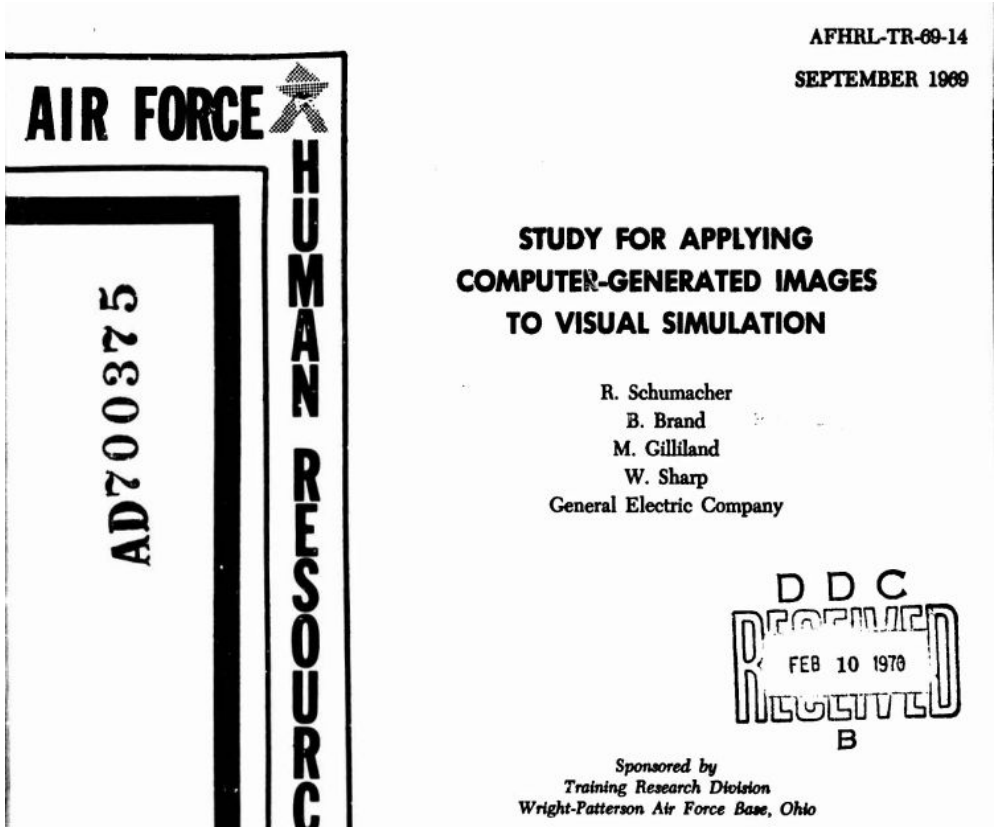
Scene Rendering Concepts

csc249

Problem Statement

Render a first person perspective of a scene.

This means placing a virtual camera within an environment and simulating light ray bounce (to various levels of detail) from the scene.



AIR FORCE HUMAN RESOURC

AD700375

AFHRL-TR-69-14

SEPTEMBER 1969

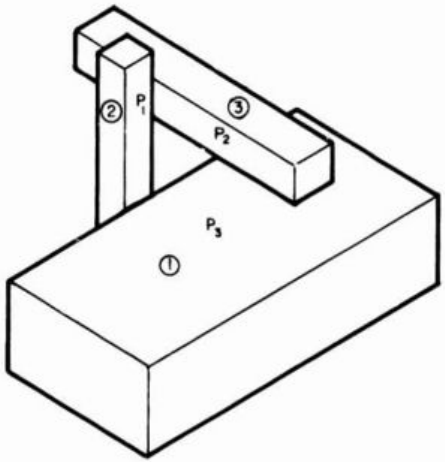
STUDY FOR APPLYING COMPUTER-GENERATED IMAGES TO VISUAL SIMULATION

R. Schumacher
B. Brand
M. Gilliland
W. Sharp
General Electric Company

DDC
RECEIVED
FEB 10 1970
B

Sponsored by
Training Research Division
Wright-Patterson Air Force Base, Ohio

Early analysis (for flight sims) proposed entirely custom hardware.



