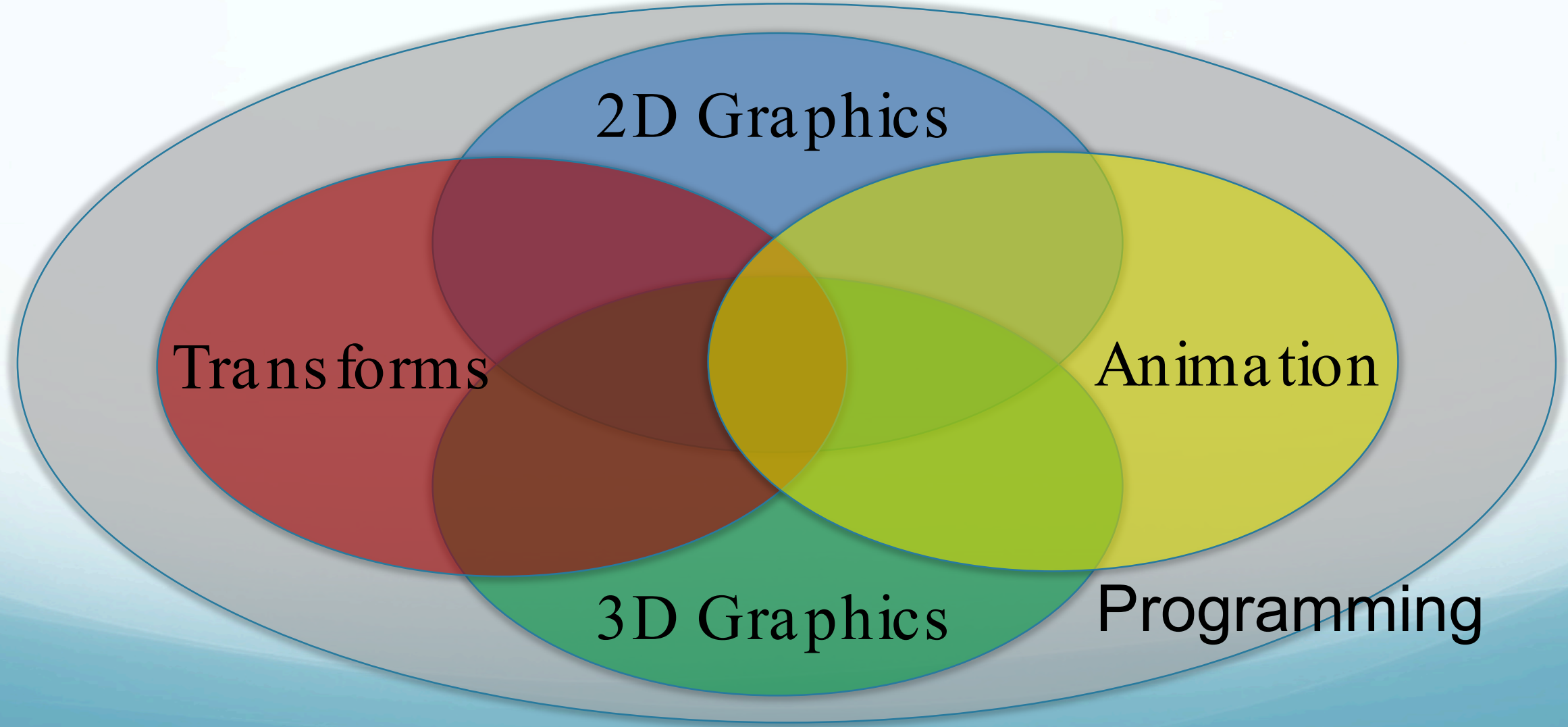
CSC 240 Computer Graphics
Video 22: Final Review

