GDC | March 21-25, 2022
San Francisco, CA

Live Long and Render!

Dan Buckstein
Engineer (Gameplay), Infinity Ward
GDC 2022
Game Career Development

#GDC22

Live Long and Render!

- **Dan Buckstein**
- Engineer (Gameplay)

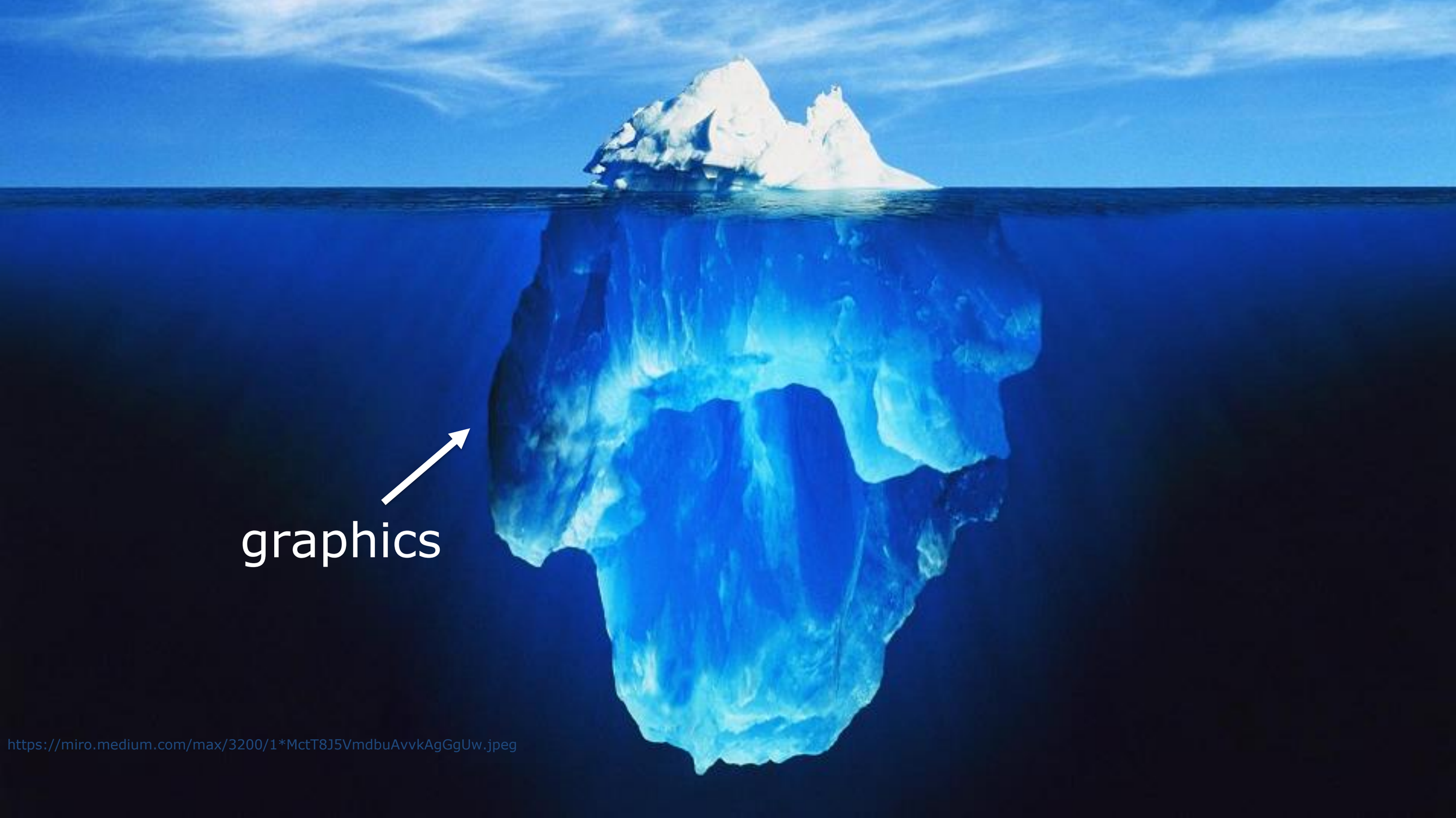
- Formerly: Associate Professor,
Game Programming, Champlain College

Session Overview

- This talk **does contain**:
  - Tales from graphics programming
  - Benefits of graphics programming
  - Tips & tools for hobby & career advancement
- This talk **does not contain**:
  - The deep and fascinating history of the GPU
  - AAA proprietary rendering technology
  - How to write the best renderer ever

Session Overview

Thoughts and opinions expressed in this talk are my own.



https://miro.medium.com/max/3200/1*MctT8J5VmdbuAvvkAgGgUw.jpeg

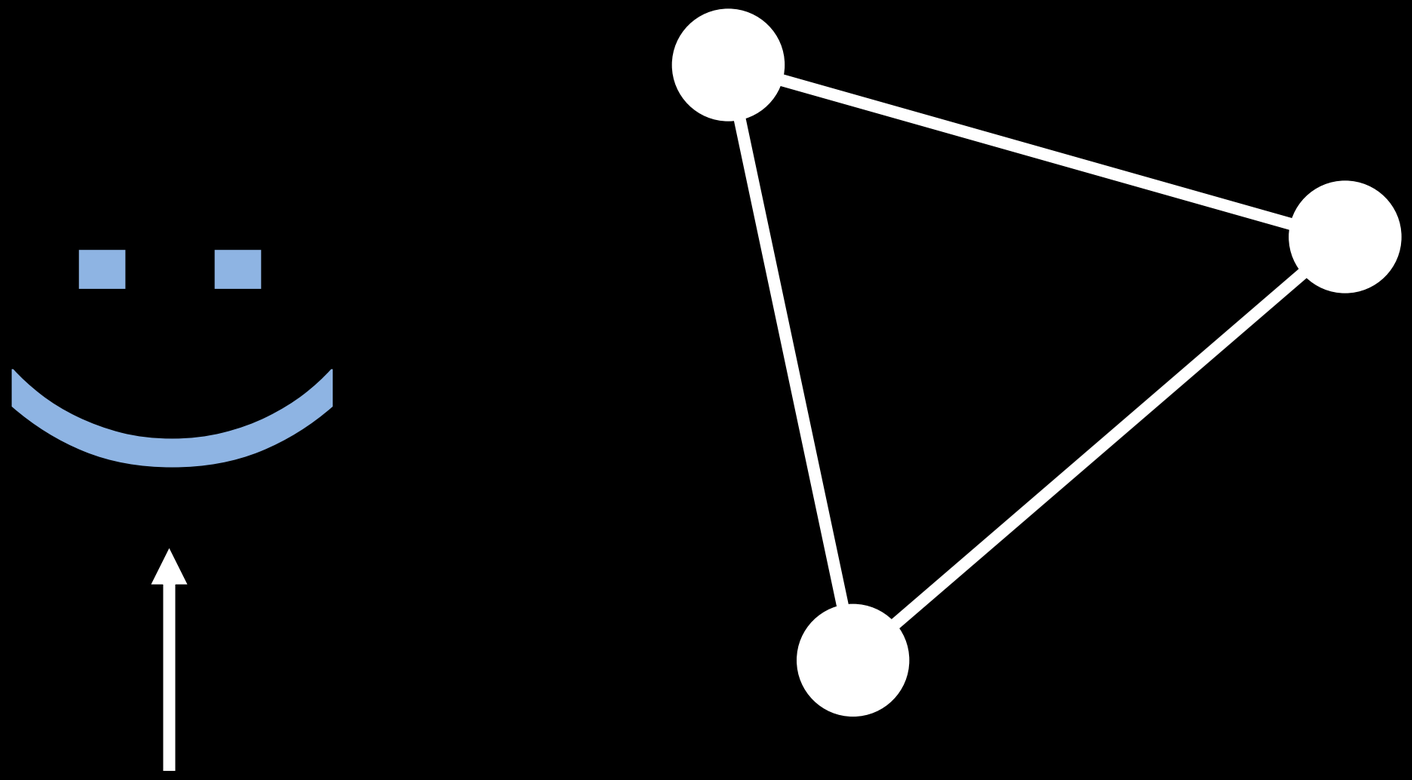
Where do I begin?

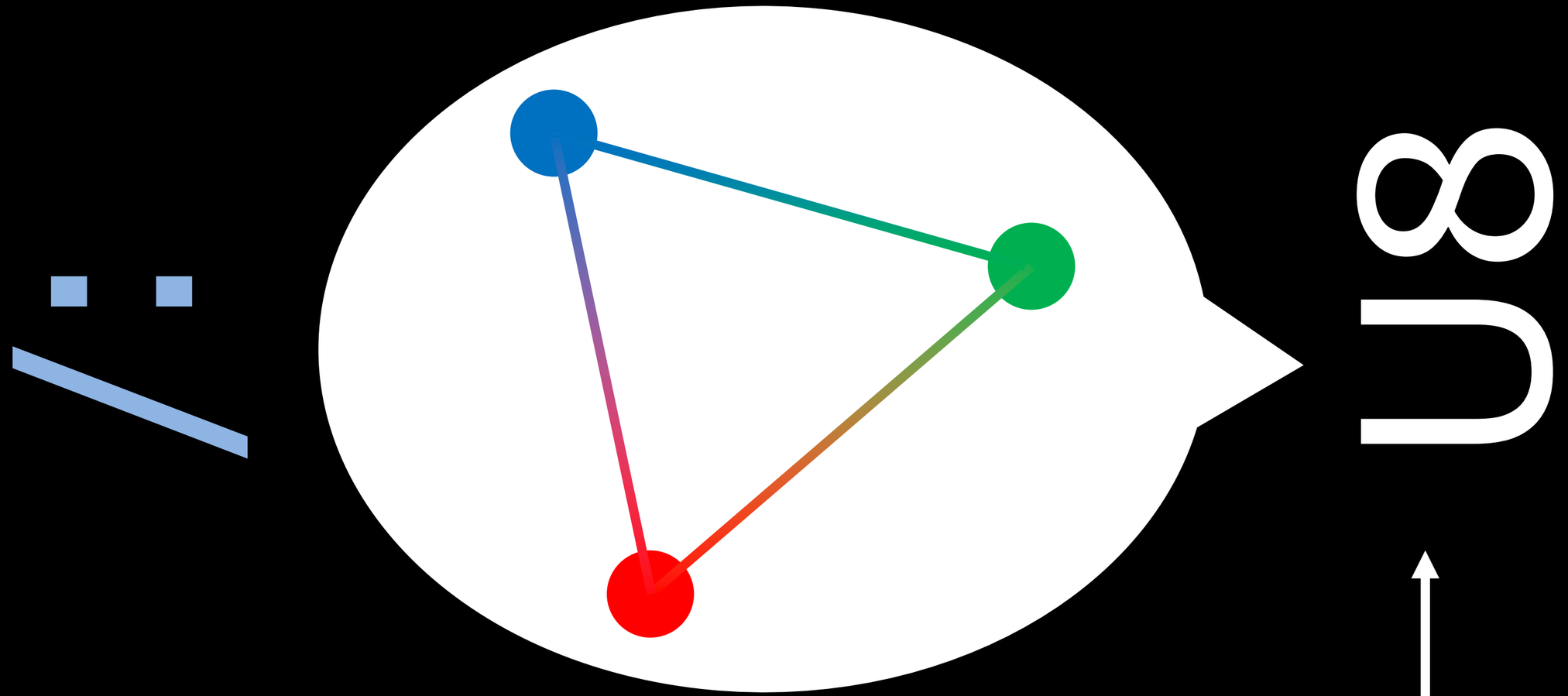
Live Long and Render!