i \ j	1	2	3
1	O	p_1	$\bar{p}_3$
2	$\bar{p}_1$	O	p_2
3	p_3	$\bar{p}_2$	O

Original Matrix

i \ j	1	2	3
1	O	1	O
2	O	O	1
3	1	O	O

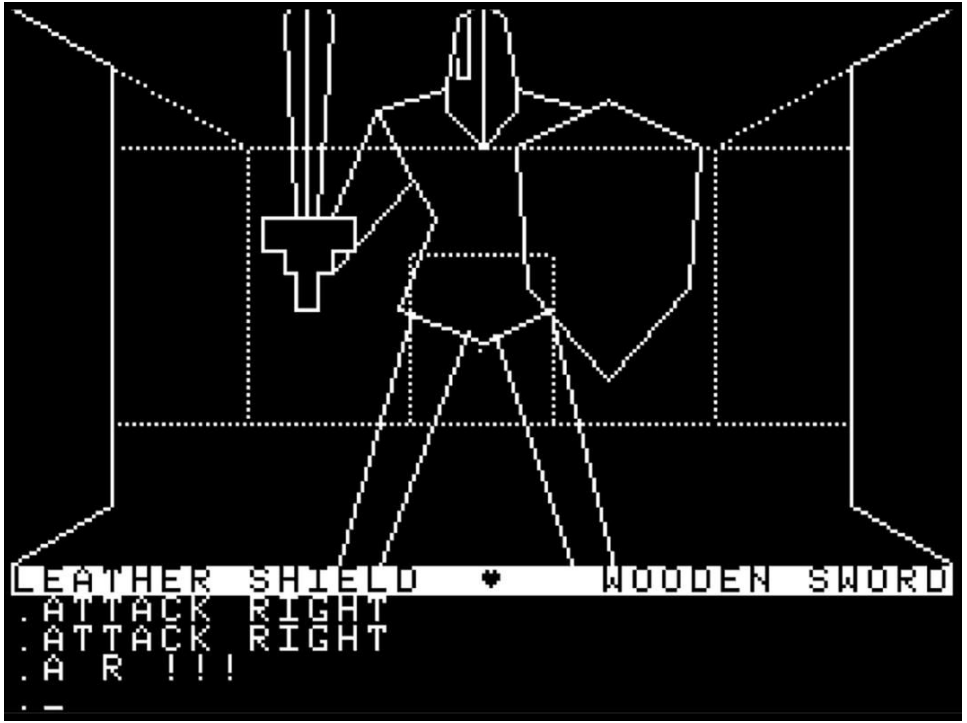
Matrix Evaluated for Vantage Point Shown

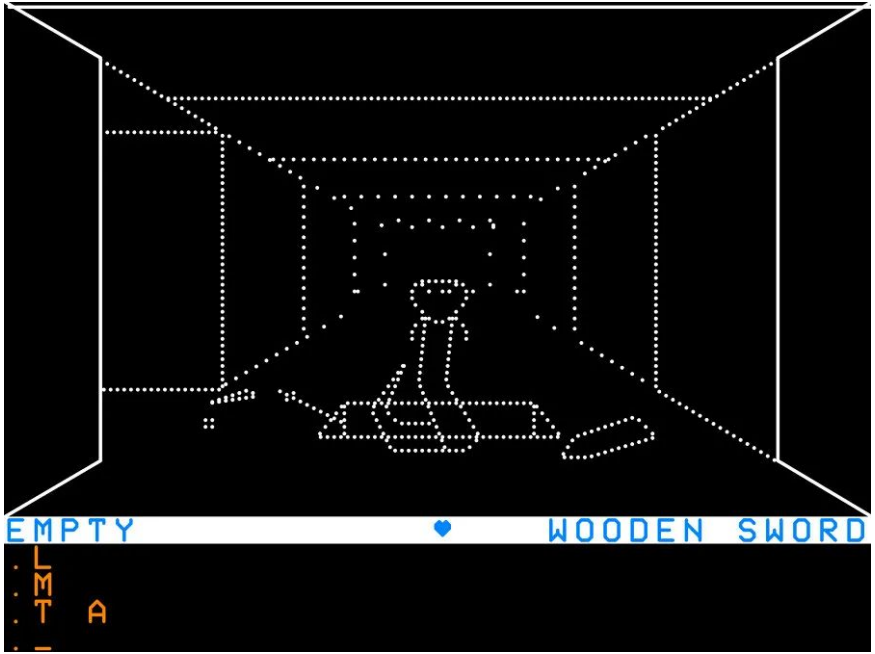
Figure 9. Priority Situation and Matrix

Matrix Multiplication – still used to find normal vectors today.

1981 - 3D Monster Maze (ZX Spectrum), ASM/BASIC.

Disassembly: http://www.fruitcake.plus.com/Sinclair/ZX81/Disassemblies/MonsterMaze.htm





1982 - Dungeons of Daggorath (TRS-80 CoCo)
http://computerarcheology.com/CoCo/Daggorath/Code.html
Vector graphics

Algorithms

(each with their own pluses and minuses)

- Painter's Algorithm
- Raytracing
- Raycasting
- Warnock's Algorithm
- Binary Space Partitioning with Z-Buffer

Types and Considerations

z-order rendering (Back to front vs. Front to back)

-> Objects must **occlude** (cover) objects behind them

Abstraction of the behavior of light

-> Most only deal with surface color

-> Dynamic vs. "Baked-In" lighting is not covered here

Painter's Algorithm

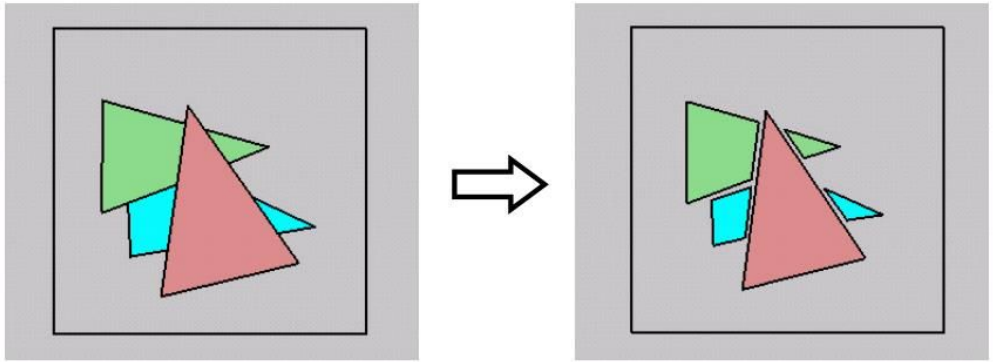
(1960s - 1970s)

z-order: Back to Front

Maximally inefficient – massive overdraw

Painter's Algorithm

- Simple approach: render the polygons from back to front, "painting over" previous polygons:



