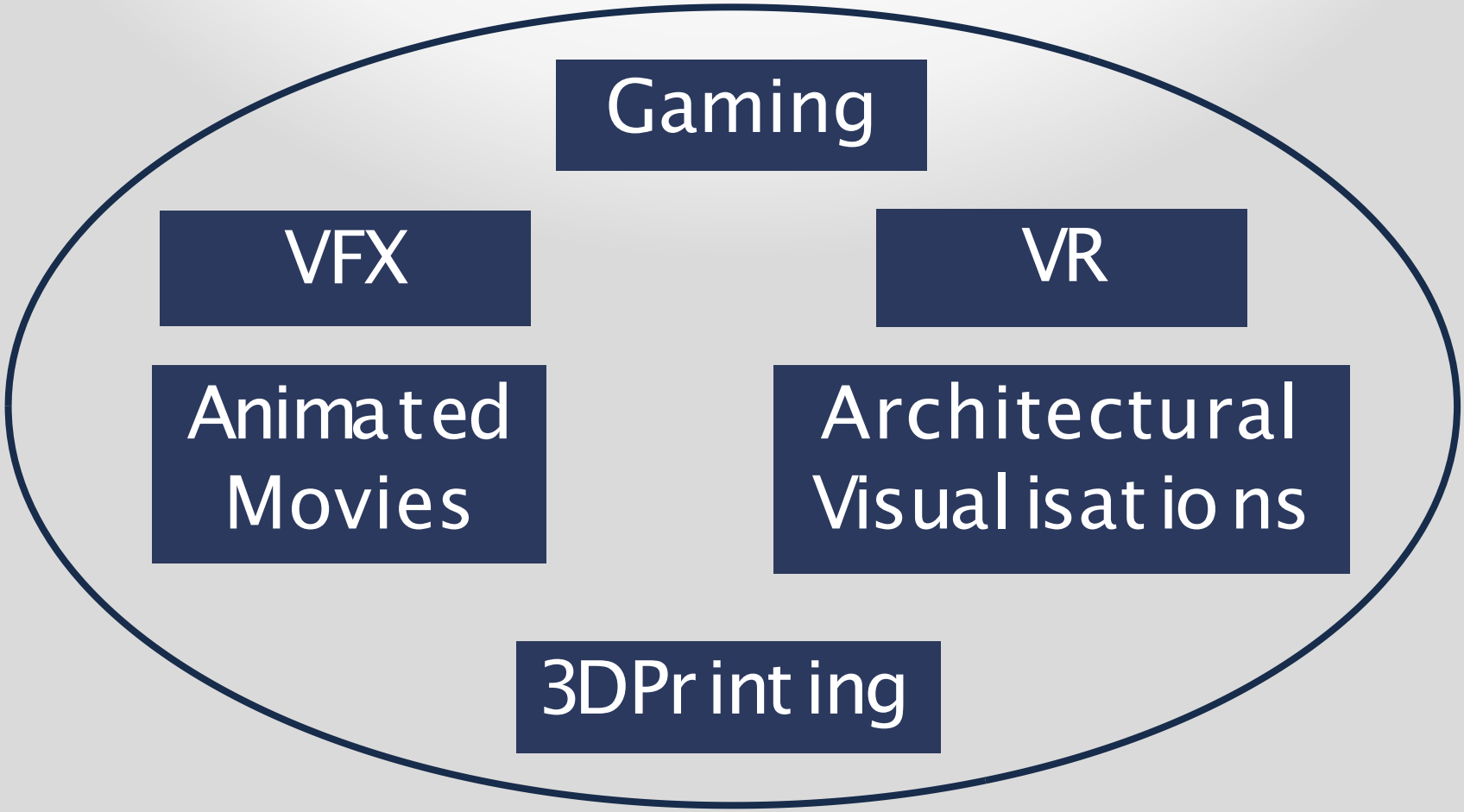
CS3005: DIGITAL MEDIA AND GAMES

3D Graphics for Games

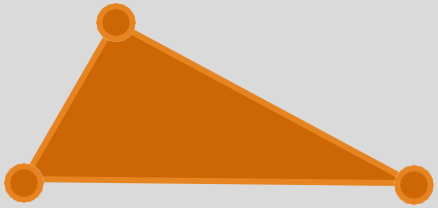
George Ghinea
Nadine Aburumman

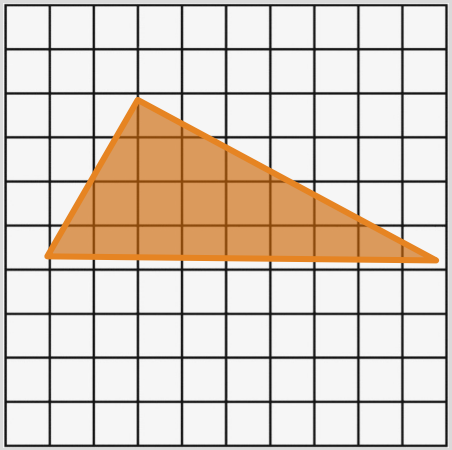


Requires Basic 3D

Requires Basic 3D

3D Graphics Pipeline





Application

Geometry

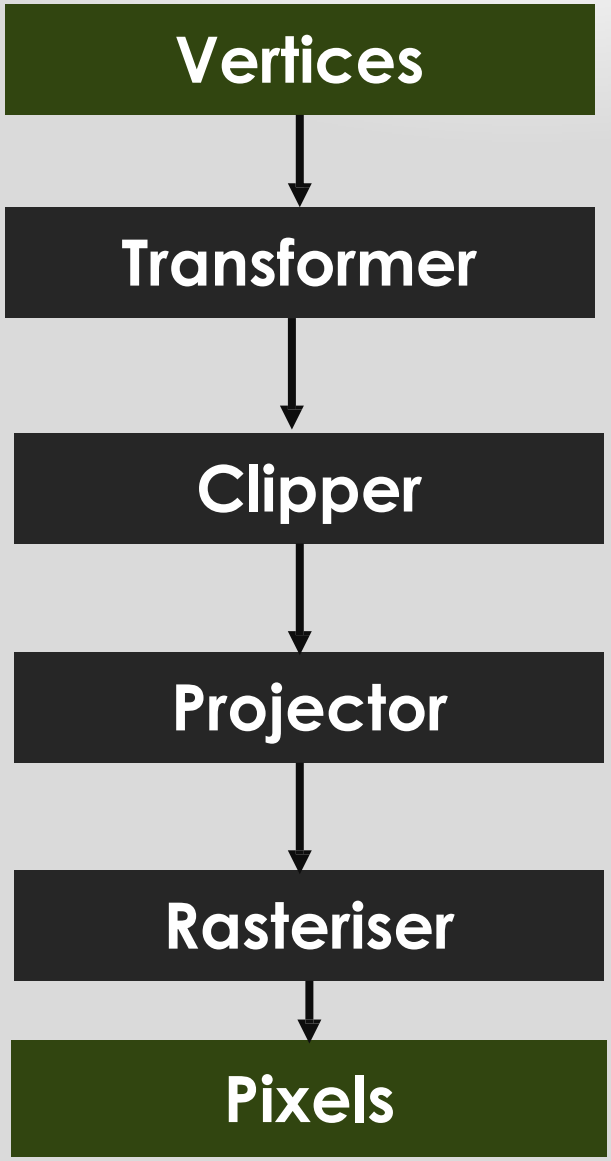
Rasterisation

Screen

Geometry Pipeline

Rasterise Triangles and Interpolate the Geometry Attributes from Screen Coordinate to Pixel

3D Graphics Pipeline



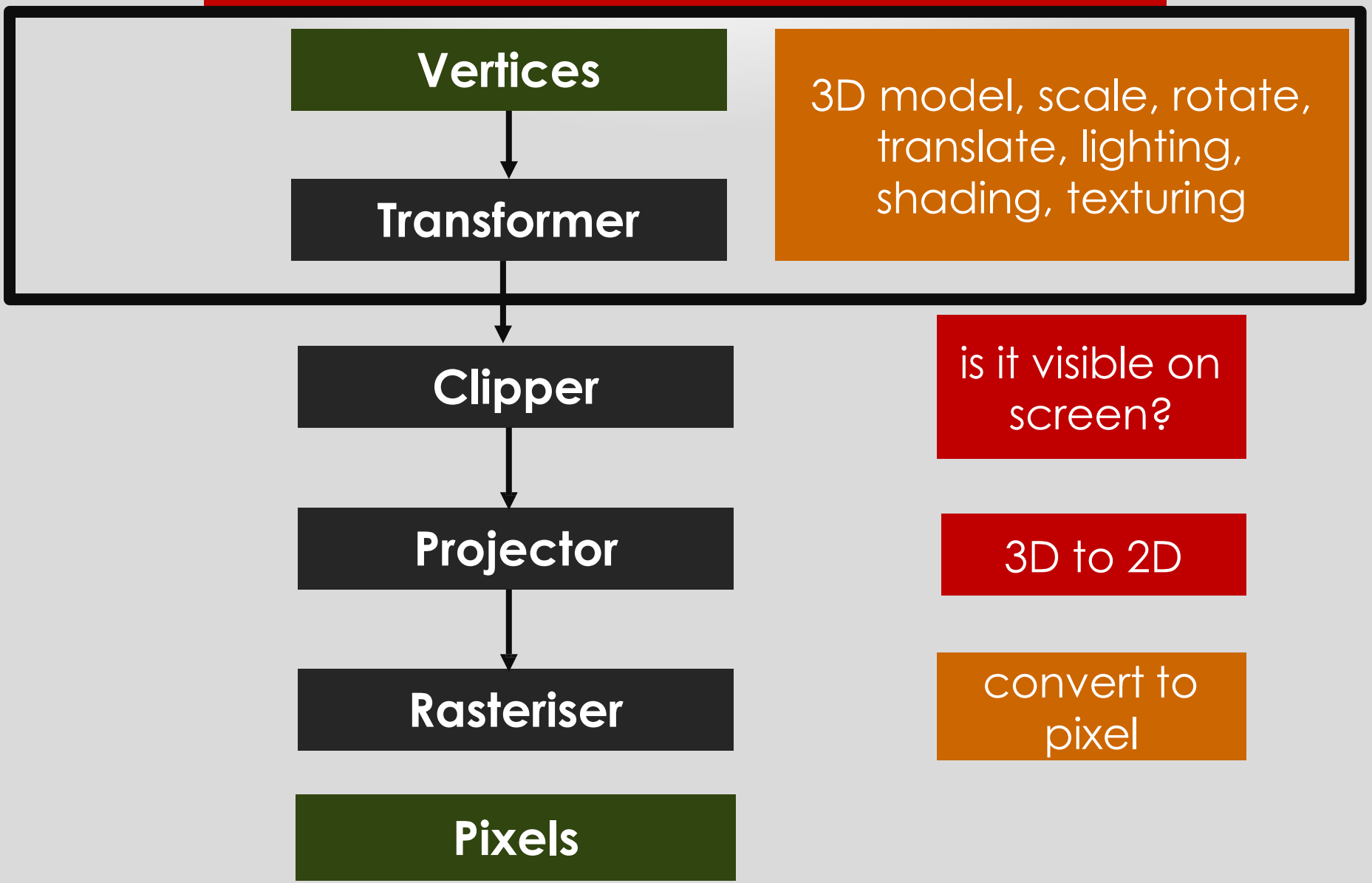
3D model, scale, rotate, translate, lighting, shading, texturing

is it visible on screen?