Nick Howe
Smith College

Some slides & content courtesy Sara Mathieson

Themes

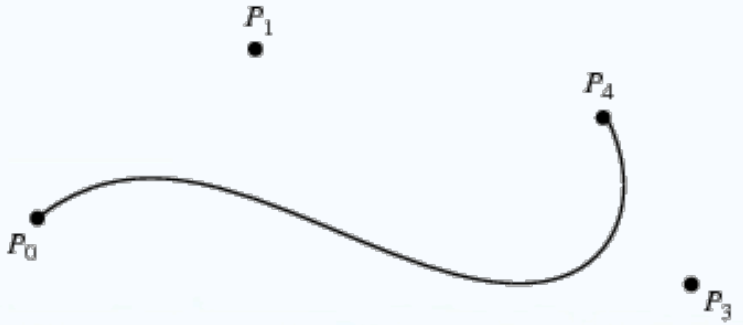

2D Graphics
Transforms
Animation
3D Graphics
Programming

Pre-Midterm Topics

Rasterization — Pixel origin conventions

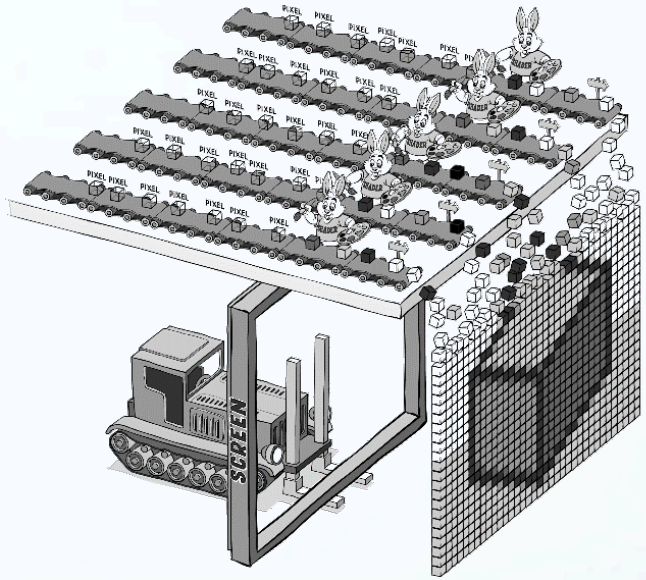
- Line drawing — Midpoint, Incremental, Anti-aliased
- Fill algorithms — Flood fill, Scan fill

