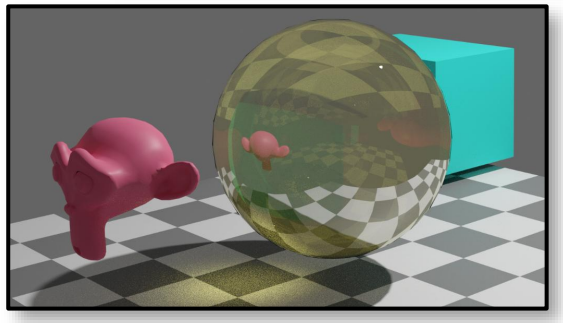
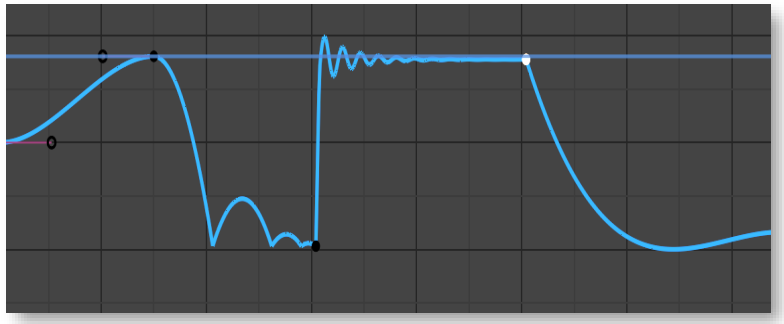
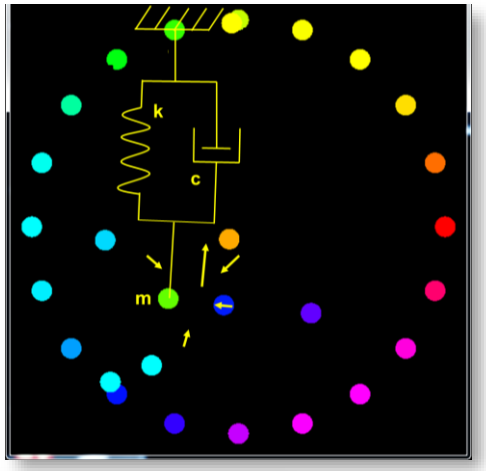
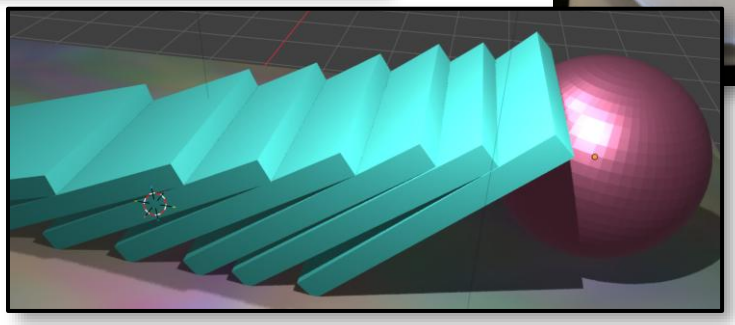
A Whirlwind Introduction to Computer Graphics!

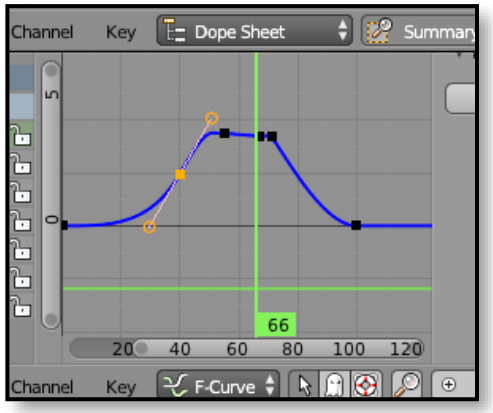
Mike Bailey
Oregon State University
mjb@cs.oregonstate.edu

