

Graphics with PICO-8



Goals

- Understand the magic behind computer graphics
- Use the PICO-8 environment to implement a graphics engine

Outline

- PICO-8 Background
- Building a graphics engine
 - 2d Wireframe
 - 3d Wireframe
 - Perspective
 - Shading
 - Texture Mapping
- Advanced techniques

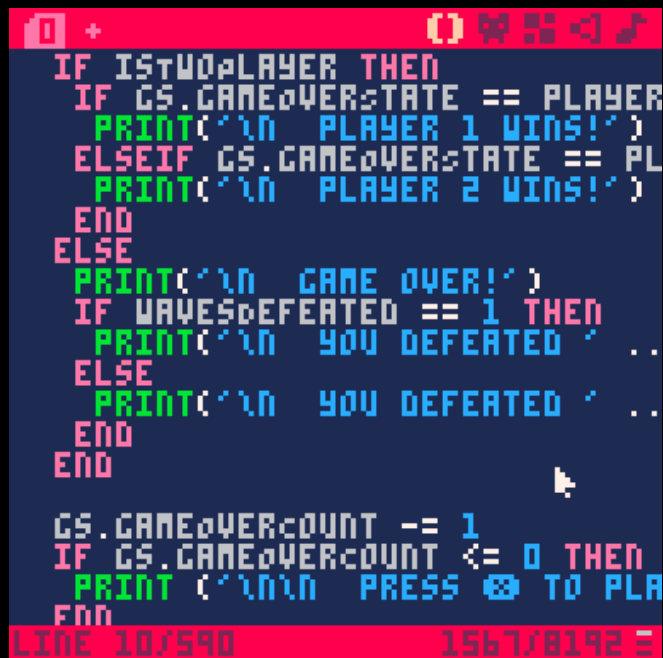
What is PICO-8?

- Minimalist Game Development Platform
- Everything you need to make a game, but nothing more



PICO-8 Features

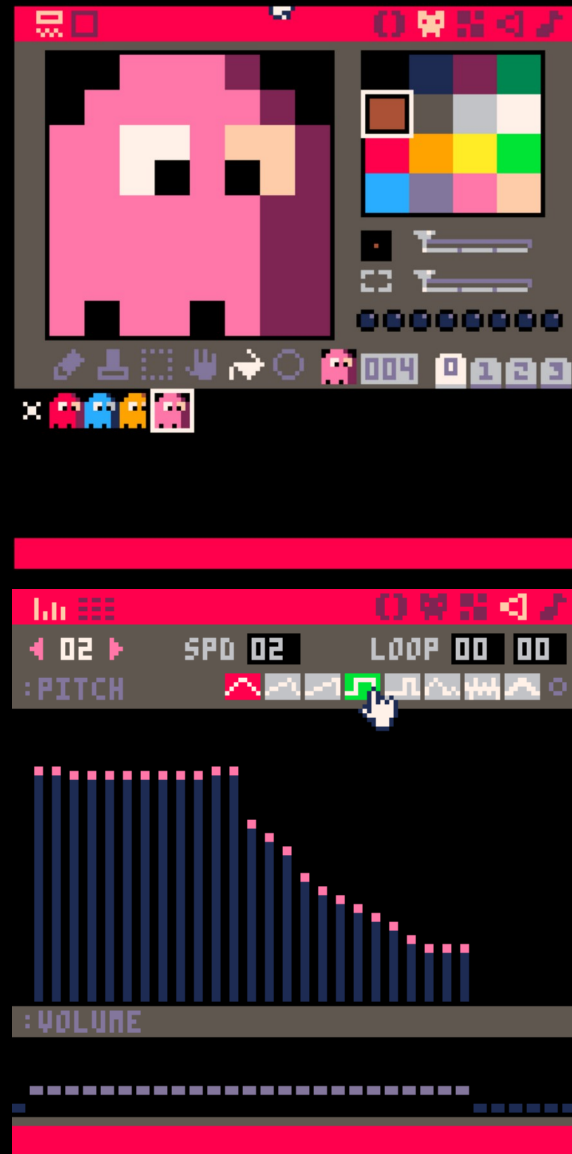
- Game Engine
- Code Editor
- Graphics Editor
- Audio Editor



```
IF ISTWOPLAYER THEN
  IF GS.GAMEOVERSTATE == PLAYER
    PRINT("\n  PLAYER 1 WINS!")
  ELSEIF GS.GAMEOVERSTATE == PL
    PRINT("\n  PLAYER 2 WINS!")
  END
ELSE
  PRINT("\n  GAME OVER!")
  IF WAVESDEFEATED == 1 THEN
    PRINT("\n  YOU DEFEATED ' ..
  ELSE
    PRINT("\n  YOU DEFEATED ' ..
  END
END
END

GS.GAMEOVERCOUNT -= 1
IF GS.GAMEOVERCOUNT <= 0 THEN
  PRINT ("\n\n  PRESS [X] TO PLA
END
END
```

LINE 10/590 1567/8192



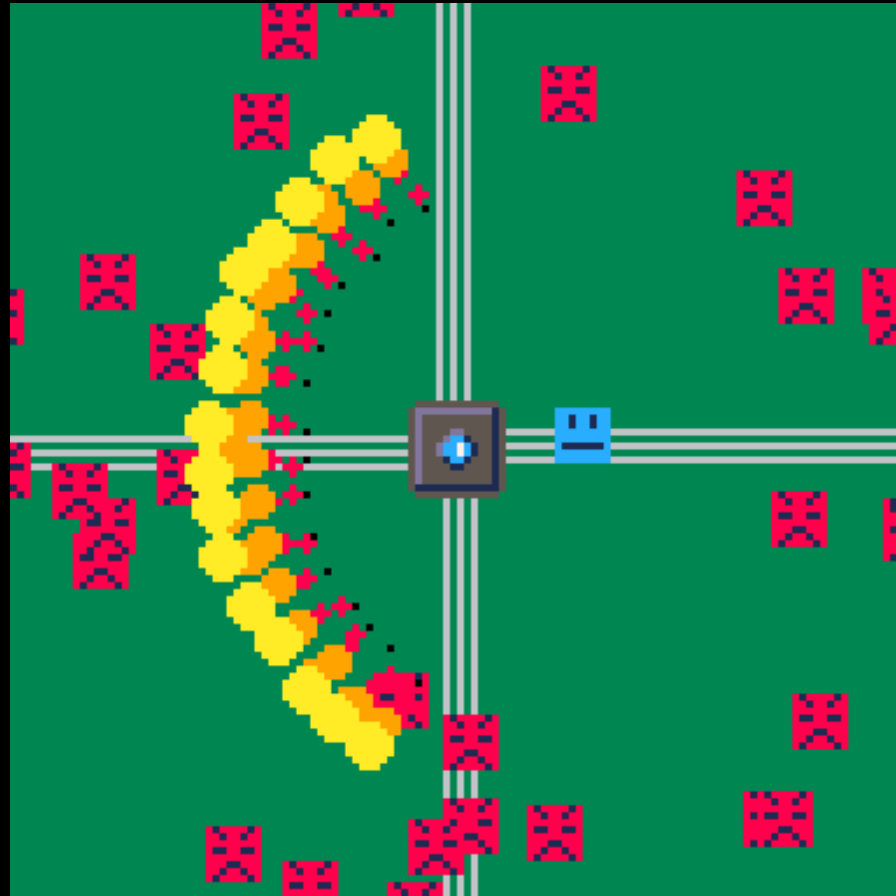
PICO-8 Limitations

- 128x128 pixel display
- 16 colors
- Limited memory, CPU, and code



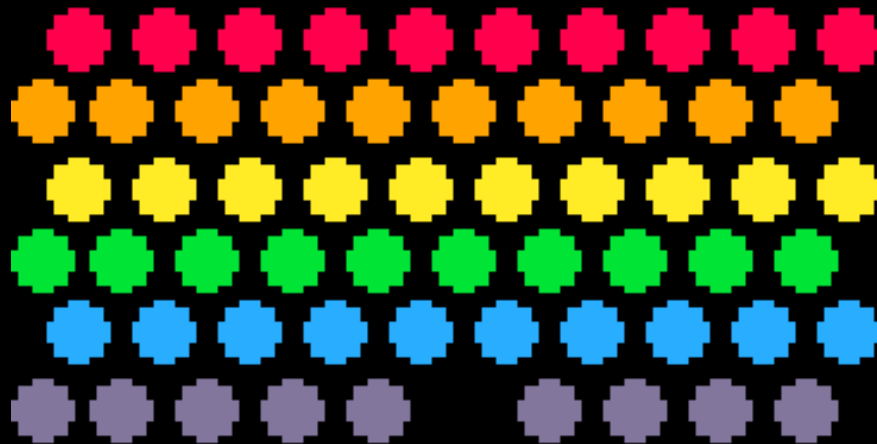
Does PICO-8 Have a Particle
System?