3D to 2D

convert to pixel

3D Graphics Pipeline

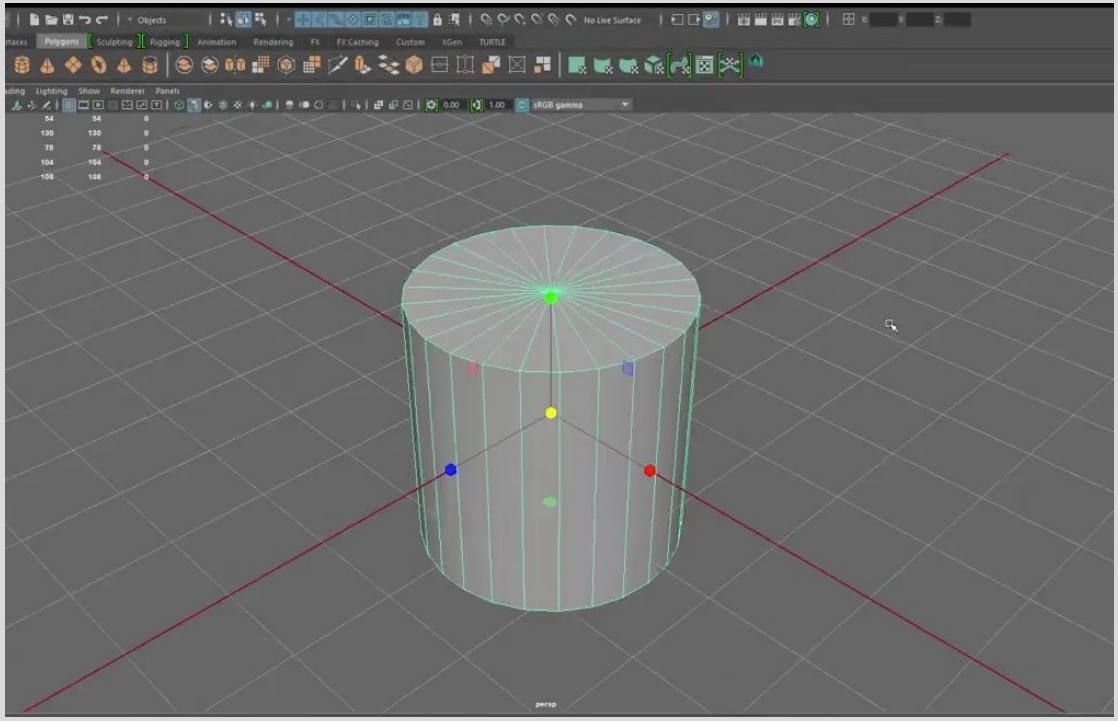


3Building Blocks of 3D

Modelling

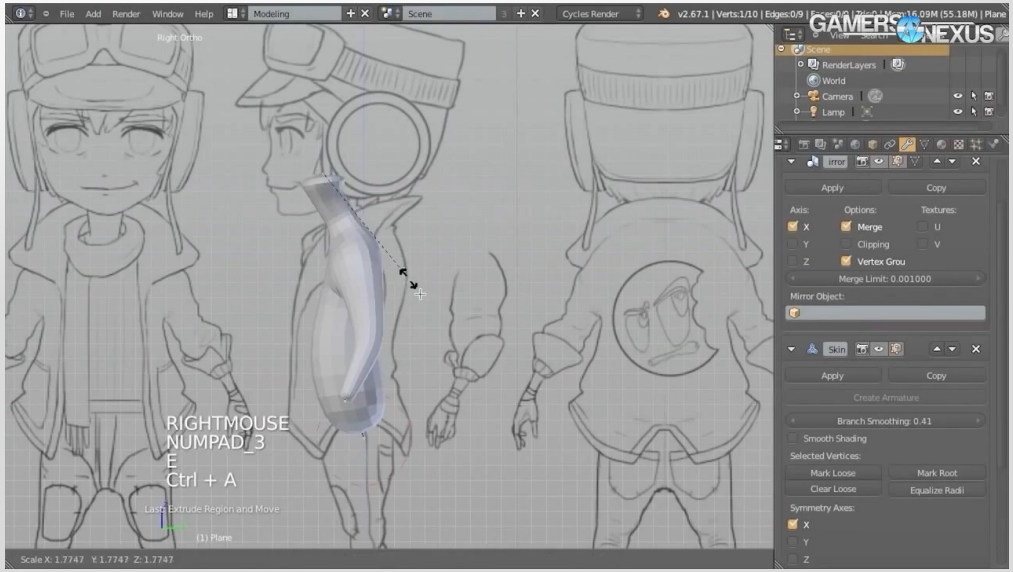
Lighting

Texturing



Modelling

3d Modelling design



3d Capture

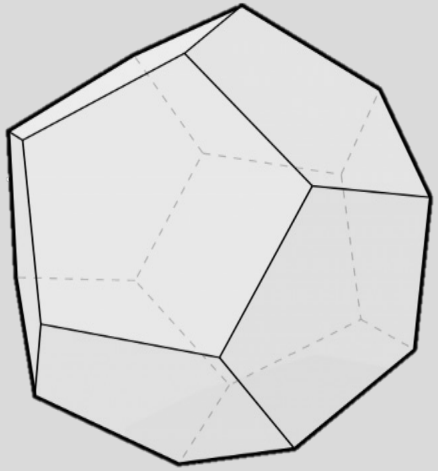
3D Geometry

3D Geometry

earth measure

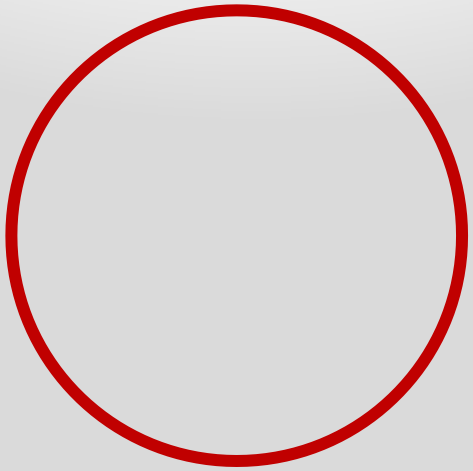
geo • metry:

- The study of shapes, sizes, patterns, and positions.



- The study of spaces where some quantity (lengths, angles, etc.) can be measured.

How can we describe geometry?



Unit Circle

How can we describe geometry?

Implicit

$x^2 + y^2 = 0$

Unit Circle

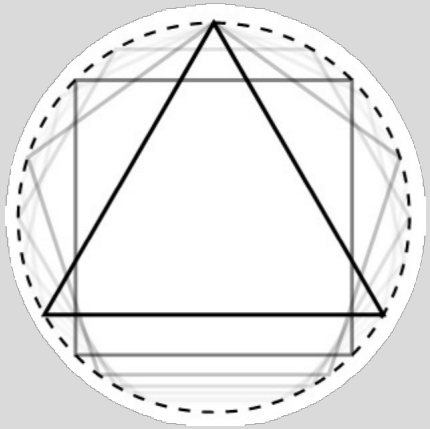
Explicit

$(\underbrace{\sin\theta}_{x}, \underbrace{\cos\theta}_{y})$

Curvature

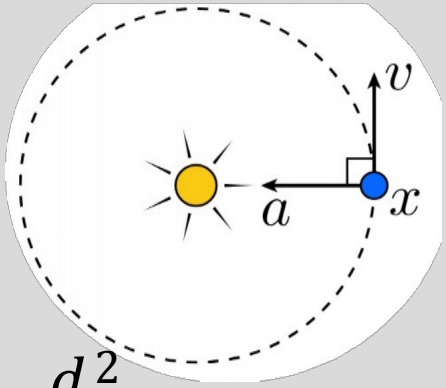
$k = 1$

Discrete



$n \to \infty$

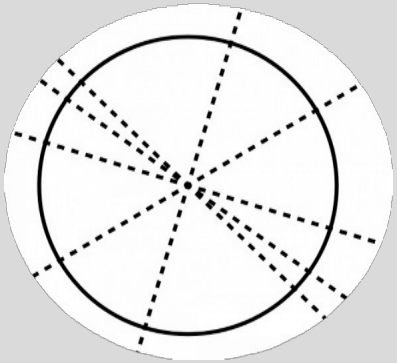
Dynamic



$$\frac{d^2}{dt^2} x = -x$$

Tomographic

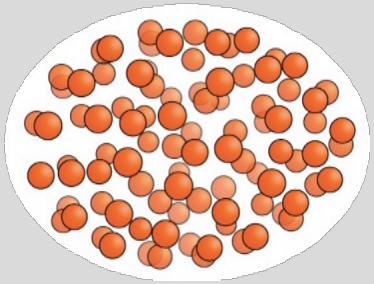
constant density

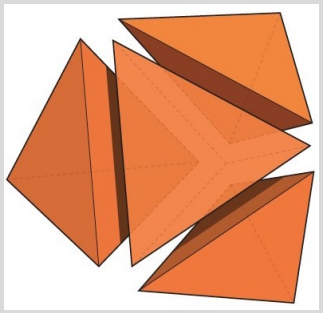


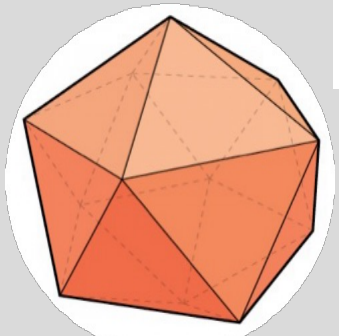
Many ways to digitally encode geometry

Explicit

- **Point cloud**
- **Polygon mesh**
- **Subdivision, NURBS**

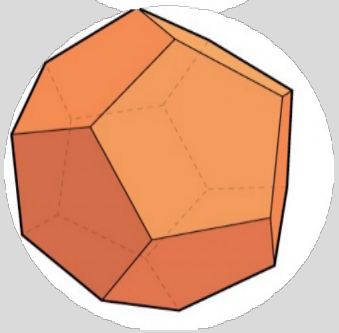


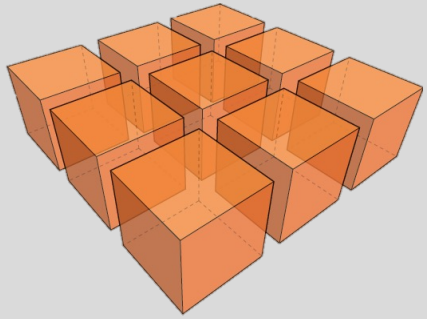




Implicit

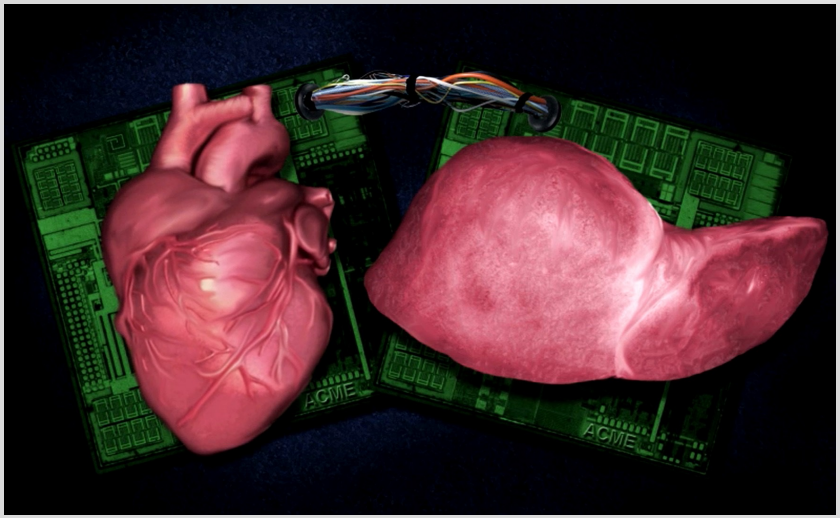
- **Level set**
- **Algebraic surface**
- **L-systems**





Given all these options, what is the best way to encode geometry on a computer?