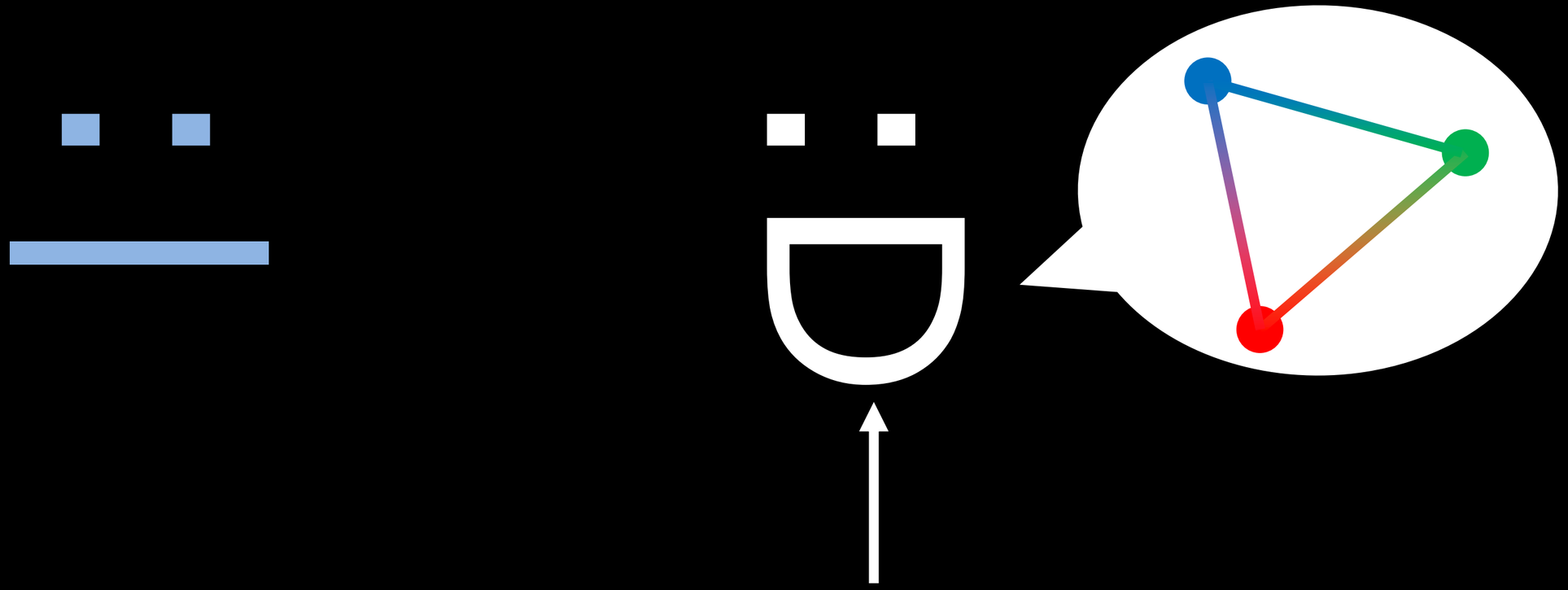
…or…

How I Learned to Stop Worrying and Love Graphics Programming

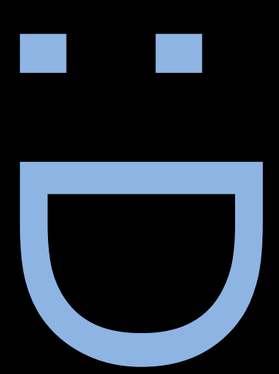
In the beginning… (2009-ish)

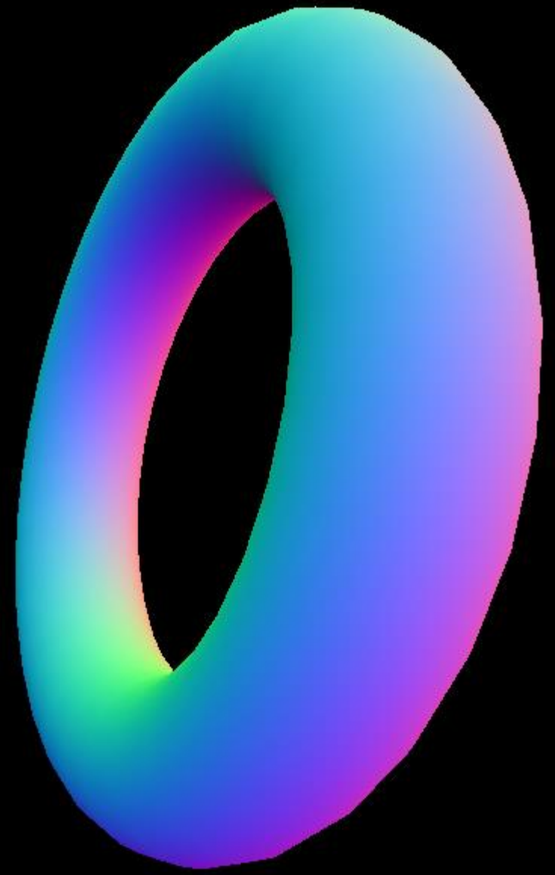


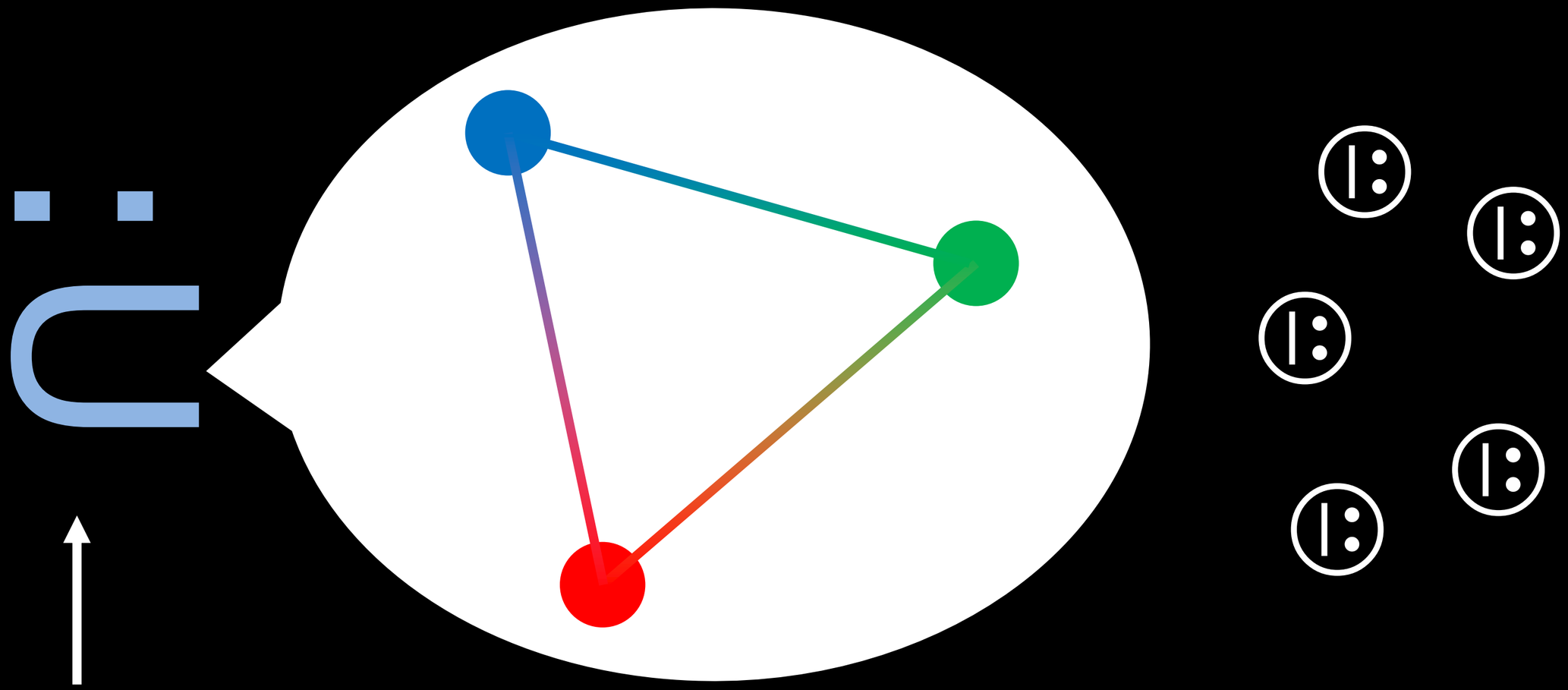

U8
prof


friend

Productive spring break!



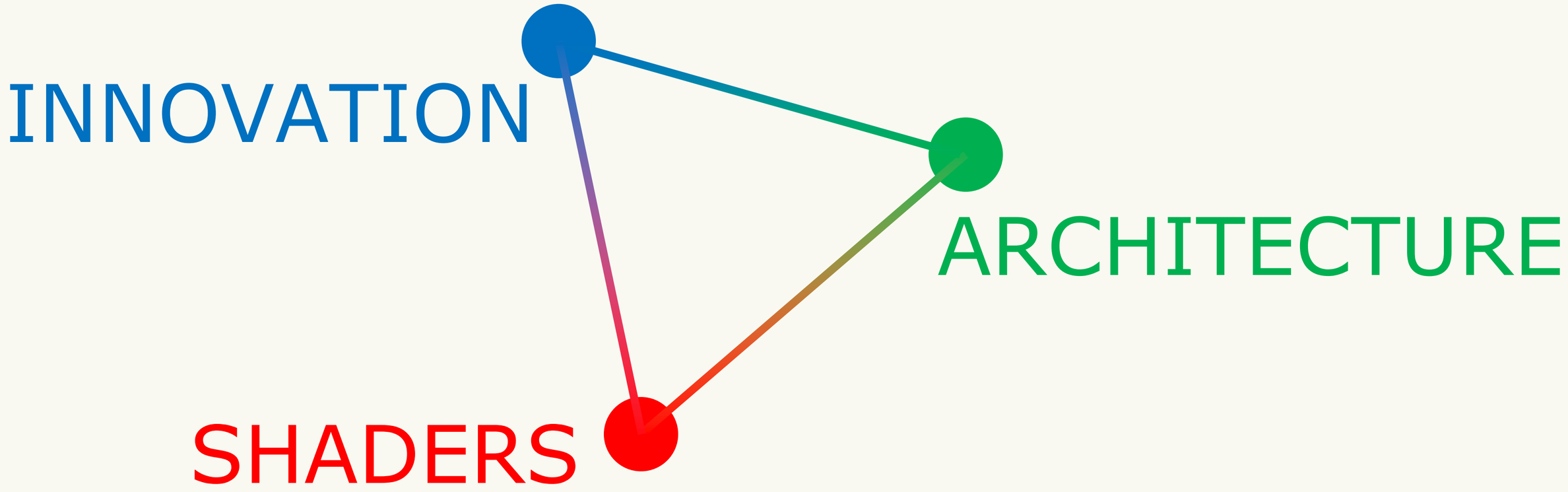




teaching assistant me

Where do I begin?

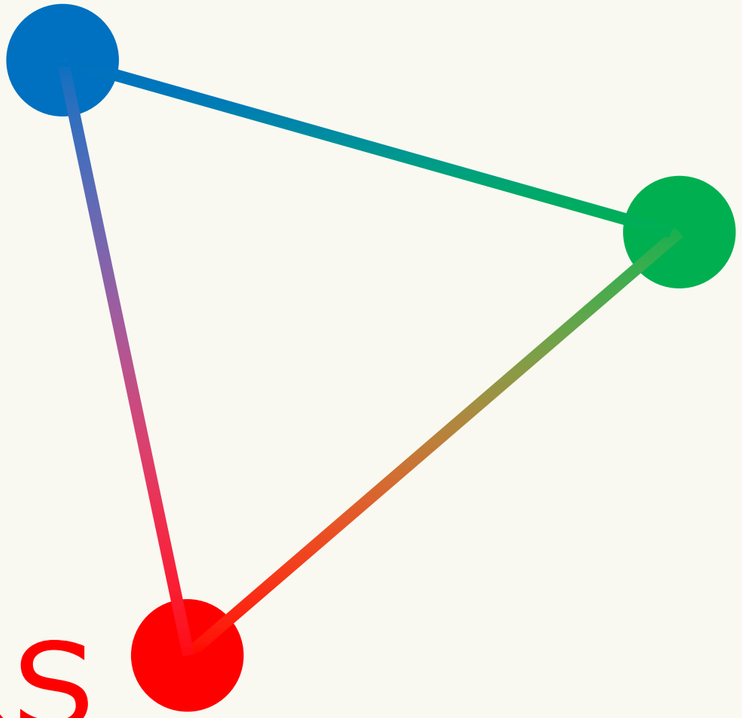
- Three main components of graphics programming:



"A Three-Course Meal in Graphics Programming Education"
D. Buckstein, GDC 2021 Education Summit

Where do I begin?

- SHADERS & VFX FIRST!!! :D



SHADERS

->Fun
->Creative
->High impact
->Low stress

"Teaching Modern Graphics: A Shader-First Approach"
S. Farooq, GDC 2019 Education Summit

Shaders first!