No! You have to build one



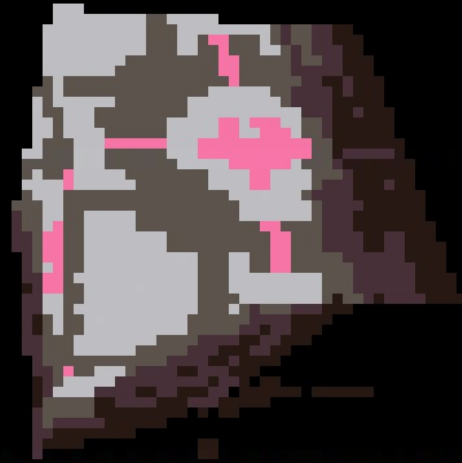
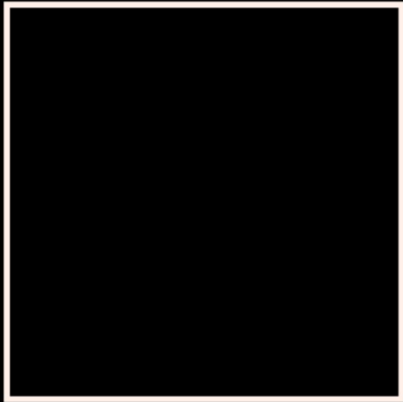
Does PICO-8 Have a Physics
Engine?

No! You have to build one!



Does PICO-8 Have a 3-D
Graphics Engine?

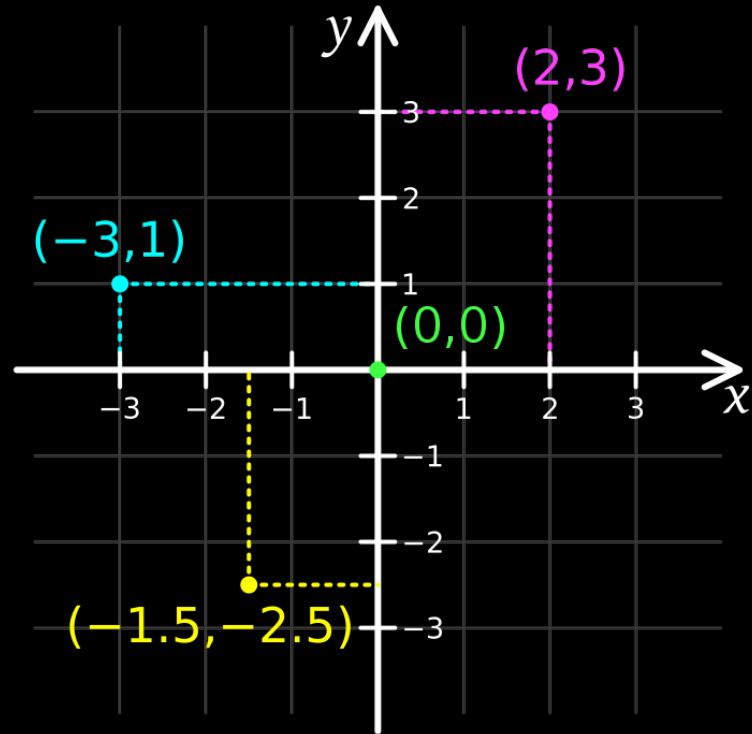
No! Let's Build One!



