @visualcap visualcapitalist.

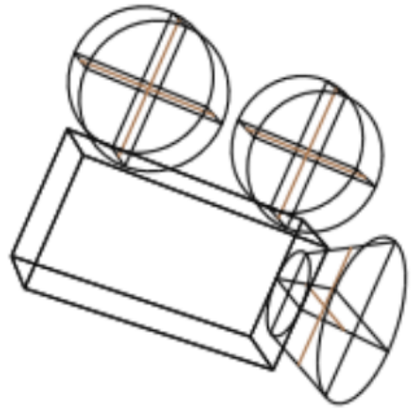
https://www.visualcapitalist.com/50-years-gaming-history-revenue-stream/

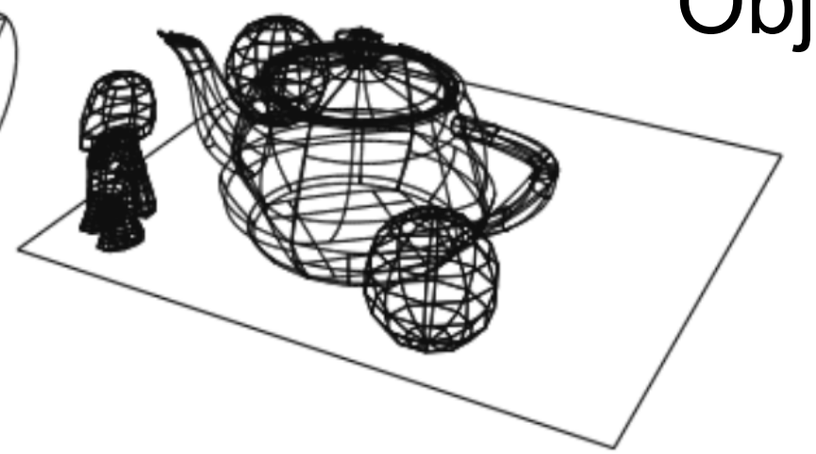
Create Virtual Scenes



Light source

Camera





Objects

Rendered Image

Course Goals

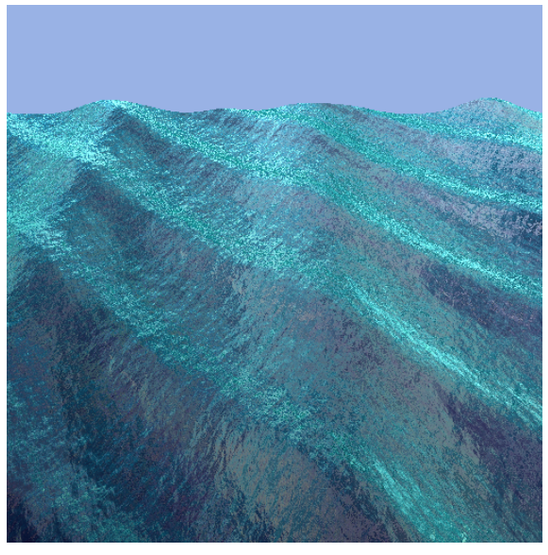
- Introduction to Computer Graphics
- Create and **position** objects in 3D
- Create materials - write **shaders**
- Visualize 3D scenes - **render** images
- Introduction to **OpenGL**
  - Create interactive graphics

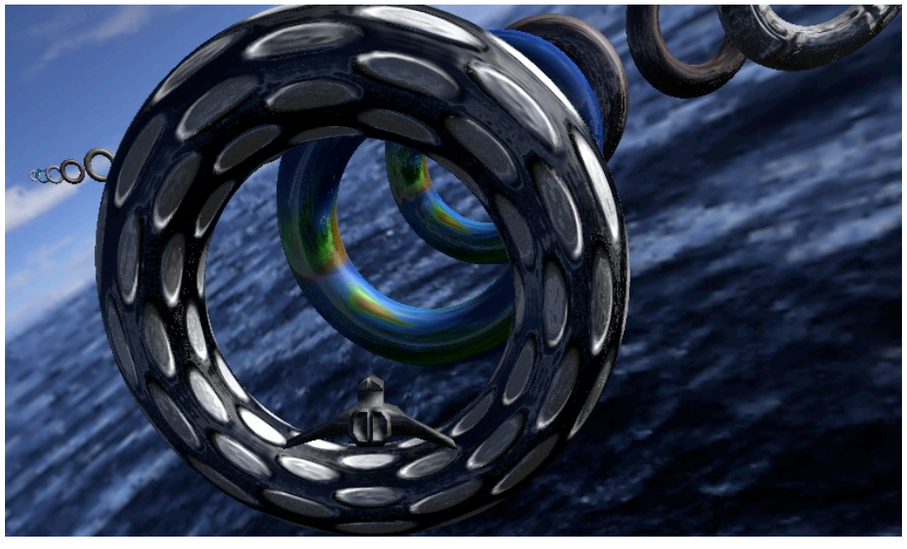
Organization

- Lectures : Theory and concepts
- Seminars
  - Applied theory & hands-on examples
- Assignments
  - Computer Graphics in practice, C++, OpenGL & GLSL
- Examination
  - All five assignments approved
  - Written exam

Assignments

- Five mandatory programming assignments
- Done in pairs - book a lab!
  - https://sam.cs.lth.se/LabsSelectSession?occasionId=777
  - Both students must be present at marking, and answer questions (including lab 5)
- Seminars: Wednesdays 15-17 in E:1406





C++ programming language

- We use the programming language C++
  - C++ is low-level, performance focused language
    - Commonly used in Graphics, Game programming
- This is NOT a programming course.
- If you are unfamiliar with C++, you will need to learn it as you go through the course.

Website

- cs.lth.se/edaf80
- Lectures will be posted online
- Online discussion using discord
- Booking system for assignment approval sessions
  - Sign up for 1 of 4 lab sessions
- Code & assignments - GitHub
- Links

Course Material

- Literature
  - Lectures & Seminars
  - No Textbook
  - Many very good graphics resources online
    - Edward Angel, Dave Shreiner, **Interactive Computer Graphics: A Top-Down Approach with Shader-Based OpenGL**, Pearson Education, 6th edition (old course textbook)
    - Graphics Codex
    - Physically Based Rendering (PBRT)
    - See webpage for other references
- Prerequisites
  - Programming (first course) & Linear Algebra

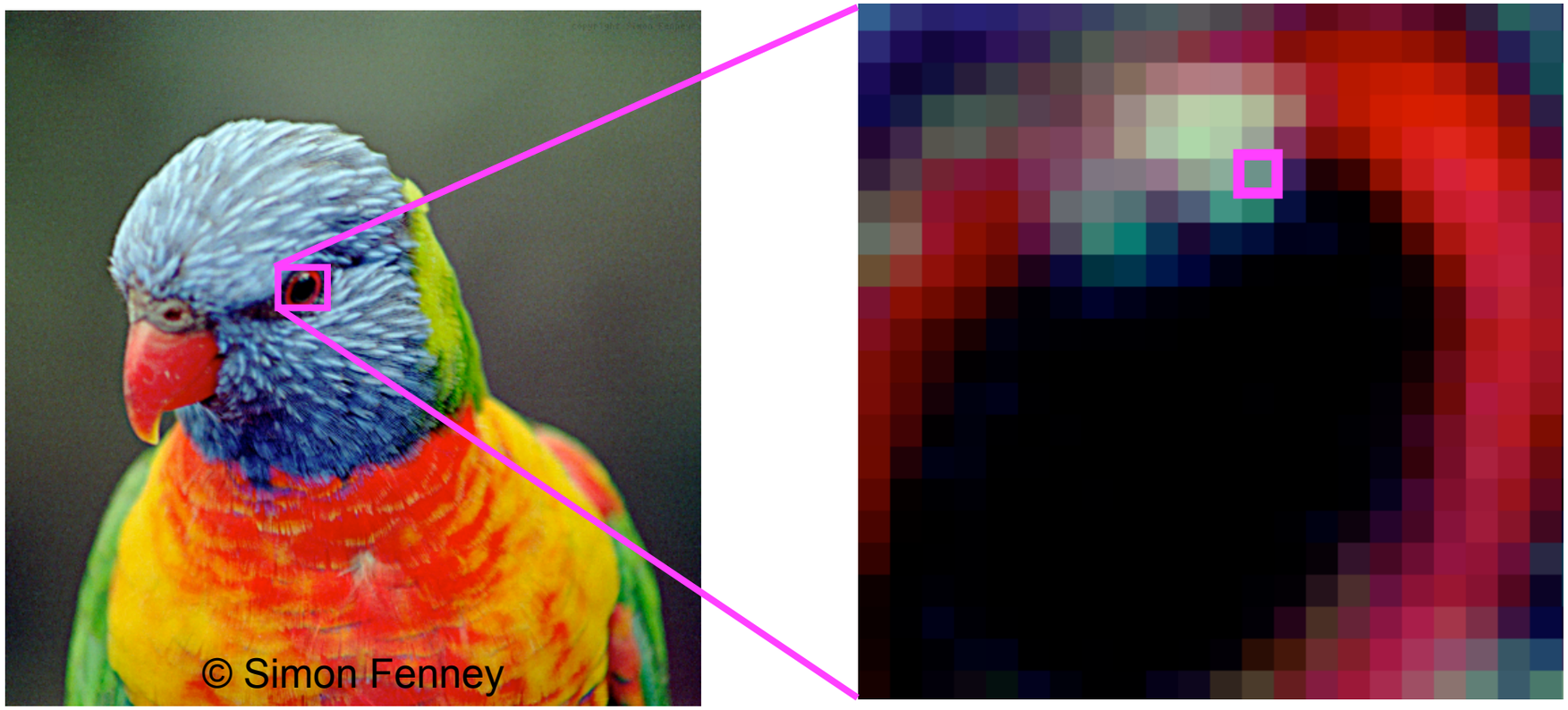
Staff

- Michael Doggett - Lectures, Seminars, Exam
    - Visiting Professor at Facebook Reality Labs, Seattle, Docent, GPU architect at ATI/AMD, Post-doc in Germany, PhD in Australia
- TAs - Rikard Olajos, Gareth Callanan, Michail Boulasikis
    - PhD students in Computer Graphics and Embedded Systems

Definitions

Pixel

- Pixel - Picture element
- Our task - compute color of each pixel



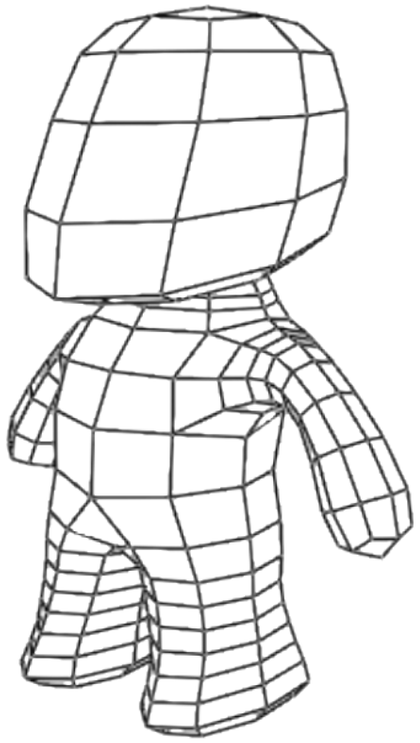
512x512 pixels

22x22 pixels

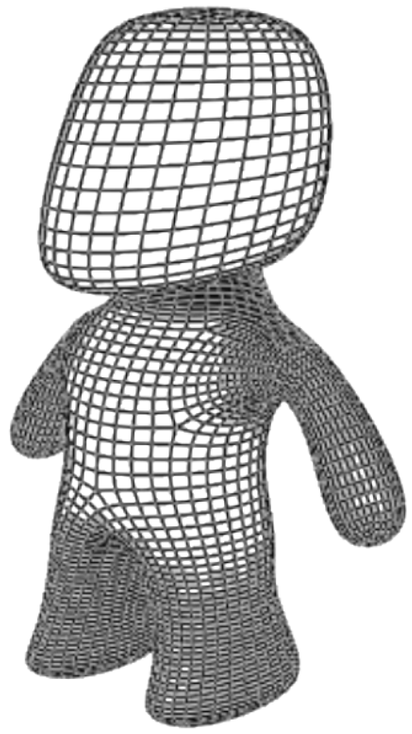
Number of Pixels

- Smartphone (1920 x 1080)
- iPad retina (2048 x1536)
- MacBook Pro retina (2880 x1800)
- 4K TV “Ultra HD” (3840 x 2160)
- Apple iMac (5120 x 2880)
- Hasselblad H4D-60: 60 Mpixel