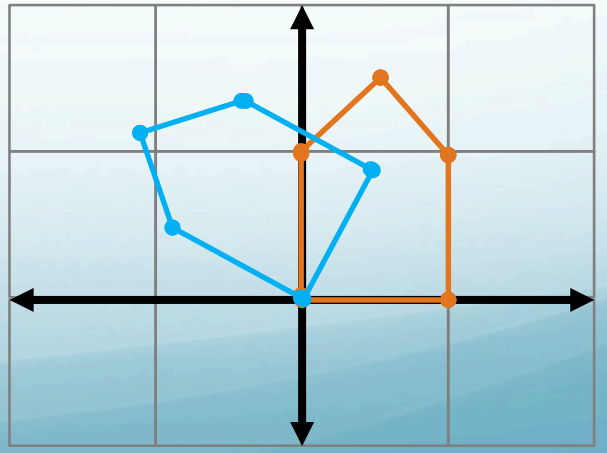
Bezier curves & splines



Transformations

- Standard transform matrices
- Homogeneous coordinates
- Matrix multiplication





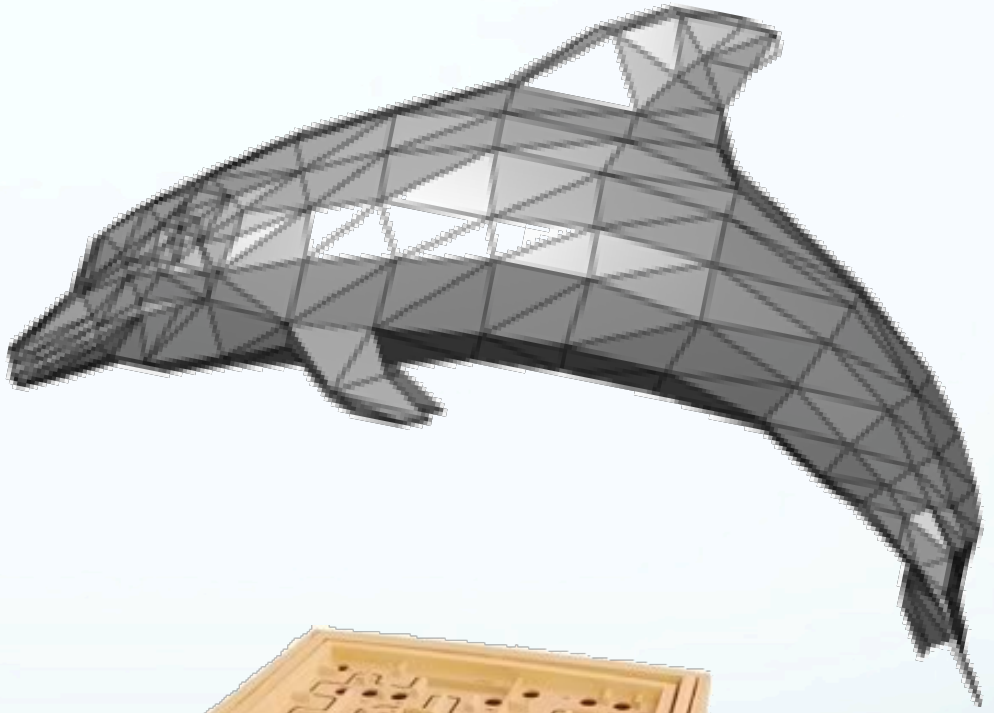
Objects in 3D

How to define

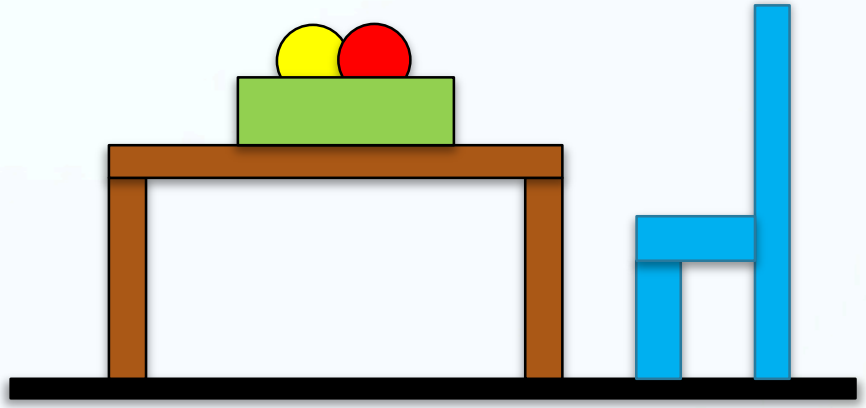
- Vertices
- Faces / triangular mesh
- Normal vectors & right-hand rule