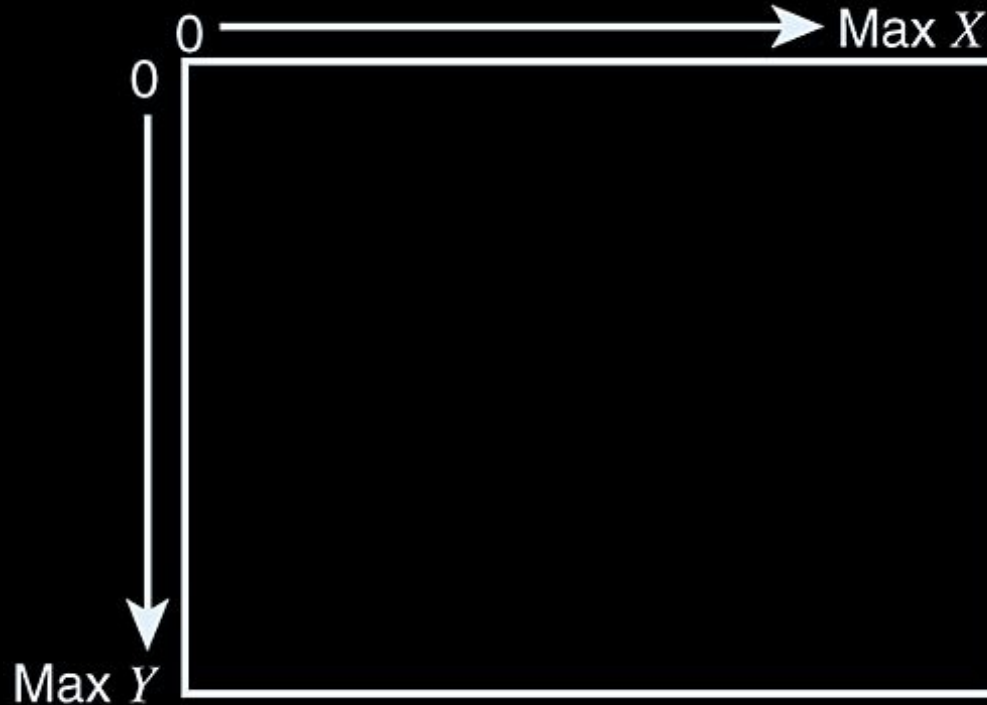
PICO-8 API

- line
- pset
- sin
- cos

Coordinates

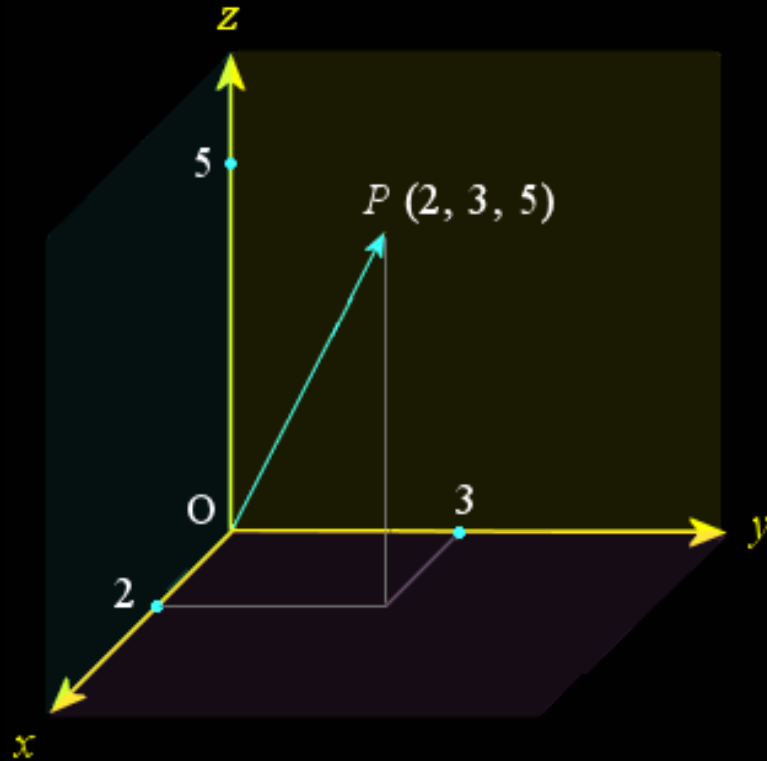


Screen Coordinates

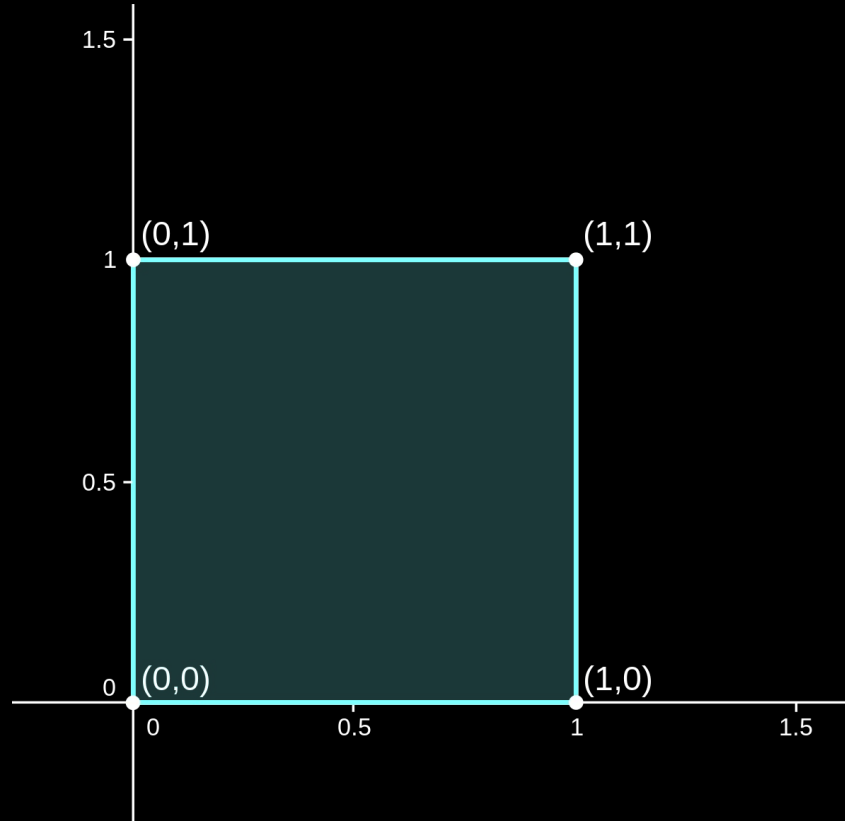


- <https://www.informit.com/articles/article.aspx?p=31554>

Vectors



Unit Square



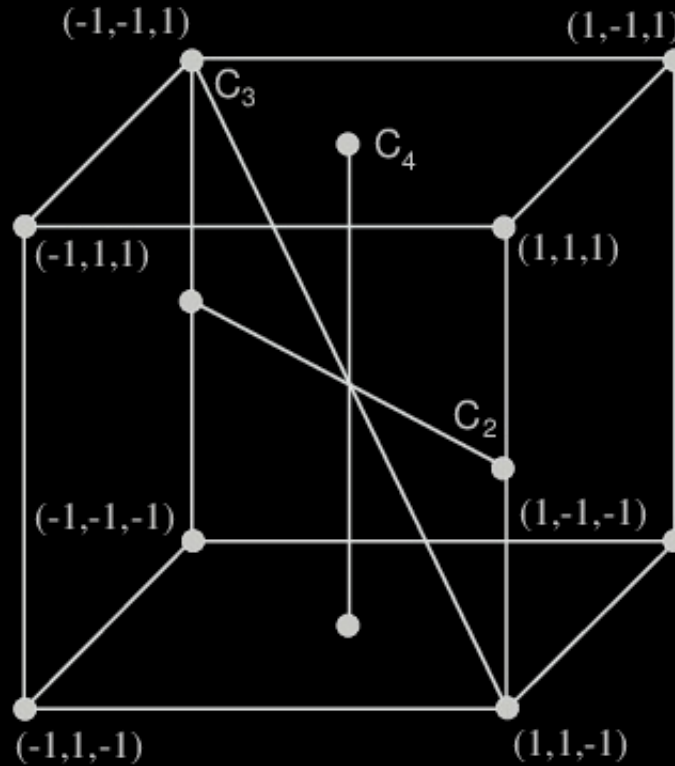
- https://www.google.com/url?sa=i&url=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FUnit_square&psig=AOvVaw39yB23xFGP7e8Pn6KL4ND8&ust=1715426469638000&source=images&cd=vfe&opi=89978449&ved=0CBQQjhxqFwoTCKC61t37goYDFQAAAAAdAAAAABAE

Transformations



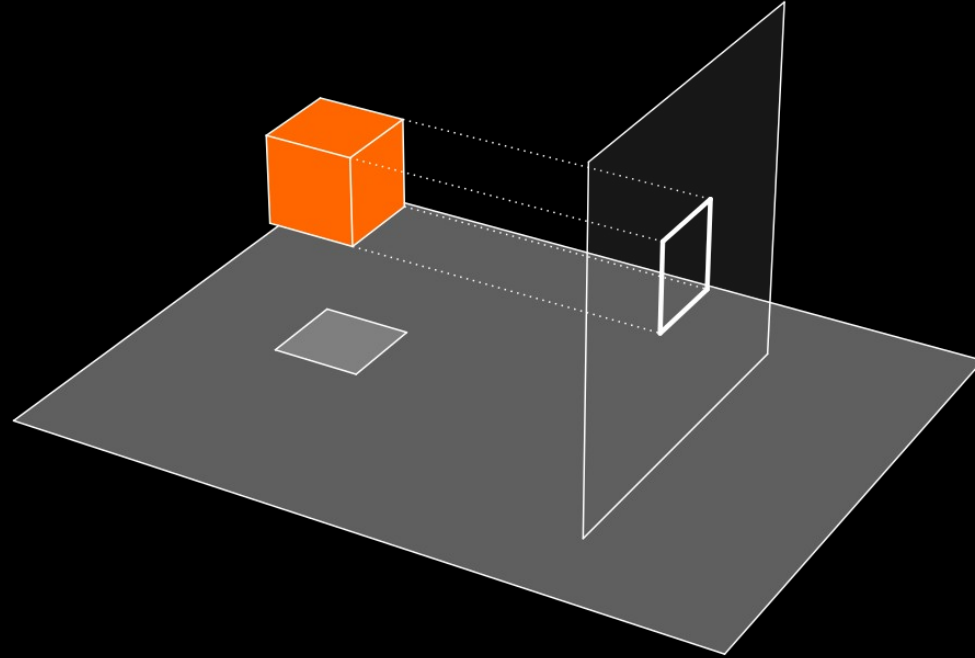
- <https://x.com/RejectedShotgun/status/1758589302840168865>

Unit Cube



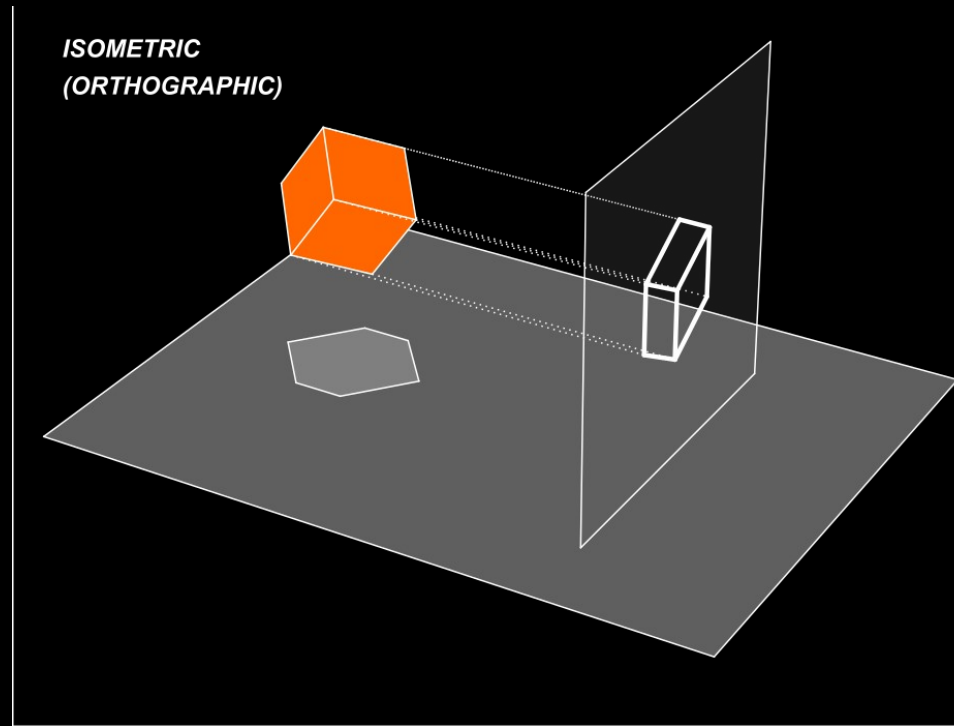
- https://www.researchgate.net/figure/The-figure-shows-the-vertices-and-corresponding-coordinates-of-the-cube-described-in-the_fig2_43164708

Orthographic Projection



https://en.wikipedia.org/wiki/Orthographic_projection#/media/File:Various_projections_of_cube_above_plane.svg
Licensed under <https://creativecommons.org/licenses/by-sa/4.0/>

Orthographic Projection



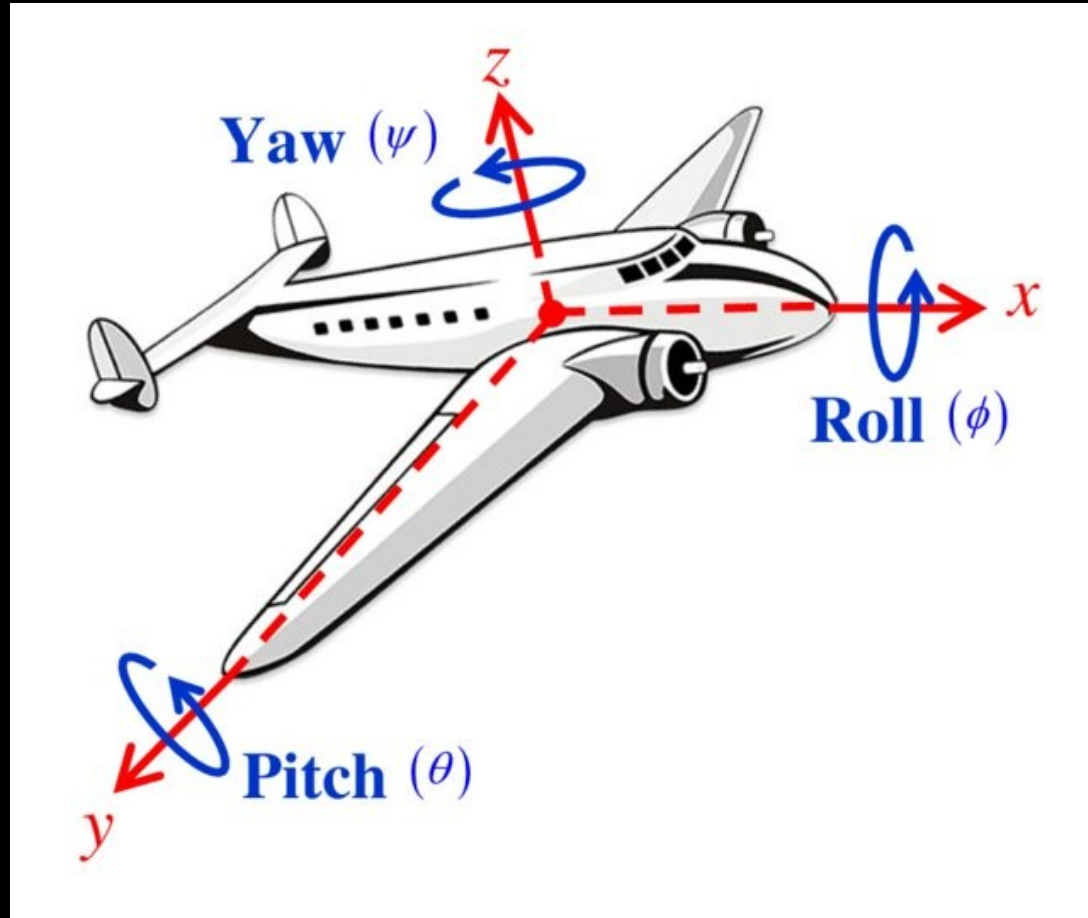
https://en.wikipedia.org/wiki/Orthographic_projection#/media/File:Various_projections_of_cube_above_plane.svg
Licensed under <https://creativecommons.org/licenses/by-sa/4.0/>