- OpenGL shader pipeline:

VS→TCS→TES→GS→FS (CS)

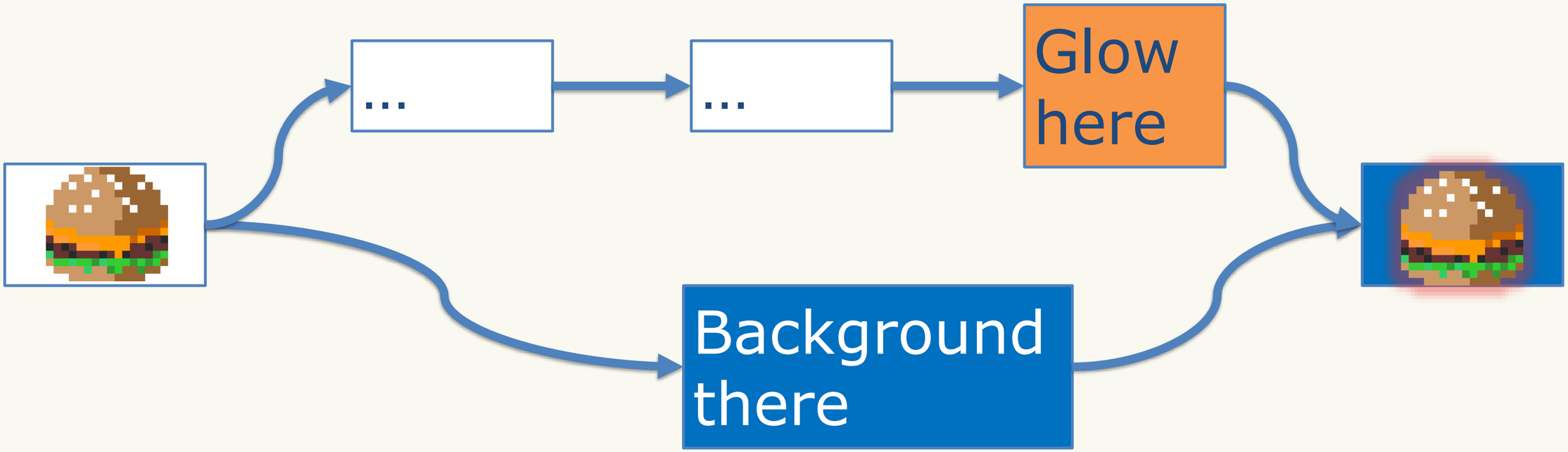
2		4		3	1	5
Vertex (attributes)		**Tessellation Ctrl.** (patches)	**Tessellation Eval.** (subdiv. out)	**Geometry** (primitives)	**Fragment** (raster out)	**Compute** (whatever)

Shaders first!

- Simplified OpenGL shader pipeline:

Shaders first!

- Tools for success (my recommendations):
- For non-programmers to get started, any visual editor!



Shaders first!

- Tools for success (my recommendations):

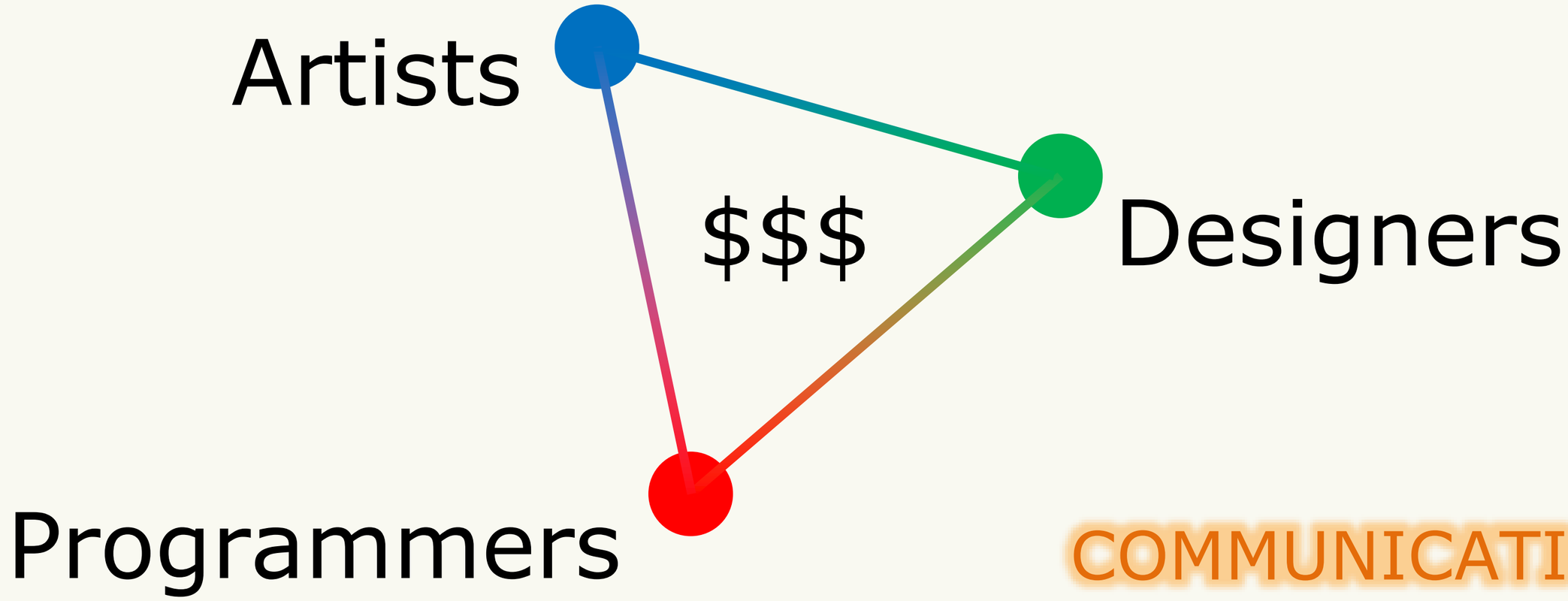
Shadertoy

shadertoy.com

shadered.org

Shaders first!

- SHADERS FOR ALL!!!



COMMUNICATION!!!

Shaders first!

- Cross-discipline in nature: engineers, artists, designers
- Understand and explain why
  - Interviews: hardest problem you've solved
- Cross-discipline goals
  - Engineers: build tools to make prototyping easier for others
  - Artists/designers: gain appreciation for how it all works
- Go to conferences and talk to people

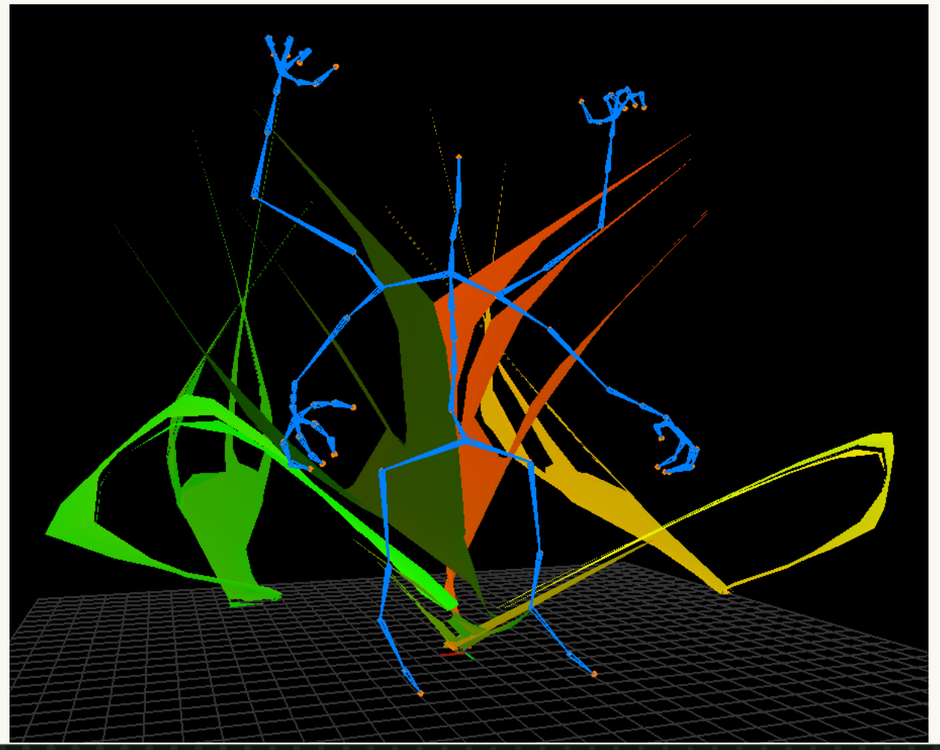
Shaders first!

- The pitfall:

It's just data!

Shaders first!

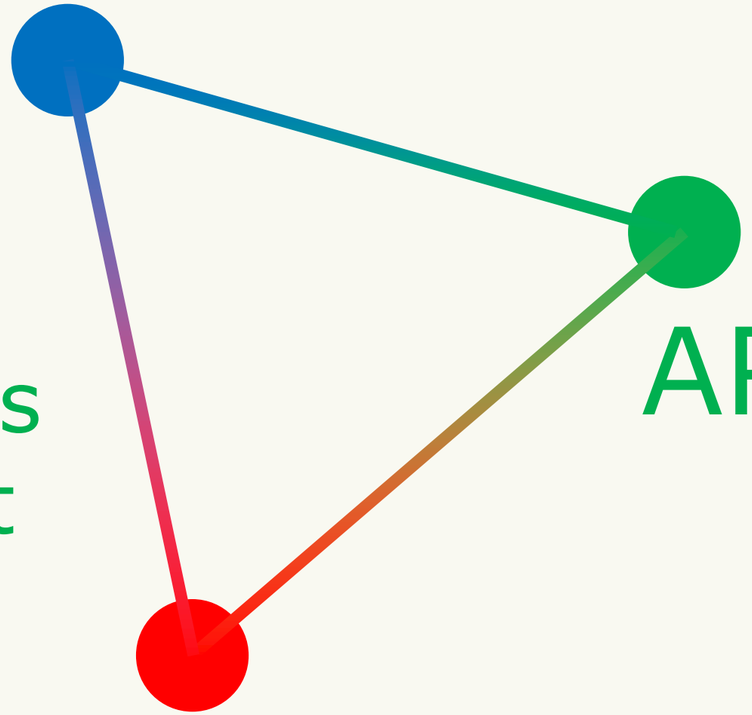
- Shaders are simply ***data readers/writers***
- They are crazy fast and they don’t give a damn



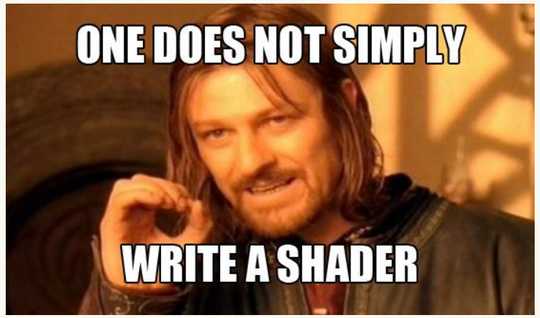
Where do I go next?

- BUILD A RENDERER!!! (or two… or six…)