Transformations in 3D

- Use in WebGL – Matrix4 defined transforms
- Use in animation



Hierarchical Models

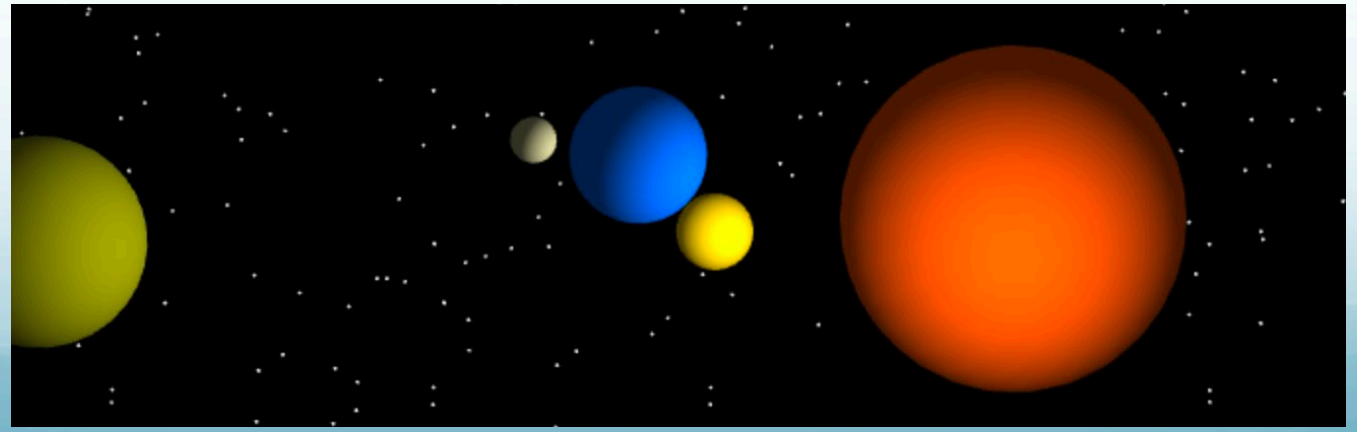


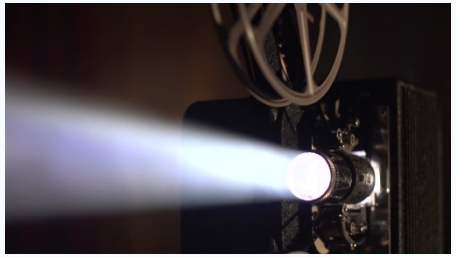
Motivation for hierarchical models

- How to design a hierarchy

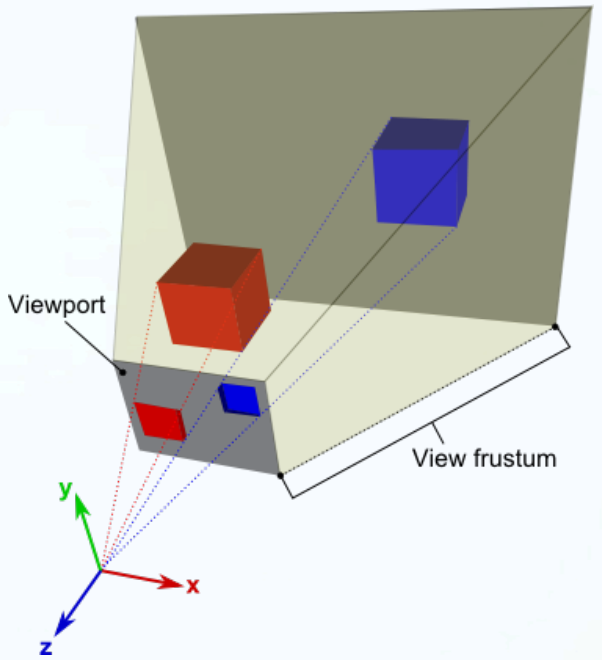
Application

- How transformations apply on hierarchy tree
- Using container objects in WebGL





Projection

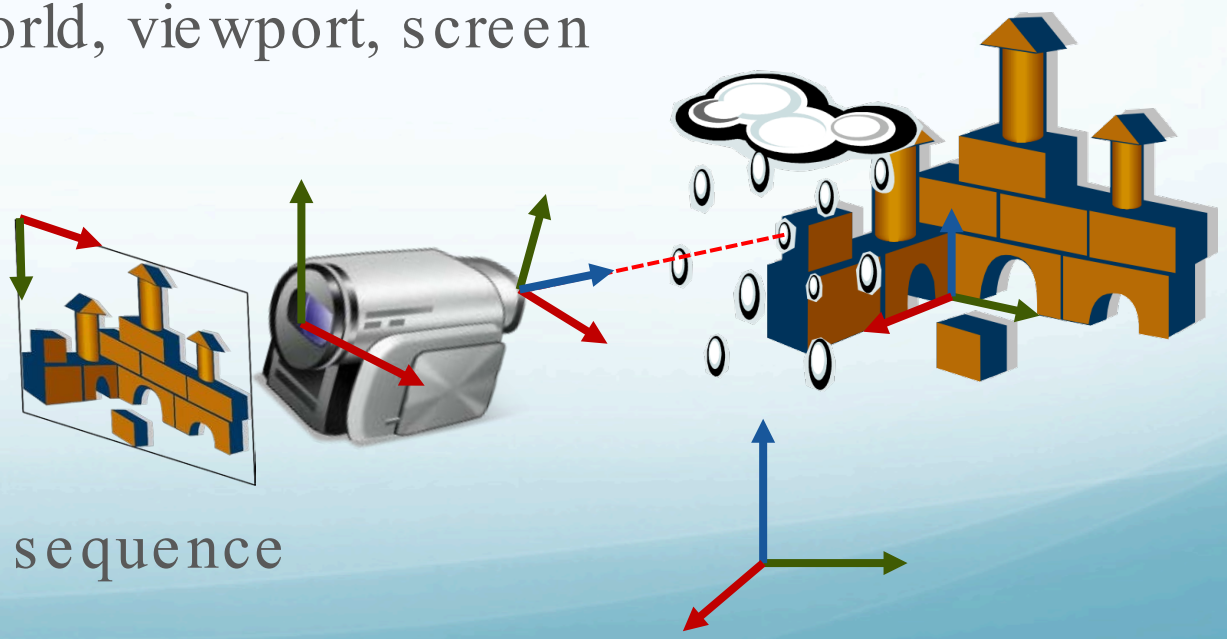


What's important about projection?

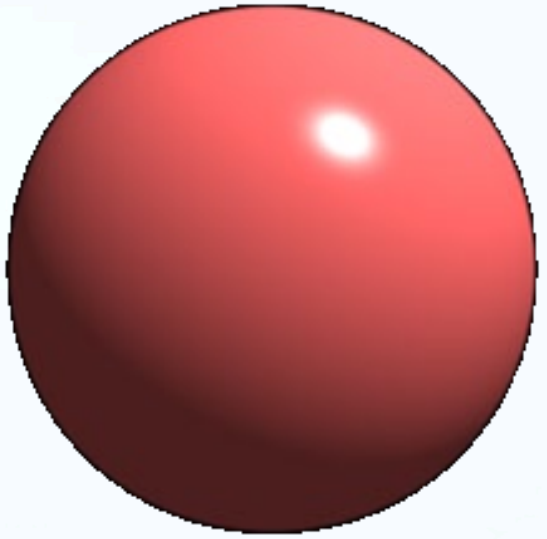
- 3D setup: scene, camera, viewport, view frustum
- Corresponding WebGL implementation
- Coordinate system conversion: world, viewport, screen