Orthographic Projection

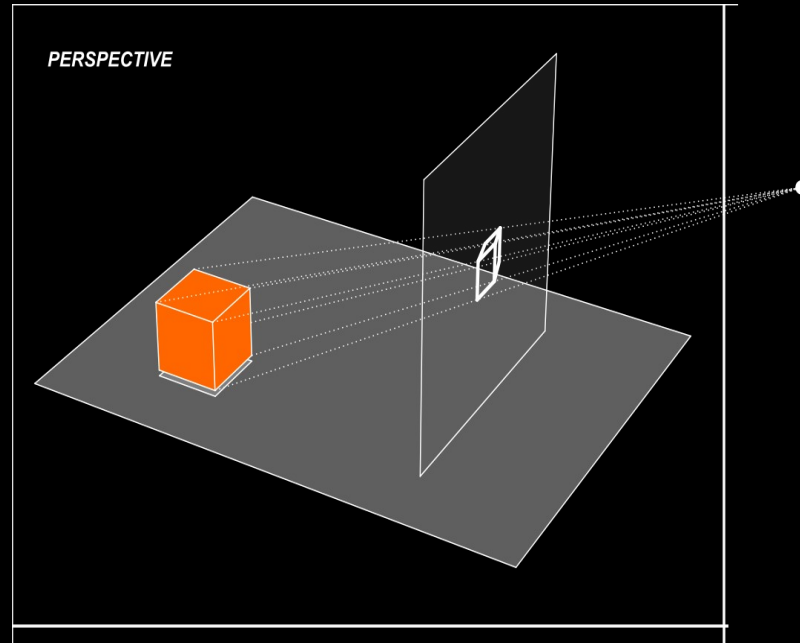


<https://www.lifewire.com/simcity-4-starting-new-city-840147>

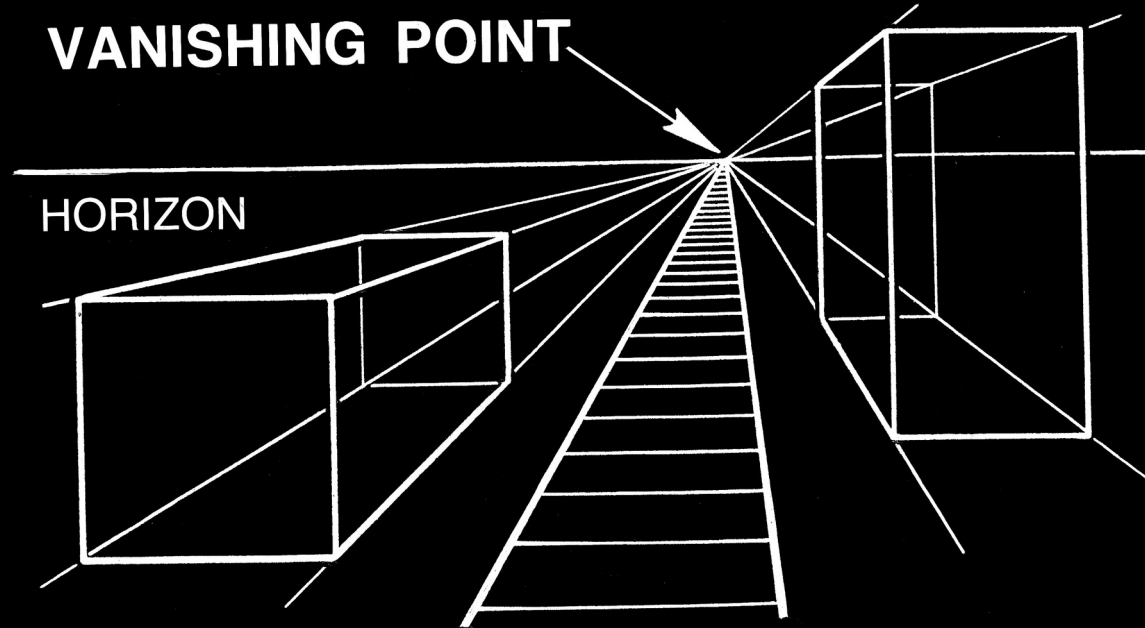
Euler Angles



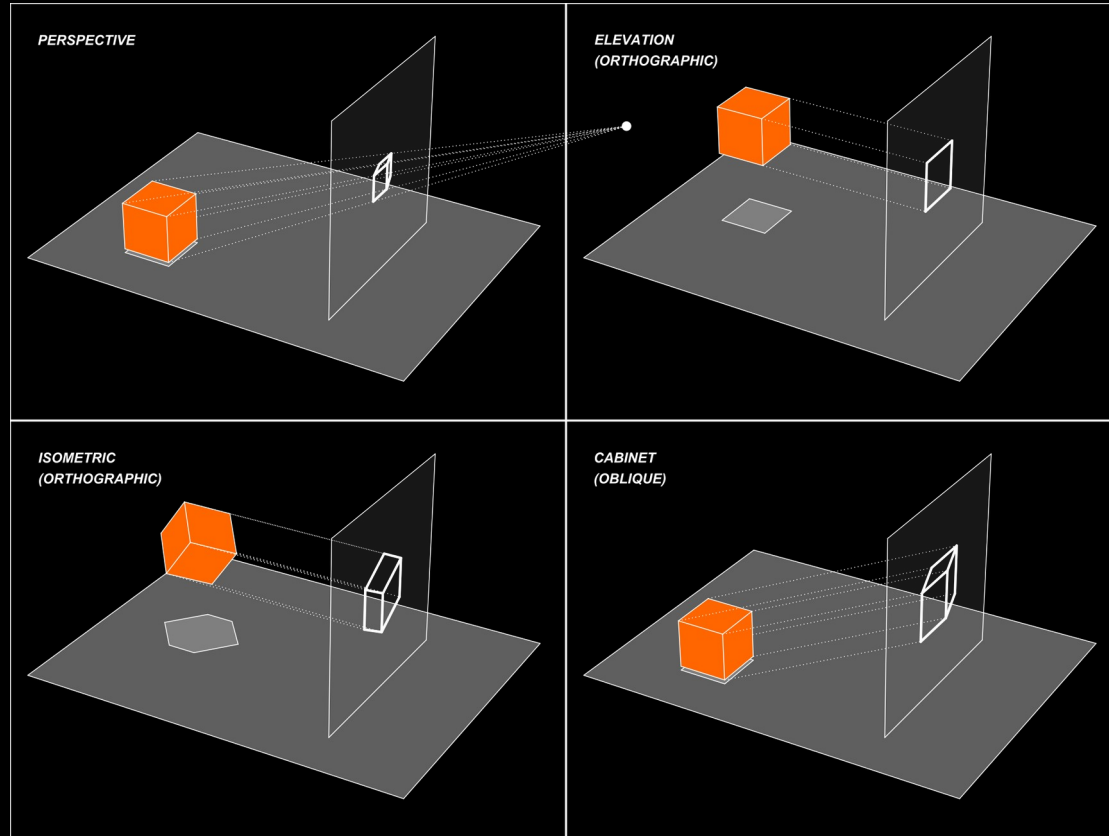
Perspective Projection



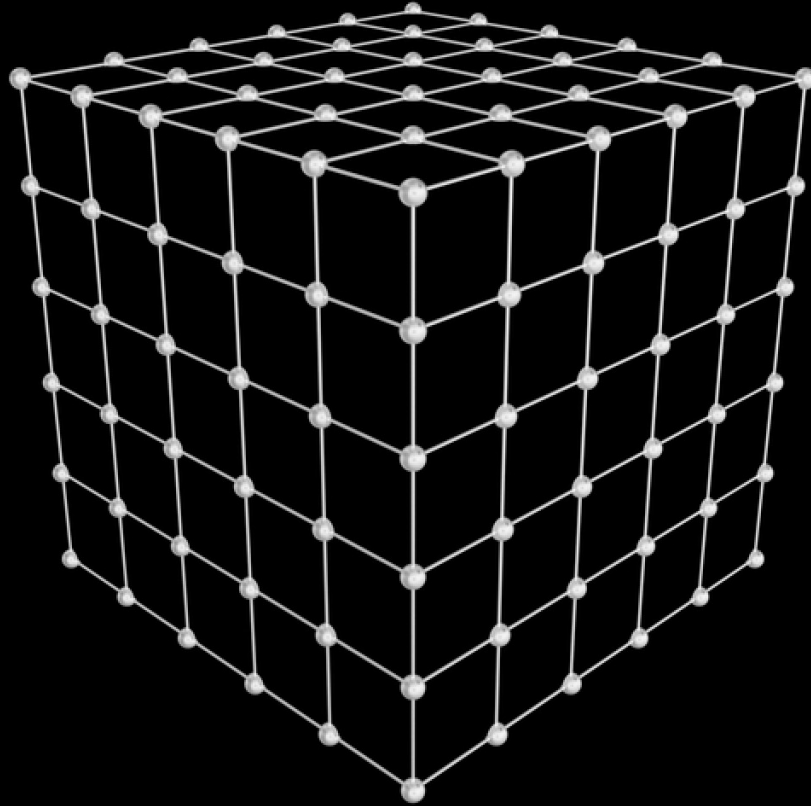
Perspective Projection



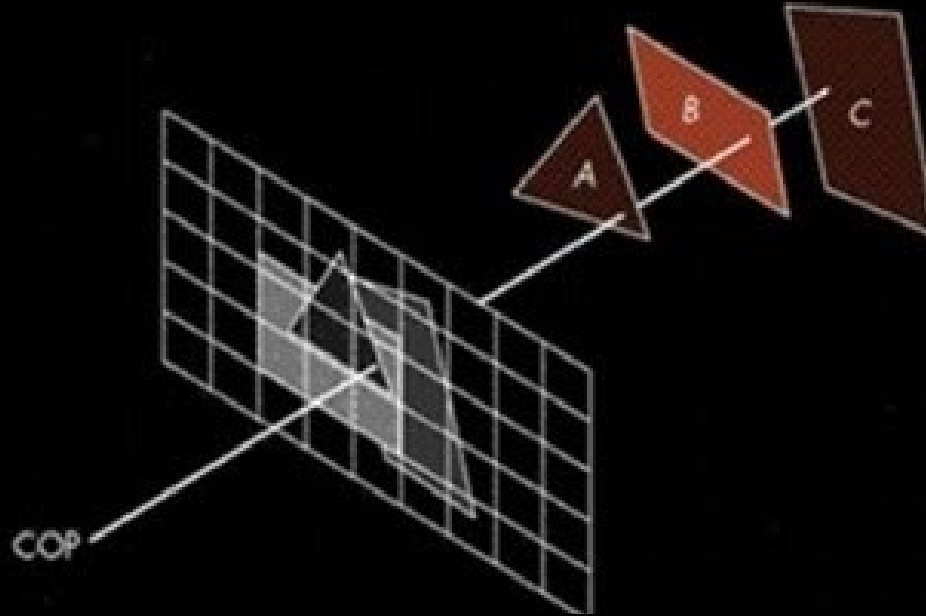
Projection



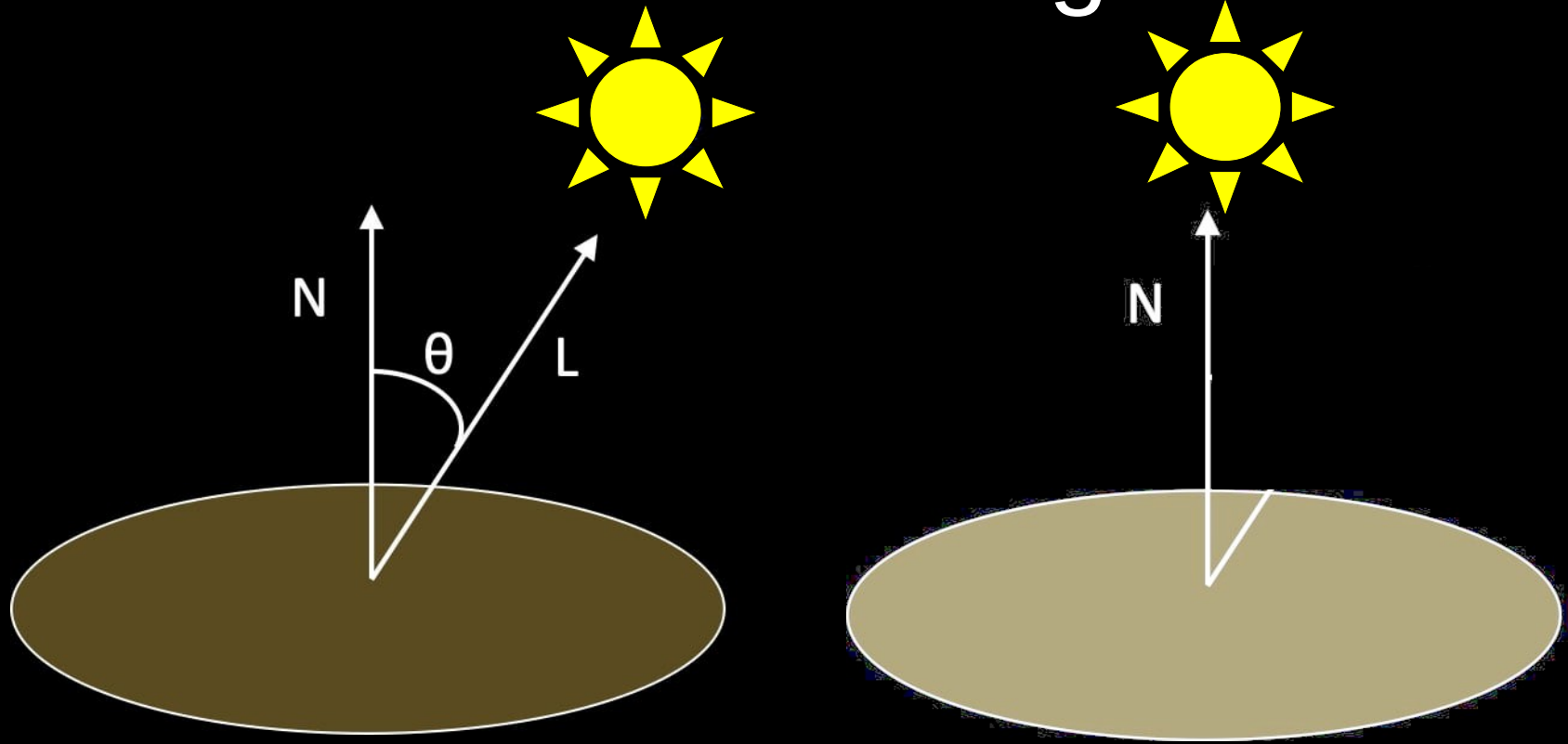
Filling a Polygon