Image Formation



Mesh



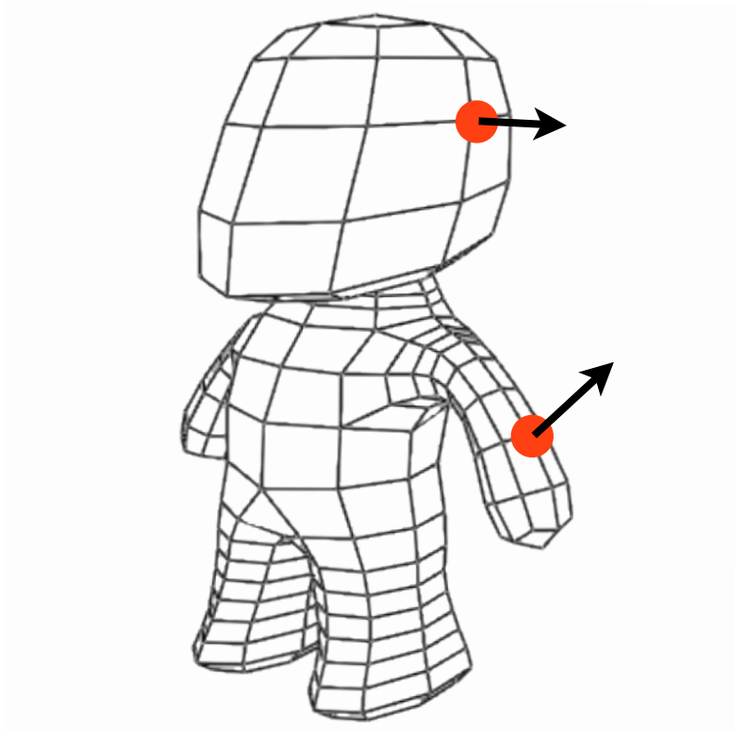
Tessellated Mesh

Lit & Shaded Mesh

We need ways to represent geometry and move objects in three-dimensional coordinate systems

Vertex

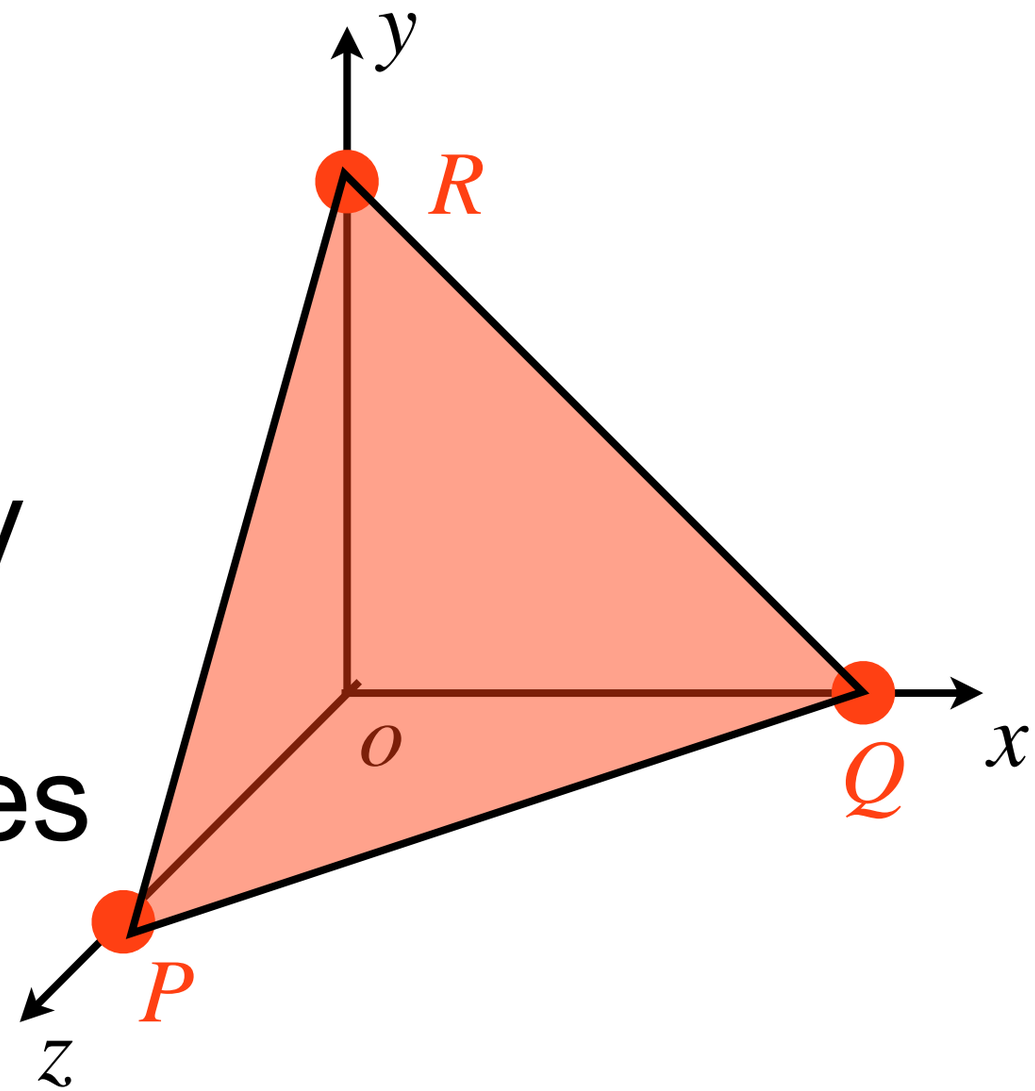
- A set of attributes describing a point in space



```
struct Vertex
{
    float x,y,z;        // pos
    float nx, ny, nz; // normal
    float r,g,b;        // color
};
```

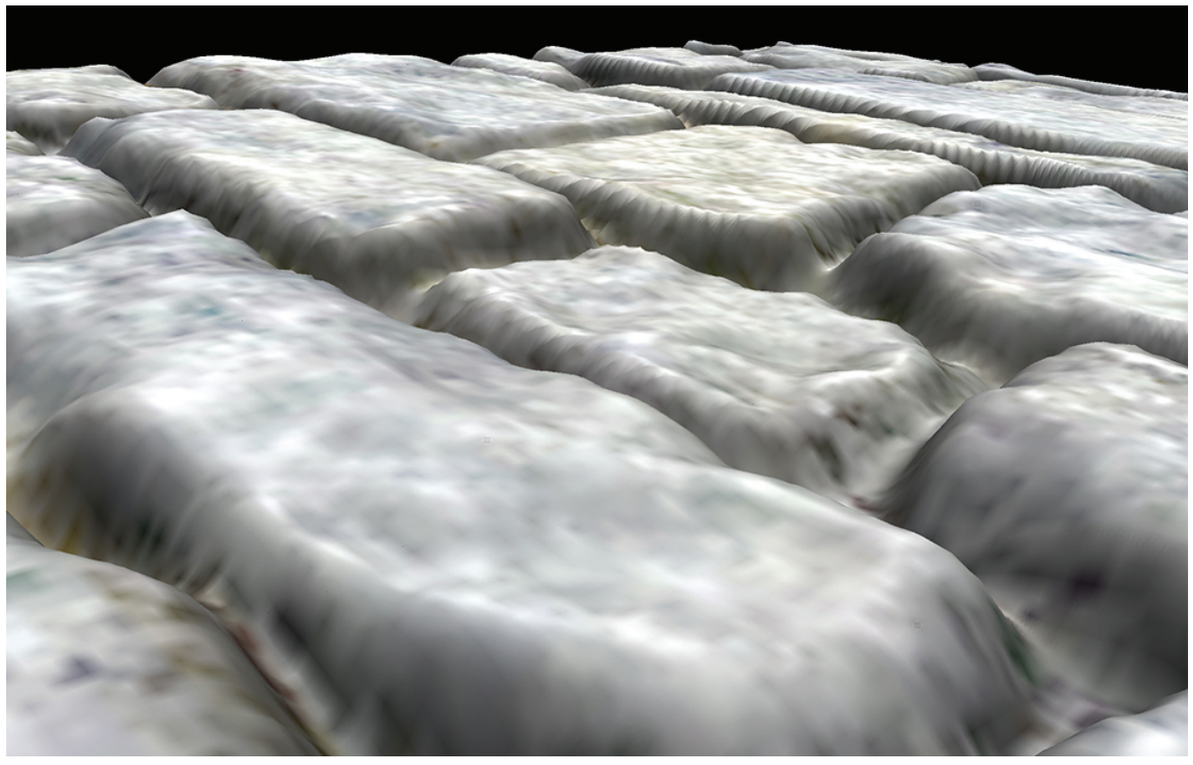
Triangle in 3D

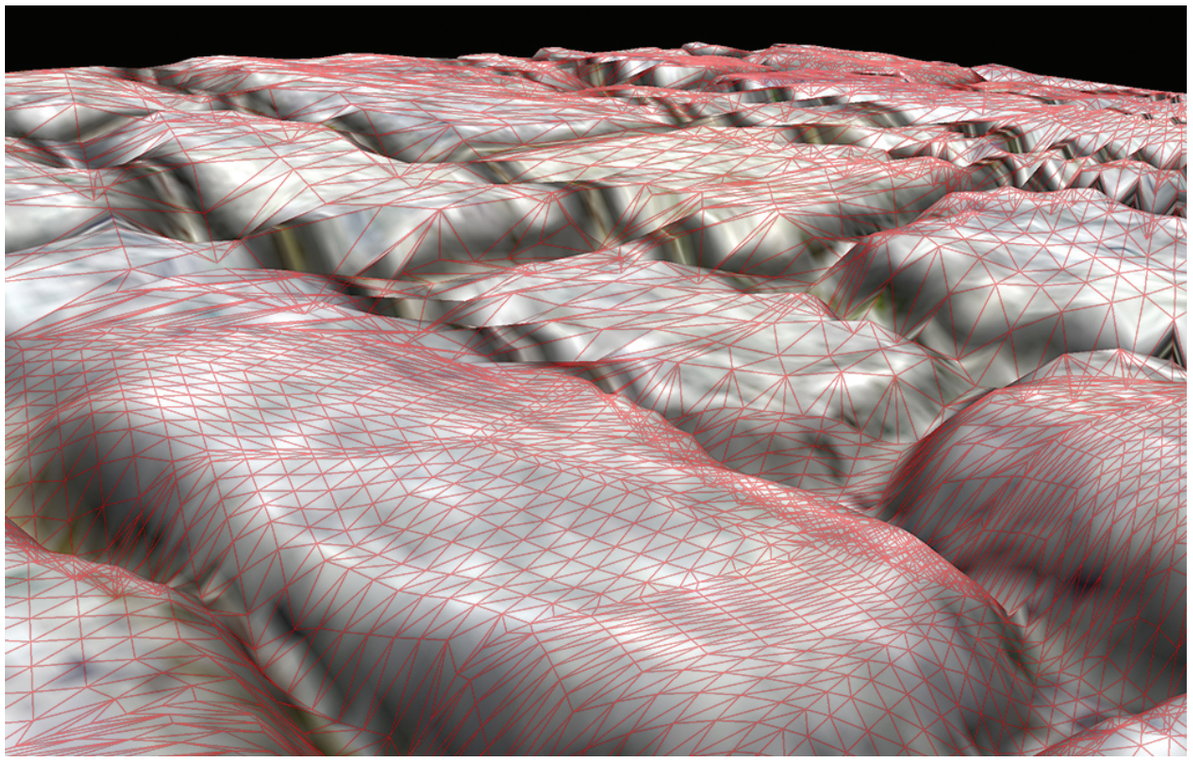
- Defined by three connected vertices
  - Here: specified in the Cartesian system
- Models are typically built from a large collection of triangles