->Engineering
->Abstraction
->Tackle hard probs
->Pull your hair out

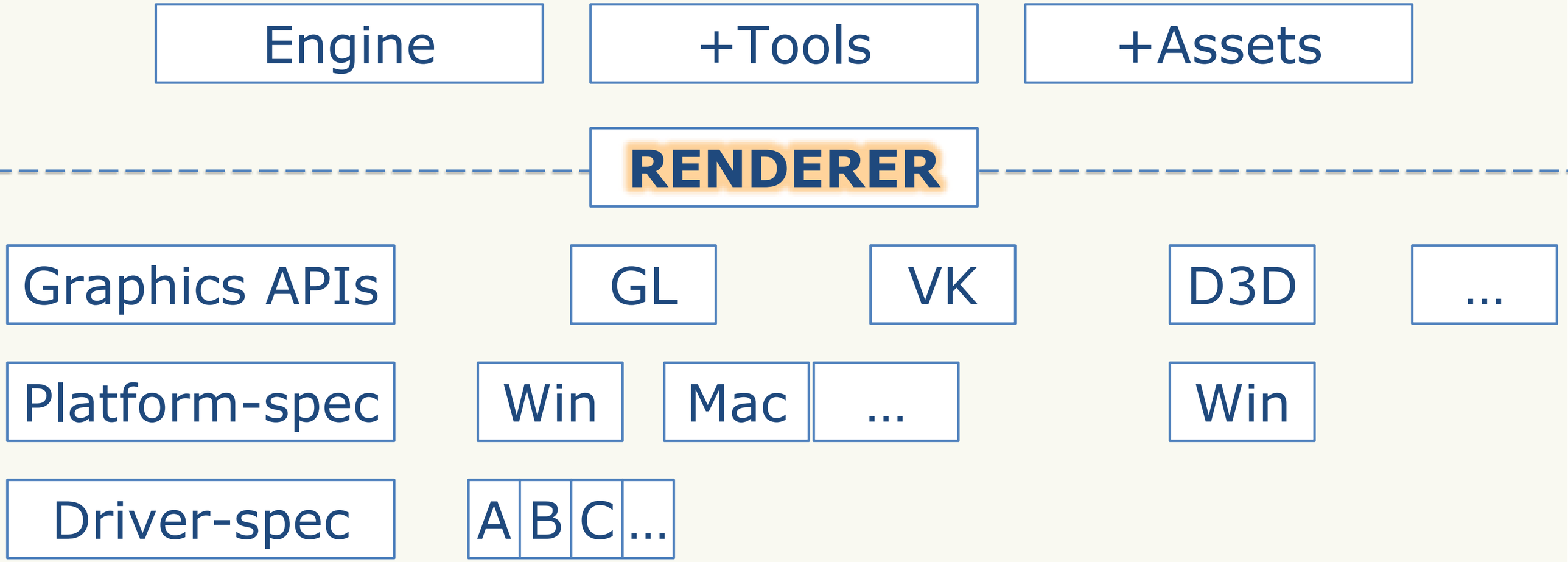


ARCHITECTURE



GL=OpenGL
VL=Vulkan
D3D=Direct3D

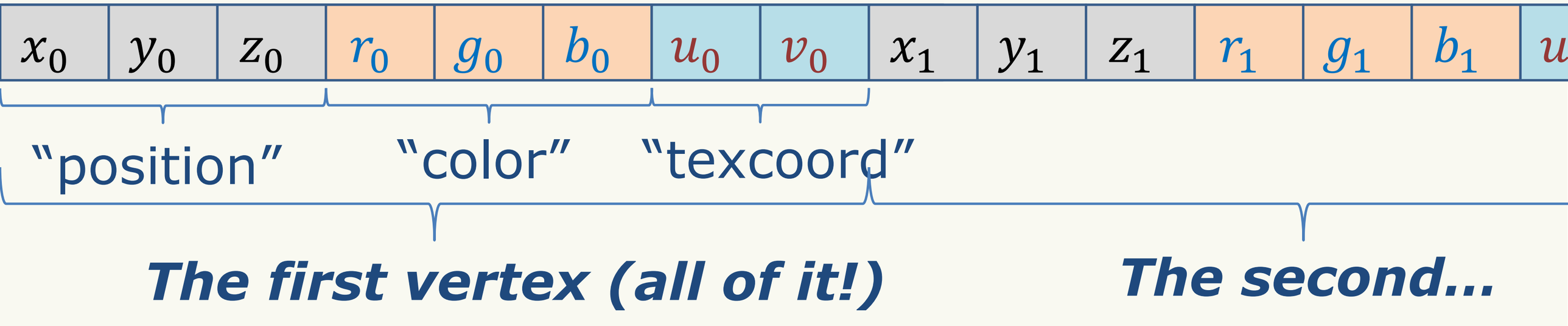
What's a renderer?



What's a renderer?

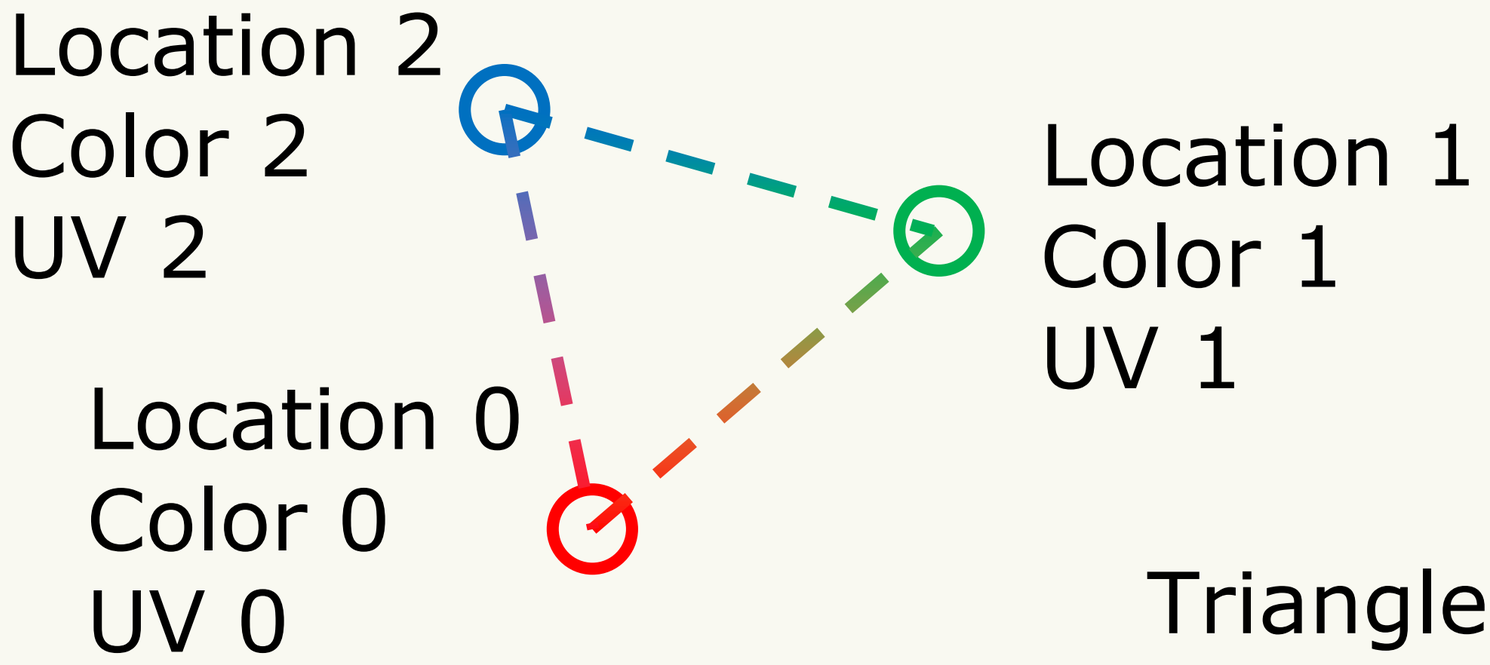
- It's just data: food for shaders
- Need to think abstractly!!!

Data in interleaved vertex buffer:



What’s a renderer?

- It’s just data: food for shaders
- Need to think abstractly!!!



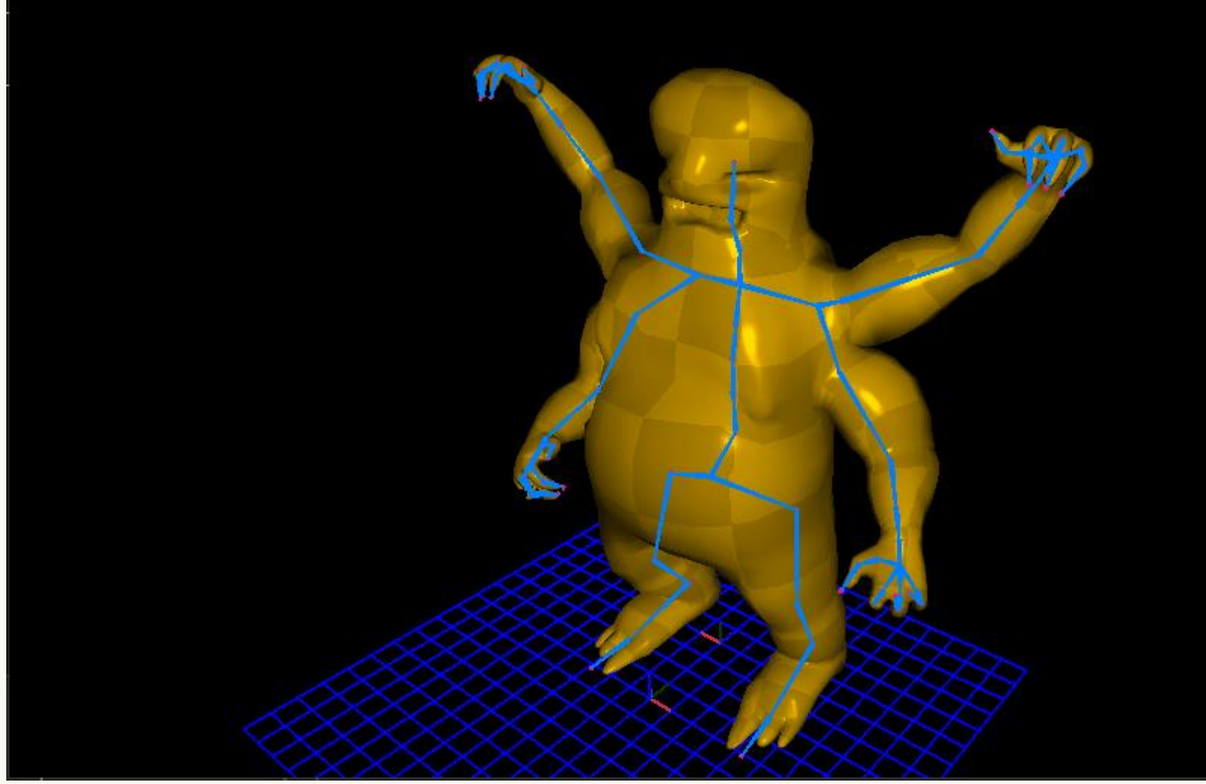
What’s a renderer?

- It’s just data: food for shaders
- Need to think abstractly!!!

https://gifimage.net/shrek-gif-image-for-whatsapp-and-facebook-16/

A brief history…

- **CBTK**
- 2012 – 2016
- Windows & iOS – C++
- OpenGL 3.3, OpenGL ES 2

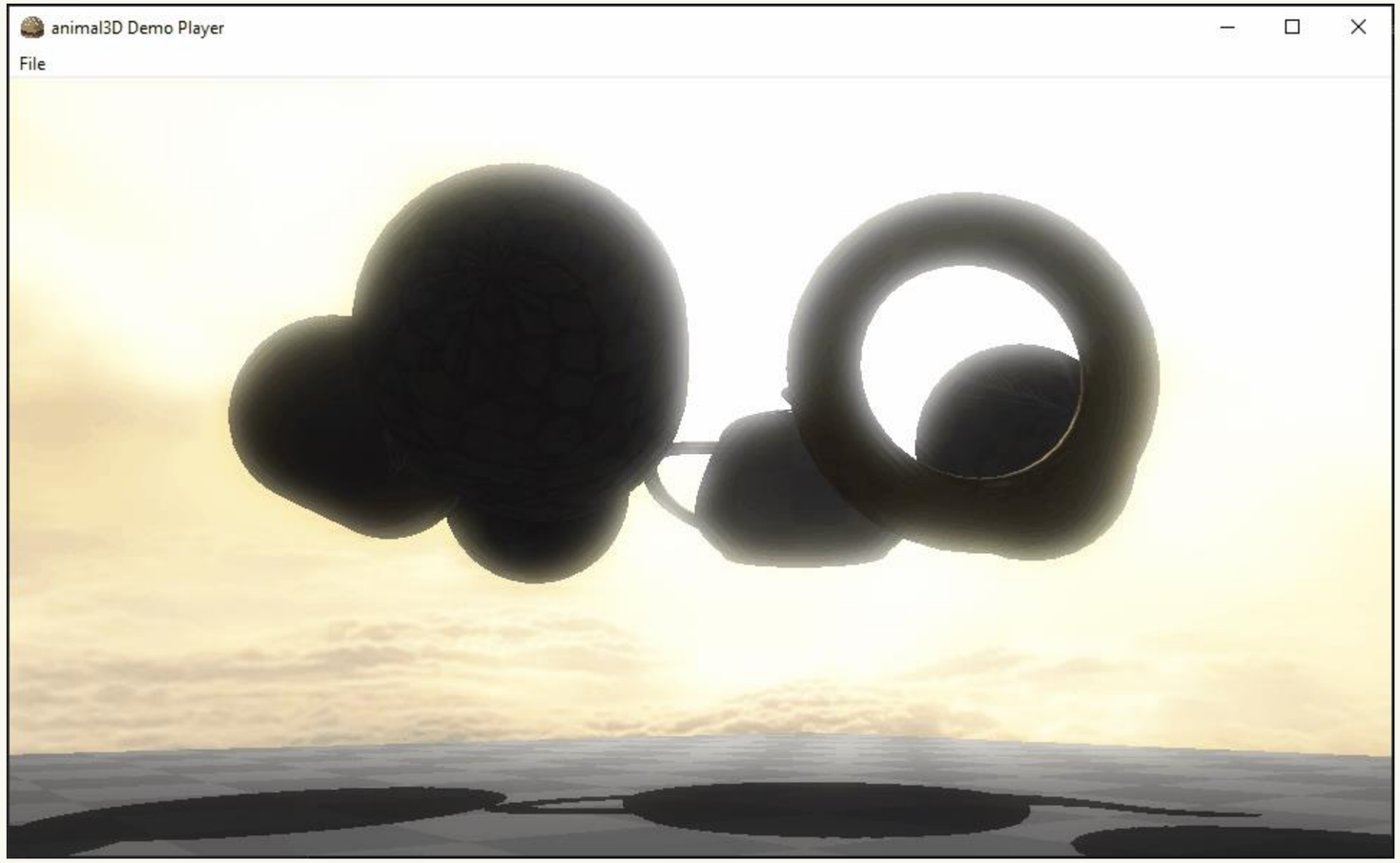


A brief history…

- **EGP**
- 2016 – 2017
- Windows – C++
- OpenGL ~~4.3~~ 4.1
- Quick Mac port

A brief history…

- ***animal3D***
- 2017 – 2021
- Windows – **C** ←
- OpenGL 4.5



A brief history…

- **Current project**
- 2020 – present
- Windows – C (want: Mac/iOS/Android)
- OpenGL, Vulkan (want: Metal/D3D)
- Relying on takeaways from animal3D