Main projection types & their math

- Perspective
- Orthographic
- Coordinate axes & transformation sequence



Shading & Lighting

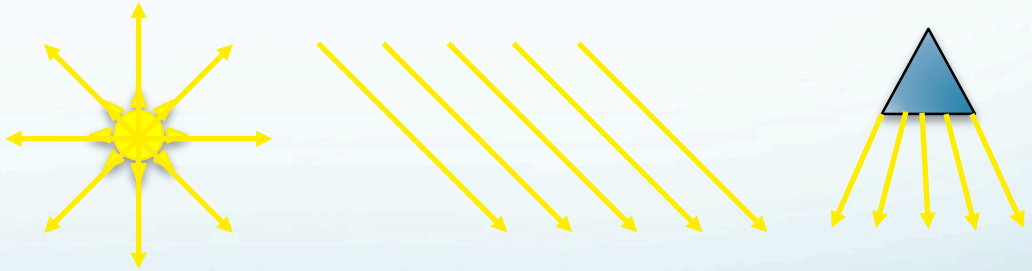


Light source concepts & interaction with objects

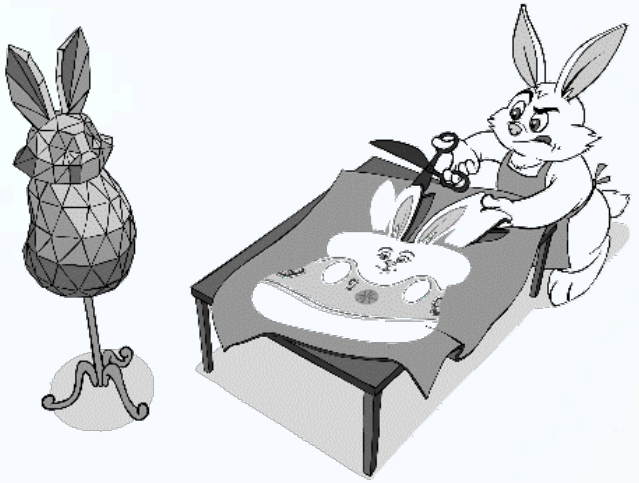
- Normal vectors, light vectors, dot products
- Diffuse and specular shading: Lambertian vs. Phong
- Three.js: four illumination modes

Types of light sources

- Ambient, point, directional, spot
- WebGL implementation in Three.js

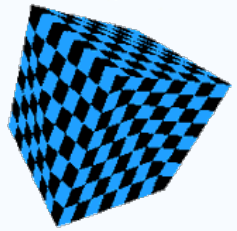


Texture Mapping



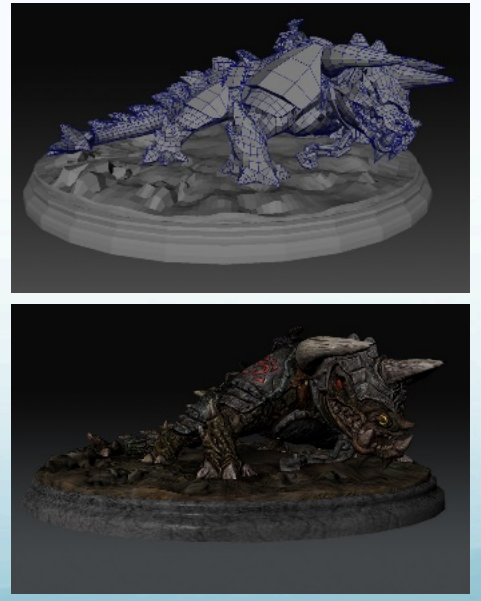
Main concept: 3D vertices → 2D UV coords

- How to specify & use in WebGL
- Perspective-correct mapping



Additional mapping applications

- Bump maps and normal maps
- Shadow mapping
- "Baking" high-poly to low-poly



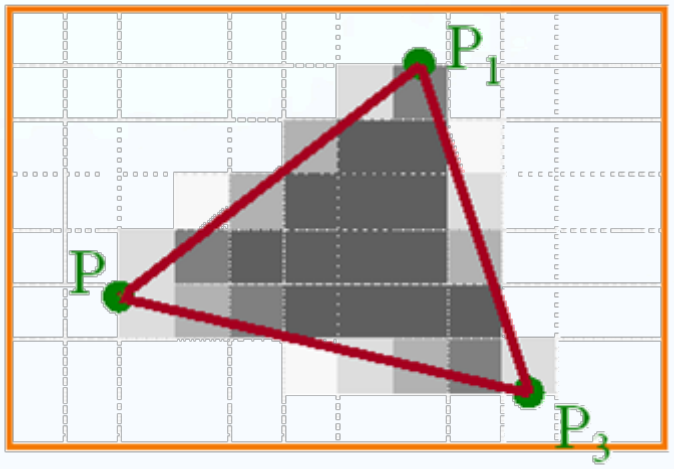
http://ithare.com/game-graphics-101-textures-uv-mapping-and-texture-filtering/