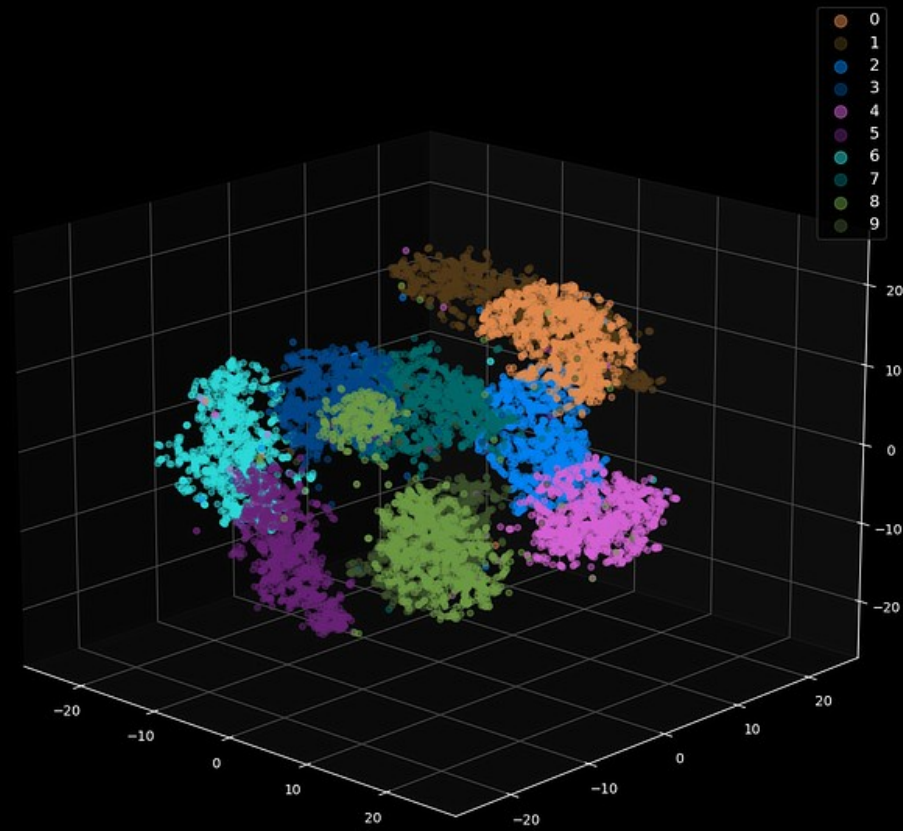
Depth Sorting



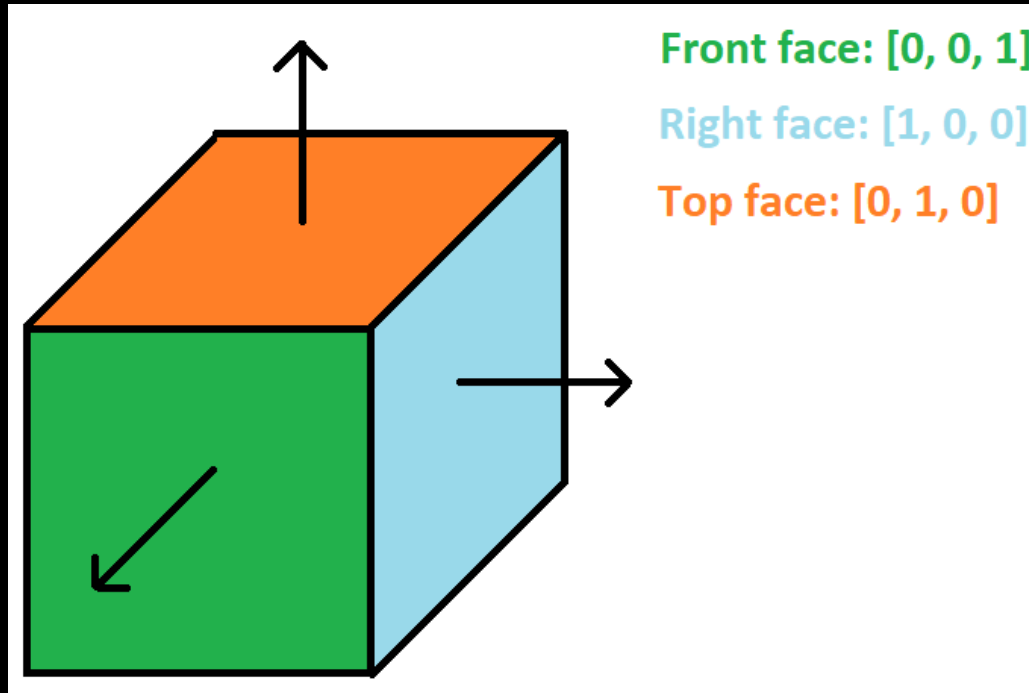
Flat Shading



Dot Product Application - AI



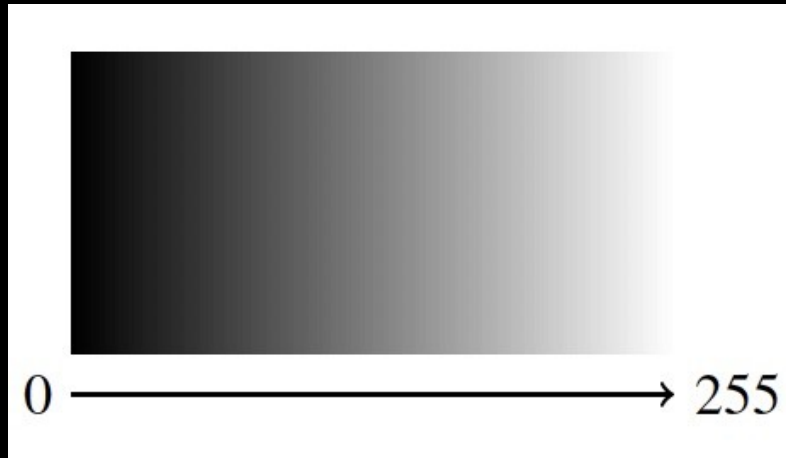
Normals



- <https://www.mbsoftworks.sk/tutorials/opengl4/014-normals-diffuse-lighting/>

Continuous Color vs. Discrete Color

- Dot product gives a value between -1 and 1
- Map this value to a color



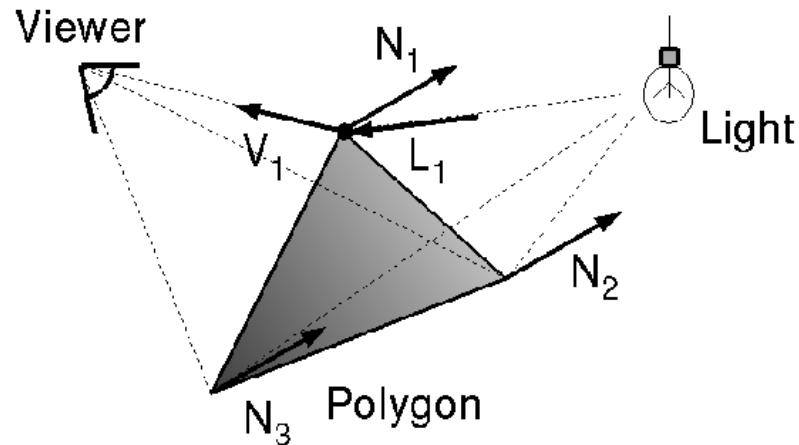