Starting over

- Two approaches:

Shoot first
Ask questions later

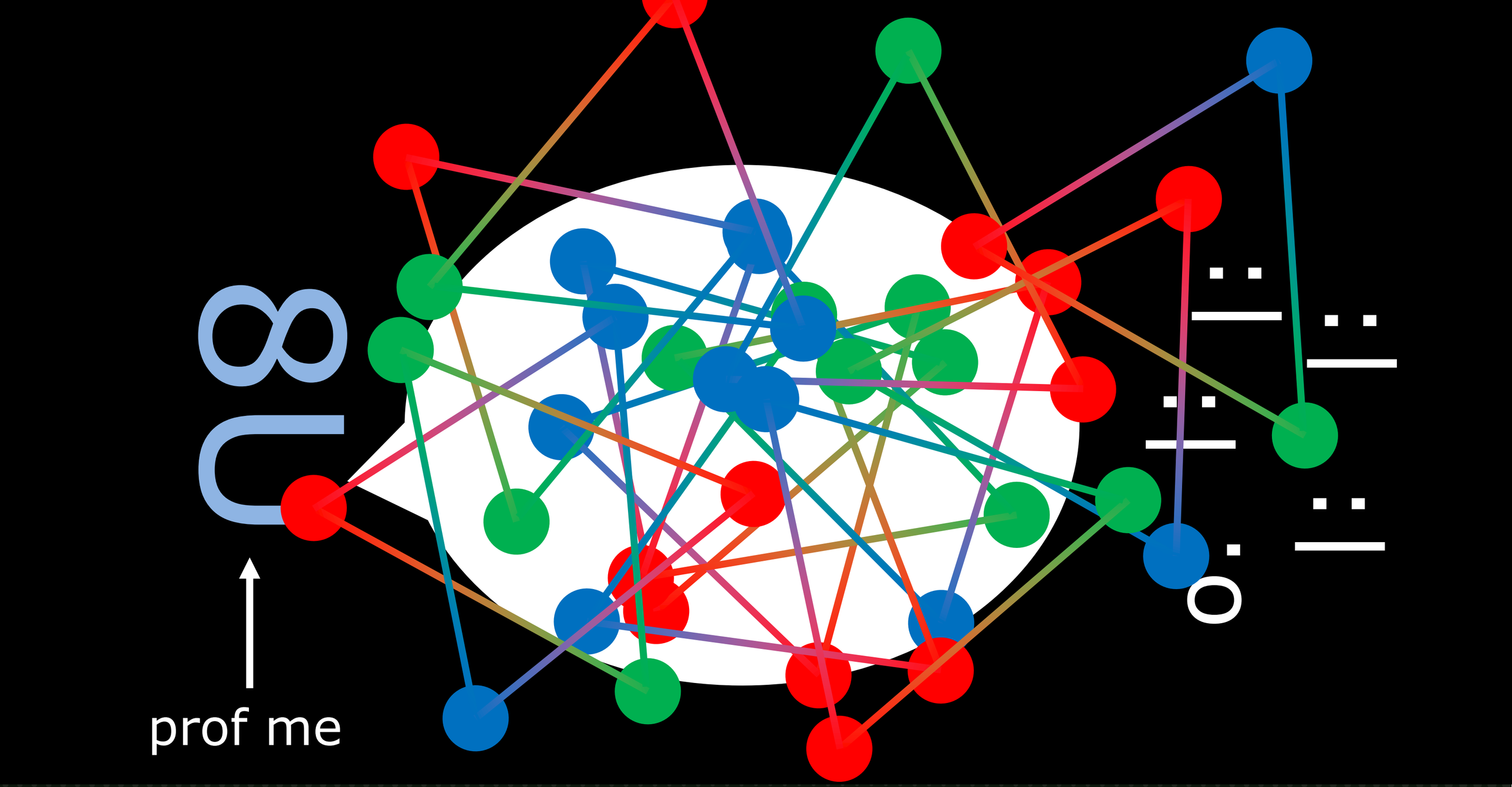
→Follow tutorials
→Essentials only
→Use existing SDKs

Ask questions first
Shoot later

→ Prerequisites
→ Prediction
→ Iteration

Starting over

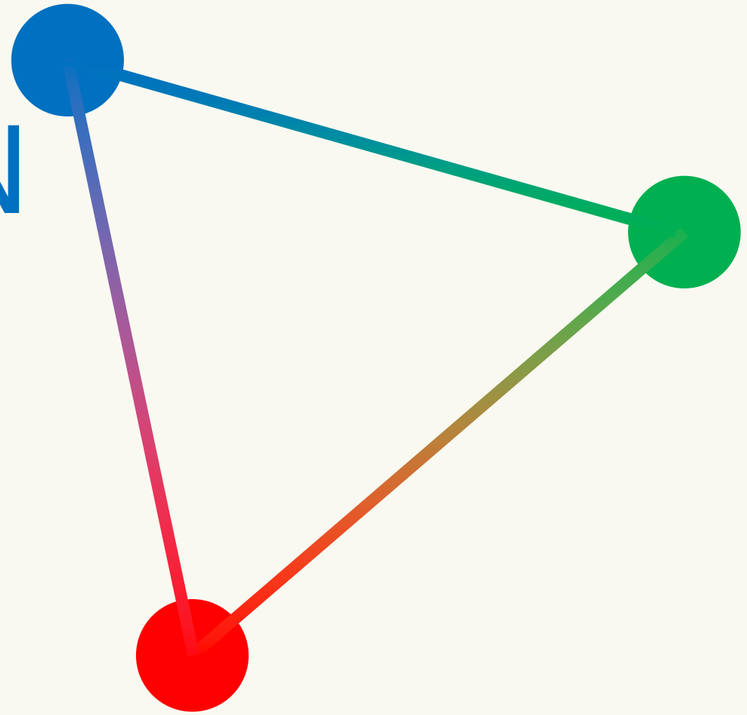
- Why/when to start over?
- Need to fulfill some greater purpose
  - E.g. teaching tools; needs of project
- Existing framework has weaknesses
  - E.g. too rigid; too many dependencies
- Iteration
  - E.g. new experiments, new setup


8
prof me

Where do I go with my renderers?

- Three main components of graphics programming:

INNOVATION

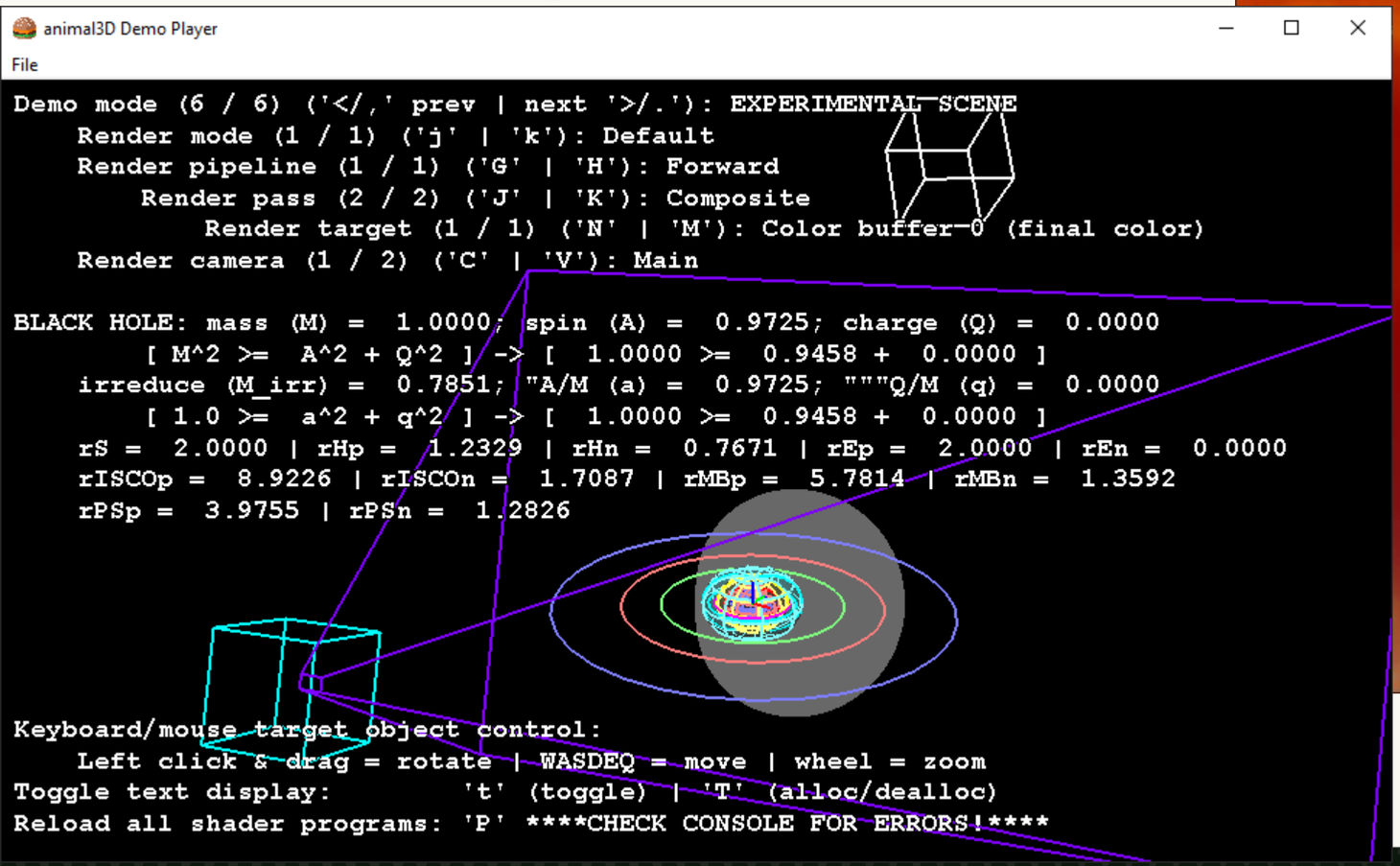


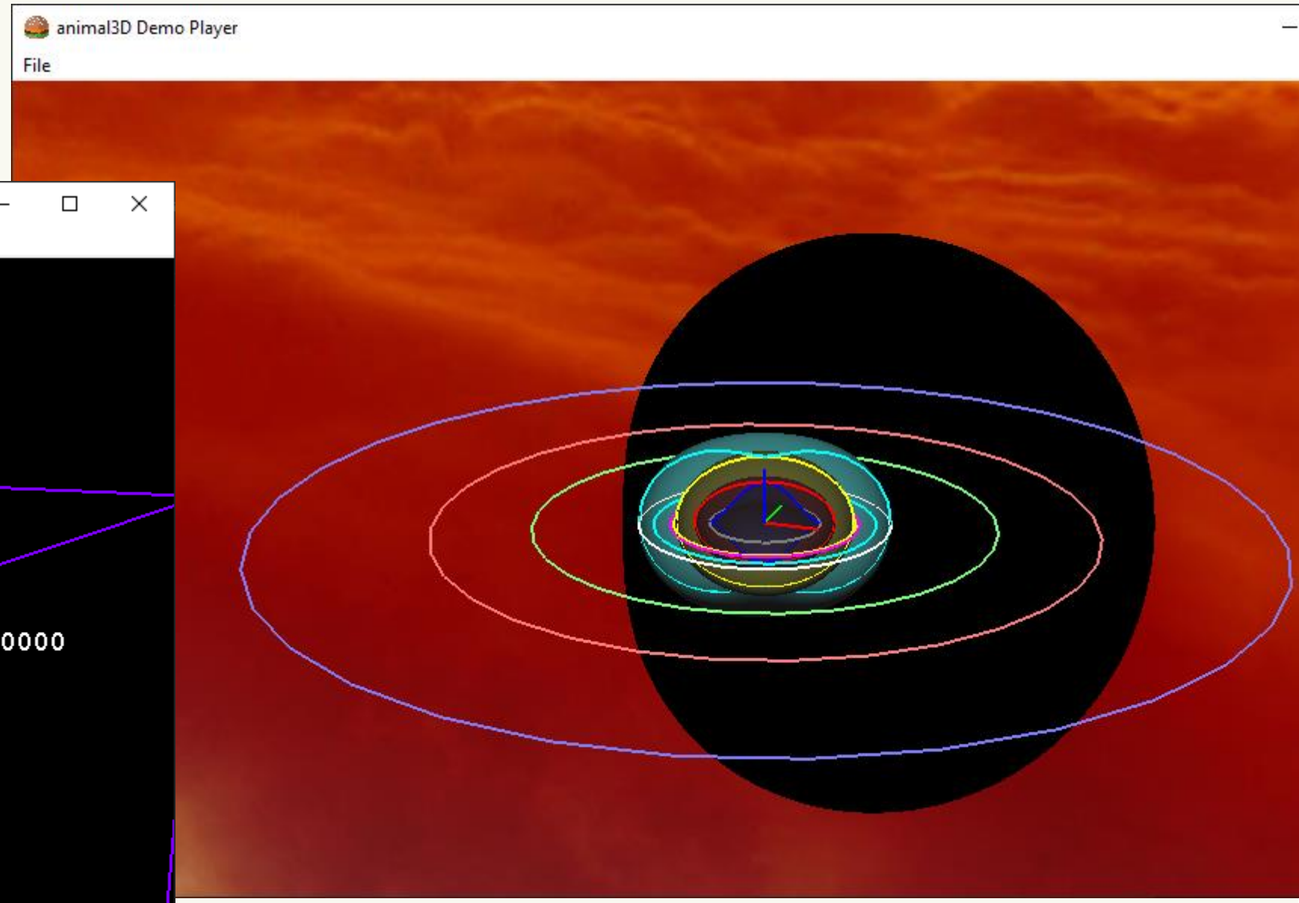
->Bend the rules
->Deep learning
->Teach others
->Get a job

CURIOSITY!!!