Rendering

Shading pipeline

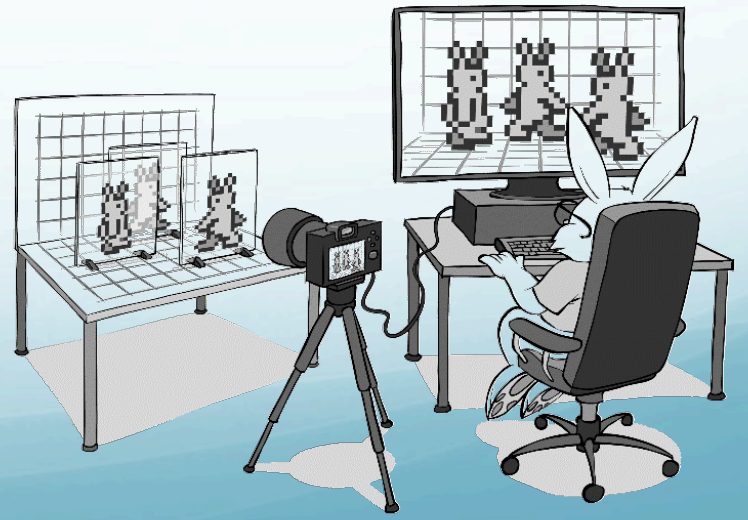
- Vertex shaders
- Fragment shaders
- Clipping & culling

Half-triangle fill
Barycentric coordinates



Depth rendering & hidden surface removal

- Painter's algorithm
- Z-buffer algorithm

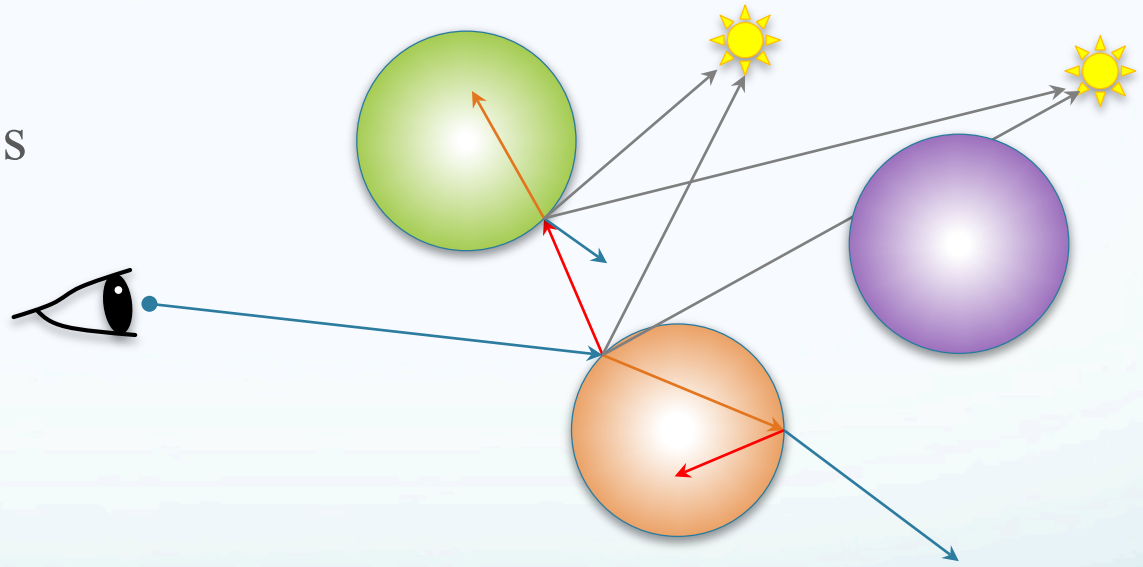


https://www.cs.princeton.edu/courses/archive/fall00/cs426/lectures/scan/sld002.htm
http://ithare.com/game-graphics-101-rendering-2d-on-gpu-shaders/