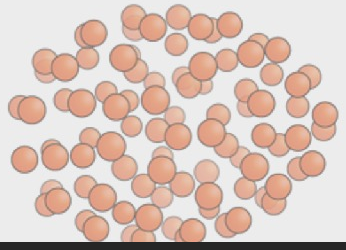
Examples of geometry



MANY WAYS TO DIGITALLY ENCODE GEOMETRY

EXPLICIT

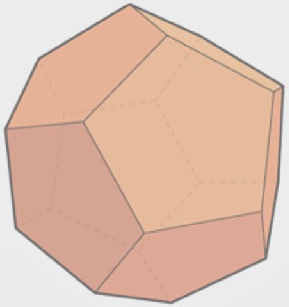
- Point cloud

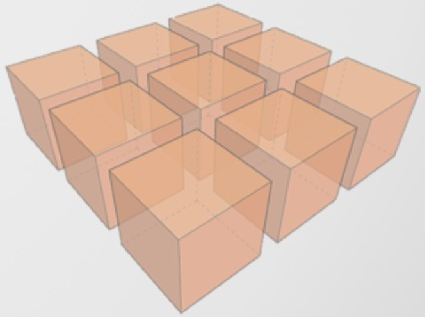


Some representations work better than others — depends on the task

IMPLICIT

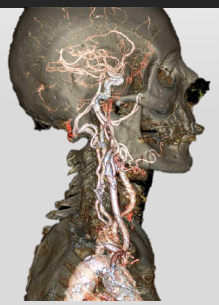
- Level set
- Algebraic surface
- L-systems





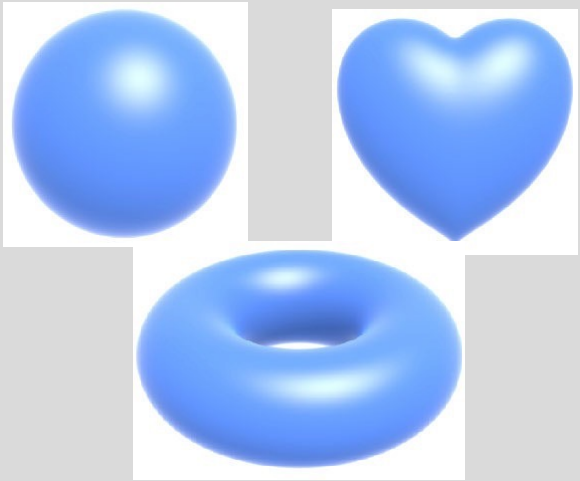
Some representations work better than others—depends on the task

Level set



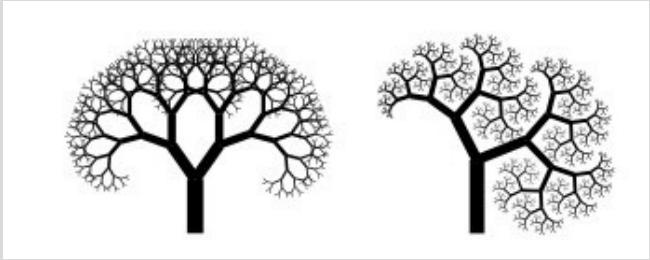
constant tissue density

Algebraic surface



polynomials

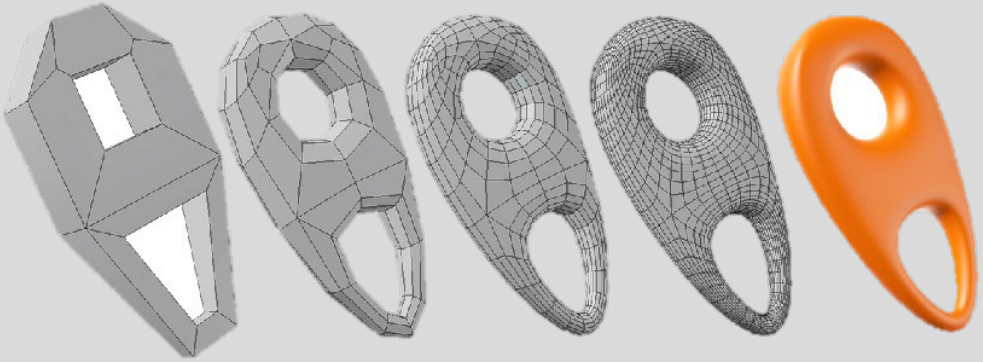
L System

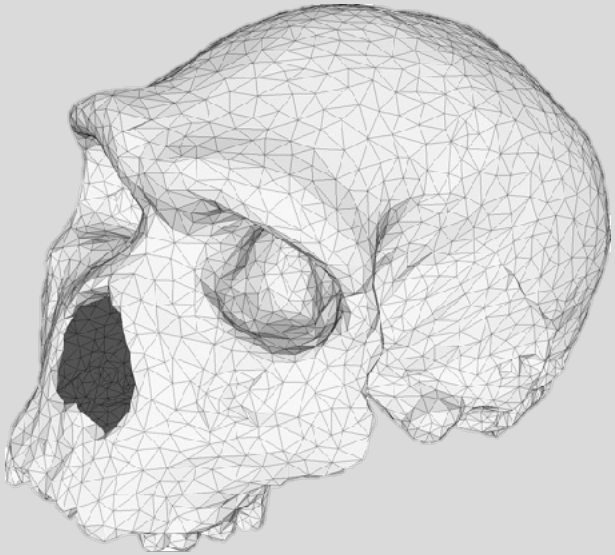


natural patterns

Explicit representation in Graphics

- Point cloud
- Polygon mesh
- Triangle mesh
- Subdivision surfaces
- NURBS

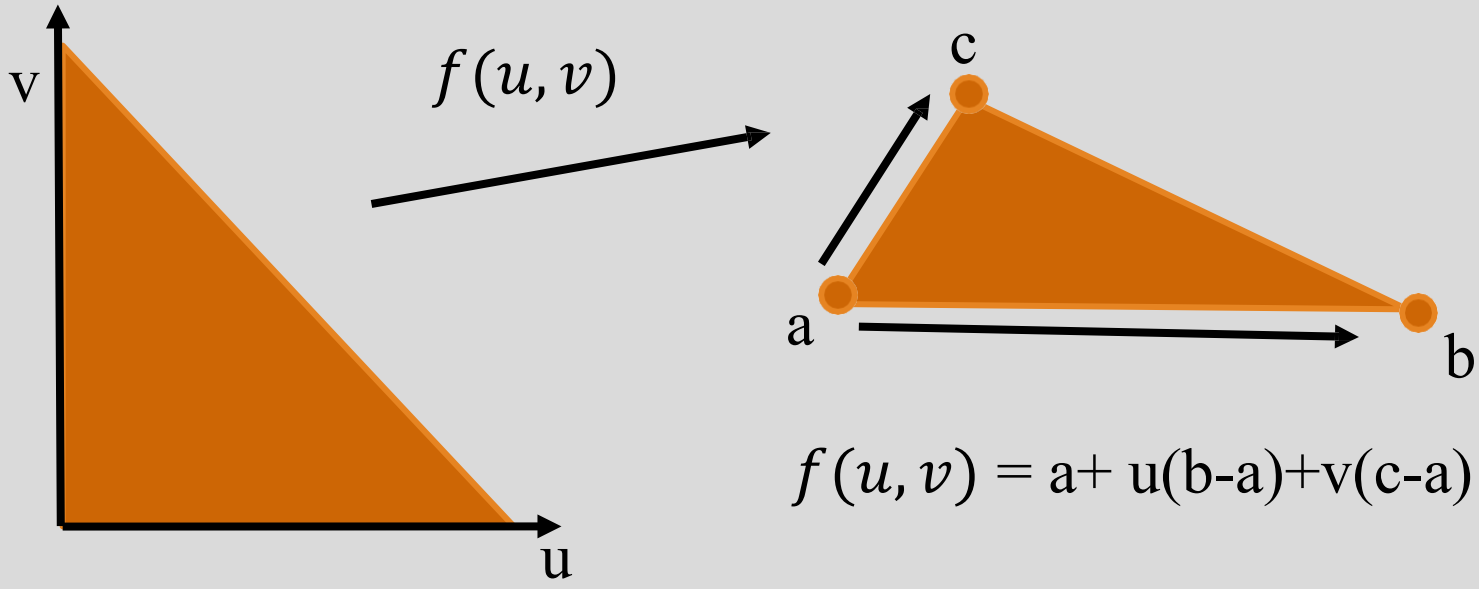




TriangleMesh

The reason why triangles are commonly used in practice is **simplicity**:

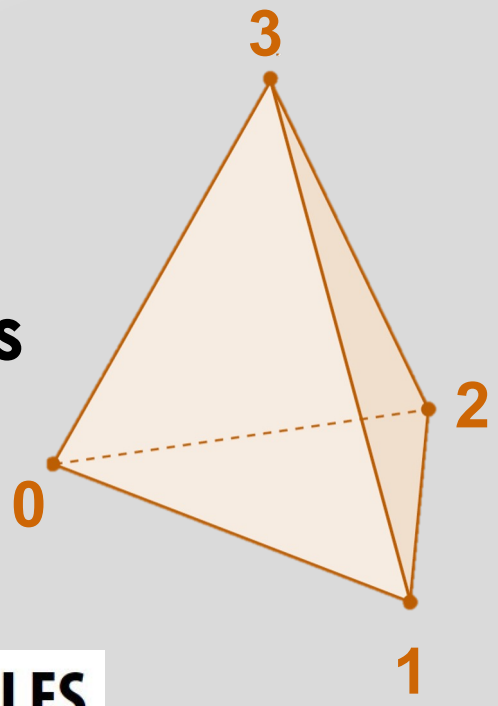
Three points in space unambiguously define a plane



$$f(u,v) = a + u(b-a) + v(c-a)$$

TriangleMesh

- Store vertices as triples of coordinates (x,y,z)
- Store triangles as triples of indices (i,j,k)
- E.g., tetrahedron