Where do I go with my renderers?

- Time for space…time…

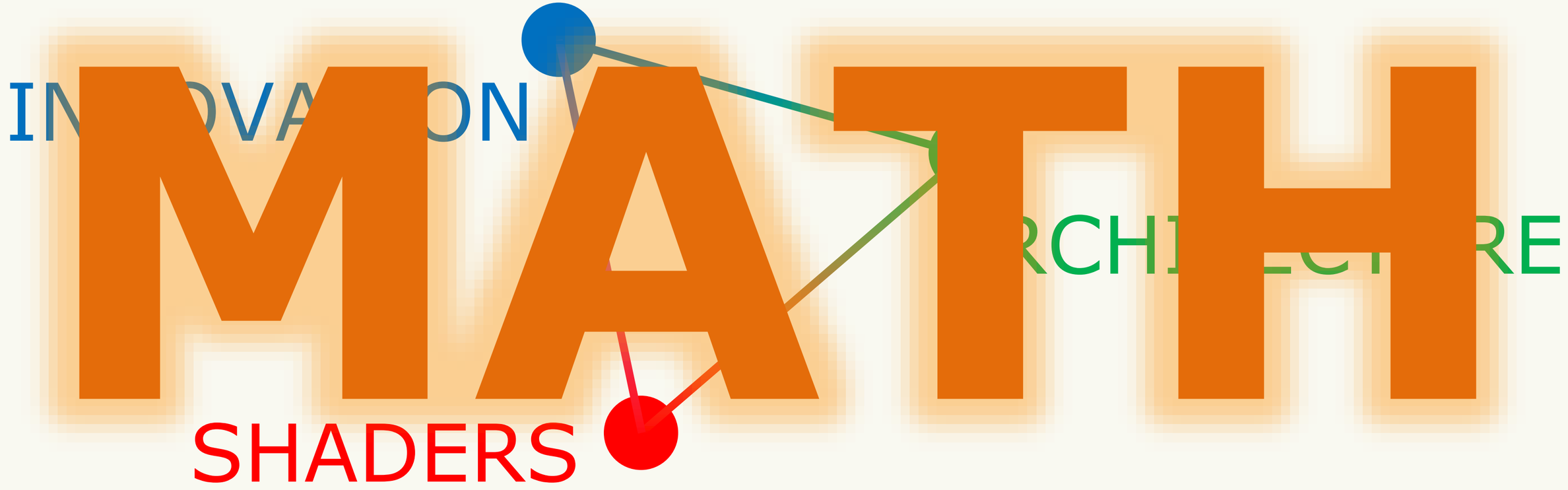




*my hobby: drawing & learning about spacetime anomalies

WOW!!!

- Now you know everything… right?

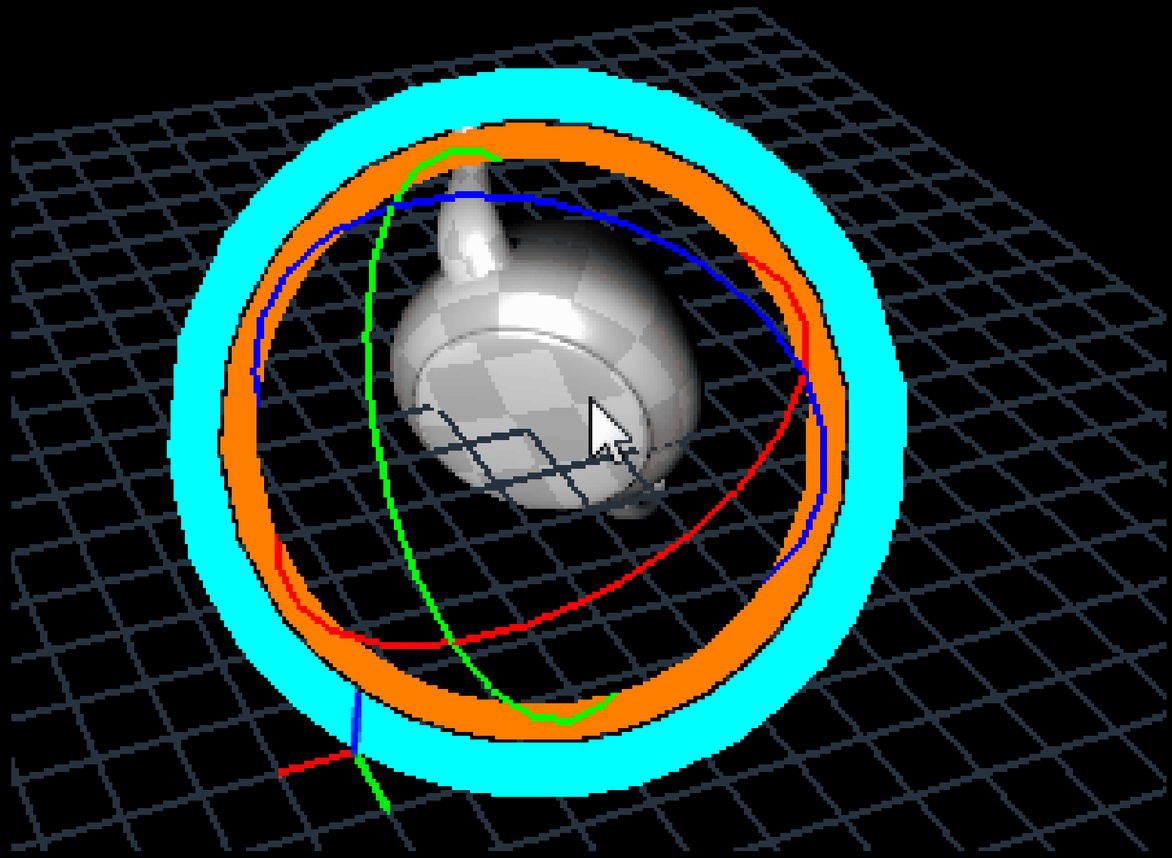


“We looked at your resume because… math.”

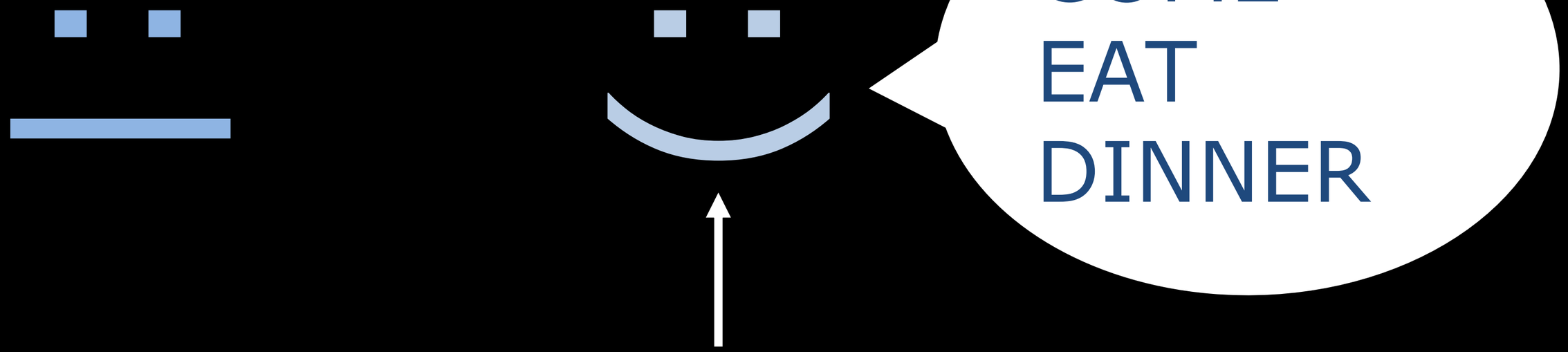
-my manager, 2021

Mathematics

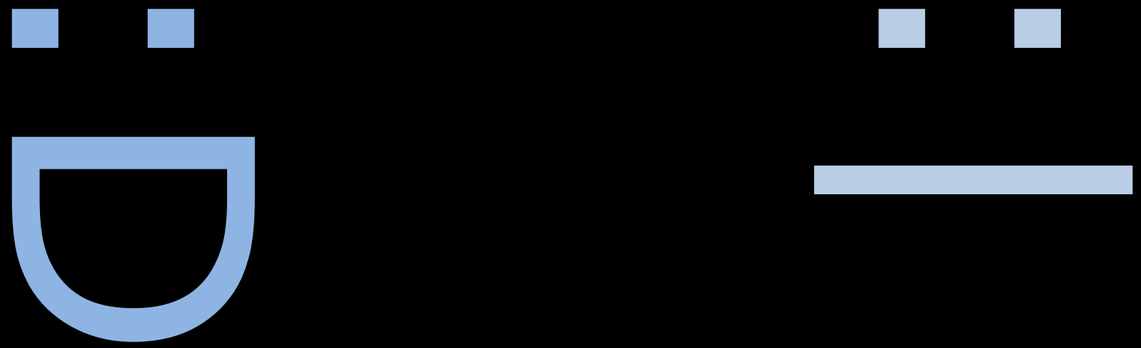
- Important types of math
  - Basic algebra: functions, trigonometry
  - Calculus: derivatives, integrals
  - Linear algebra: vectors, matrices, quaternions (bonus)
- Misconception: total pre-requisite of graphics?
  - It definitely helps to know some, but…
- Actually: great way to learn math through application
- Interested in a math problem? Visualize it!