  - Draw blue, then green, then pink
- *Will this work in general?*

Painter's Algorithm

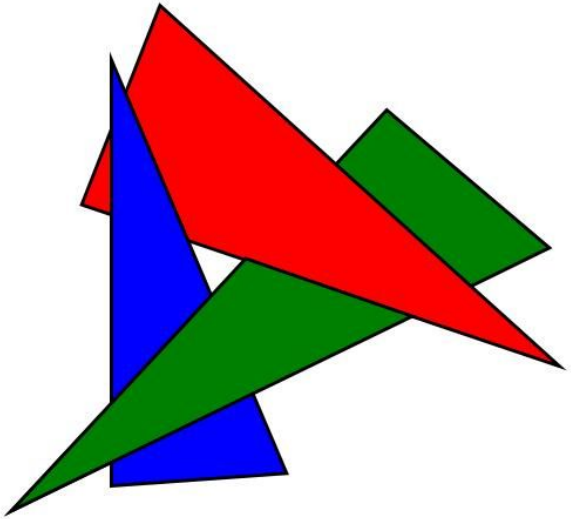
(1960s - 1970s)

z-order: Back to Front

Maximally inefficient – massive overdraw

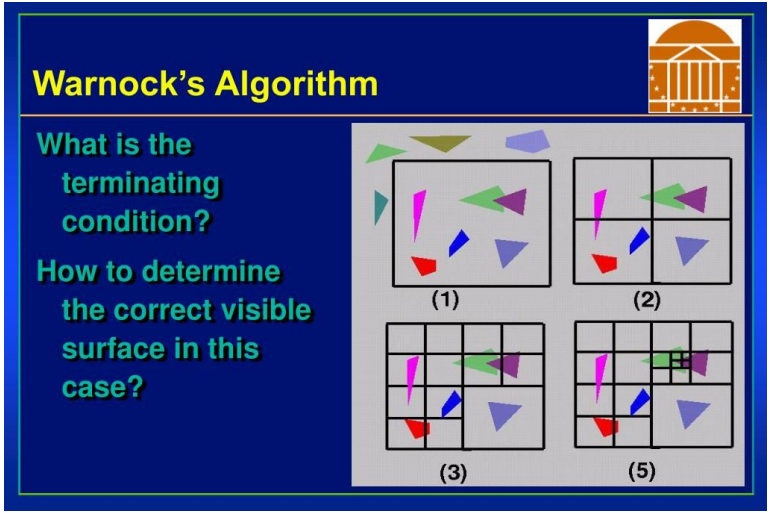
Painter's Algorithm

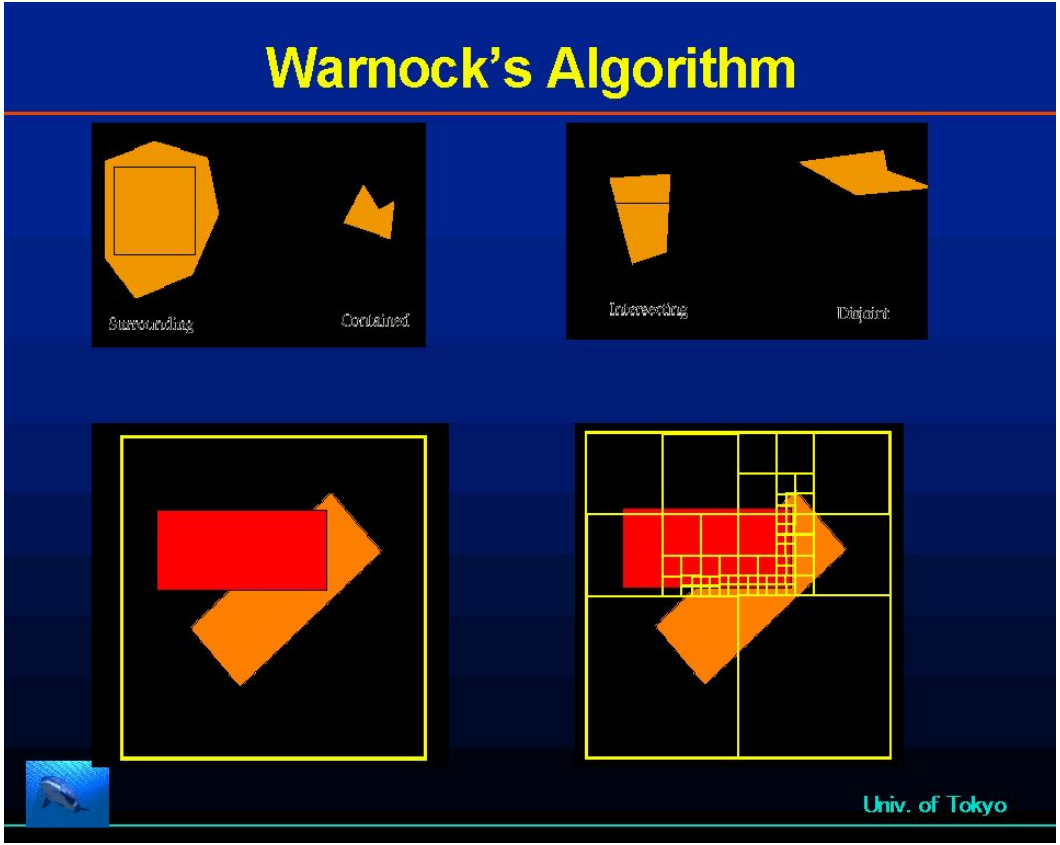
- Sometimes there is NO ordering that produces correct results!!!