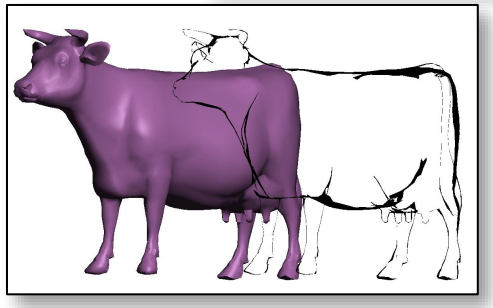












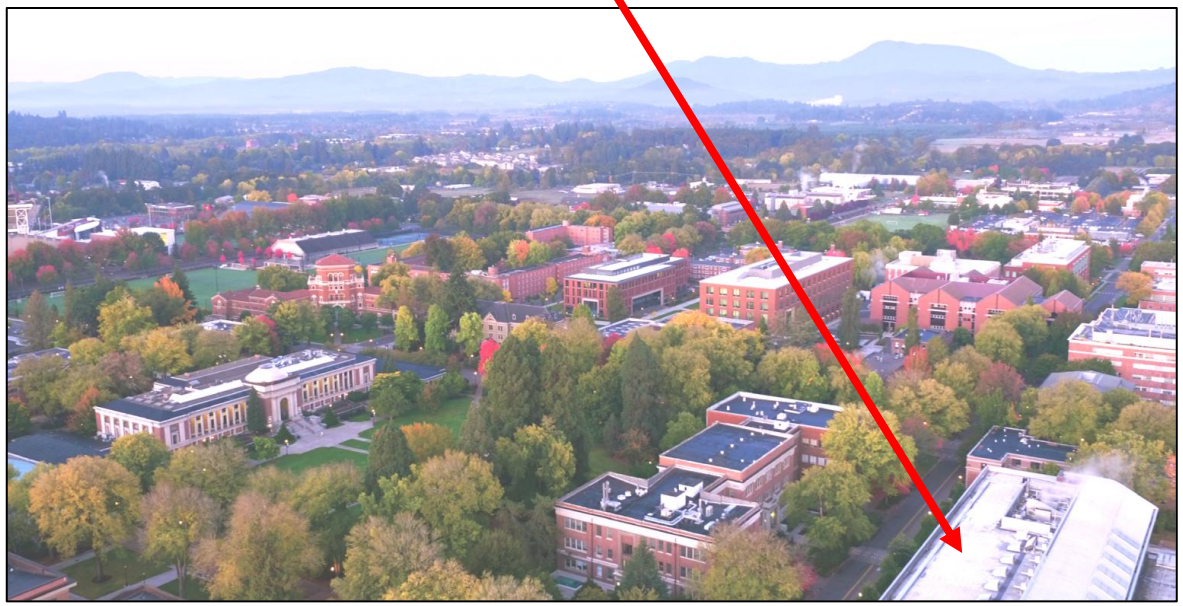
http://cs.oregonstate.edu/~mjb/whirlwind

SIGGRAPH 2026
Los Angeles 19–23 JUL

Mike Bailey

Oregon State University
Computer Graphics

- Professor of Computer Science, Oregon State University
- Has had over 15,000 students in his university classes
- Has taught over 100 conference and workshop short courses in CG and Visualization
- mjb@cs.oregonstate.edu



http://cs.oregonstate.edu/~mjb/whirlwind

http://cs.oregonstate.edu/~mjb/whirlwind

Course Learning Objectives

At the end of this course, you will know:

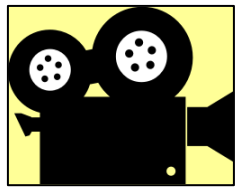
- The meaning of a lot of the jargon describing the amazing things at SIGGRAPH 2026. We call this “buzzword compliant”. ☺
- Some of what it took to make the images and animations that you will see
- How to find references for further study