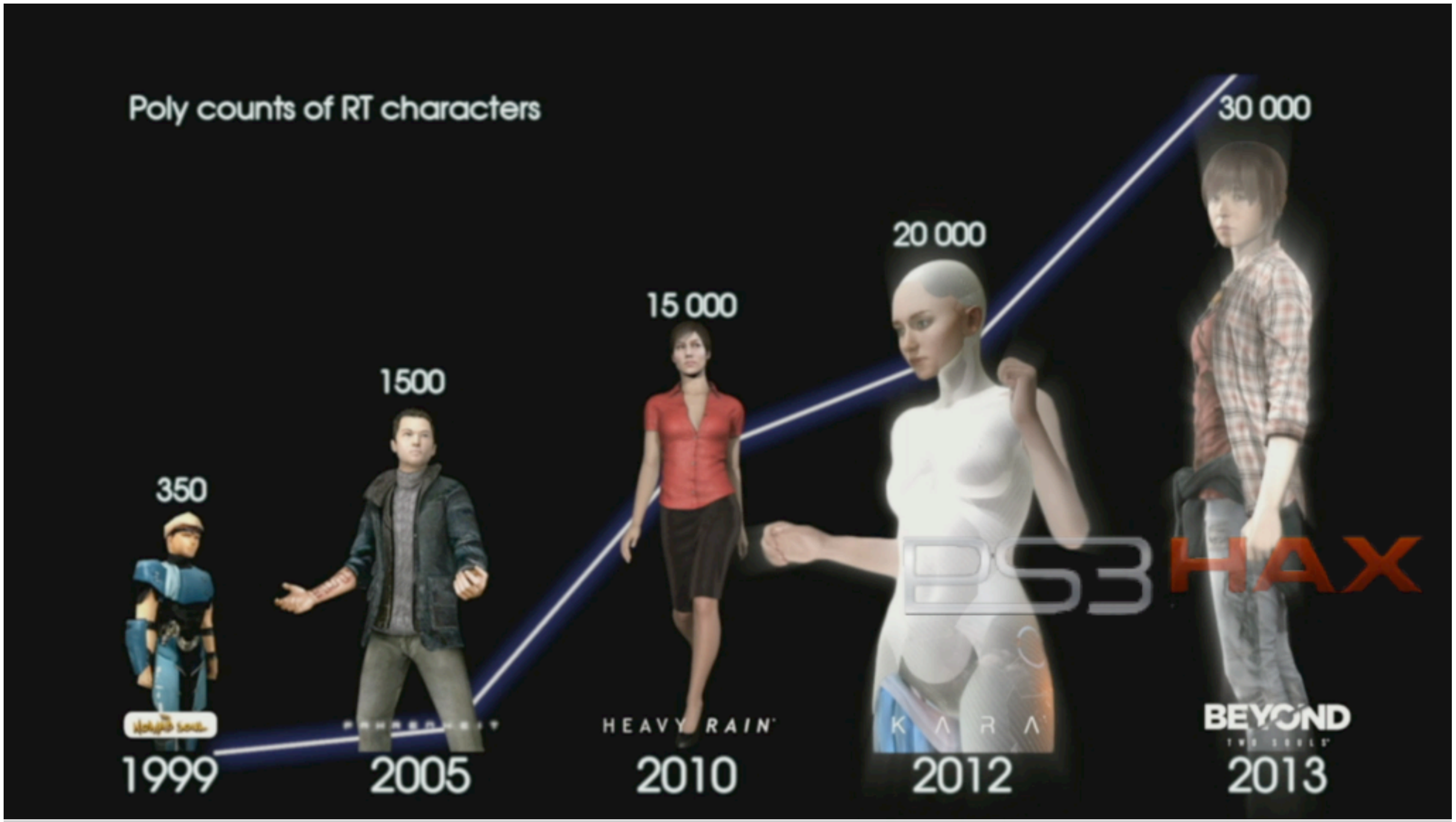
Geometry Meshes

- Model complex shapes from triangles





Poly counts of Real-Time characters



Source: Sony

Materials

- Determine the appearance of objects
  - How objects interact with light
- Specified as “small” programs called **shaders**

Simple Pixel Shader

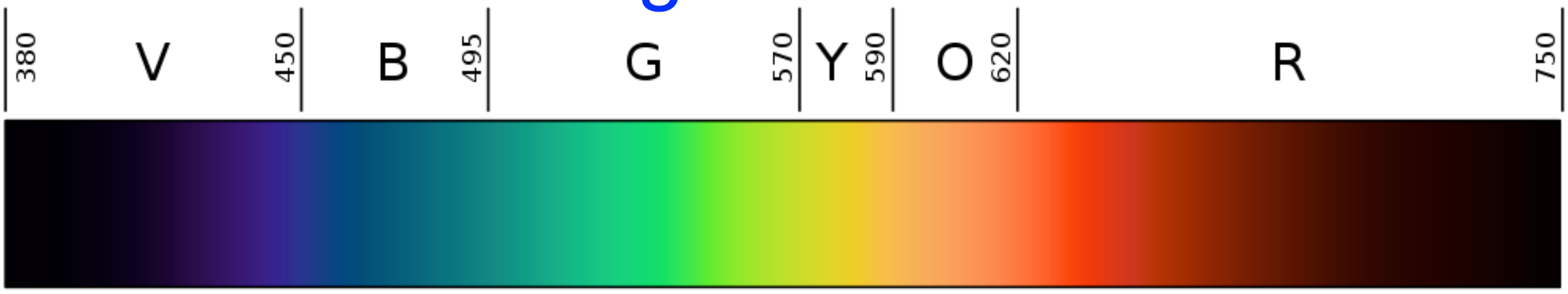
- Set pixel color to red

```
out vec4 fColor;

void main()
{
	fColor = vec4(1,0,0,1);
}
```

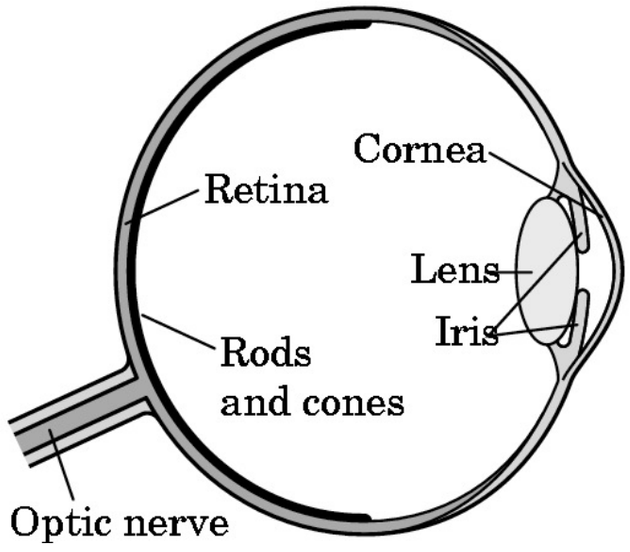
Light

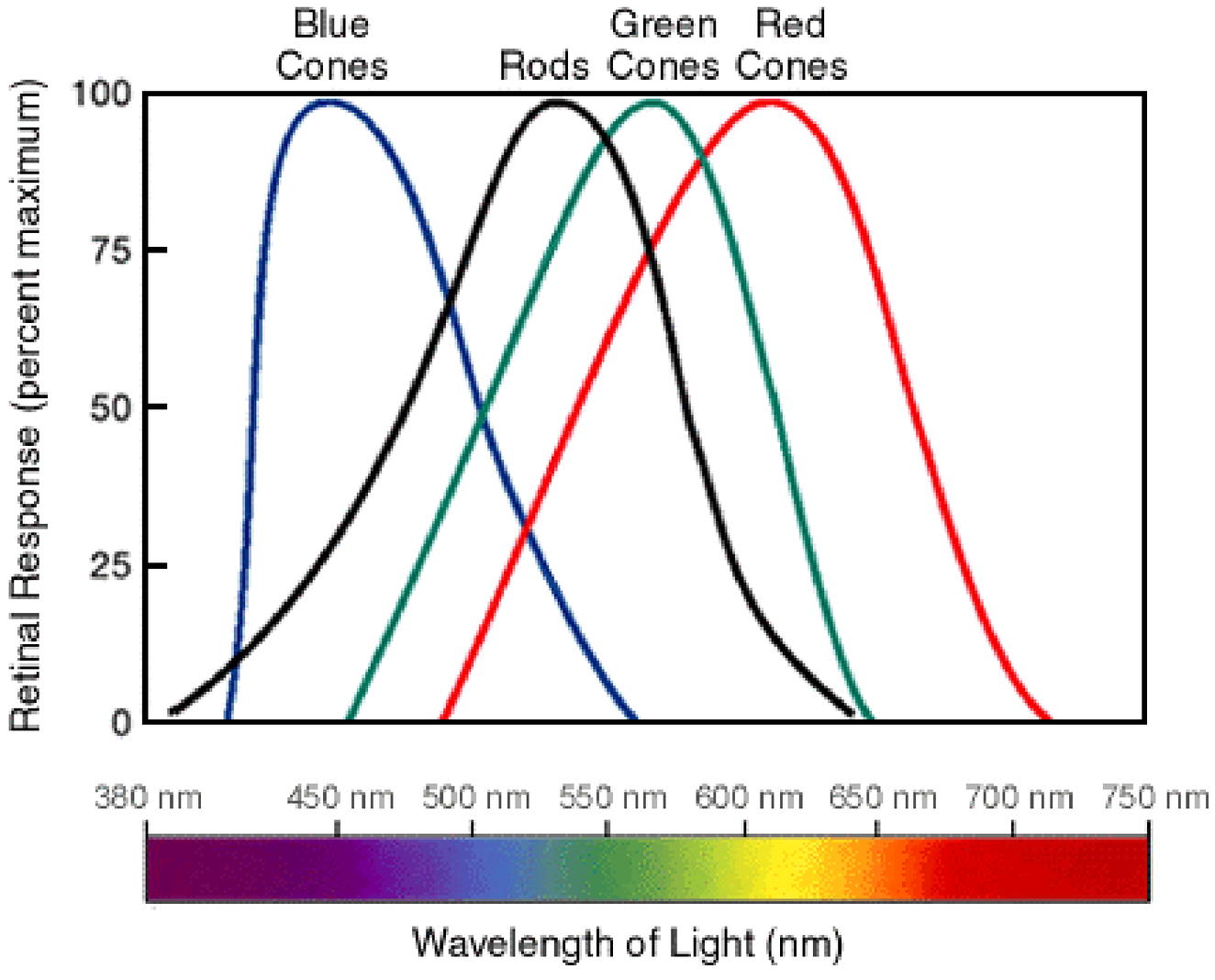
- Light is the part of the electromagnetic spectrum that causes a reaction in our visual systems
- Wavelengths in 380-750 nanometers
- Long wavelengths appear as reds and short wavelengths as blues