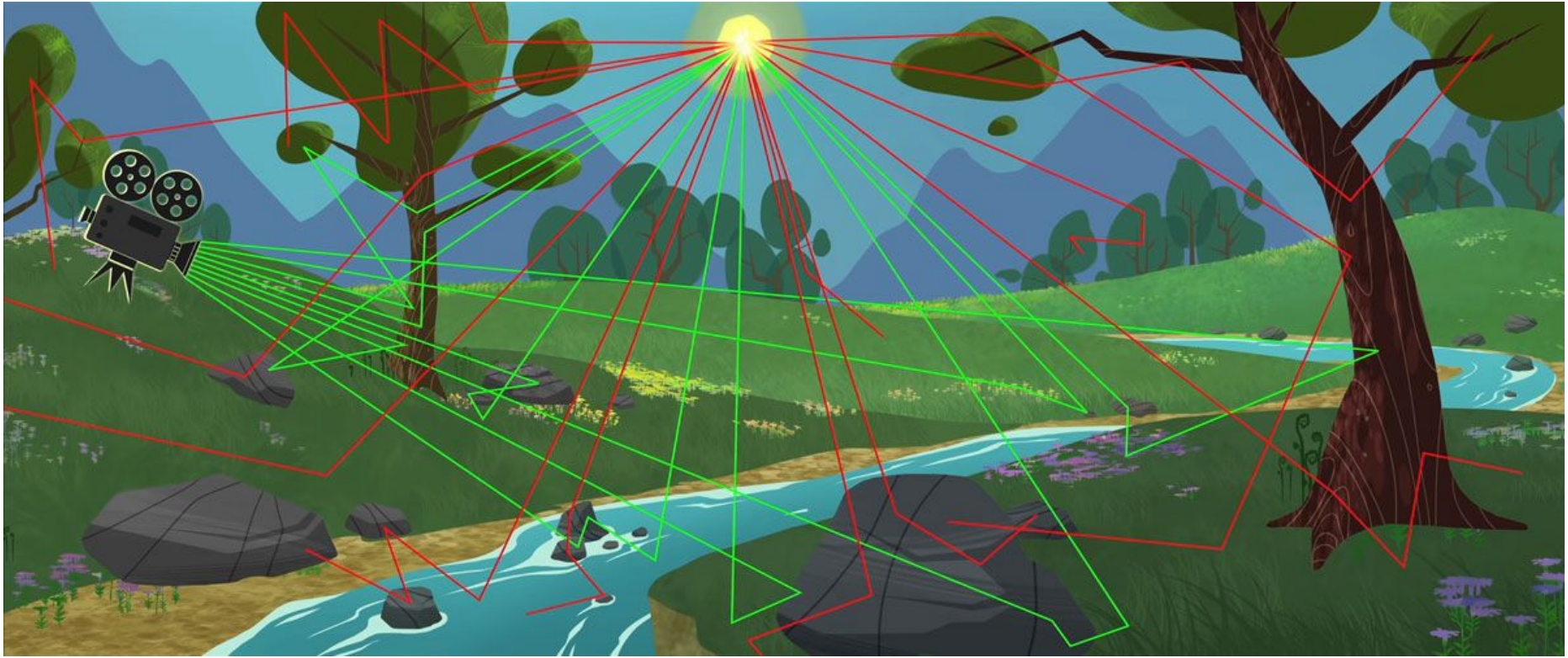
Warnock's Algorithm (1969)

- elegant scheme based on a powerful general approach common in graphics: *if the situation is too complex,* ***subdivide***
  - start with a *root viewport* and a list of all primitives (polygons)
  - then recursively:
    - clip objects to viewport
    - if number of objects incident to viewport is zero or one, visibility is trivial
    - otherwise, subdivide into smaller viewports, distribute primitives among them, and recurse