http://cs.oregonstate.edu/~mjb/whirlwind

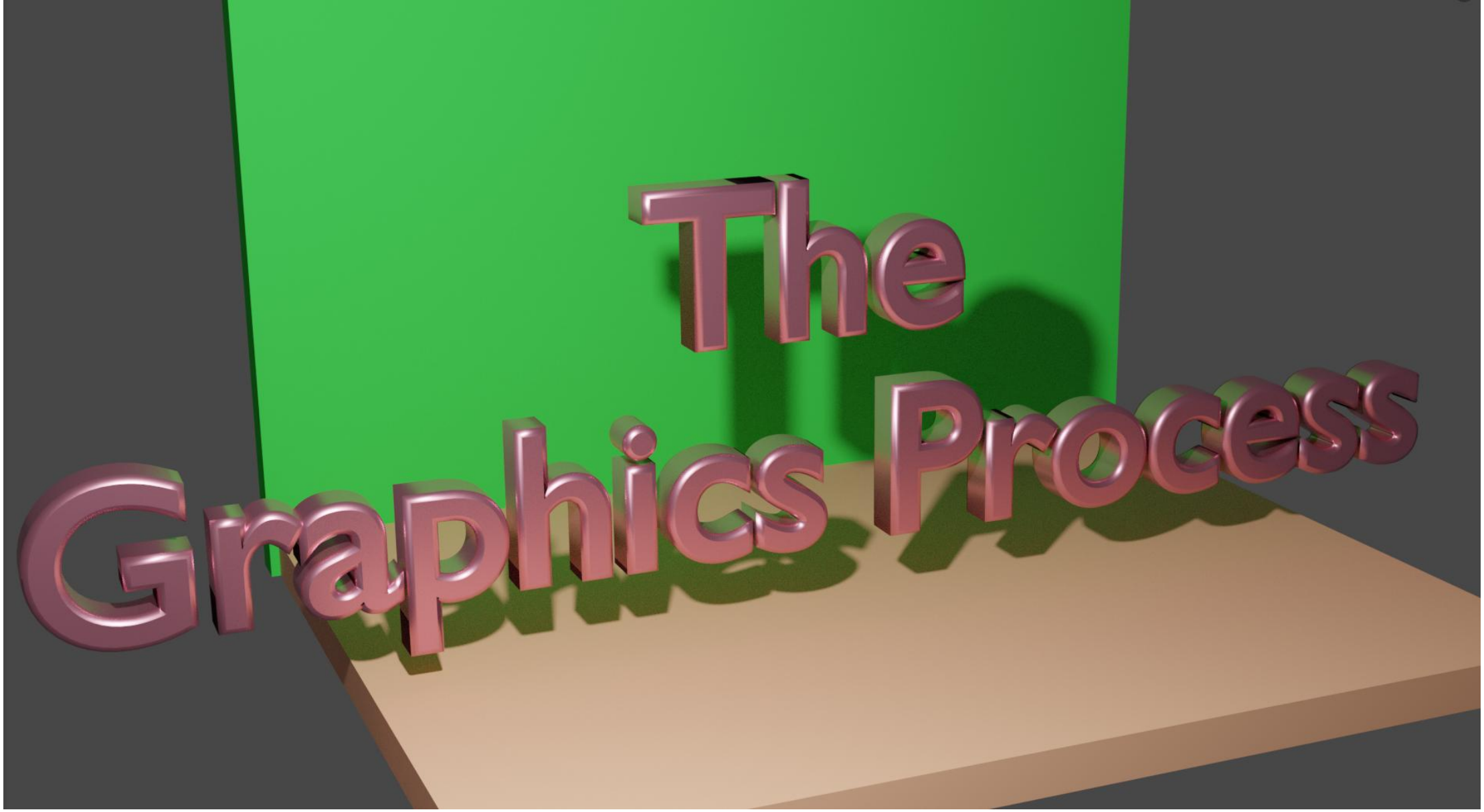
Schedule

1. 0:05 How the computer graphics pieces fit together
2. 0:20 Modeling
3. 0:20 Animation
4. 0:30 Rendering
5. 0:05 Finding More Information
6. 0:10 Q&A

http://cs.oregonstate.edu/~mjb/whirlwind

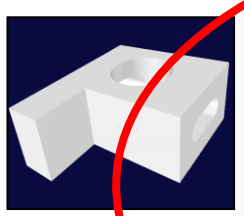


When you see a symbol like this, it means that there is a video on the Whirlwind page that you can watch for further information



What are all the pieces that go into making the graphics you will be see?
What does it take to make them?