- https://creativecomputing.ca/04/4_1_Grayscale_Colour.html

Gouraud Shading

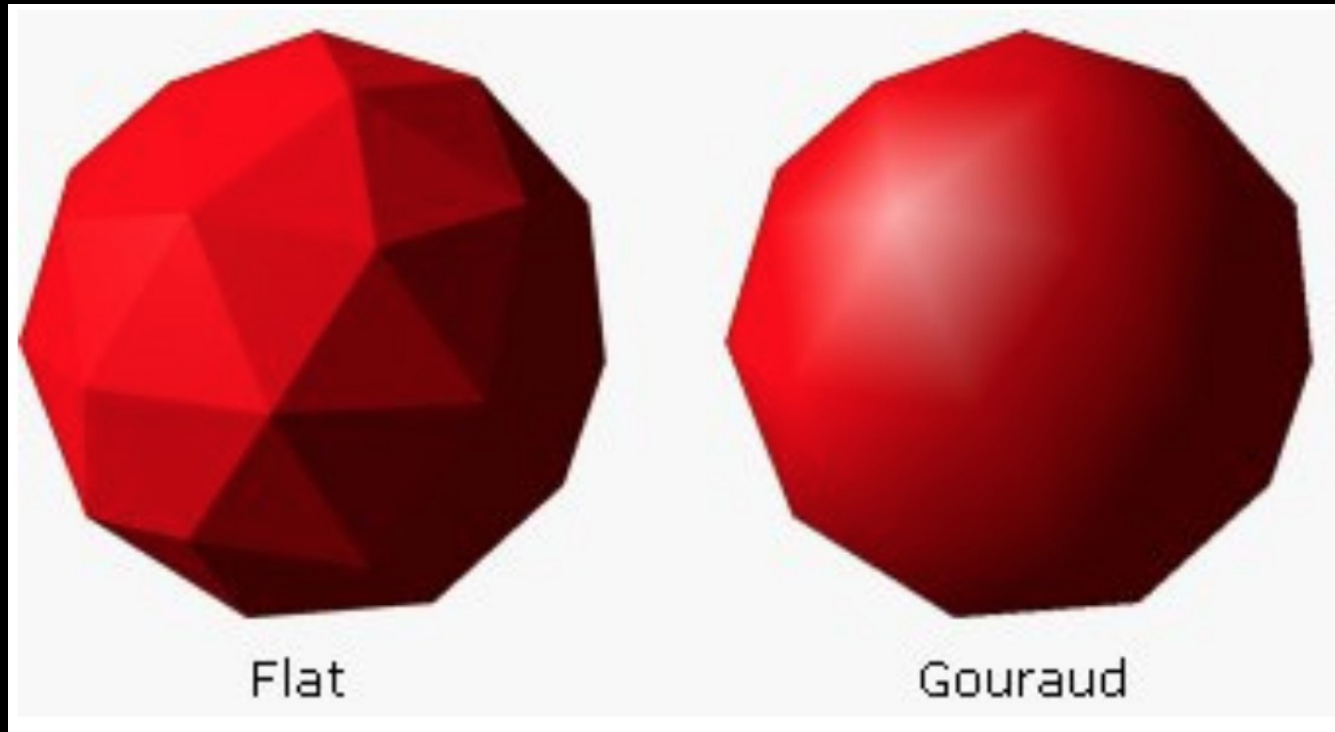
Gouraud Shading



- Method 1: One lighting calculation per vertex
 - Assign pixels inside polygon by interpolating colors computed at vertices

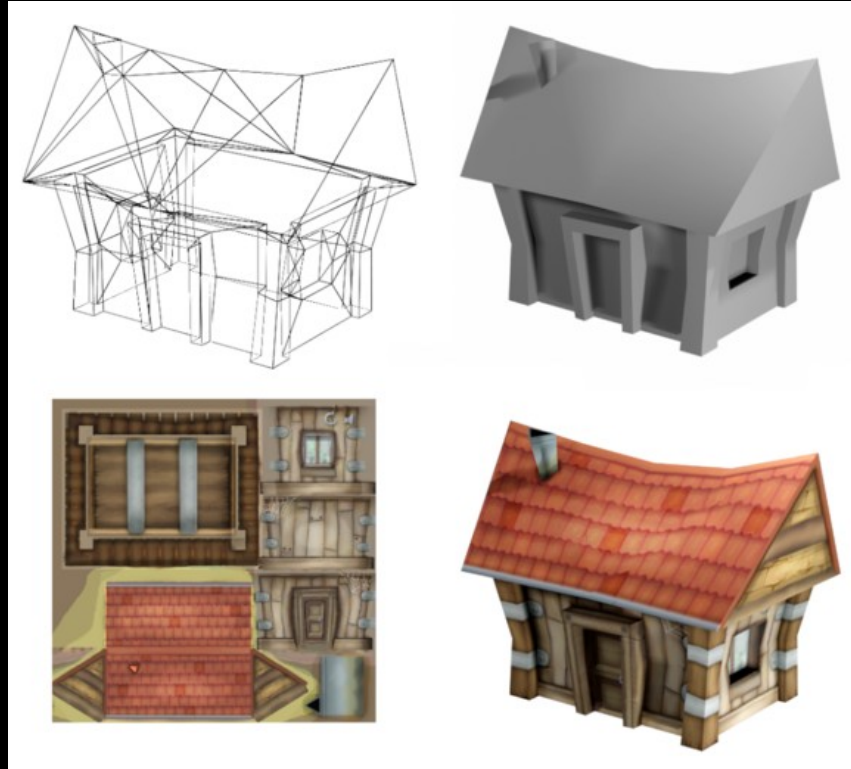


Gouraud Shading



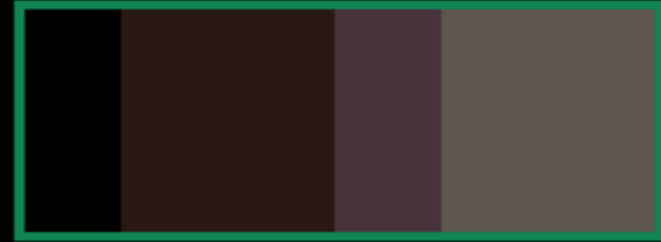
- <https://computergraphics.stackexchange.com/a/10846>