Warnock's may potentially subdivide to subpixel size



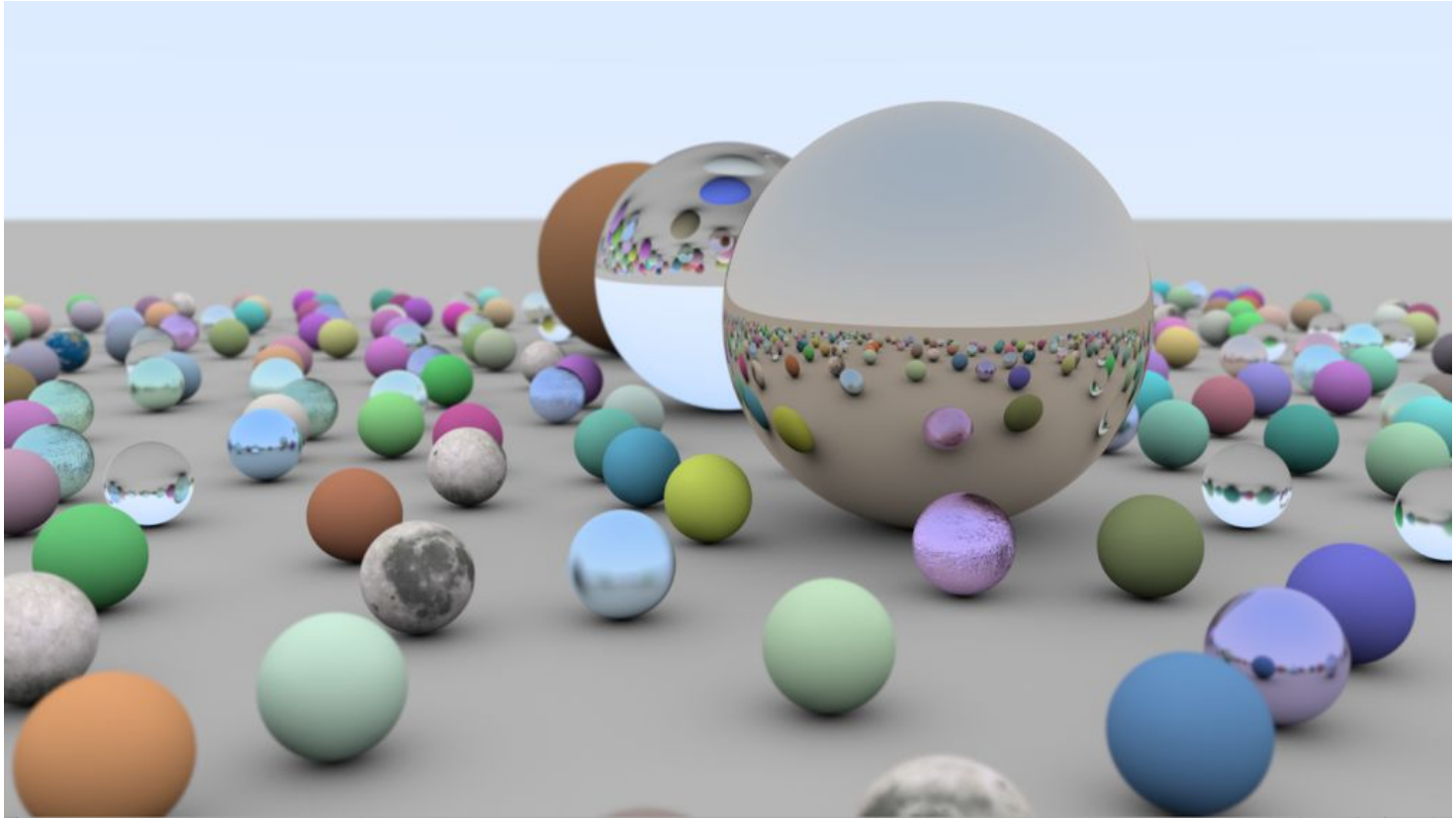
Raytracing - Simulate light ray bounce

Raytracing

For each **pixel** in the camera's view:

- Trace a **light ray** out to infinity,
- Reflect or refract the light ray as needed (may bounce many times)

Until very recently (2020+), not used in real-time applications



Raytracing Example

Raycasting (per-pixel)

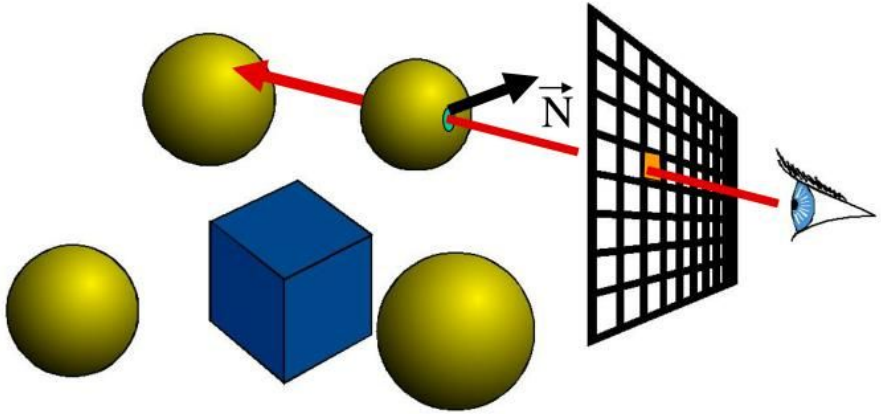
For each **pixel** in the camera view:

- Cast one ray into a two-dimensional scene
- Record the distance and material the ray collides with
- Render the pixel column based on these factors.

Definitions vary, but generally raytracing involves simulating reflection, transparency, and light scatter – per-pixel raycasting does not.