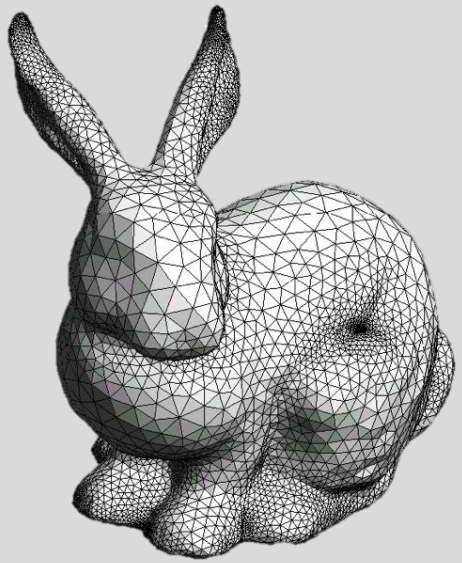


	VERTICES			TRIANGLES		
	x	y	z	i	j	k
0:	-1	-1	-1	0	2	1
1:	1	-1	1	0	3	2
2:	1	1	-1	3	0	1
3:	-1	1	1	3	1	2

TriangleMesh

```
# vertex count = 2503
# face count = 4968
v -3.4101800e-003 1.3031957e-001 2.1754370e-002
v -8.1719160e-002 1.5250145e-001 2.9656090e-002
v -3.0543480e-002 1.2477885e-001 1.0983400e-003
v -2.4901590e-002 1.1211138e-001 3.7560240e-002
v -1.8405680e-002 1.7843055e-001 -2.4219580e-002
v 1.9067940e-002 1.2144925e-001 3.1968440e-002
v 6.0412000e-003 1.2494359e-001 3.2652890e-002
v -1.3469030e-002 1.6299355e-001 -1.2000020e-002
v -3.4393240e-002 1.7236688e-001 -9.8213000e-004
v -8.4314160e-002 1.0957263e-001 3.7097300e-003
v -4.2233540e-002 1.7211574e-001 -4.1799800e-003
v -6.3308390e-002 1.5660615e-001 -1.3838790e-002
v -7.6903950e-002 1.6708033e-001 -2.6931360e-002
v -7.2253920e-002 1.1539550e-001 5.1670300e-002
```

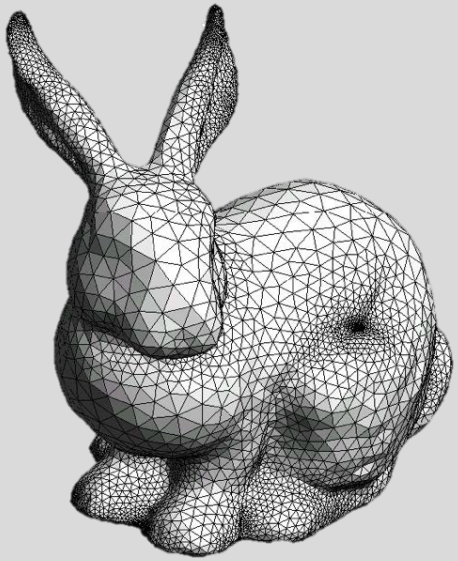
```
f 420 1071 1065
f 264 281 254
f 298 309 281
f 368 366 367
f 368 396 366
f 1639 1564 1139
f 560 48 47
f 82 471 212
f 25 38 37
f 202 1206 1080
f 264 298 281
f 298 331 309
f 309 331 366
```

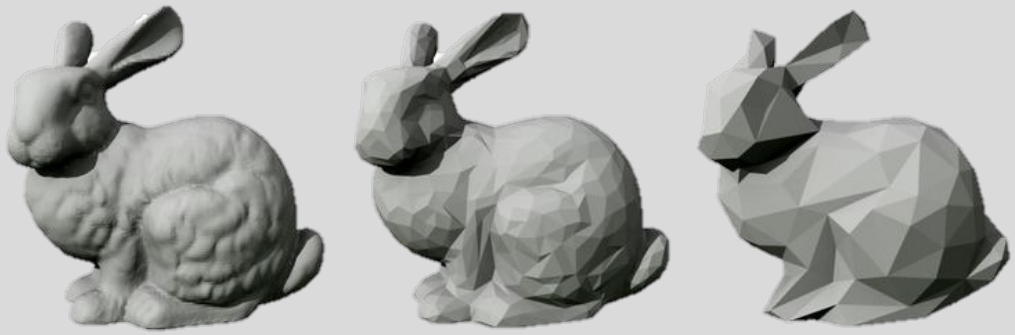


OBJ File

TriangleMesh

Game designers have to carefully balance **model fidelity vs polygon count**, because if the count goes too high, the **framerate** of an animation drops below what users perceive as smooth.





3D Transformations

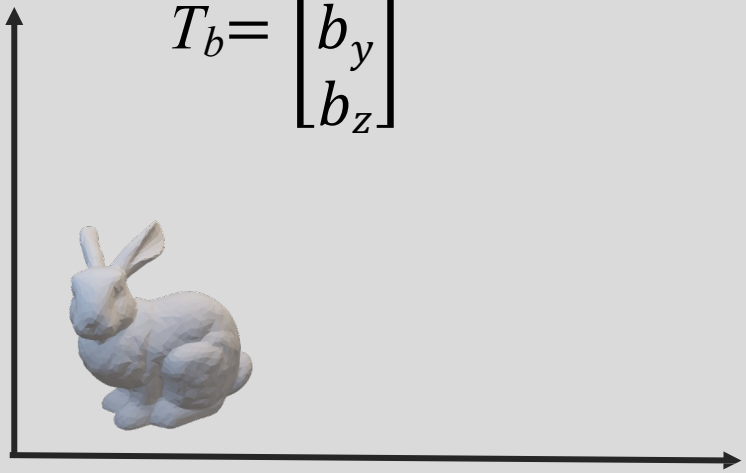
Translation	Position
Scaling	Size
Rotation	orientation

represent 3D transformations as matrices

3D Transformations

Translation

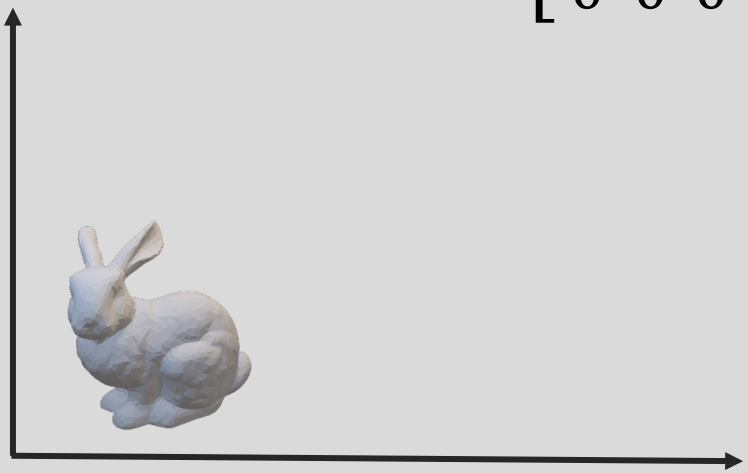
$$T_b = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix}$$



$$\mathrm{v} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Homogeneous Coordinates

$$\mathrm{T}_b = \begin{bmatrix} \boldsymbol{I} & T_b \\ \mathbf{0}^T & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & b_x \\ 0 & 1 & 0 & by \\ 0 & 0 & 1 & b_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & b_x \\ 0 & 1 & 0 & by \\ 0 & 0 & 1 & b_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$\mathrm{v}' = T_b v$$

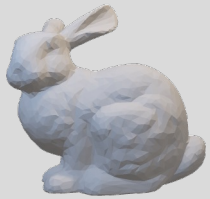
3D Transformations

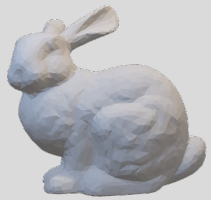
Scaling

$$S_s = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & S_z \end{bmatrix}$$

$$S_s = \begin{bmatrix} S_s & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Homogeneous Coordinates





$$v = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$v' = S_s v$$

3D Transformations

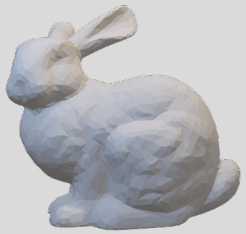
Rotation

$R_{yz}(\theta)$: rotation about x axis by θ

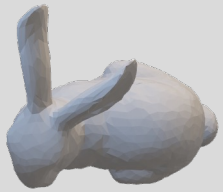
$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Homogeneous Coordinates



$$v = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$



$\theta = 90°$

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

$$v' = R_x v$$

3D Transformations

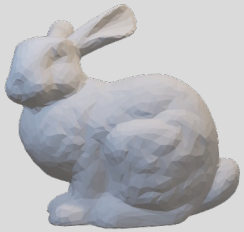
Rotation

$R_{zx}(\theta)$: rotation about y axis by θ

$$\begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_y = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

Homogeneous Coordinates



$\theta = 90°$

$$v = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

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

$$v' = R_y v$$

3D Transformations

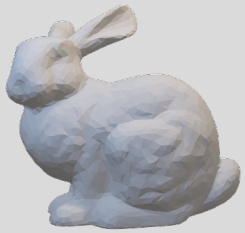
Rotation

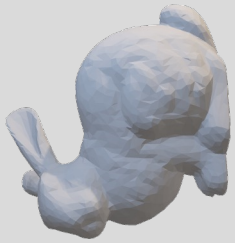
$R_{xy}(\theta)$: rotation about z axis by θ

$$R_z = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Homogeneous Coordinates





$\theta = 90°$

$$v = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

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

$$v' = R_z v$$

why this is so beautiful ?

we can represent translation as a matrix, we can compose it with other transformations

Example: translate 10 units down the Z then rotate 90° around X

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 10 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(90°) & -\sin(90°) & 0 \\ 0 & \sin(90°) & \cos(90°) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

why this is so beautiful ?

we can represent translation as a matrix, we can compose it with other transformations

Example: translate 10 units down the Z then rotate 90° around X

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 10 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 10 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

WHY THIS IS SO BEAUTIFUL ?

we can represent translation as a matrix, we can compose it with other transformations

Example: translate 10 units down the Z then rotate 90° around X

Caution: matrix multiplication is NOT commutative!