Texture Mapping



- https://en.wikipedia.org/wiki/Texture_mapping

Discrete Color Part 2

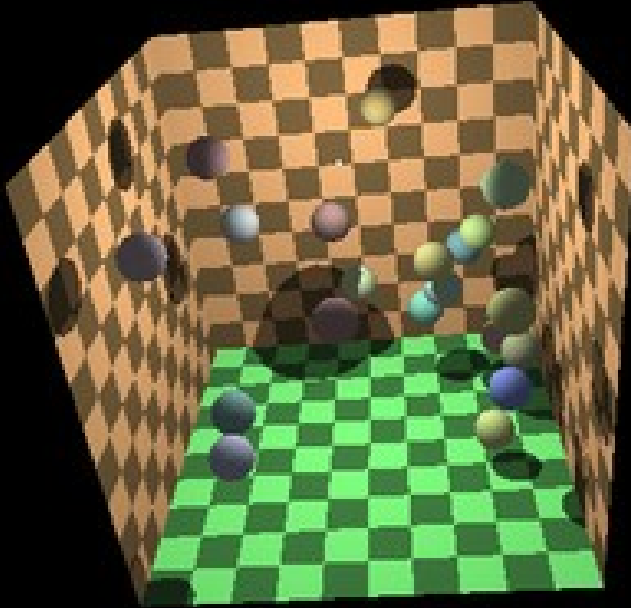


- <https://hackernoon.com/pico-8-lighting-part-1-thin-dark-line-8ea15d21fed7>

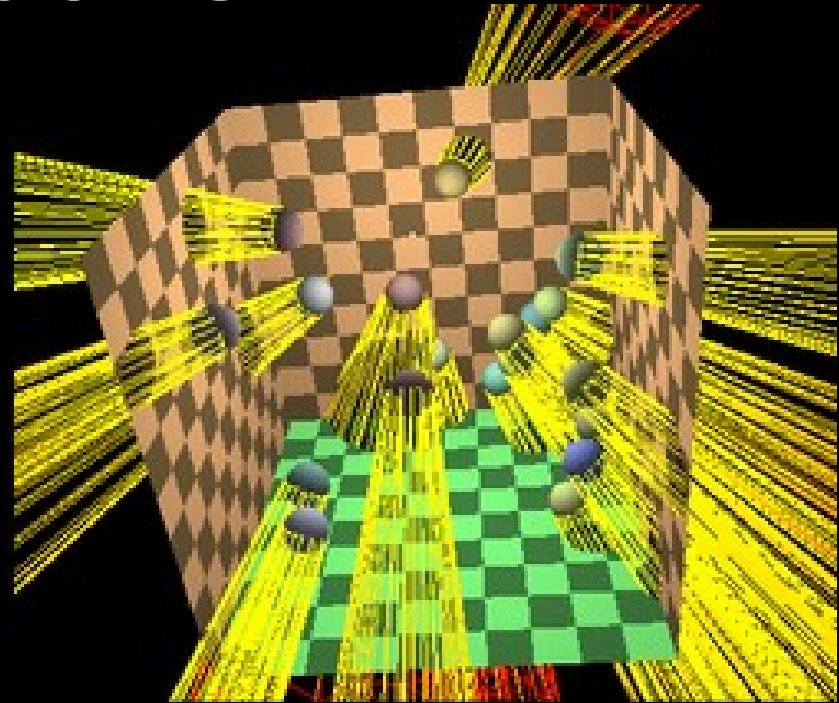
Limitations

- Shadows
- Transparency
- Reflections

Shadows



shadowed scene



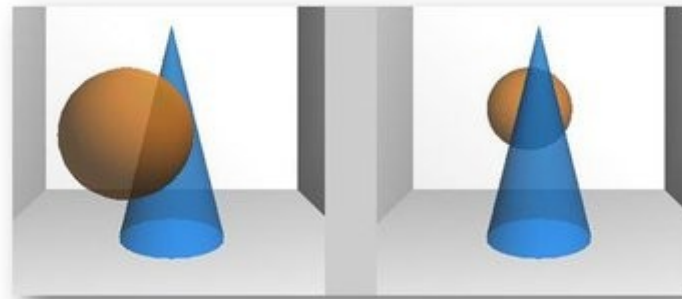
wireframe shadow volumes

- https://www.google.com/url?sa=i&url=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FShadow_volume&psig=AOvVaw2B6zZ1mybl4l4YQVbp-QLd&ust=1715427785110000&source=images&cd=vfe&opi=89978449&ved=0CBQQjhqFwoTCMDSmfGAg4YDFQAAAAAdAAAAABAR

Transparency

Transparency

See-through objects



Object order affects final color
Blending non commutative

$$\alpha_1 C_1 + (1 - \alpha_1) \alpha_0 C_0 \neq \alpha_0 C_0 + (1 - \alpha_0) \alpha_1 C_1$$

Depth test
discards objects

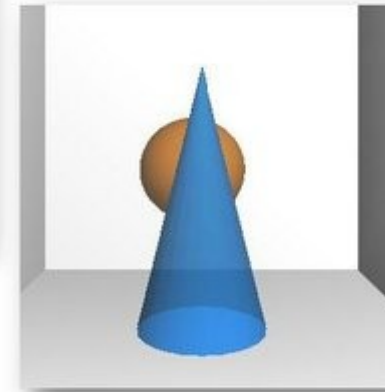


Image Synthesis – WS 07/08

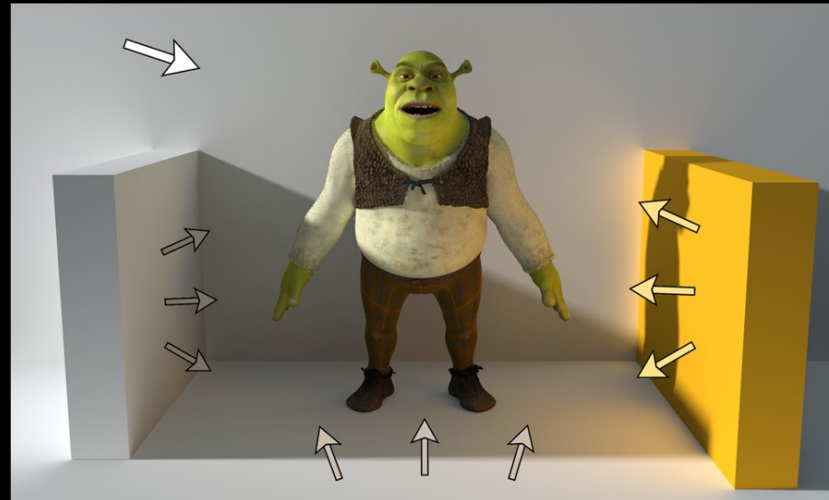
Dr. Jens Krüger – Computer Graphics and Visualization Group