Ray Casting

```
For every pixel
  Construct a ray from the eye
  For every object in the scene
      Find intersection with the ray
      Keep if closest
   Shade depending on light and normal vector
```



Finding the intersection and normal is the central part of ray casting

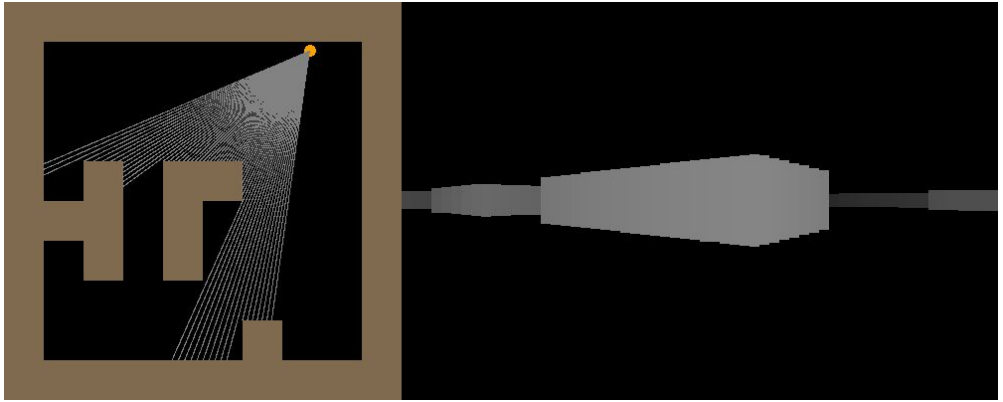
Raycasting (per-column)

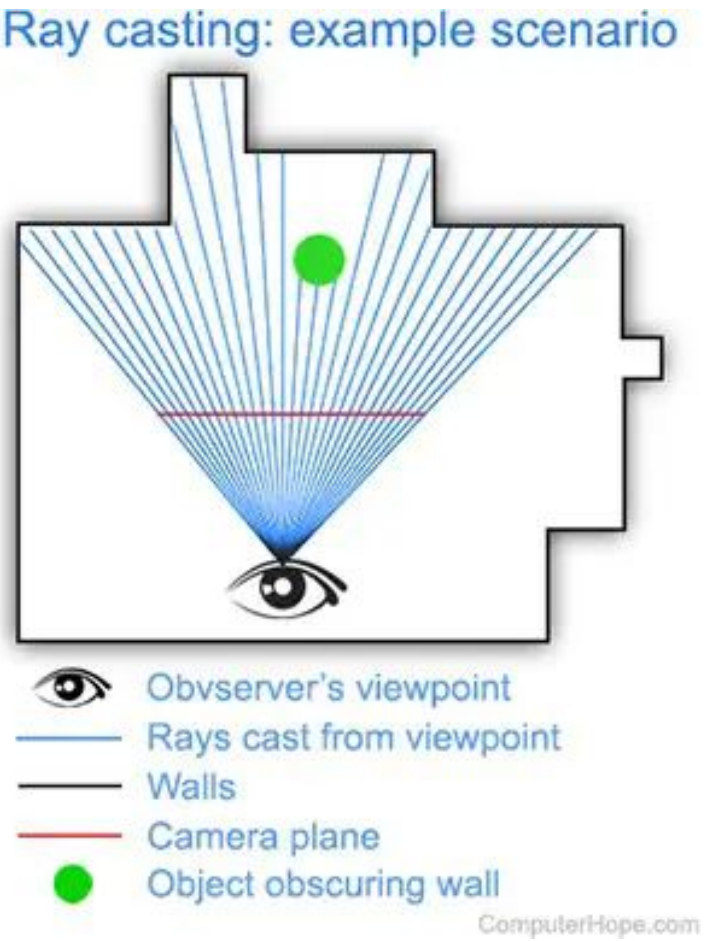
For each **pixel column** in the camera view:

- Cast one ray into a two-dimensional scene
- Record the distance and material the ray collides with
- Render the pixel column based on these factors.

Much faster than per-pixel, as the calculations occur once per column (Y-pixel) of the view area.)

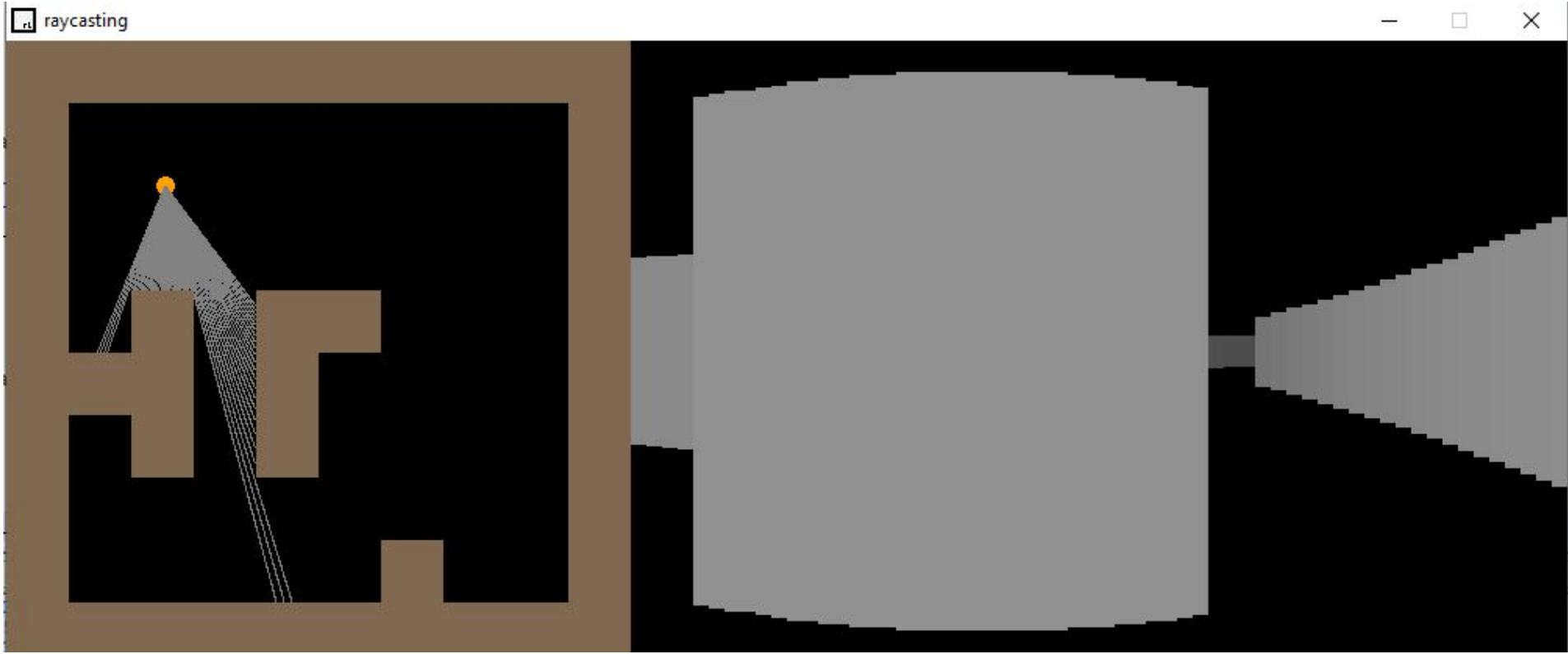
(640 x 480 -> 640 rays, instead of 307,200.)