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 10 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 10 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

World Coordinates

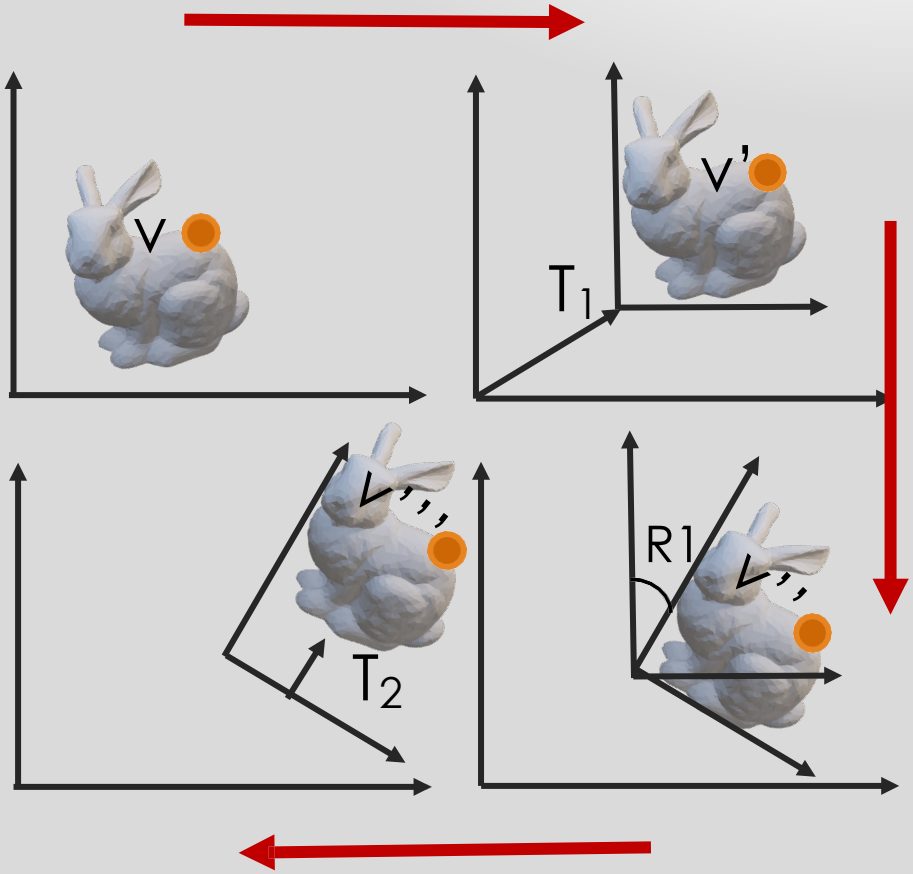
🎮 **We have been discussing transformations as transforming points.**

🎮 **Always need to think of the transformation in the world coordinate system.**

🎮 **Useful to think of them as a change in coordinate system.**

🎮 **Model objects in a local coordinate system, and transform into the world system.**

World Coordinates



$$v'' = T_1 v$$
$$v''' = T_1 R_1 v$$
$$v'''' = T_1 R_1 T_2 v$$

- Concatenate local transformation matrices from left to right
- Can obtain the local world transformation matrix
- v',v'',v''' are the world coordinates of p after each transformation

Transformations

Quiz

🕹 I sat in the car, and realized the side mirror is 0.4m on my right and 0.3m in my front

🕹 I started my car and drove 5m forward, turned 30 degrees to right, moved 5m forward again, and turned 45 degrees to the right, and stopped

🕹 What is the position of the side mirror now, relative to where I was sitting in the beginning?

The side mirror position is locally (0.4,0.3)

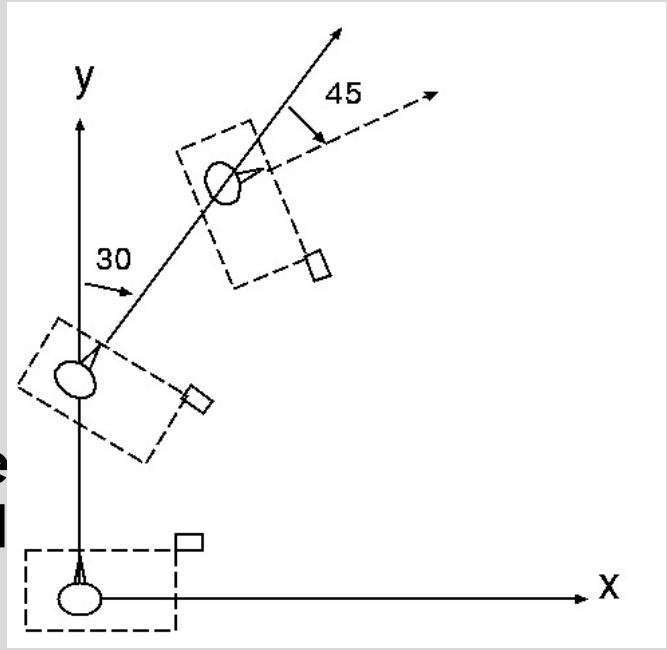
Transformations

Quiz

The local-to-global transformation matrix at the last configuration of the car is

$$TR_1TR_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 5 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{\sqrt{3}}{2} & \frac{1}{2} & 0 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 5 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The final position of the side mirror can be computed by TR1TR2 p which is around (2.89331, 9.0214)