Color Perception

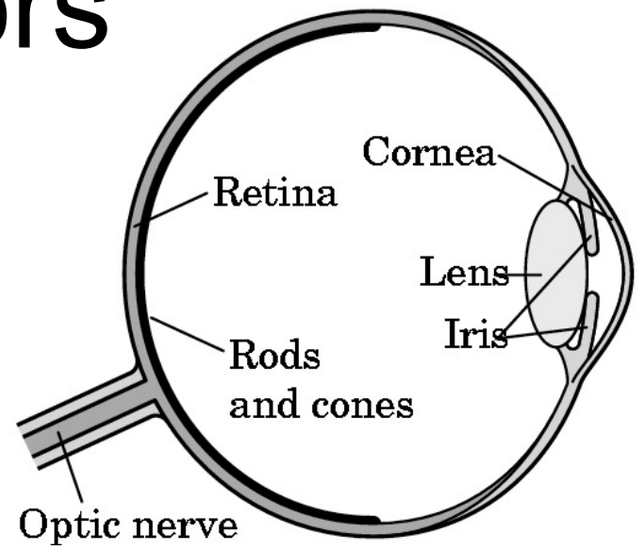
- Color stimulates cones in the retina
- Three different kinds of cones





Human Visual System (HVS)

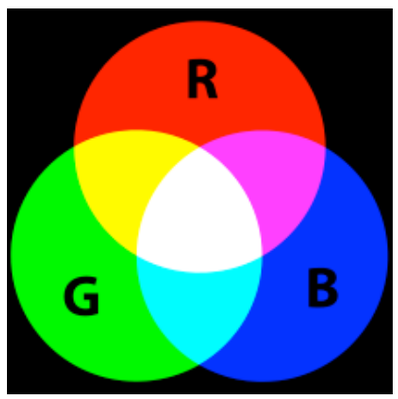
- HVS has two types of sensors
  - **Rods**: monochromatic, night vision
  - **Cones**: Color sensitive, three types of cones
    - Only three values (the *tristimulus* values) are sent to the brain



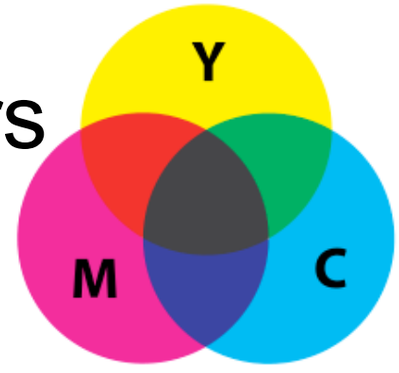
- Need only match these three values
- Need only three primary colors: RGB

Color

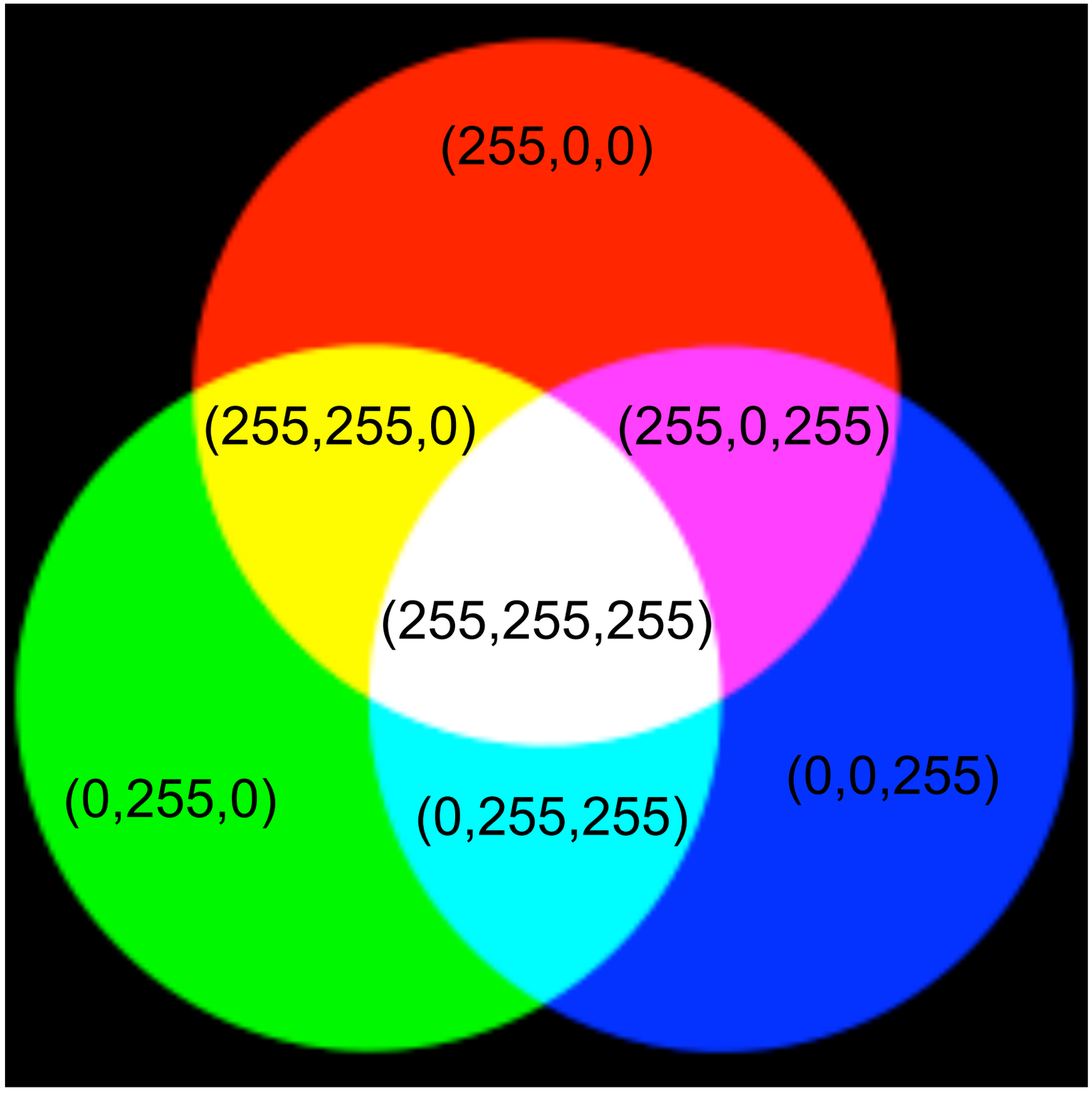
- Additive color
  - Form a color by adding amounts of three primaries: Red (R), Green (G), Blue (B)
  - Often stored using 8 bits per primary which gives $(2^8)^3$ = 16.8M unique colors



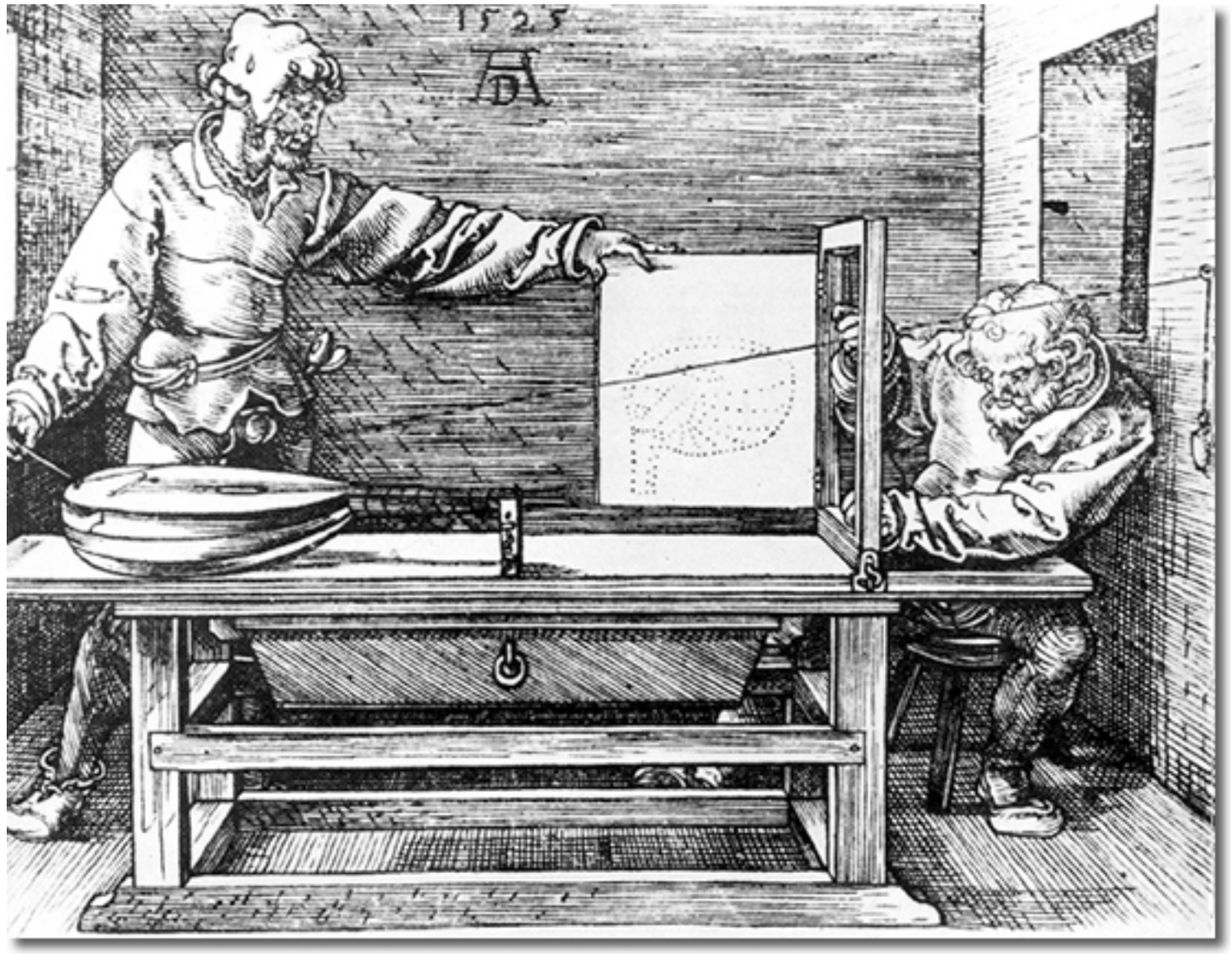
- Subtractive color
  - Form a color by filtering white light with Cyan (C), Magenta (M), and Yellow (Y) filters
  - Light-material interactions, printing