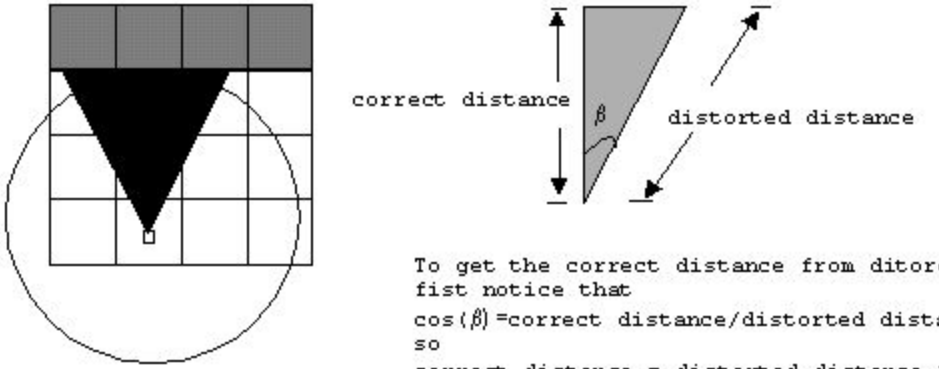
https://github.com/justinac0/raylib-raycaster

How to remove the distortion.



Raycasting per column will "fisheye" w/o correction

How to remove the distortion.



To get the correct distance from ditored distance
fist notice that
$\cos(\beta)$ =correct distance/distorted distance
so
correct distance = distorted distance * $\cos(\beta)$

Raycasting per column will "fisheye" w/o correction

Wolfenstein 3D (iD Software)

Relevant video clip: https://youtu.be/hYMZsMMlubg?t=290 (to 6:30)

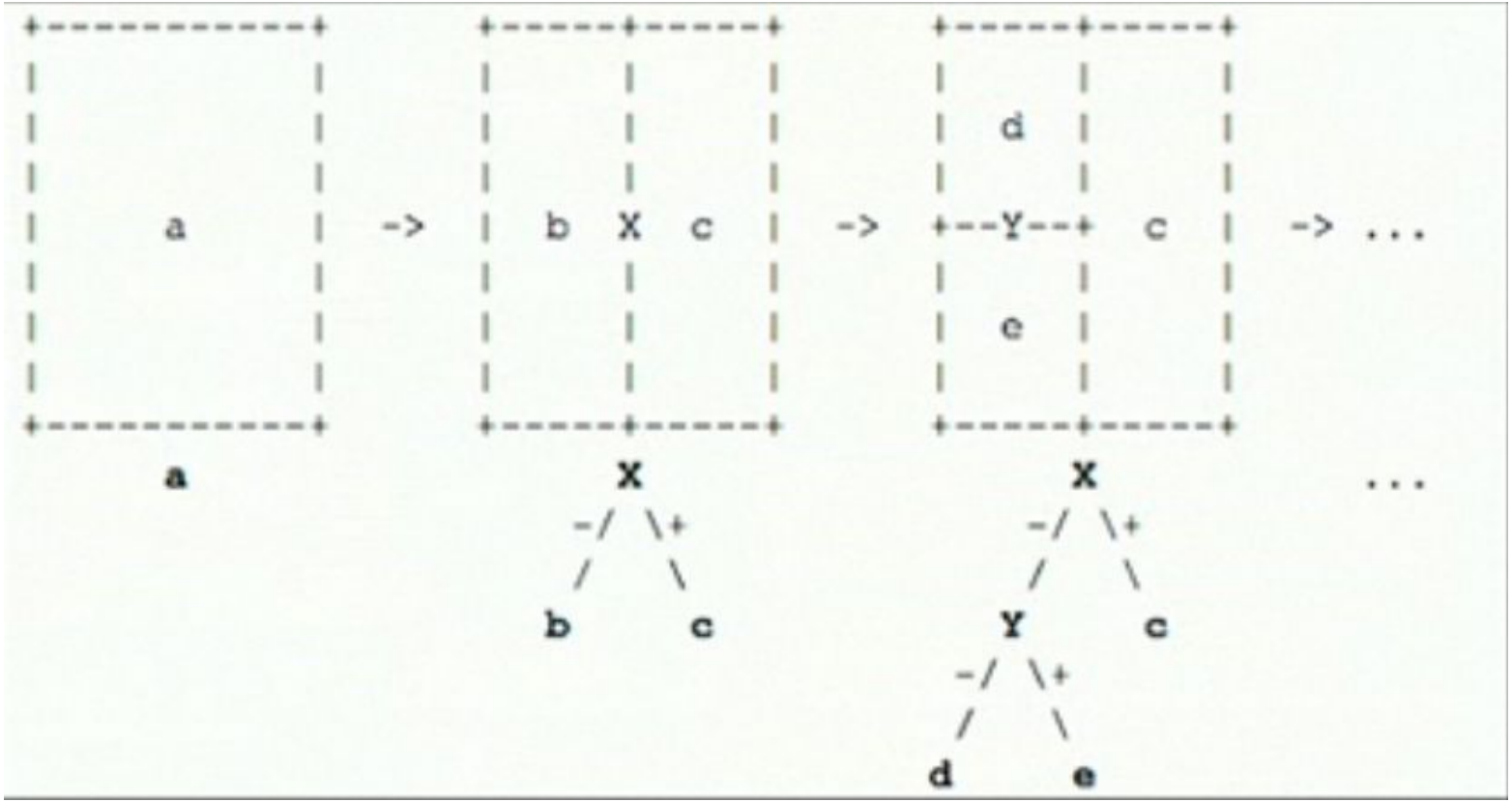
Binary Space Partitioning (BSP)