The Process



3D Geometric Models

Surface Information

Texture Information

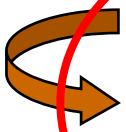
Lighting Information

Modeling

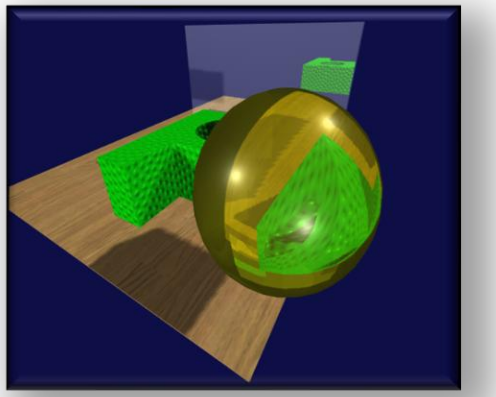
Rendering

Drawing

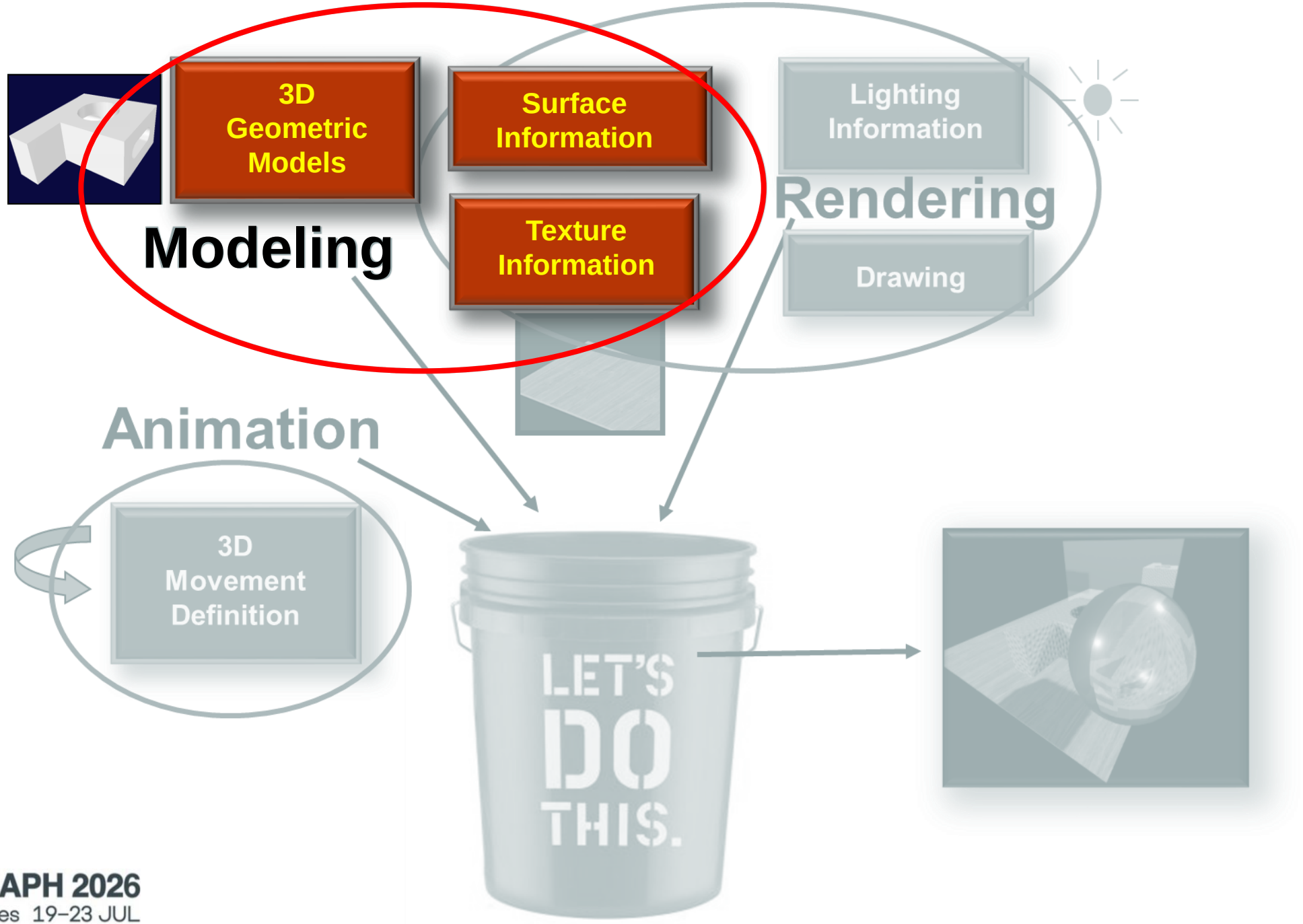
Animation



3D Movement Definition



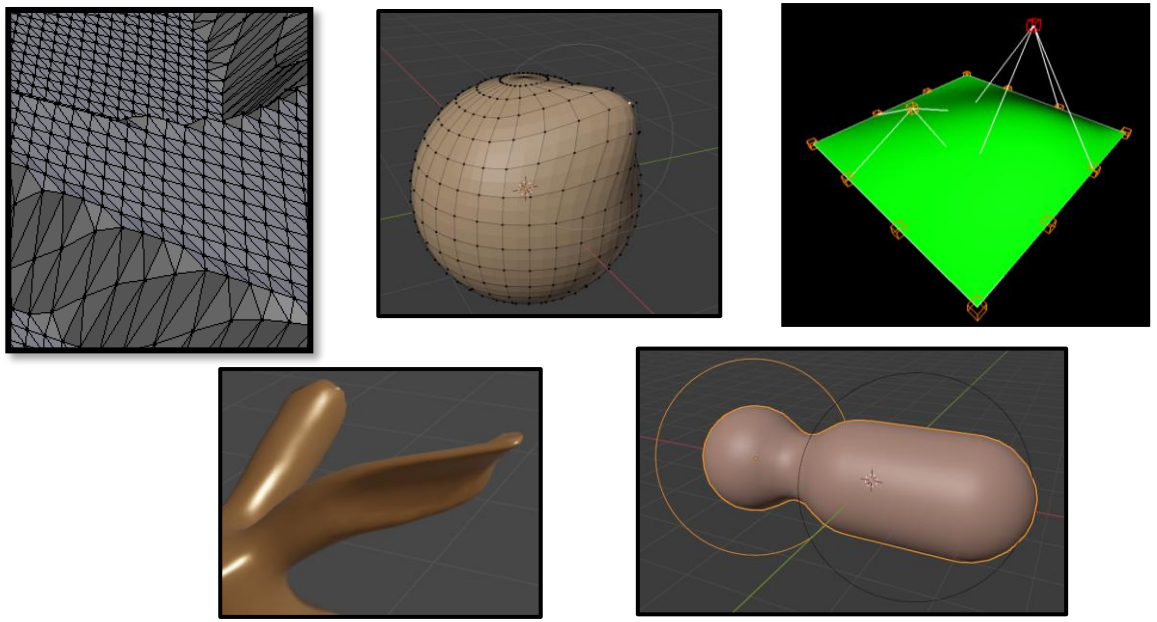
Creating 3D Geometry


3D Geometric Models
Surface Information
Texture Information
Modeling
Lighting Information