Ray Tracing

Ray casting

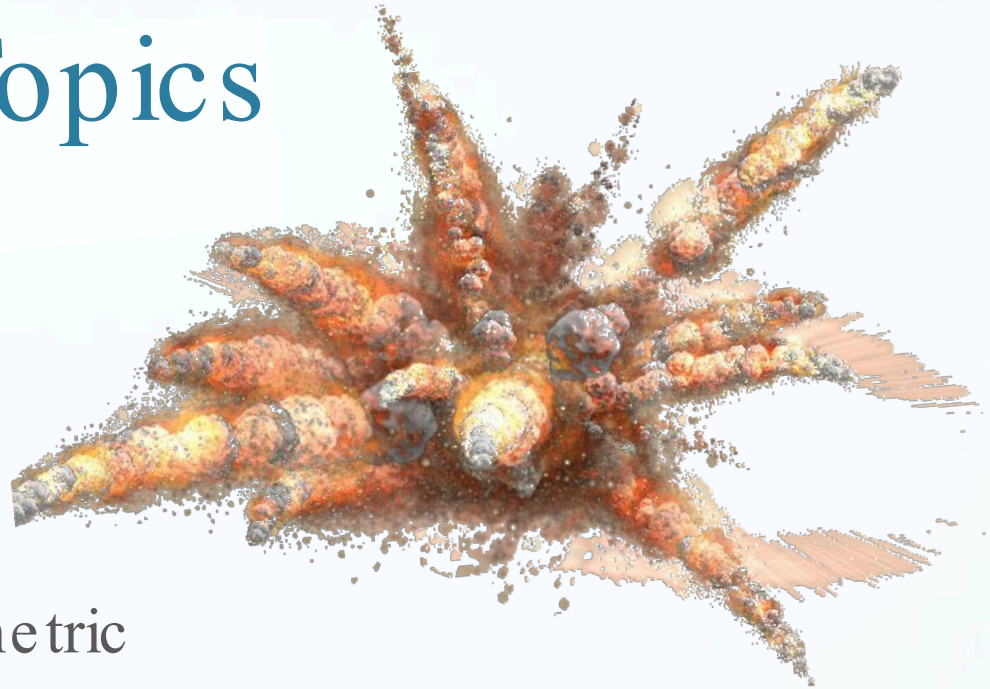
- Ray representation & math
- Intersection with spheres & triangles
- Use for collision detection



Ray tracing

- Viewport scanning & pixel math
- Recursive tracing: transmission, reflection, shadows
- Stochastic & multiray methods

Advanced Topics



Particle systems & their stages

- Emission
- Simulation: physics, etc.
- Rendering: static/animated, discrete/volumetric

Subsurface scattering

- Depth map method
- Texture diffusion method

https://kaitlynplyley.com/2012/07/18/disneys-brave-new-hair/

https://colinbarrebrisebois.com/2011/03/07/gdc-2011-approximating-translucency-for-a-fast-cheap-and-convincing-subsurface-scattering-look/


Thank you!

Video 1A Review

After this video, you should be able to:

- Identify applications of computer graphics
- Distinguish between raster and vector image formats
- Define terms: **pixel** and image **resolution**
- Use RGB triplets to represent different colors
- Describe the axis configuration for standard screen coordinates
- Work with points in both pixel center origin and corner origin conventions

Video 1B Review

After this video, you should know how to:

- Build a basic HTML page that incorporates a graphics canvas
- Identify open/close tag pairs in HTML & check proper nesting
- Define functions and variables in Javascript
- Write simple loops and conditions in Javascript
- Color individual pixels of the graphics canvas in arbitrary colors

Video 2 Review

After this video, you should know how to:

- Give pseudocode for a simple line algorithm, and implement it if need be
- Given endpoints of a line, determine the proper loop for rendering it
- Carry out by hand the calculations for the simple line algorithm
- Give pseudocode for the midpoint line algorithm
- Explain the advantages of the midpoint algorithm over the simple one
- Define antialiasing and how it applies to drawing lines.

Video 3 Review

After this video, you should know how to:

- Define simple, complex, convex, and concave polygons
- Compute the center "pizza slice" angle of an n-sided regular polygon
- Compute the coordinates of a vertex given the center point and angle
- Draw a sequence of lines in Javascript
- Write a function to draw a regular polygon

Video 4 Review

After watching this video, you should be able to…

- Define the 2D fill operation
- Determine 4-connected and 8-connected regions
- Design a recursive function with stop condition & simplification
- Implement a recursive 2D fill algorithm
- Pseudocode & simulate a sweep-based 2D fill algorithm
- Explain the advantages of the sweep-based fill.

Video 5 Review

After watching this video, you should be able to…

- Determine the magnitude and direction of a given vector
- Generate a unit vector from an ordinary vector, or from a 2D angle
- Recognize a matrix and note its dimensions
- Perform both addition and scalar multiplication on matrices and vectors
- Perform and implement matrix multiplication, where possible
- Envision vectors as a combination of unit basis vectors.