Indirect Lighting

Direct Lighting Only

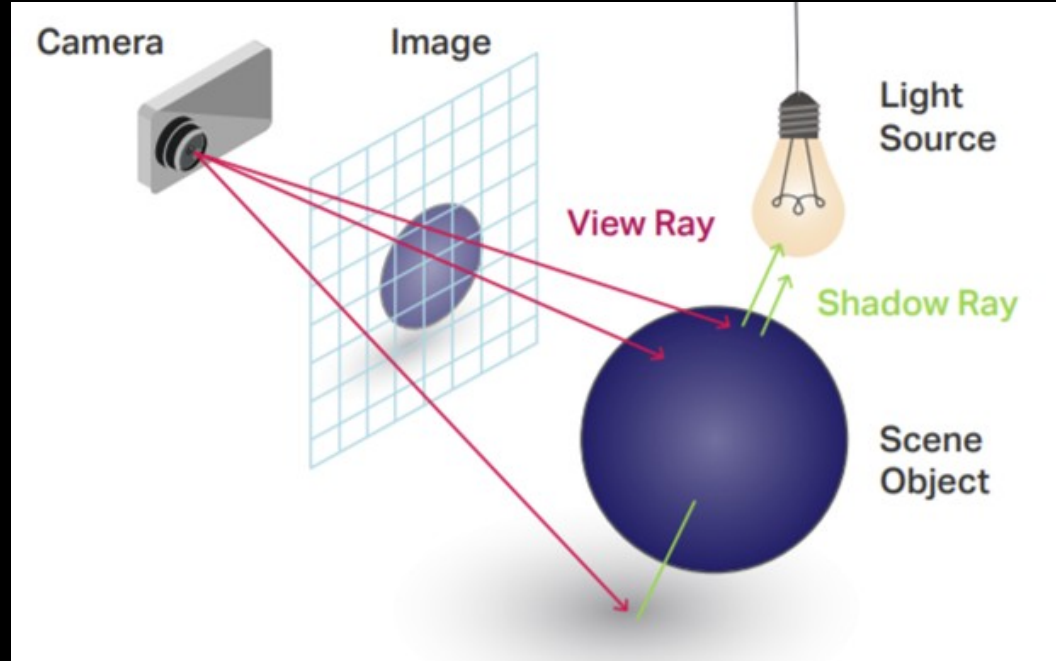


Direct + Indirect Lighting

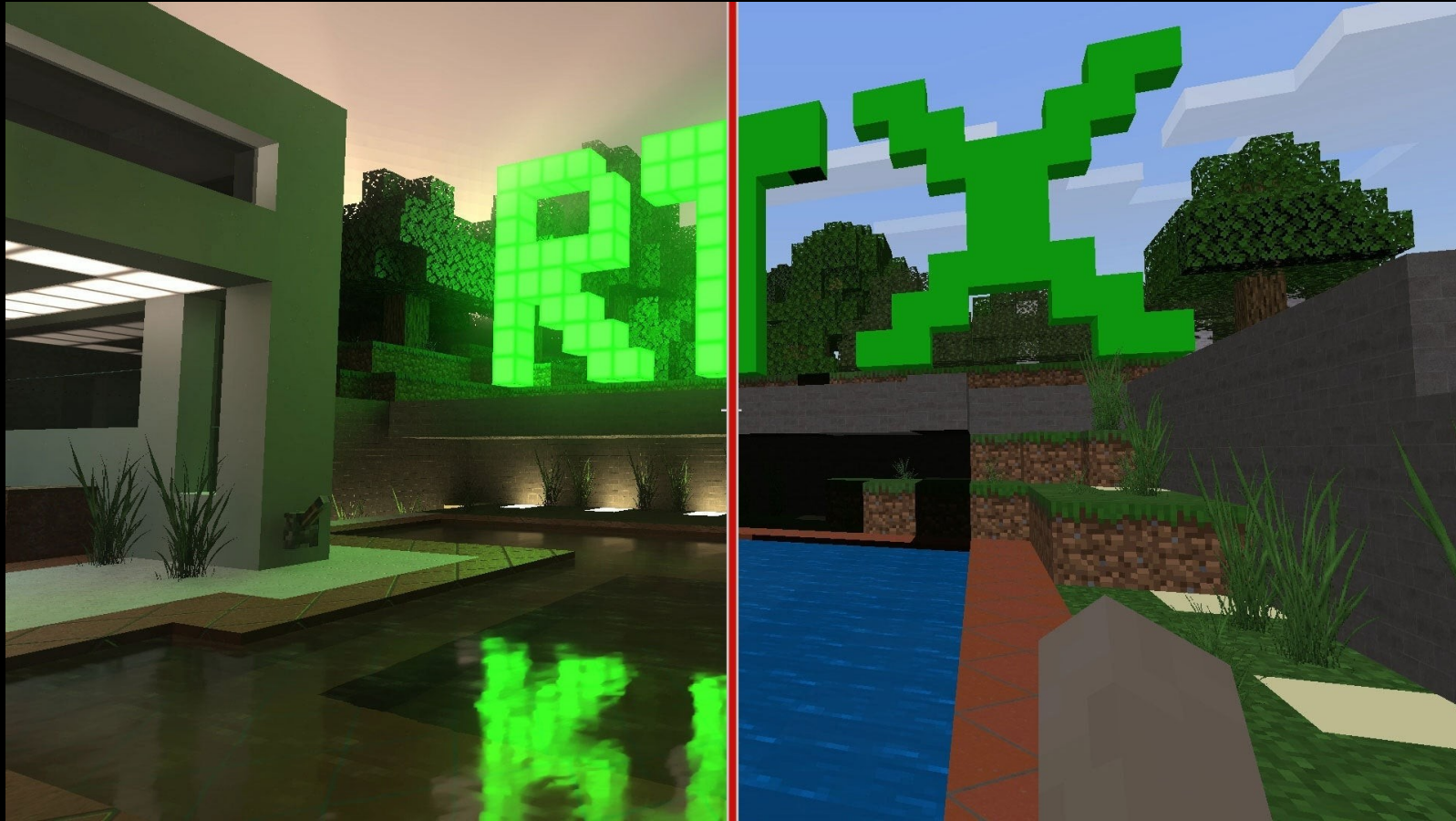


- <https://colinbarrebrisebois.com/2015/11/06/finding-next-gen-part-i-the-need-for-robust-and-fast-global-illumination-in-games/>

Ray Tracing

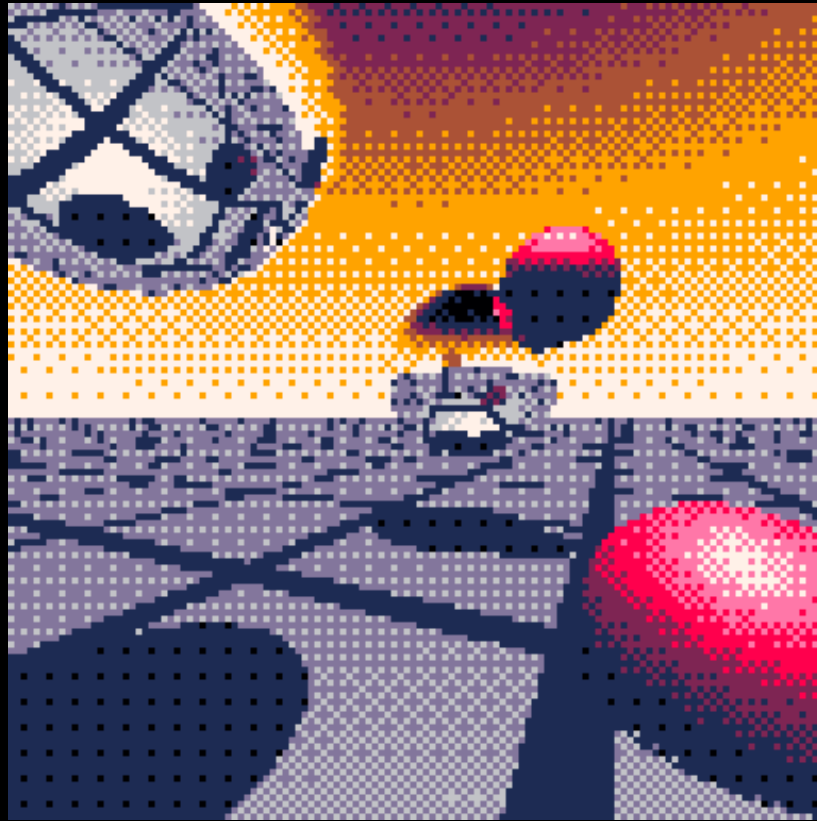


Ray Tracing



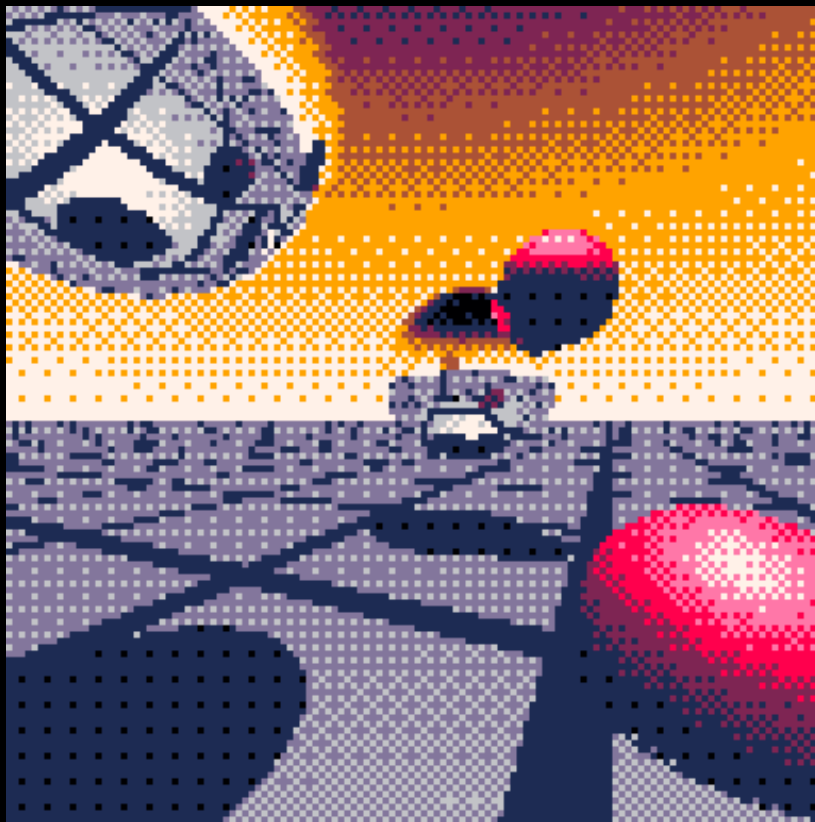
- <https://www.ign.com/articles/what-is-ray-tracing>

Ray Tracing



- <https://www.lexaloffle.com/bbs/?pid=59480>

Questions?



- <https://www.lexaloffle.com/bbs/?pid=59480>