Rendering
Drawing
Animation
3D Movement Definition
LET'S DO THIS.

What do we mean by “Modeling”?

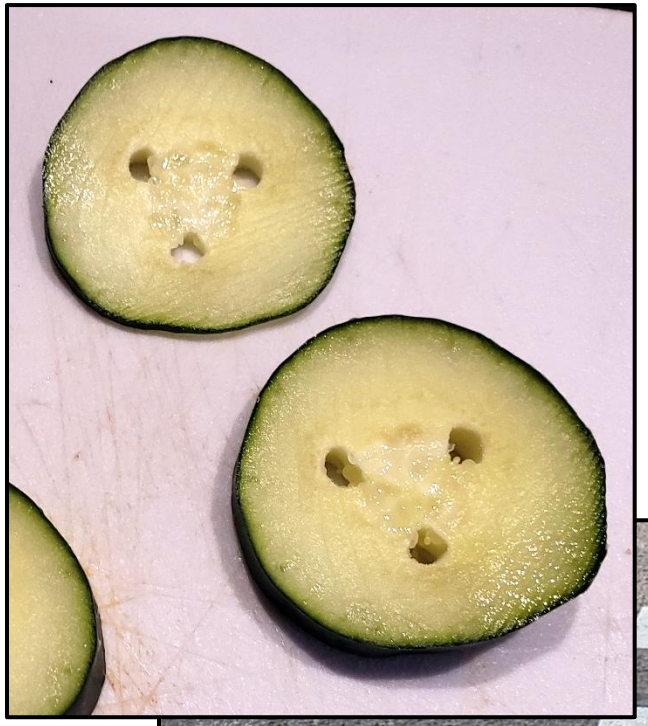
In computer graphics applications, how we model geometry depends on what we would like to use the geometry for:

- Looking at its appearance?
- Interacting with its shape?
- How does it interact with its environment?
- What is its surface area and volume?
- Does it need to be 3D-printed?
- Etc.