RGB Color

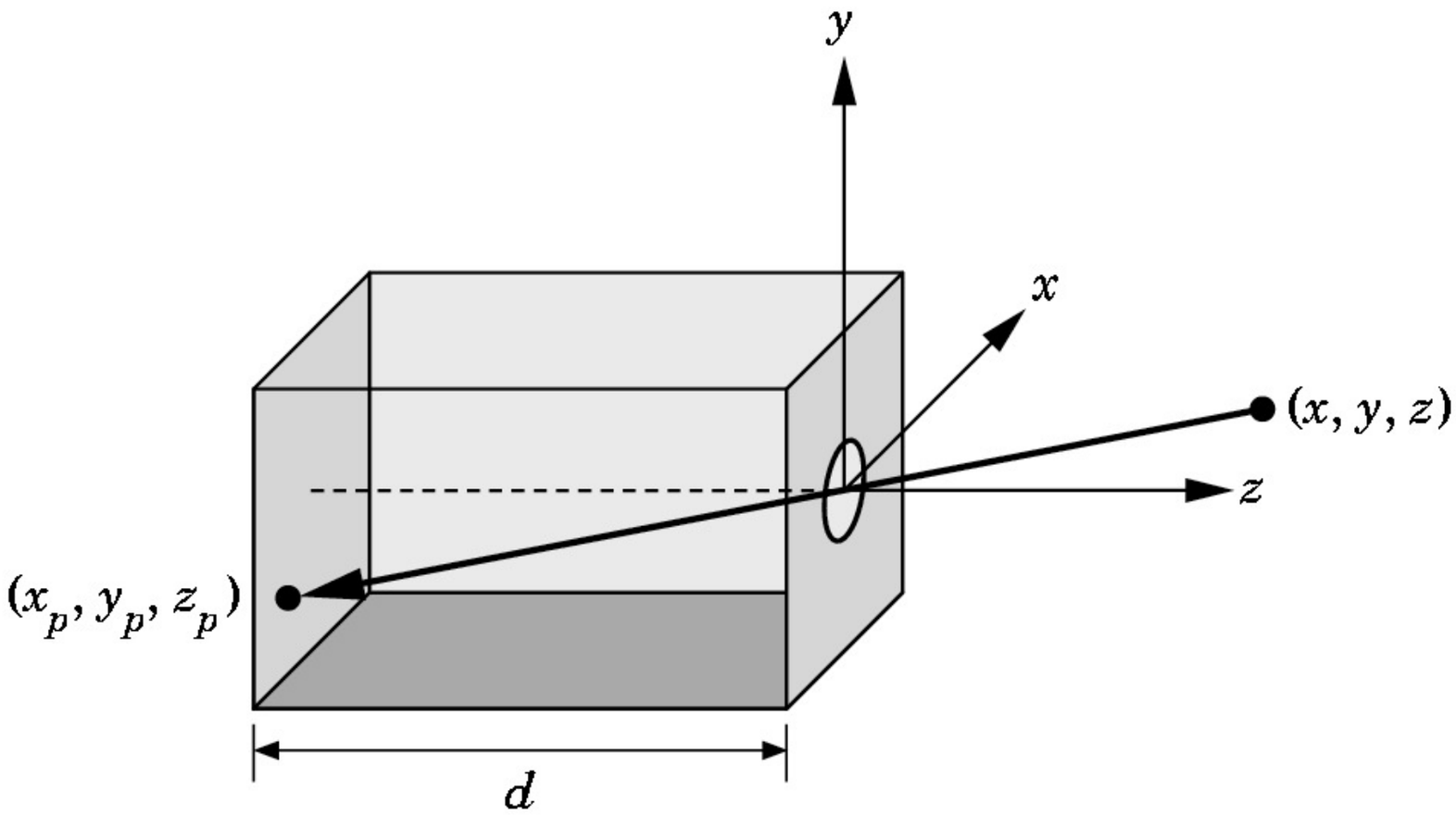


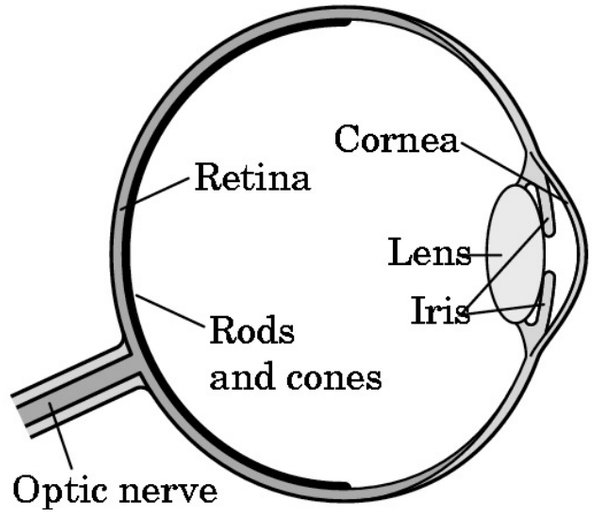
Rendering a 2D image



Perspective drawing in the Renaissance: “Man drawing a lute” by Albrecht Dürer, 1525

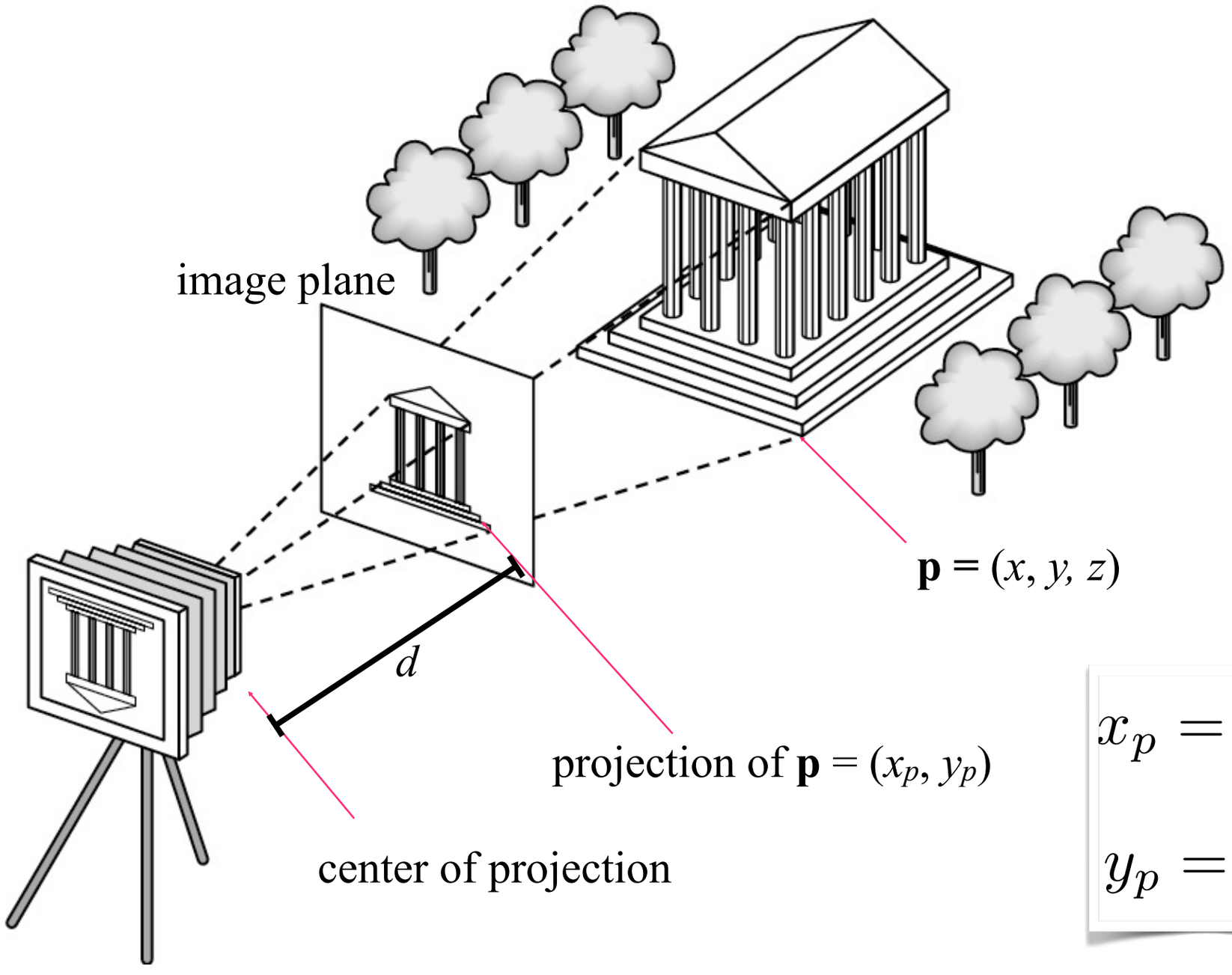
Pinhole Camera





- Projection of a 3D point (x,y,z) on image plane: $x_p = -d\frac{x}{z}, \quad y_p = -d\frac{y}{z}$
  - Equal triangles: $\frac{x}{z} = \frac{x_p}{z_p} \Leftrightarrow x_p = z_p\frac{x}{z} = -d\frac{x}{z}$

Synthetic Camera Model



$$x_p = d\frac{x}{z}$$

$$y_p = d\frac{y}{z}$$

Image Formation

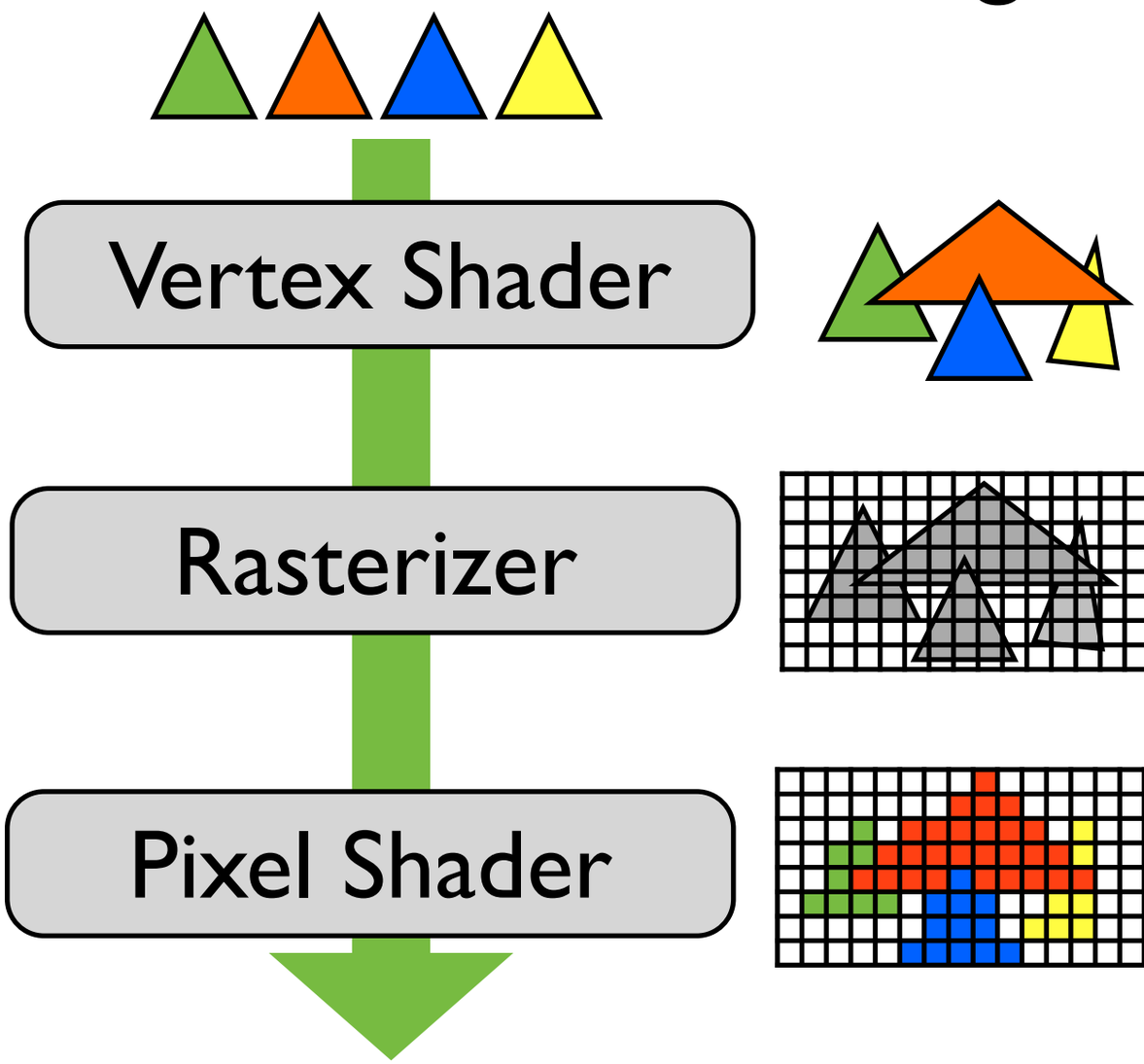
- Create geometric models
  - Position the models in a 3D scene
  - Assign materials to each model
  - Add lights & position a virtual camera
- For each pixel - find visible object
- Compute color of pixel based on the visible object's material and light