Video 6 Review

After watching this video, you should be able to…

- Define a transformation and describe why they are useful
- Describe 6 major transformation types qualitatively
- Express 6 major transformation types numerically as a matrix
- Convert a regular 2D vector into homogeneous coordinates
- Convert a 2x2 transformation matrix into a 3x3 homogenous equivalent
- Compose transformations by multiplying their matrices

Video 7 Review

After watching this video, you should be able to…

- List three types of coordinate systems used in graphics
- Describe the transforms used to relate the different coordinate systems
- Recognize that the same effect can be achieved through either transform
- Know how the current transform affects objects added to a scene
- Manipulate the current transform to achieve desired effects
- Understand that transforms accumulate through object part hierarchies

Video 8 Review

After watching this video, you should be able to…

- Define a Bézier curve of any order
- Compute the point on a Bézier curve at position t.
- Define a spline curve and give two example types
- Identify the necessary conditions for a Bézier spline to be smooth
- Construct a cubic spline given control points & a polynomial curve fitter
- Recognize other applications of splines

Video 9 Review

After watching this video, you should be able to…

- Explain the motivation for line clipping
- Compute Cohen-Sutherland endpoint codes, given a point & viewport
- Use the codes to determine whether a segment is visible, and/or whether clipping is required
- Clip line segments as needed according to the viewport boundaries
- Demonstrate the Sutherland-Hodgman polygon clipping algorithm by hand

Video 10 Review

After watching this video, you should be able to…

- List factors that influence the formation of images
- Identify different coordinate systems used during 3D rendering
- Define two types of projections & express as projection matrices
- Describe the view frustum and its purpose
- Project points numerically from 3D to 2D under both orthographic and perspective projection

Video 11 Review

After watching this video, you should be able to…

- Set up a web page for 3D rendering using Three.js
- Add a camera, lights, and visible objects to your scene
- Make informed choices about the different available options
- Create custom object geometries and combine them with materials
  - Write code to define vertices and combine them into faces
  - Control the directionality of triangular faces

Video 12 Review

After watching this video, you should be able to…

- Write a callback function to perform animation in Three.js
- Express 3D transformations mathematically as 4D homogeneous matrices
- Express 3D rotation as a composition of R_x, R_y, and R_z component rotations
- Apply translation, scaling and rotation to 3D objects in Three.js
- Use `lookAt` as an alternative to component rotations

Video 13 Review

After watching this video you should be able to…

- Propose sensible groupings of objects and object parts
- Build scenes as hierarchies of Object3D
- Employ object hierarchies to efficiently produce a desired animation

Video 14 Review

After watching this video you should be able to…

- Define a BRDF and explain how it is measured
- List and describe the four types of light transmission in Three.js
- Compute how geometry interacts with light position in diffuse shading
- Describe and apply three shading algorithms, along with their different schemes for handling surface normal
- Understand how Three.js computes the final color of a pixel
- Write Three.js programs that control shading to achieve desired effects

Video 15 Review

After watching this video you should be able to…

- Describe the stages of the graphics pipeline & what each does
- Implement a half-triangle fill algorithm for a polygon shader
- Compute barycentric coordinates and use them to find weighted averages
- Map textures to objects using a texture image and *uv* coordinates
- Explain the advantages of a texture mipmap
- Implement texture mapping in Three.js

Video 16 Review

After watching this video you should be able to…

- Explain the need for perspective correction in texture mapping
- Compute perspective-corrected UV coordinates
- Understand the normal map and bump map formats
- Use normal maps and/or bump maps to produce textured shading effects
- Describe the shadow mapping algorithm
- Carry out shadow computations in a simple model

Video 17 Review

After watching this video, you should be able to…

- List three major strategies for hidden surface removal
- Identify polygons outside the view or that face away from a camera
- Understand the z buffering algorithm and its limitations
- Apply offsets in Three.js to mitigate z fighting
- Perform basic operations in Blender
- Work with models from 3D archives

Video 18 Review

After watching this video, you should be able to…

- Define ray tracing and describe how it differs from ordinary rendering
- Give pseudocode & computations for a simple raycasting algorithm
- Understand recursive ray tracing and how to follow light rays in a scene
- Compute the formula for the ray that travels between two given endpoints
- Compute the intersection between a ray and a plane

Next time: more intersections

Video 19 Review

After watching this video, you should be able to…

- Compute the intersection of a ray and a sphere
- Compute the intersection of a ray and a triangle
- Use a simple technique to rule out intersections with many objects at once
- Compute the new direction vector of a mirror reflection
- Compute the new direction vector of a transmitted ray with refraction
- Describe how to improve image quality by tracing additional rays

Video 20 Review

After watching this video, you should be able to…

- Give two reasons for detecting collisions
- Describe two simple methods for checking possible collisions
- Describe and implement an algorithm for detailed 3D collision checks
- Identify important elements in any successful game

Video 21 Review

After watching this video, you should be able to…

- Define subsurface scattering and identify cases where it is important for realism
- Describe two techniques for simulating subsurface scattering in rendered images
- Define a particle system in graphics and list three common uses
- Describe how the state model and its update governs the behavior of a particle
- Explain how similar mathematical processes can generate very different effects such as fire, water, hair, and cloth