Want to experiment with some free modeling programs?
Want some notes to help you get started?
http://cs.oregonstate.edu/~mjb/blender
http://cs.oregonstate.edu/~mjb/tinkercad

First of all, realism can be *optional*.
For example, there are a lot of ways to “Model a Face”

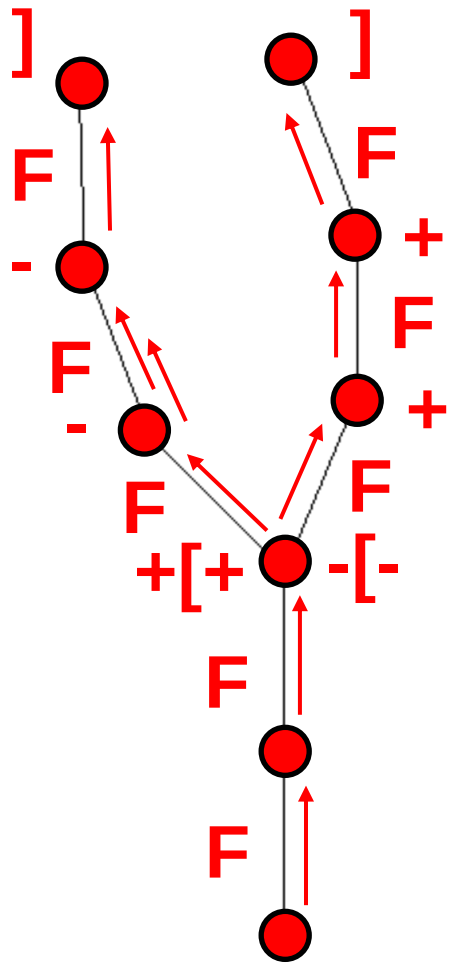


Modeling 3D Line Geometry: L-Systems

Introduced and developed in 1968 by Aristid Lindenmayer, L-systems are a way to apply grammar rules for generating fractal (self-similar) geometric shapes. For example, take the string:

"FF+[+F-F-F]-[-F+F+F]"