Challenges

- Create geometric models 1M triangles
  - Position the models in a 3D scene (Move 1M tris)
  - Assign materials to each model
  - Add lights & position a virtual camera
- For each pixel (5M), compute which triangle (of the 1M tris) that is visible
- Compute color of pixel (5M) based from the object's material and light

Graphics Hardware - GPU

- **Pipeline** that accelerates the costly tasks of rendering



Move geometry

Compute visibility per pixel

Compute color per pixel

Graphics layers

3D Engines Unreal Unity

Graphics APIs DirectX OpenGL DirectX12 Vulkan

Hardware CPU GPU

Bonobo

EDAF80

Real-time vs Offline

- Real-time
  - Render image in ~16 ms (60 FPS)
  - Instant feedback
  - User interactions
- Offline (feature films)
  - Each image may take hours or days
  - Photorealism
  - No user interaction

Real-Time

- Unity Enemies demo
- UE5 Matrix demo
- Sharkmob: Vampire: The Masquerade – Bloodhunt
- Massive/Ubisoft : Avatar: Frontiers of Pandora

Offline

Disney Pixar Lightyear

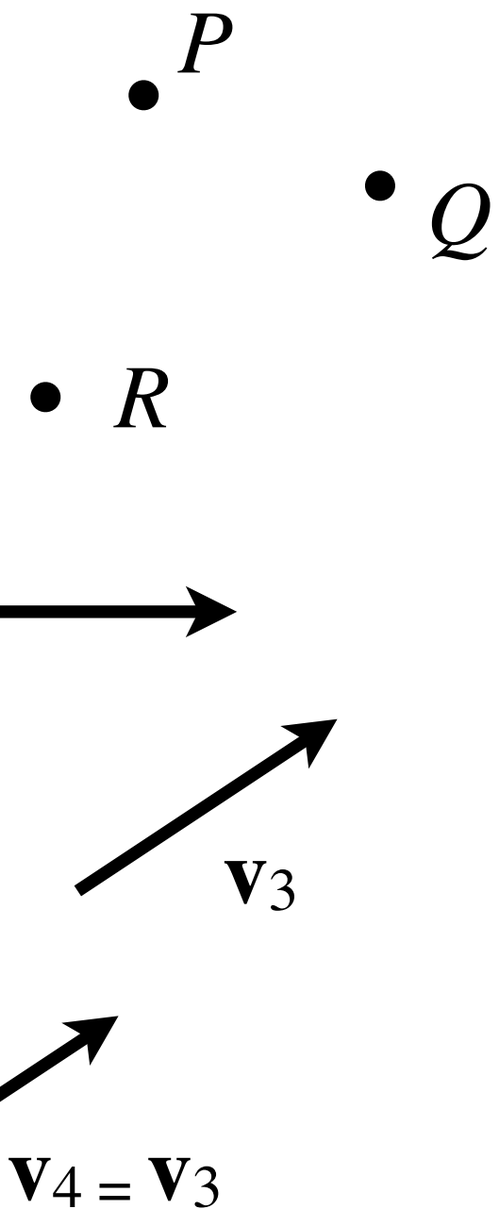
Linear Algebra

Linear Algebra

- Points vs Vectors
- Angles between vectors
- Dot (scalar) product
- Cross product
- Coordinate system

Points & Vectors

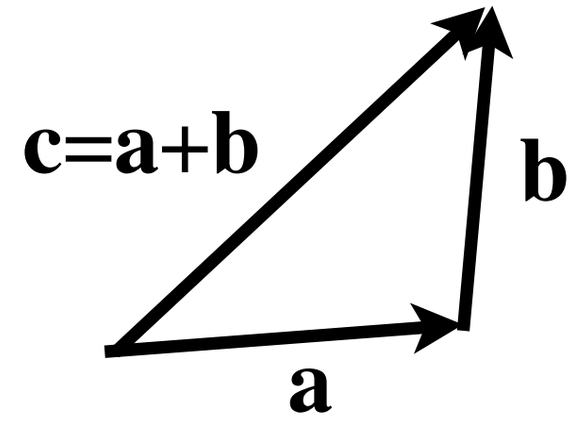
- A **point** is a location in space



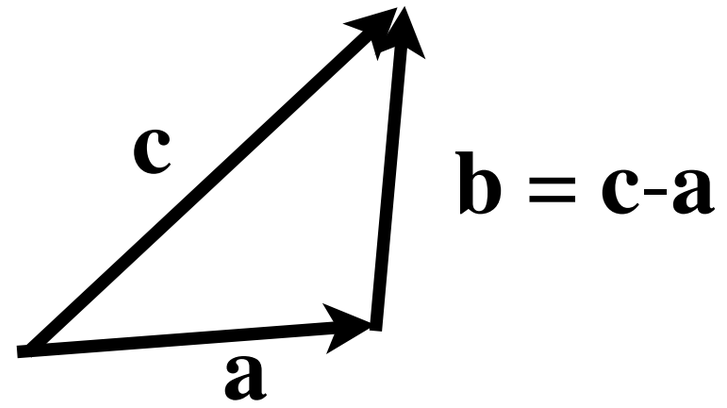
- A **vector** represents a direction and magnitude

Basic Operations

- Vector-vector addition

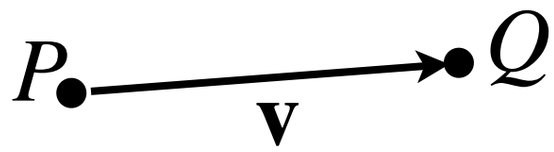


- Vector-vector subtraction



- Point-point subtraction

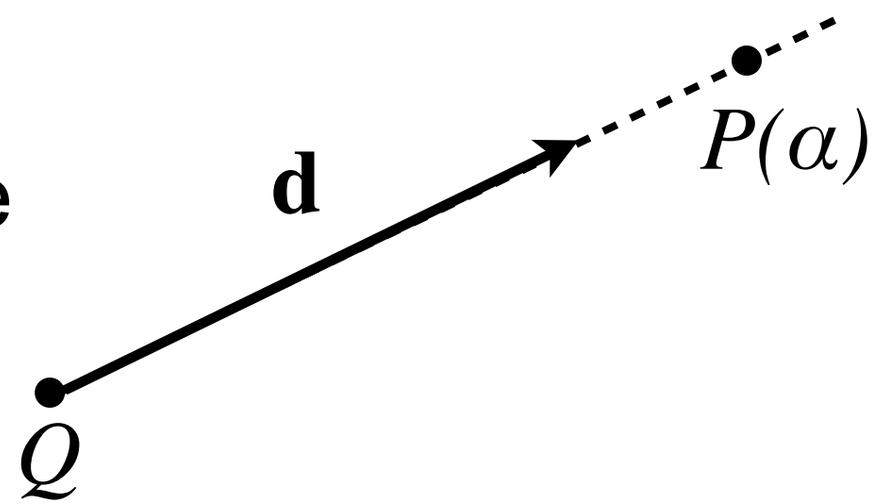
$$\mathbf{v} = Q\text{-}P \Leftrightarrow Q = P + \mathbf{v}$$



Lines

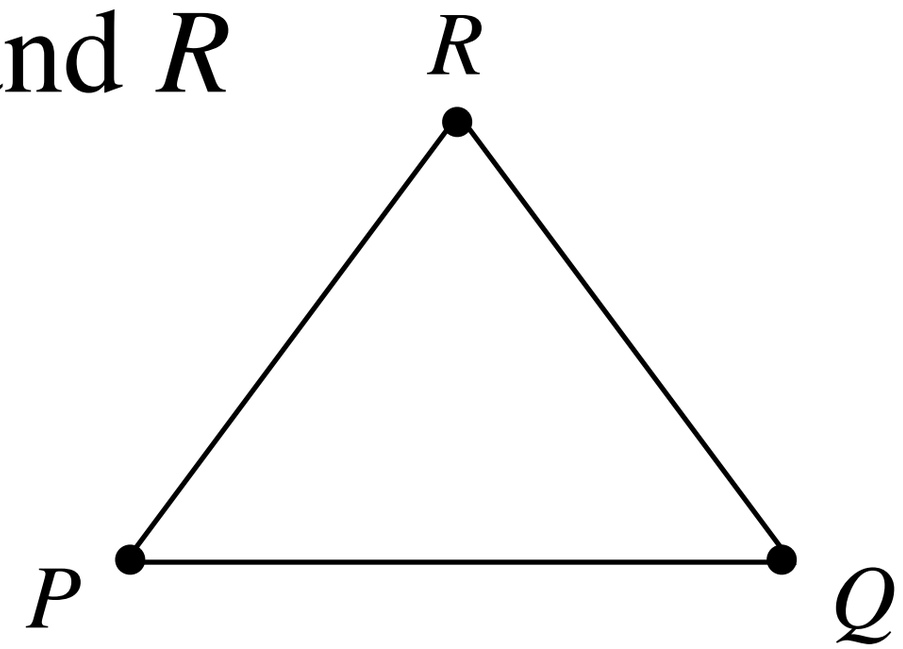
- Parametric form
  - Start with a point Q and a vector $\mathbf{d}$
  - If $\alpha>0$, the line is called the **ray** from Q in the direction $\mathbf{d}$
  - α is a scalar parameter that determines how far we have travelled along the line

$$P(\alpha) = Q + \alpha\mathbf{d}$$



Triangle

- Defined by three points P, Q and R

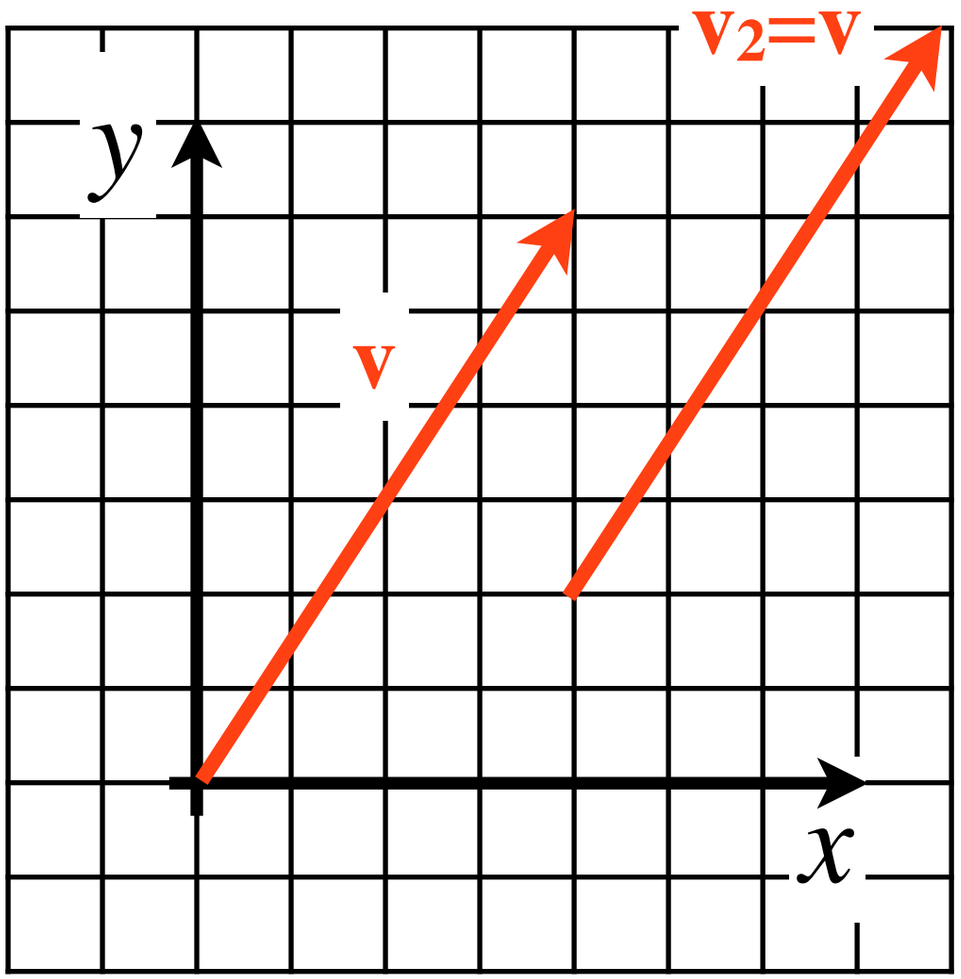


- Points inside triangle: $wP + uQ + vR$
- u,v,w : barycentric coordinates

$$u + v + w = 1$$

$$u, v, w \geq 0$$

Cartesian Coordinates



$$\mathbf{v} = 4\mathbf{x} + 6\mathbf{y}$$

$$= 4\begin{bmatrix} 1 \\ 0 \end{bmatrix} + 6\begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 6 \end{bmatrix}$$