3 Building Blocks of 3D

Modelling

Lighting

Texturing

3 Building Blocks of 3D

Done

Lighting

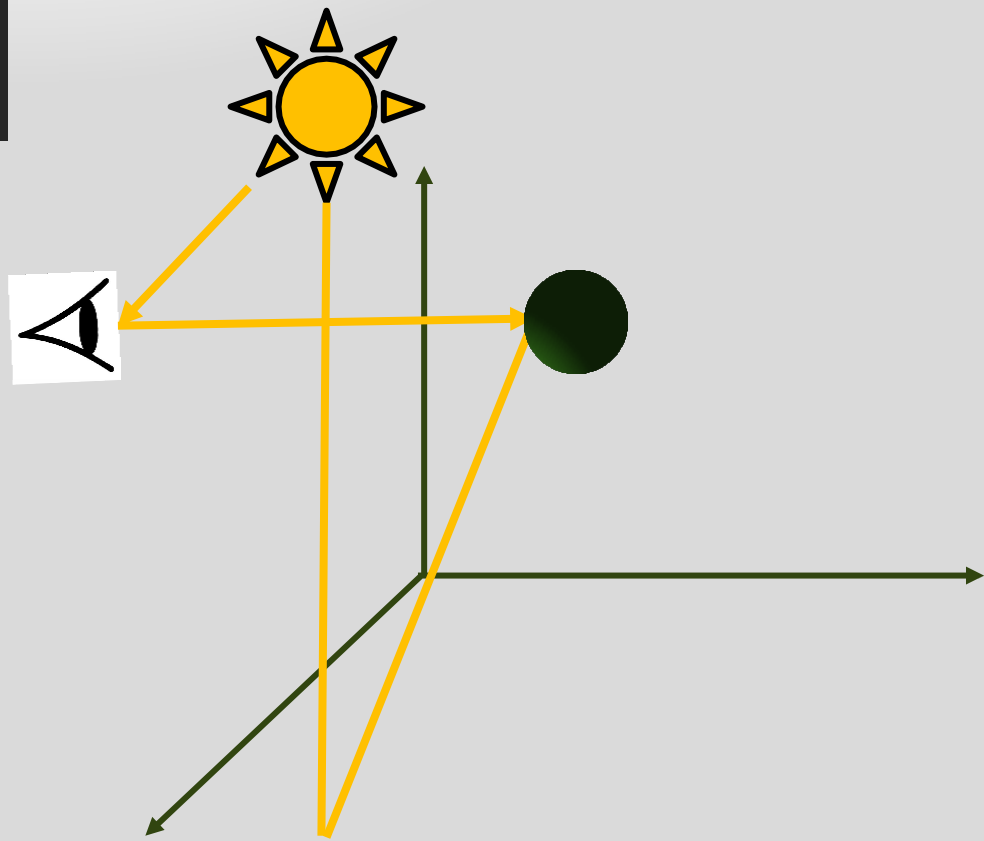
Texturing

Lighting

Lighting and Shading

lighting equation

Emissive+
Ambient +
Diffuse +
Specular+



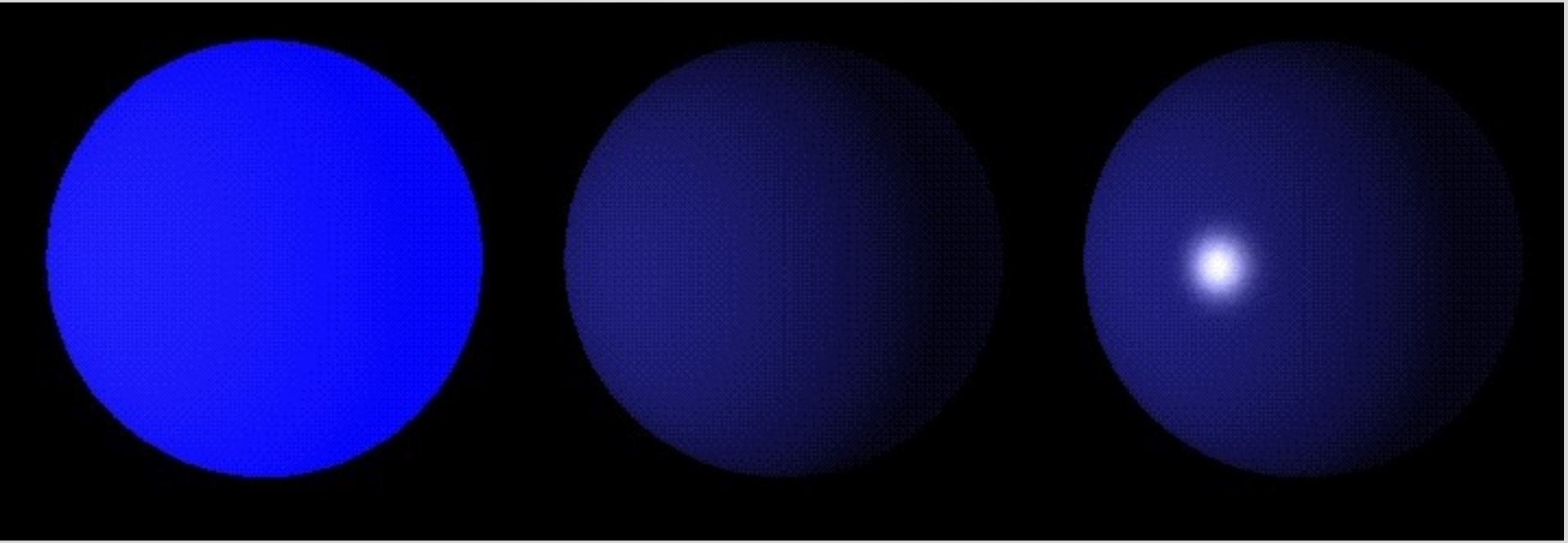
lighting Model

https://www.gabrielgambetta.com/computer-graphics-from-scratch/light.html

Lighting and Shading

Emissive

Lighting and Shading

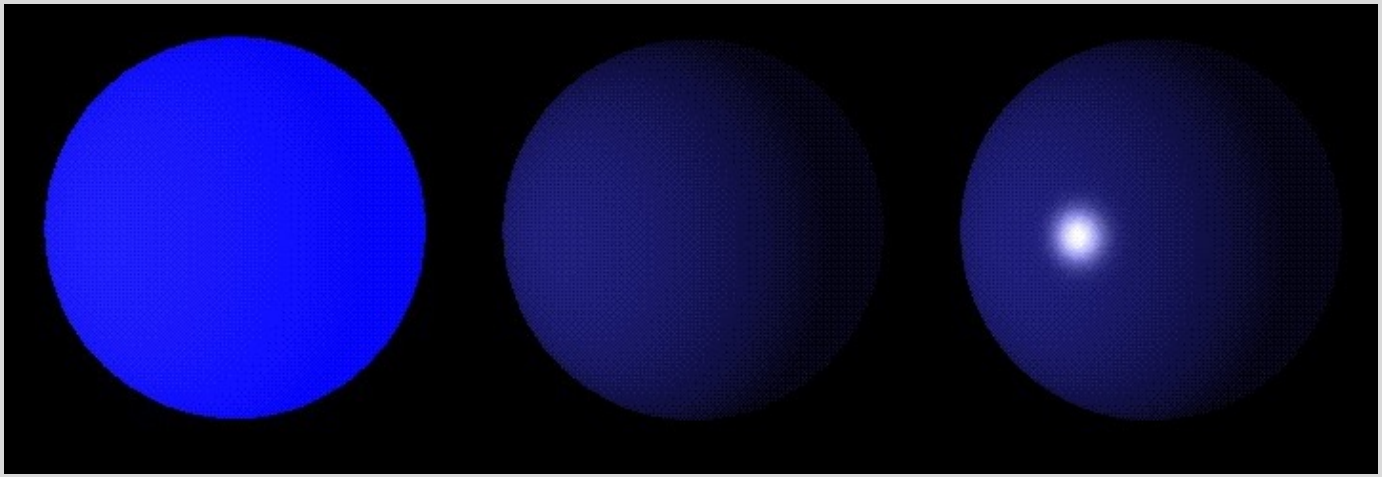


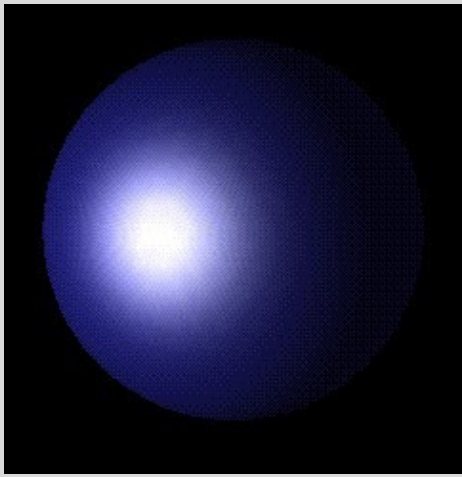
Ambient

Diffuse

Specular

Lighting and Shading





Ambient | Diffuse | Specular | Result

Lighting and Shading

lighting equation

The diffuse and specular components are computed based on the lights in the scene, while emissive and ambient are essentially independent.

The diffuse term can be thought of as a flat matte finish, and specular can be thought of as the shine finish of an object.

In terms of computation, the specular term is also affected by the camera angle.

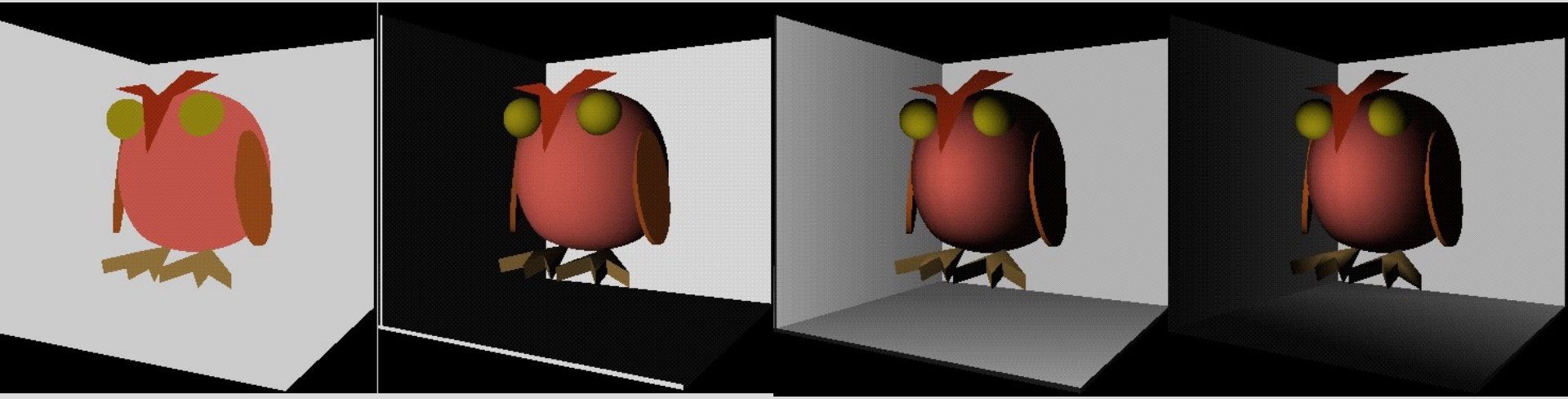
Lighting and Shading

Types of light sources

- **Ambient light:** no identifiable source or direction.
- **Point source:** given only by point.
- **Directional light:** given only by direction.
- **Spotlight:** from source in direction.
  - Cut-off angle defines a cone of light.
  - Attenuation function (brighter in centre).

Lighting and Shading

Types of light sources



Ambient light | Directional light | Point source | Spotlight

Lighting and Shading

Shading Models

- Flat Model
- Gouraud Model
- Phong Model

Lighting and Shading

Flat Model

We define a normal at each polygon (not at each vertex)

Lighting: Evaluate the lighting equation at the center of each polygon using the associated normal

Shading: Each sample point on the polygon is given the interpolated lighting value at the vertices (i.e., a total hack)

Lighting and Shading

Gouraud Model

Define a normal vector at each *vertex*

Lighting: Evaluate lighting equation at each vertex using associated normal vector

Shading: Each sample point's color on polygon is interpolated from color values at polygon's vertices

An issue with Gouraud shading is that specular highlights appear to "jump" as the object rotates.

Lighting and Shading

Phong Model

Each vertex has an associated normal vector

Lighting: Evaluate lighting equation at each vertex using associated normal vector

Shading:

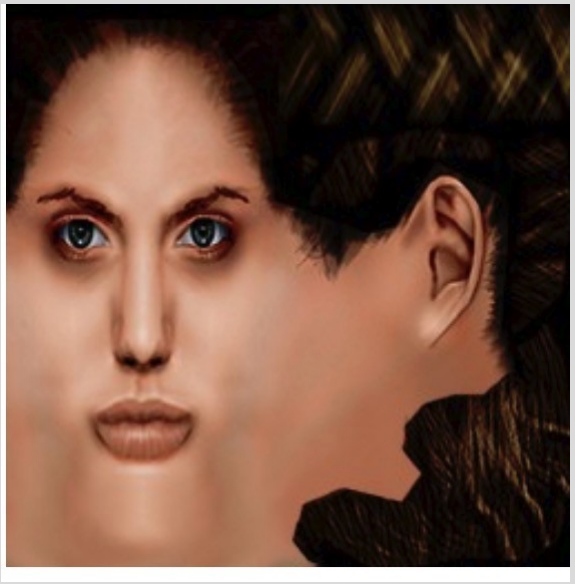
For every sample point on the polygon we interpolate normals at vertices of the polygon and compute color using lighting equation with the interpolated normal at each interior pixel

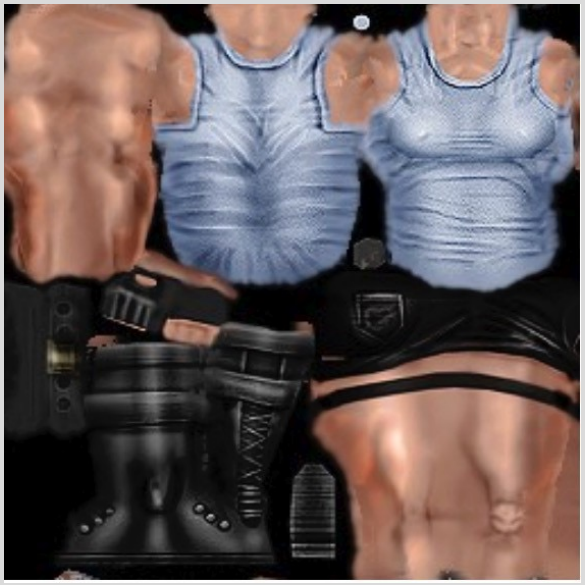
Lighting and Shading

Flat Model	Gouraud Model	Phong Model

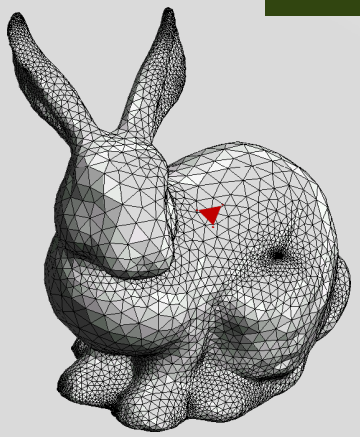
Texturing

Texturing

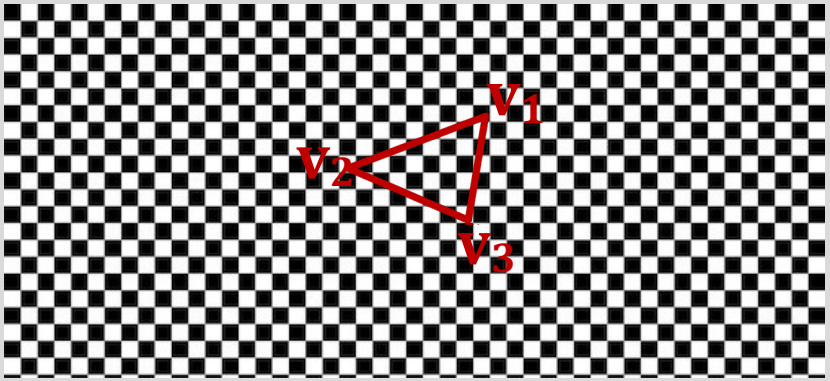




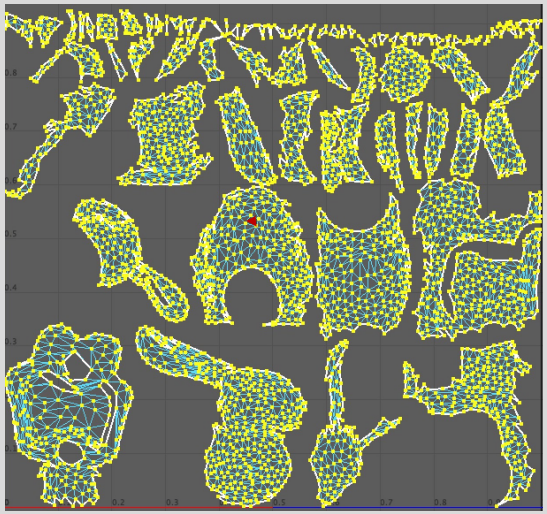
Texturing



- For each triangle in the model establish a corresponding region in the phototexture