Proposed in a white paper many years before being implemented by John Carmack for DOOM.

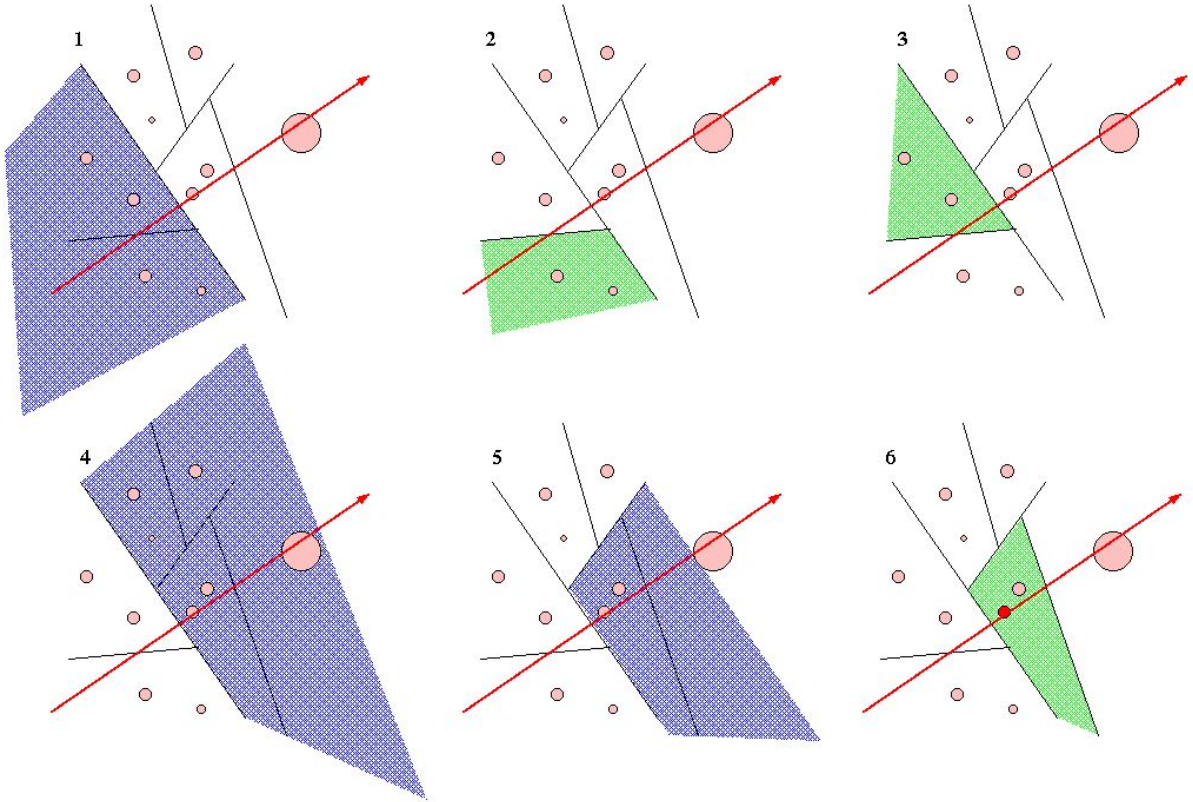
When a scene composed of polygons ('level') is created,

- Divide based on "splitters" (lines in the polygons)
- Recursively subdivide until all subsectors are convex polygons.
- These are ordered in a binary tree by relative distance.

Explained in a 1 dimensional version: https://youtu.be/sFSLY7n3YsM?t=53



Binary Space Partitioning



Binary Space Partitioning

DOOM (iD Software)

Relevant video clip: https://youtu.be/hYMZsMMlubg?t=1049

17:29 - 20:20