Ken Shoemake’s arcball paper: http://graphicsinterface.org/wp-content/uploads/gi1992-18.pdf


COME
EAT
DINNER
significant
other

$$\text{slerp}_{q_0,q_1}(t) = \frac{\sin([1-t]\Omega)\, q_0 + \sin([t]\Omega)\, q_1}{\sin \Omega}$$



Mathematics

- Solve random but relevant problems
- E.g. prove the quadratic formula:

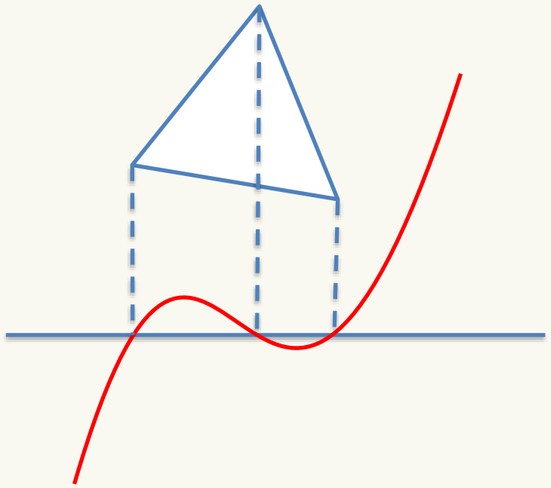
$$0 = ax^2 + bx + c \qquad x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

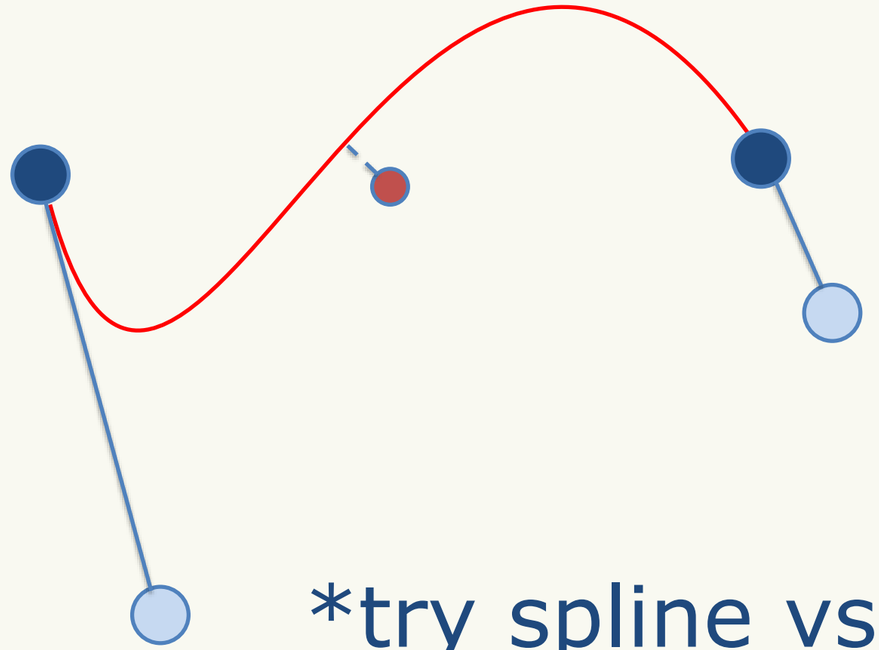
- Then optimize it:

$$\text{If } \; a = 1 \; \text{ and } \; b = 2B \; \text{ then } \; x = -B \pm \sqrt{B^2 - c}$$

Mathematics

- Solve random but relevant problems
- E.g. cubic roots: $0 = ax^3 + bx^2 + cx + d$