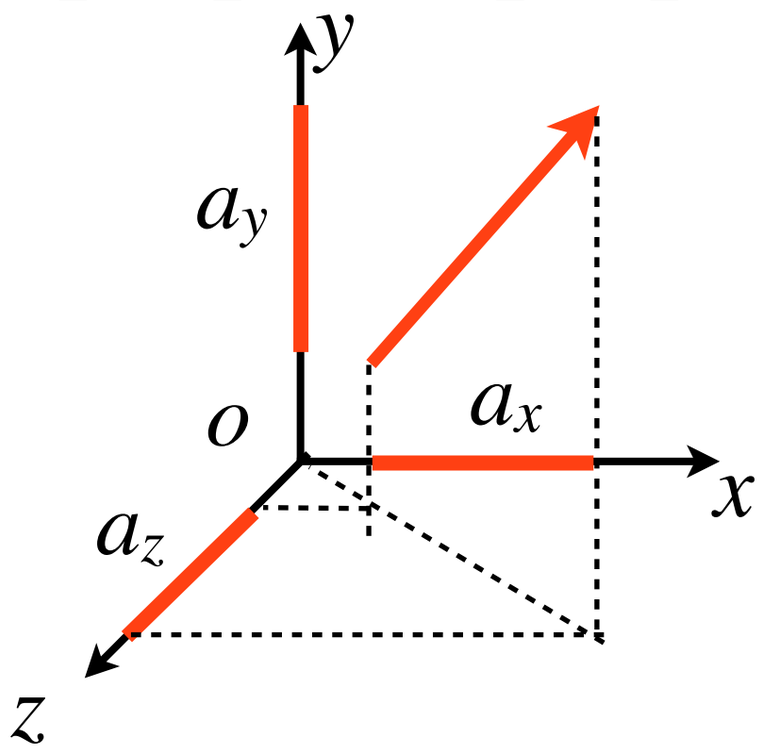
$$|\mathbf{v}| = \sqrt{(4^2 + 6^2)}$$

Cartesian Coordinates in 3D

- Express vector in terms of three orthonormal basis vectors

$$\mathbf{a} = a_x \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + a_y \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + a_z \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}$$

Basis



In this basis, the vector $\mathbf{a}$ can be expressed as

$$\mathbf{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}$$

Vector notation

- To avoid clutter, we introduce the notation:

$$\mathbf{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = (a_x, a_y, a_z)$$

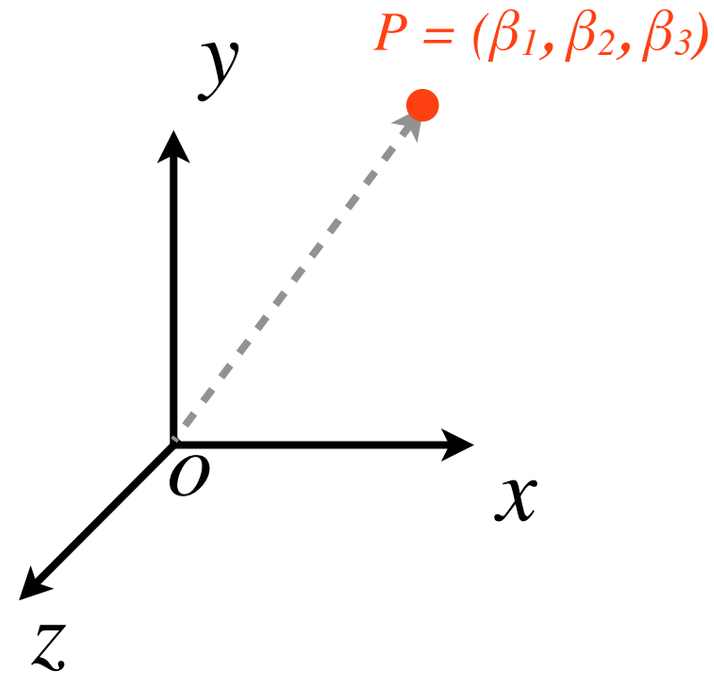
- Note that $\mathbf{a} = (a_x, a_y, a_z)$ is still a **column-vector**, expressed in a certain basis

Points in 3D

- Coordinate **frame**
  - Basis vectors + **origin**

$$\begin{aligned} \mathbf{v}_1 &= (1,0,0) \\ \mathbf{v}_2 &= (0,1,0) \\ \mathbf{v}_3 &= (0,0,1) \end{aligned}$$

$$\boxed{\mathbf{o} = (0,0,0)}$$

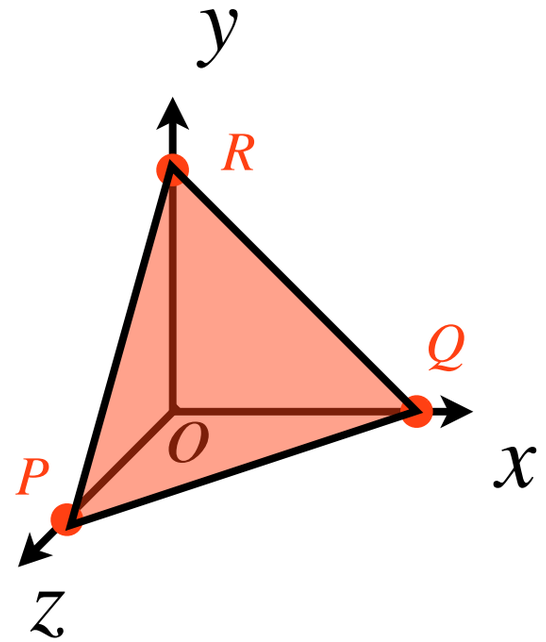


- In this frame, a **point** is given by:

$$\begin{aligned} P &= \mathbf{o} + \beta_1\mathbf{v}_1 + \beta_2\mathbf{v}_2 + \beta_3\mathbf{v}_3 \\ &= \beta_1(1,0,0) + \beta_2(0,1,0) + \beta_3(0,0,1) \\ &= (\beta_1, \beta_2, \beta_3) \end{aligned}$$

Example: Triangle in 3D

- Defined by three points,
  $P = (0,0,1)$
  $Q = (1,0,0)$
  $R = (0,1,0)$
  specified in the Cartesian system



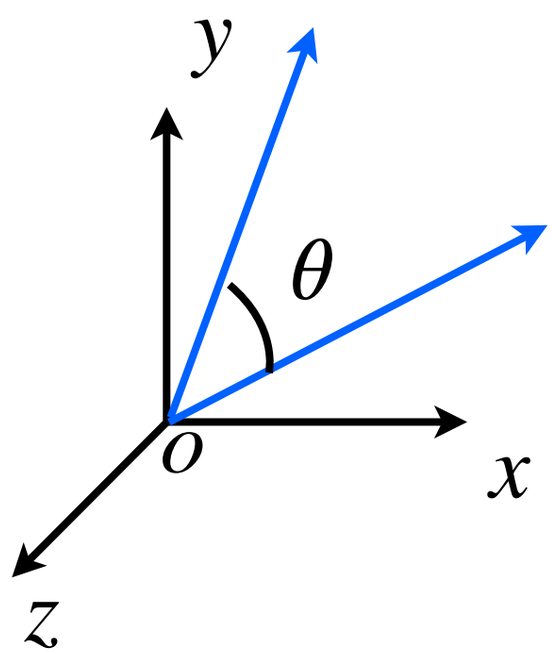
$$P = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + 0 \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + 0 \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + 1 \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Euclidean 3D Space

- Coordinate **frame**
  - Basis vectors + **origin**

$$\begin{aligned} \mathbf{v}_1 &= (1,0,0) \\ \mathbf{v}_2 &= (0,1,0) \\ \mathbf{v}_3 &= (0,0,1) \end{aligned}$$

$$\boxed{\mathbf{o} = (0,0,0)}$$

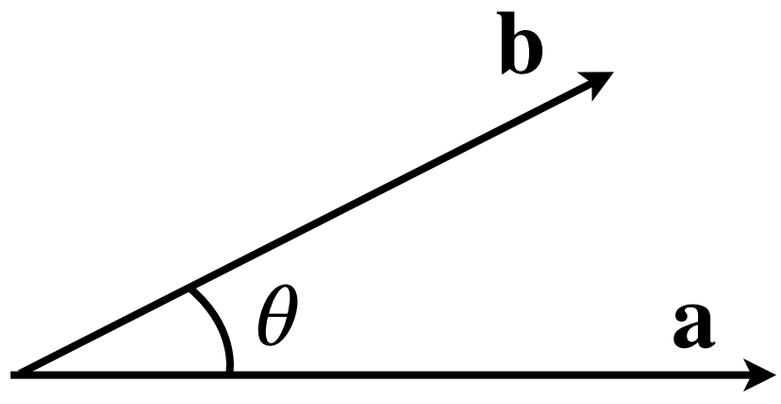


- The Cartesian coordinates express the three basis vectors, origin at (0,0,0)
- How to define the angle between two vectors in this coordinate frame?

Dot Product (scalar product)

- Given two 3D vectors $\mathbf{a} = (a_x, a_y, a_z)$ and $\mathbf{b} = (b_x, b_y, b_z)$, the dot product is given by

$$\mathbf{a} \cdot \mathbf{b} = a_x b_x + a_y b_y + a_z b_z = |\mathbf{a}||\mathbf{b}| \cos\theta$$



- Angle between $\mathbf{a}$ and $\mathbf{b}$: $\theta = \arccos\left(\frac{\mathbf{a} \cdot \mathbf{b}}{|\mathbf{a}||\mathbf{b}|}\right)$

Shadertoy Rainforest

https://www.shadertoy.com/view/4ttSWf

making of https://youtu.be/BFld4EBO2RE

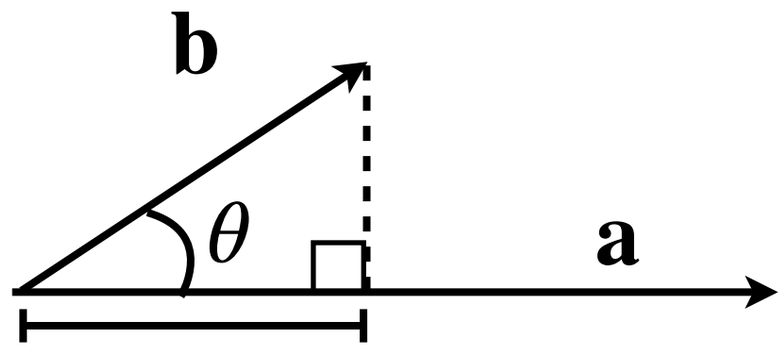
Dot Product (scalar product)

- Use cases

$$\mathbf{a} \cdot \mathbf{b} = a_x b_x + a_y b_y + a_z b_z = |\mathbf{a}||\mathbf{b}| \cos\theta$$