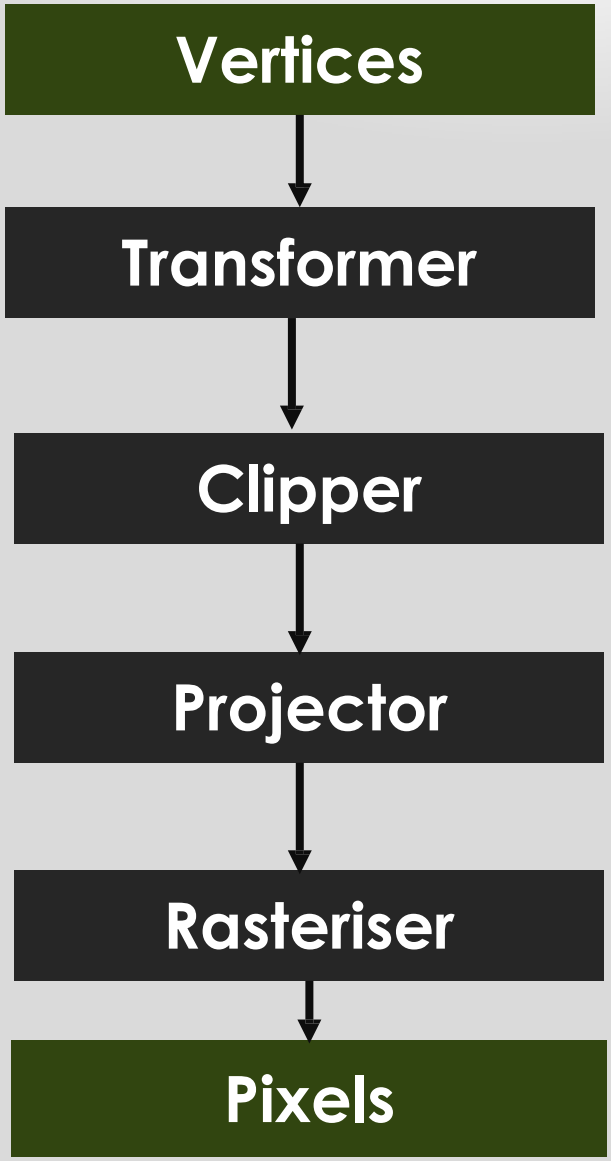
2D image



Texture map

- During rasterization interpolate the coordinate indices into the texture map

3D Graphics Pipeline



3D model, scale, rotate, translate, lighting, shading, texturing

is it visible on screen?

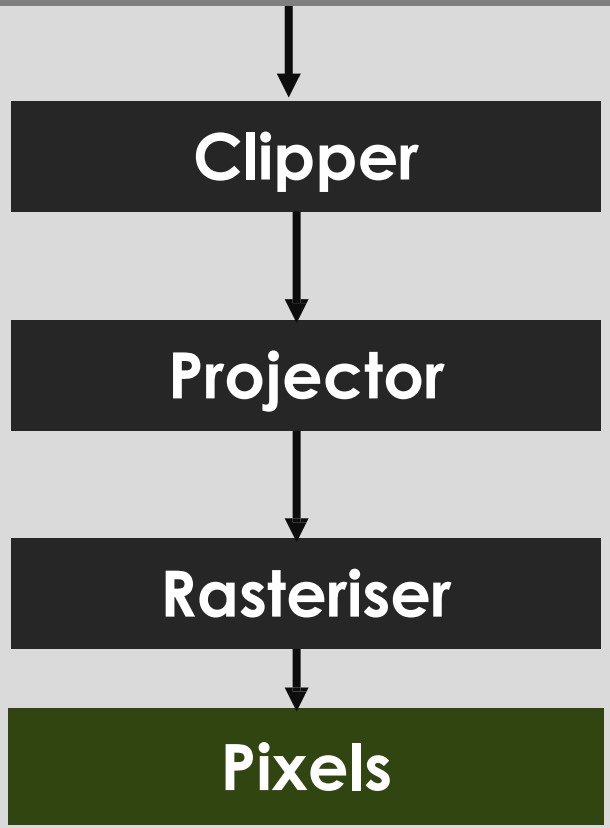
3D to 2D

convert to pixel

Viewing and Projection

Done

3D graphics pipeline



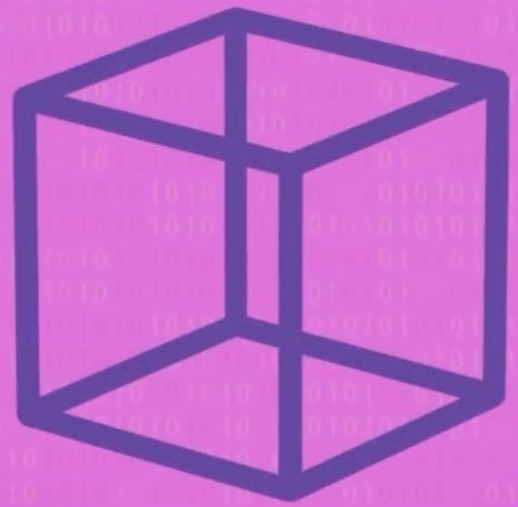
is it visible on screen?

3D to 2D

convert to pixel

Viewing and Projection

Projection

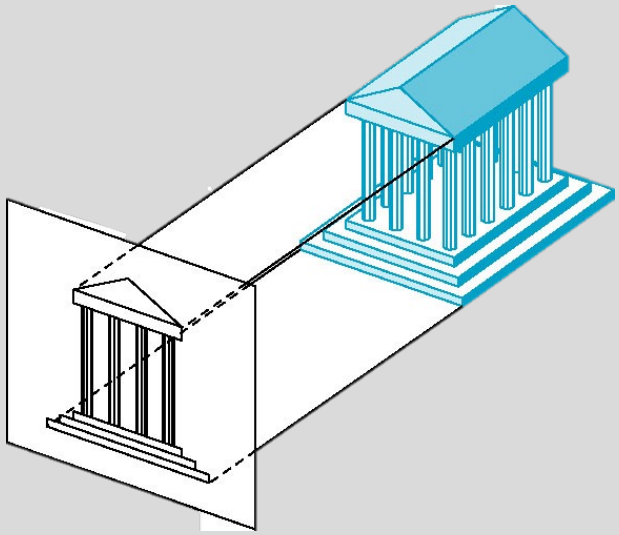


Viewing and Projection

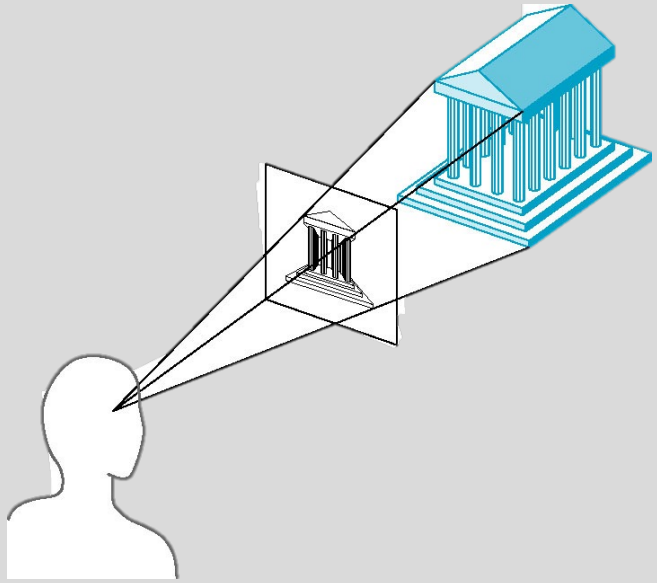
Projection

Complex transformation [Angel Ch. 5]

Orthographic Projection



Perspective Projection



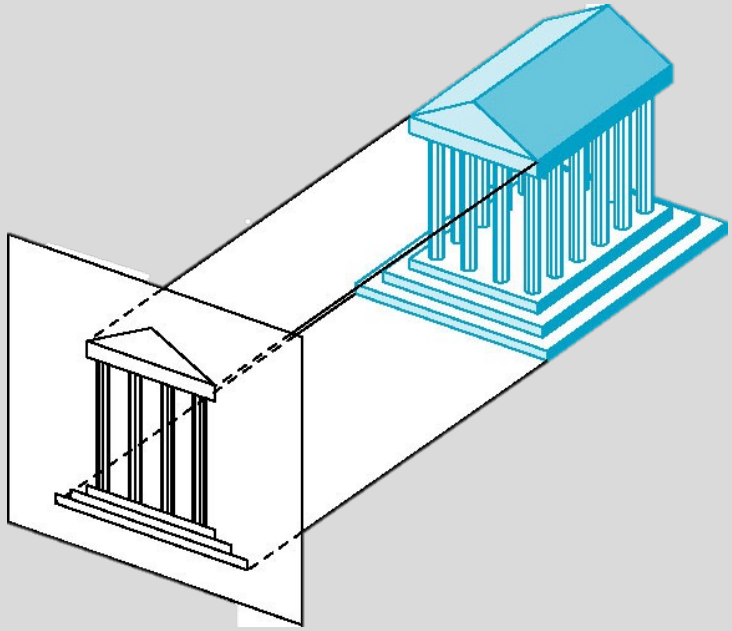
Viewing and Projection

Projection

Orthographic Projection

- projection that maintains parallel lines but provides no sense of depth.

- It is used in the engineering fields to create accurate renderings of models. They maintains parallel lines but provide no sense of depth. All vertices are projected straight onto a viewing window



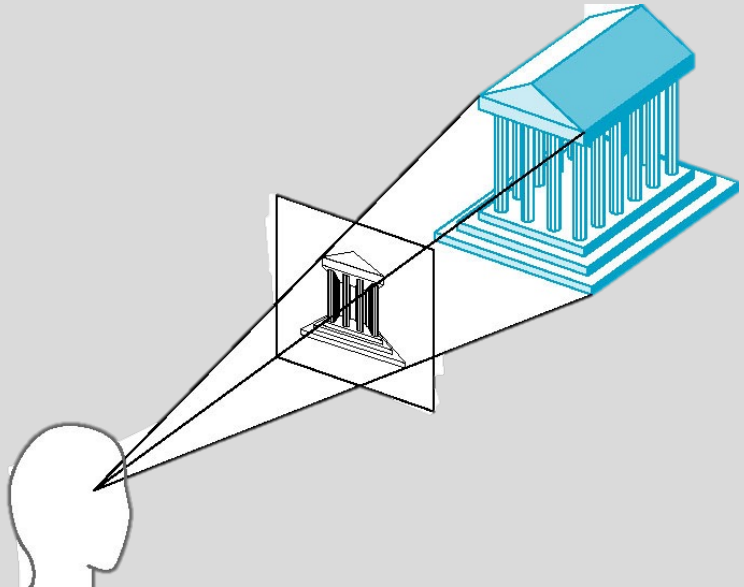
Viewing and Projection

Projection

Perspective Projection

- projection provides for a sense of depth, but parallel lines are skewed toward vanishing points

- a virtual scene to make it appear like a view from a real-world camera. Objects further from the camera appear to be smaller and all lines appear to project toward vanishing point which skew parallel lines. Perspective projections are almost always used in gaming, movie special effects, and visualizations of virtual worlds.