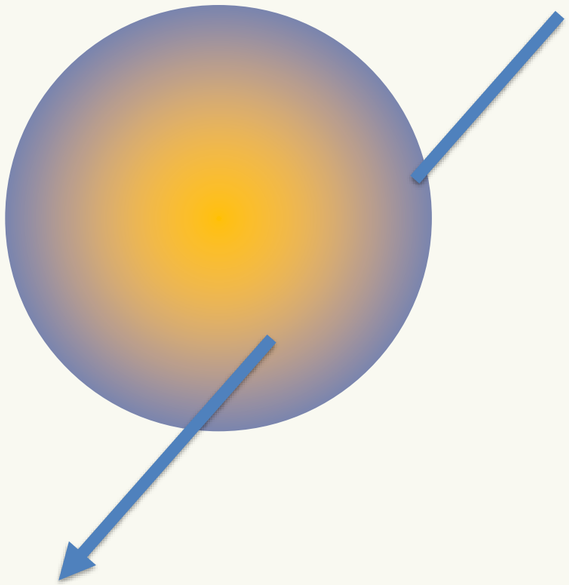


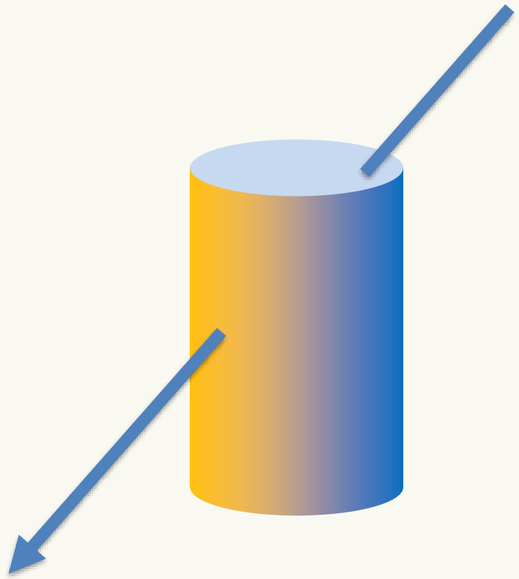


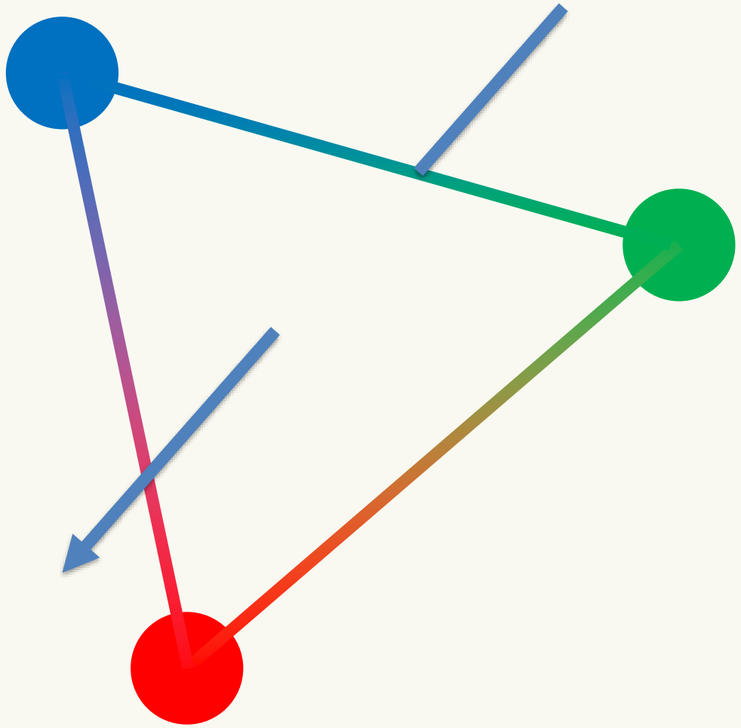
*try spline vs plane

Mathematics

- Solve random but relevant problems
- E.g. ray vs shapes & convex hulls

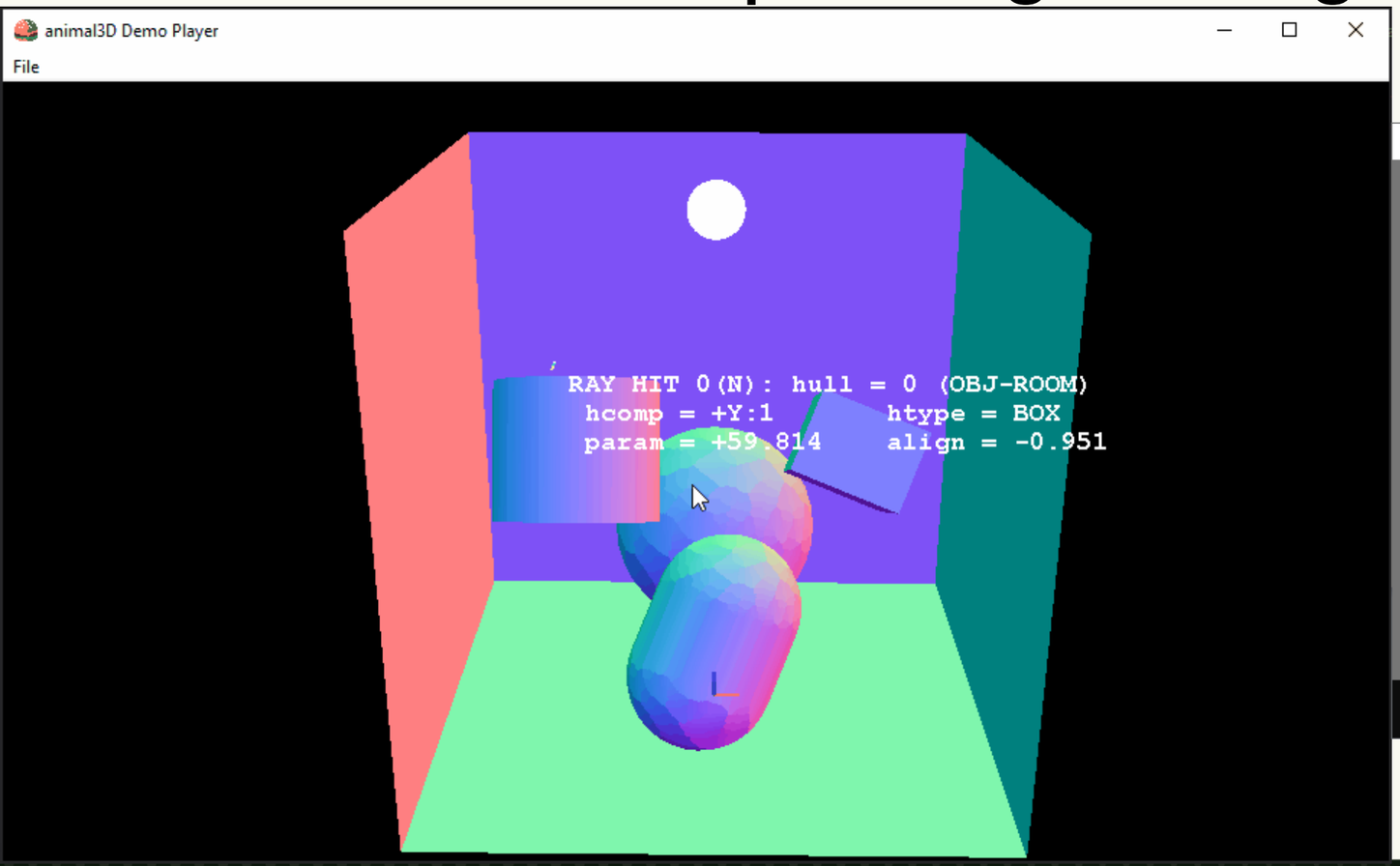




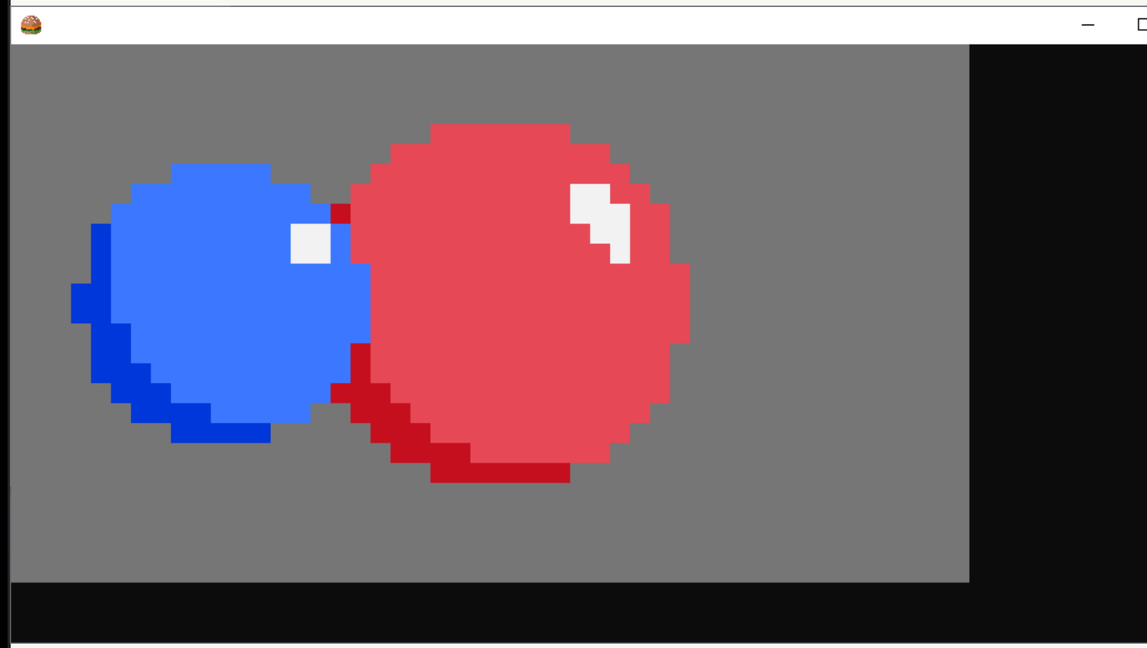


Mathematics

- Learn modern paradigms: e.g. raytracing



←ray debugging



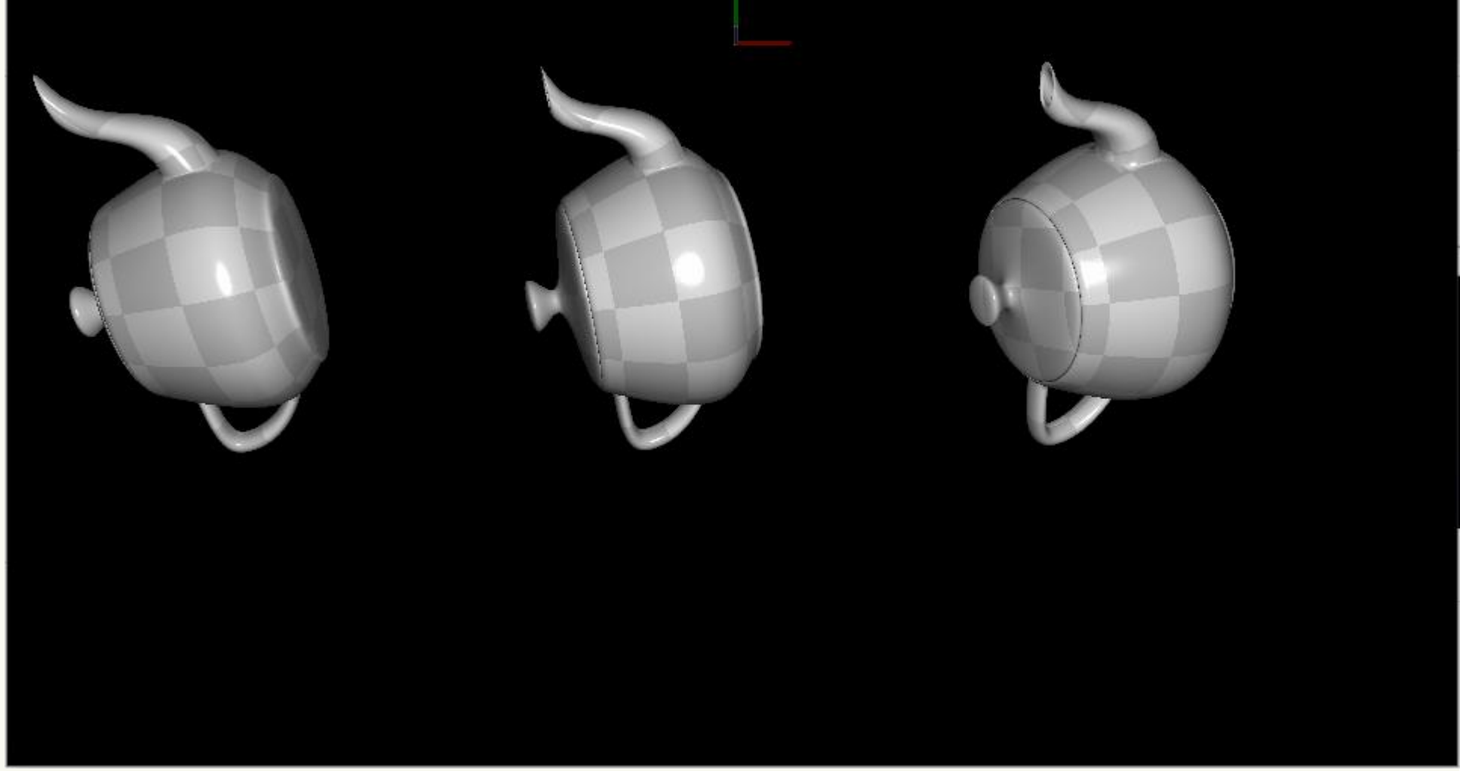
^using console as canvas

Mathematics

- E.g. quaternions!!!

vs

matrices!!!

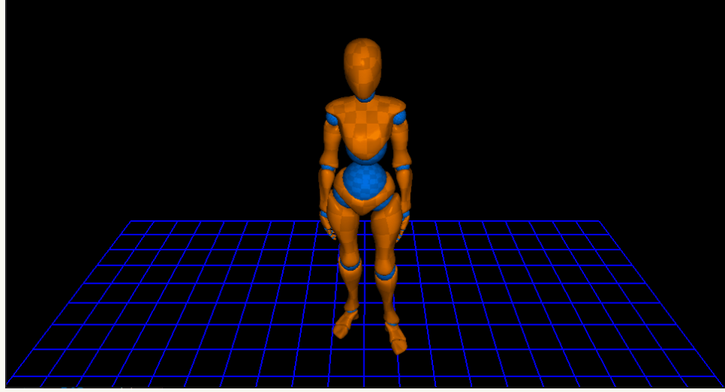


*left: interpolating matrix results in scale and skew
*middle: quaternion slerp results in smooth rotation
*right: dual quaternion sclerp results in arc motion

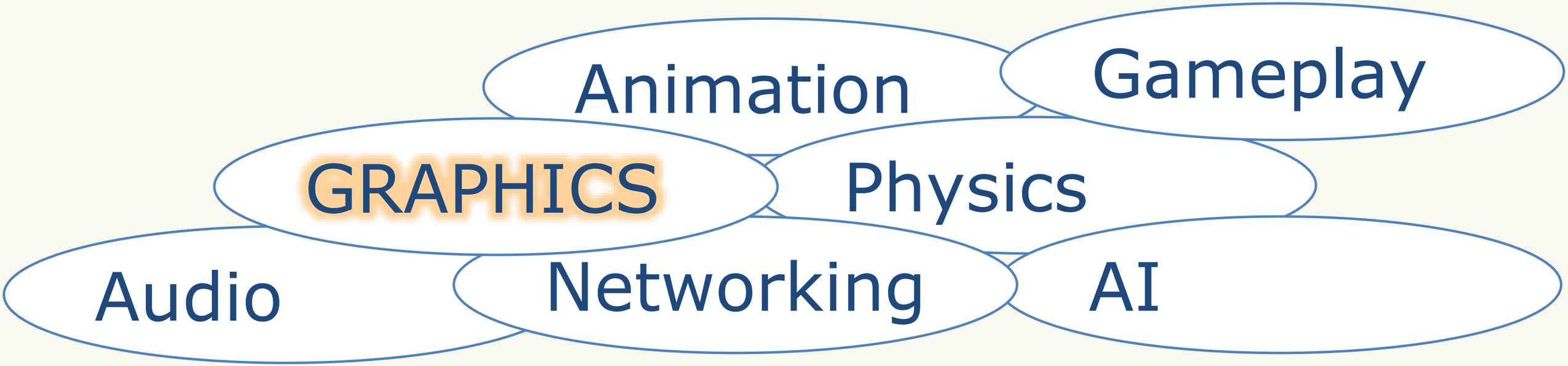
Mathematics

Write your own math library

Job Opportunities

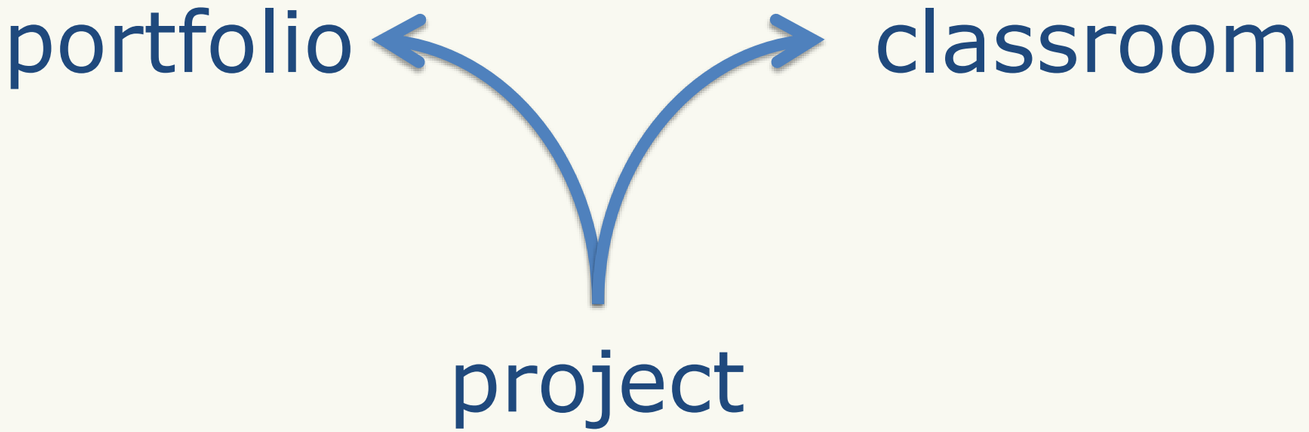


- Start here, go anywhere



Academia to Industry

- Balance research with practice



- Note to industry: not all academics are researchers
  - (For those looking to switch, focus on portfolio!)

Learning

- Write EVERYTHING down
  - Personal writeups and articles
- Comment all code
  - For future you
- Write a blog
- Publish your work (e.g. on GitHub, conference talks)

Transferrable Skills!!!

- Math
- Abstraction
- Patience
- Curiosity
- Communication
- Learning

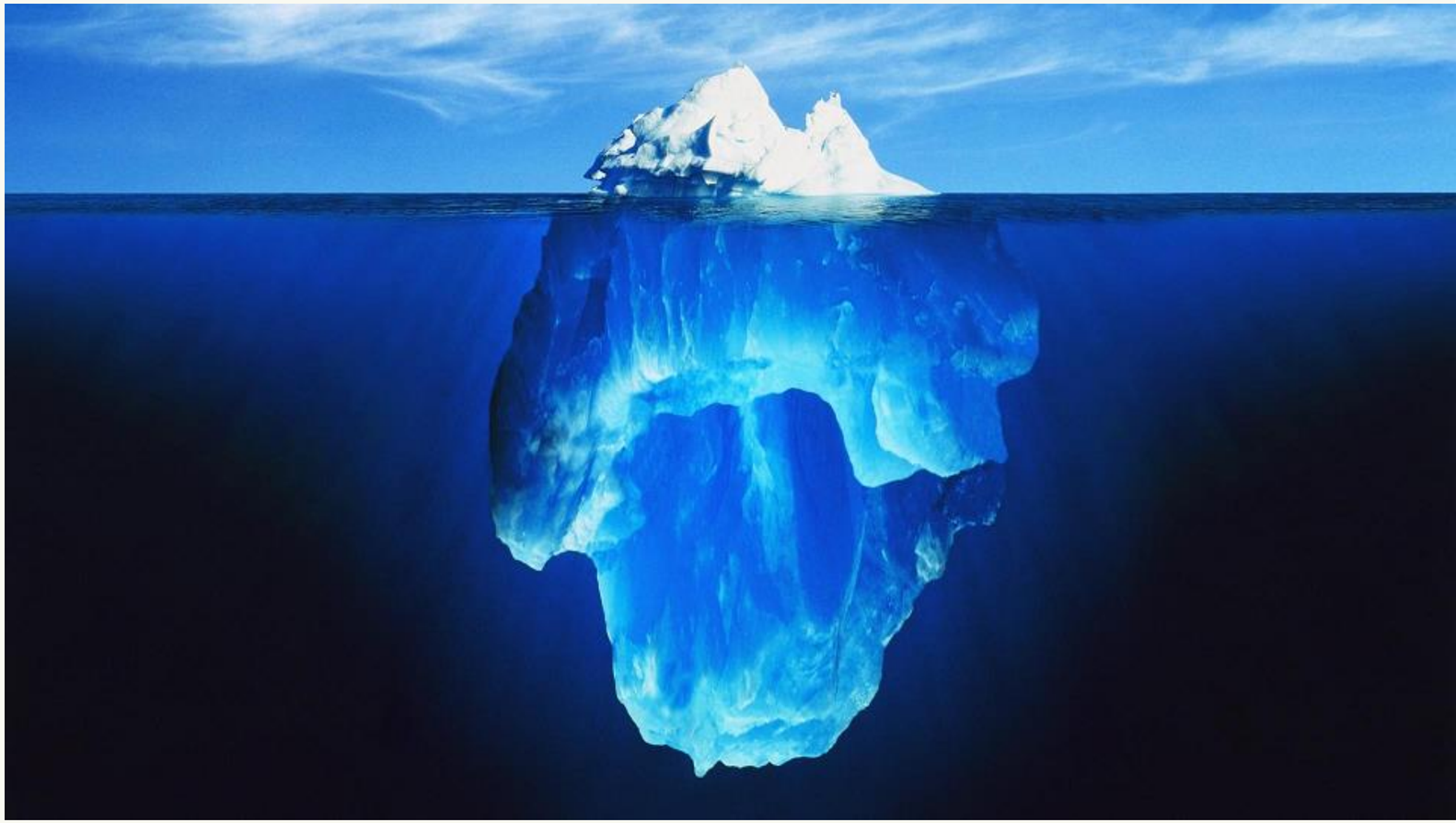
Perspectives

- What I think I know

vs

- What I know I don’t

Live Long and Render!



Resources 4 U

- Shadertoy: shadertoy.com
- SHADERed: shadered.org
- animal3D: github.com/dbuckstein/animal3D-SDK-Source
- Teaching materials: github.com/dbuckstein/teaching
- Ray Tracing in One Weekend: raytracing.github.io
- Vulkan Tutorial: vulkan-tutorial.com

- Twitter, LinkedIn, GitHub: **dbuckstein**

We’re hiring!!!

- Infinity Ward is hiring Engineers across our four studio locations: Los Angeles, Austin, Mexico City & Krakow!
- For more information, please visit:

careers.infinityward.com

Thank You & Enjoy Your GDC ☺

- Special thanks:
  - GDC
  - Infinity Ward & Activision
  - Family, friends, colleagues & mentors