  - The projection of $\mathbf{b}$ on $\mathbf{a}$

$$|\mathbf{b}| \cos\theta = \frac{\mathbf{a} \cdot \mathbf{b}}{|\mathbf{a}|}$$



  - Square magnitude of vector

$$\mathbf{a} \cdot \mathbf{a} = |\mathbf{a}|^2 \Leftrightarrow |\mathbf{a}| = \sqrt{\mathbf{a} \cdot \mathbf{a}}$$

  - $\mathbf{a}$ and $\mathbf{b}$ orthogonal if

$$\mathbf{a} \cdot \mathbf{b} = 0$$

Cross Product

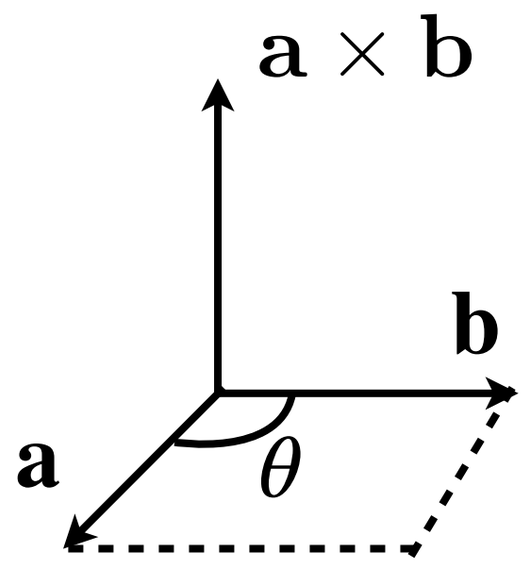
- Given two 3D vectors $\mathbf{a} = (a_\mathrm{x}, a_\mathrm{y}, a_\mathrm{z})$ and $\mathbf{b} = (b_\mathrm{x}, b_\mathrm{y}, b_\mathrm{z})$, the cross product is given by

$$\mathbf{a} \times \mathbf{b} = (a_y b_z - a_z b_y, a_z b_x - a_x b_z, a_x b_y - a_y b_x)$$

- Signed area of the parallelepiped spanned by vectors $\mathbf{a}$ and $\mathbf{b}$:

$$|\mathbf{a} \times \mathbf{b}| = |\mathbf{a}||\mathbf{b}| \sin\theta$$

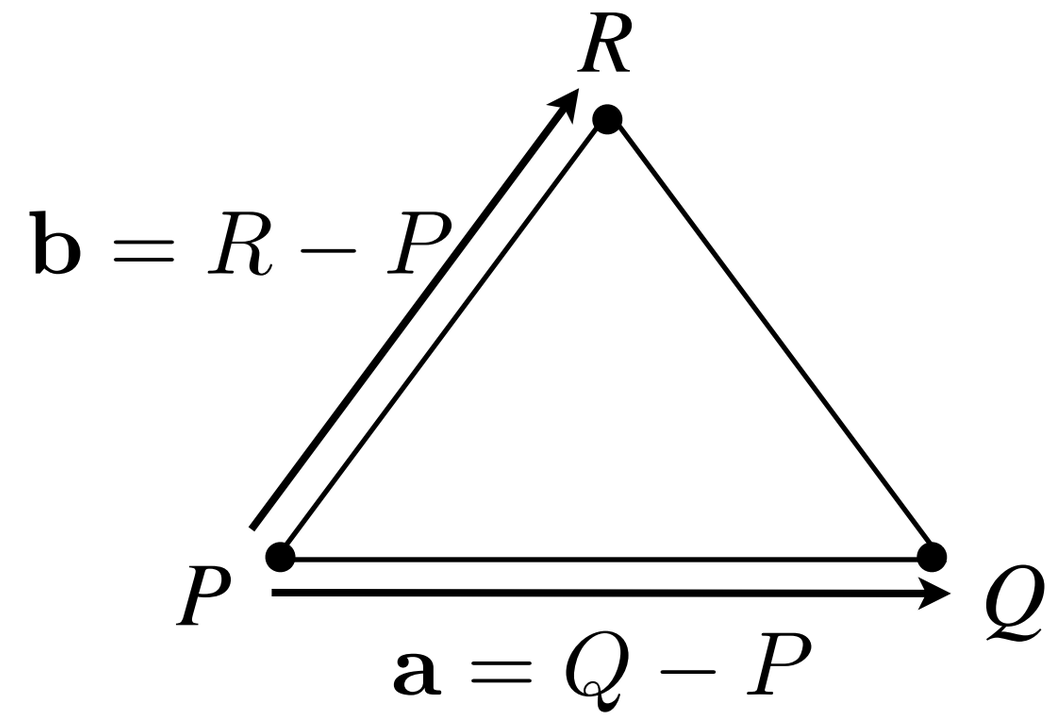
- $\mathbf{a} \times \mathbf{b}$ is orthogonal to both $\mathbf{a}$ and $\mathbf{b}$
- $\mathbf{a} \times \mathbf{a} = \mathbf{0}$
- Note: $\mathbf{a} \times \mathbf{b} = -\mathbf{b} \times \mathbf{a}$



Triangle Normal

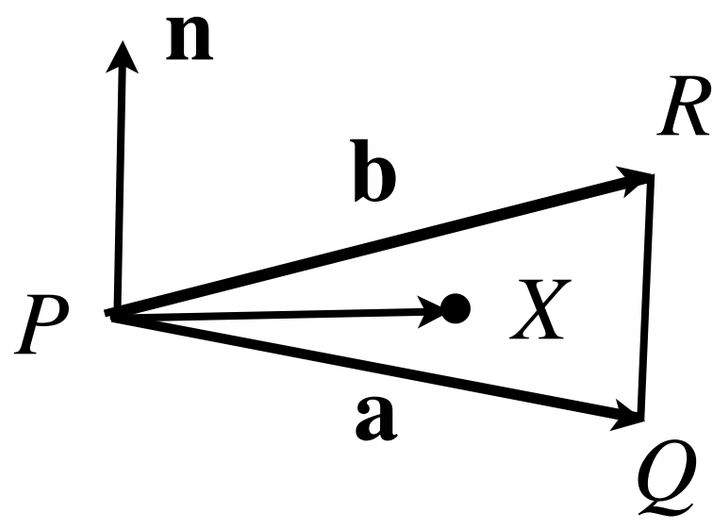
- Introduce two edge vectors $\mathbf{a}, \mathbf{b}$
- Triangle face normal

$$\mathbf{n} = \frac{\mathbf{a} \times \mathbf{b}}{|\mathbf{a} \times \mathbf{b}|}$$



- Plane equation:
  - X belongs to the triangle plane if

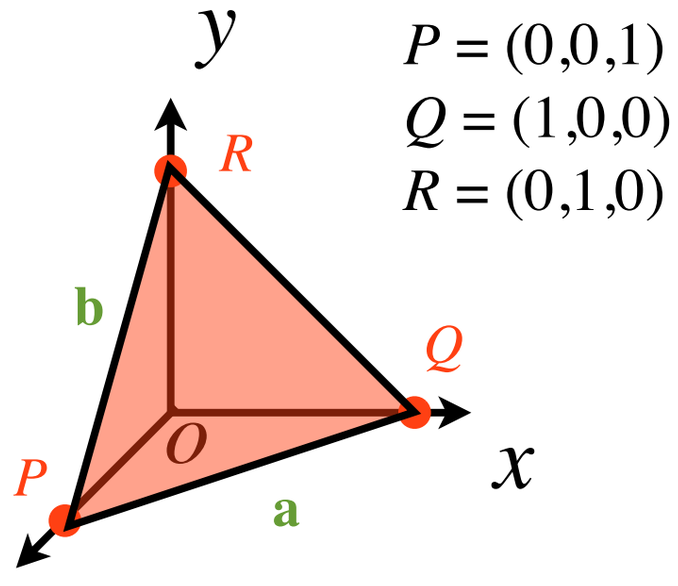
$$\mathbf{n} \cdot (X - P) = 0$$



Example: Triangle in 3D

- Edge vectors

$$\begin{aligned}\mathbf{a} &= Q - P = (1, 0, -1)\\ \mathbf{b} &= R - P = (0, 1, -1)\end{aligned}$$



- Face normal

$$\mathbf{n} = \frac{\mathbf{a} \times \mathbf{b}}{|\mathbf{a} \times \mathbf{b}|} = \frac{(1, 1, 1)}{|(1, 1, 1)|} = \frac{1}{\sqrt{3}}(1, 1, 1)$$

- Plane: $\mathbf{n} \cdot (X - P) = 0 \qquad X = (x, y, z)$

$$x + y + z - 1 = 0$$

Matrices

- We will use
  - Matrix-matrix multiplication: $\mathbf{A}\,\mathbf{B} = \mathbf{C}$
    - Concatenate transforms, change basis
  - Matrix-vector multiplication $\mathbf{A}\,\mathbf{x} = \mathbf{y}$
    - Transform vectors and points

Matrix-vector multiplication

$$A\mathbf{x} = \mathbf{y}$$

$$\begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_{00}x + a_{01}y + a_{02}z \\ a_{10}x + a_{11}y + a_{12}z \\ a_{20}x + a_{21}y + a_{22}z \end{bmatrix}$$

Matrix-matrix multiplication

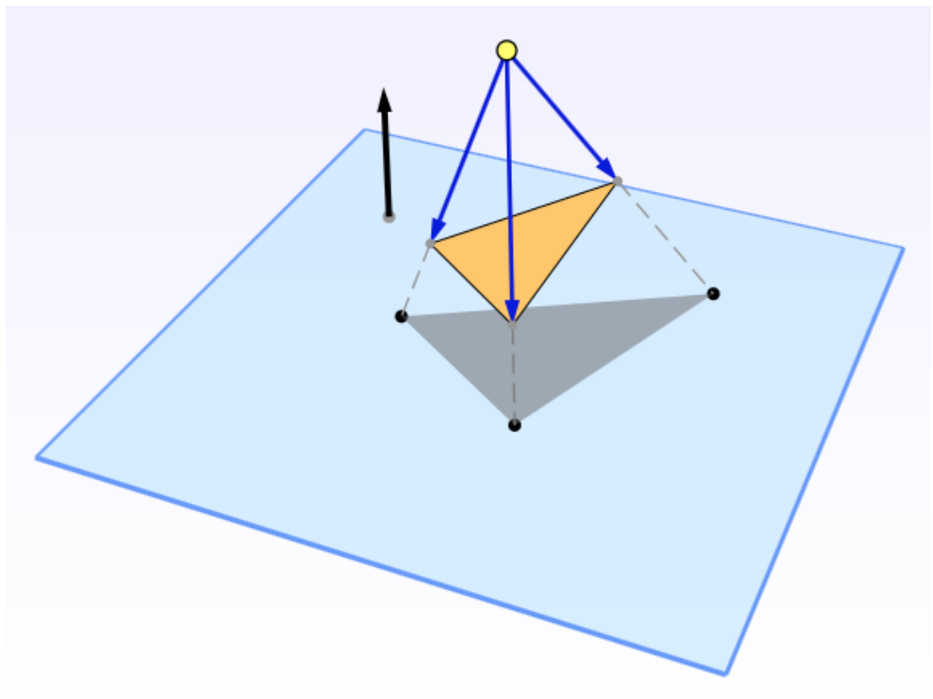
$$AB = C$$

$$\begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} b_{00} & b_{01} & b_{02} \\ b_{10} & b_{11} & b_{12} \\ b_{20} & b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} c_{00} & c_{01} & c_{02} \\ c_{10} & c_{11} & c_{12} \\ c_{20} & c_{21} & c_{22} \end{bmatrix}$$

$$c_{ij} = \sum_{k=0}^{2} a_{ik} b_{kj}$$

Online Book on Linear Algebra

- immersive linear algebra
  - http://immersivemath.com/
  - by J. Ström, K. Åström, and T. Akenine-Möller
  - Interactive figures



Next lecture

- Transforms
- Coordinate Spaces
- Homogeneous Coordinates
- Seminar: OpenGL intro & C++ basics