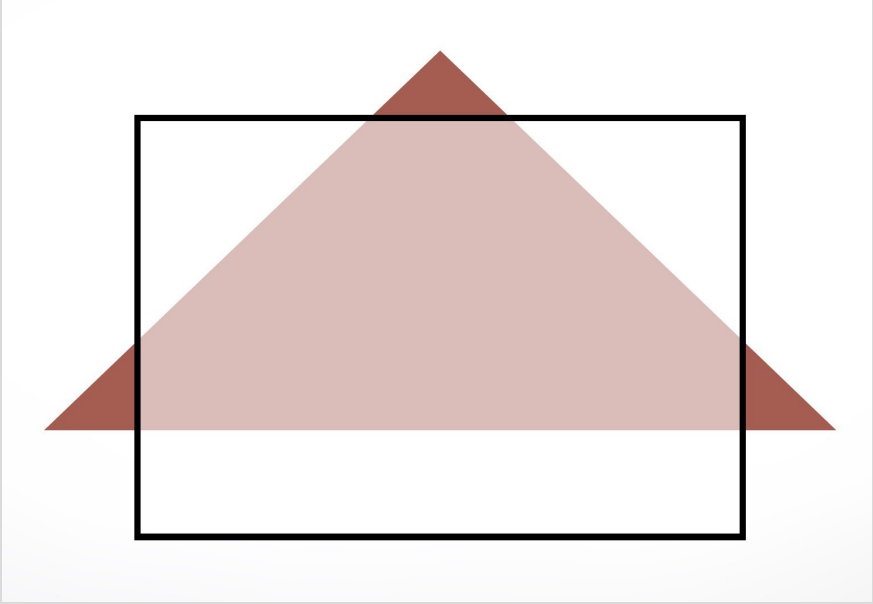


Viewing and Projection

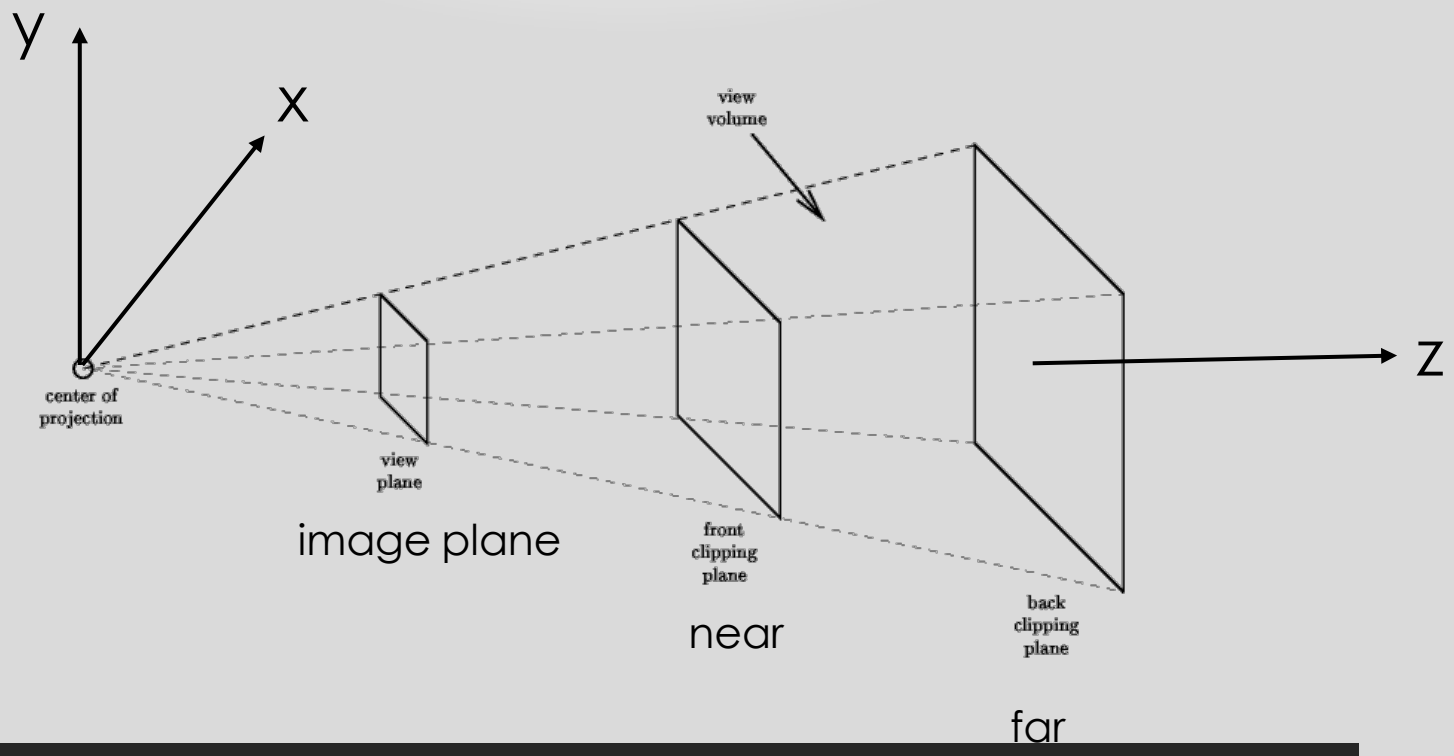
Clipping

Mostly automatic (must set viewport)

2D Clipping Problem



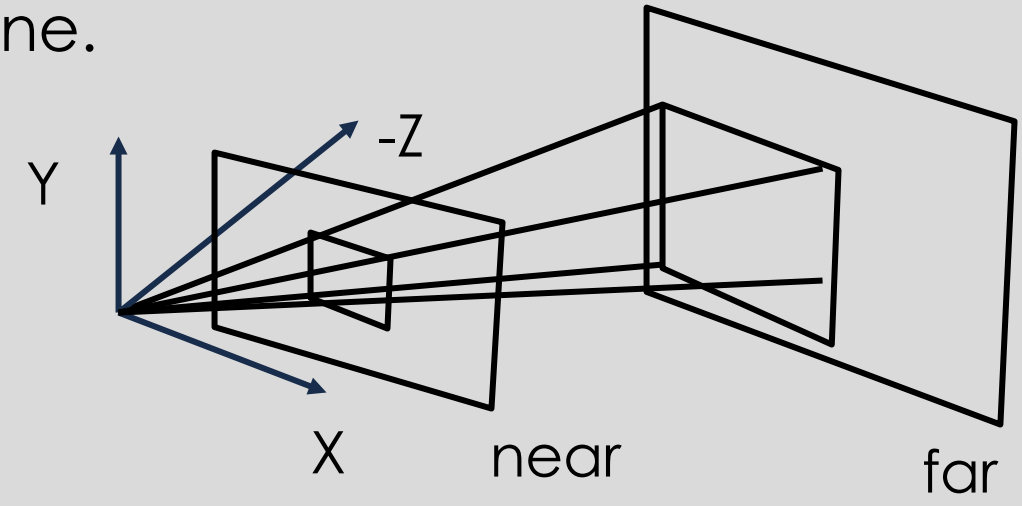
Viewing and Projection



Clipping is tricky because of frustum shape

Viewing and Projection

- Define the "view frustum" (6 parameters).
- Assume origin is the view point.
- Near and far planes (planes parallel to XY plane in the negative Z axis).
- Left, right, top, bottom rectangle defined on the near plane.



Rendering

is the automatic process of generating a photorealistic or non-photorealistic image from a 2D or 3D model

Rendering

The rendering equation is an integral equation based on light and is very computationally demanding

Aliasing

Rendering

The rendering equation is an integral equation based on light and is very computationally demanding

Aliasing

Aliasing: the process by which smooth curves or lines become jagged due to the resolution of the graphics device or insufficient data to represent the curves

Rendering

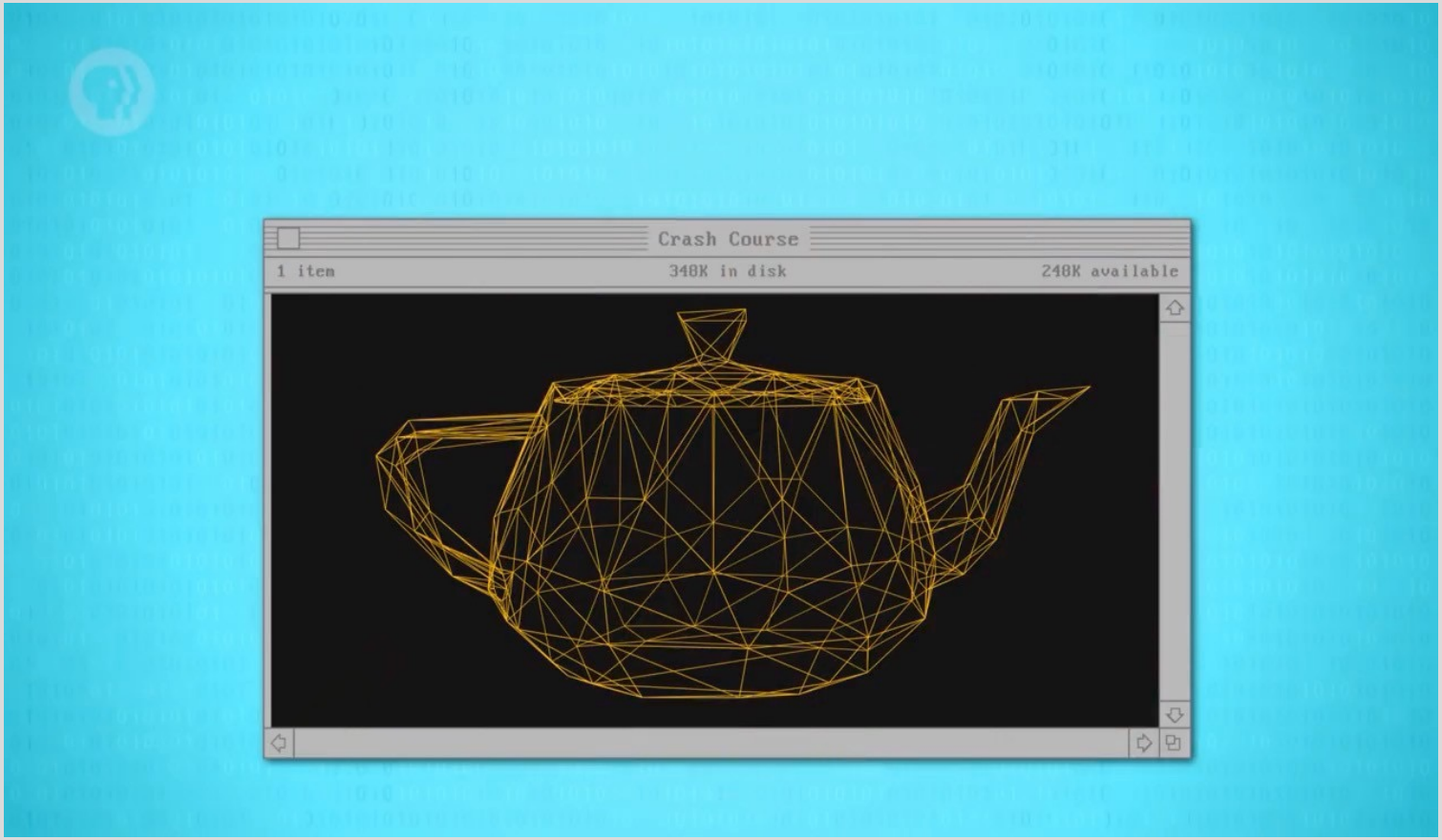
The rendering equation is an integral equation based on light and is very computationally demanding

Aliasing

Anti-aliasing is a technique for minimizing the distortion artefacts

Rendering

The rendering equation is an integral equation based on light and is very computationally demanding



https://www.youtube.com/watch?v=TEAtmCYYKZA

CS3005: DIGITAL MEDIA AND GAMES

Questions?

Email: George.Ghinea@brunel.ac.uk