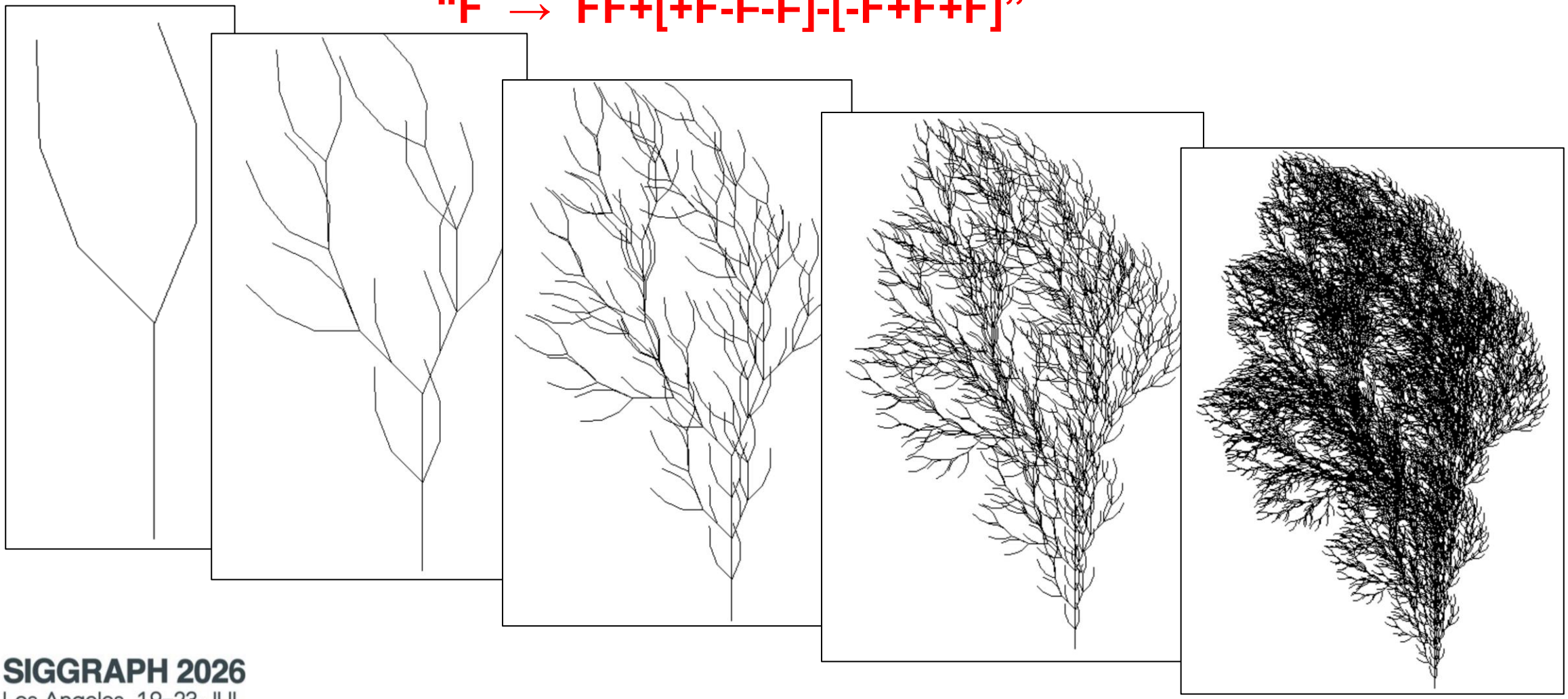
F	move forward one step
+	turn right
-	turn left
[	save position
]	restore position



Modeling 3D Line Geometry: L-Systems

But the *real* fun comes when you call that string recursively. For every **F**, replicate it with that entire string but with smaller geometry:

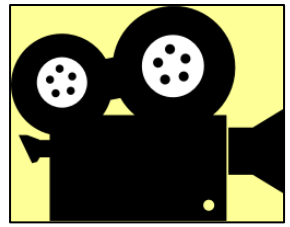
"F → FF+[+F-F-F]-[-F+F+F]"



Modeling 3D Line Geometry: L-Systems

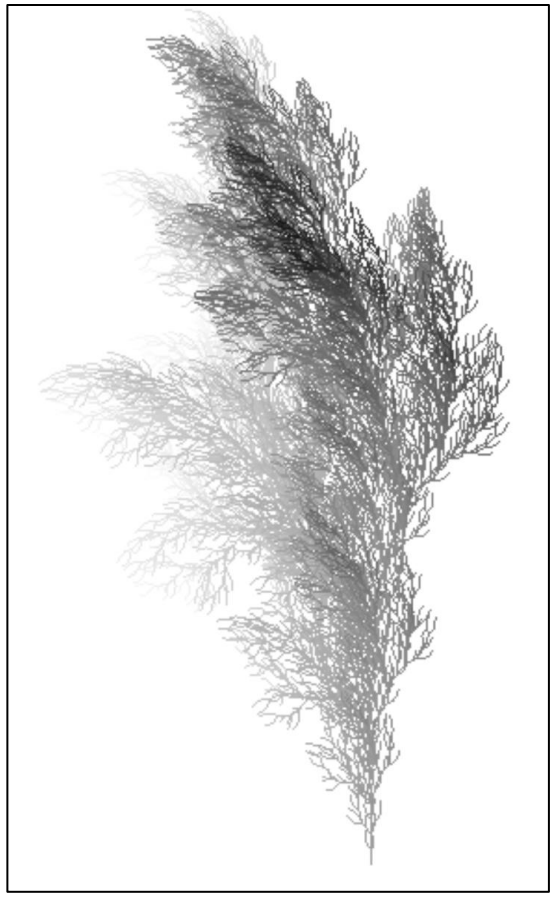
And, of course we can introduce more grammar to swing it into 3D

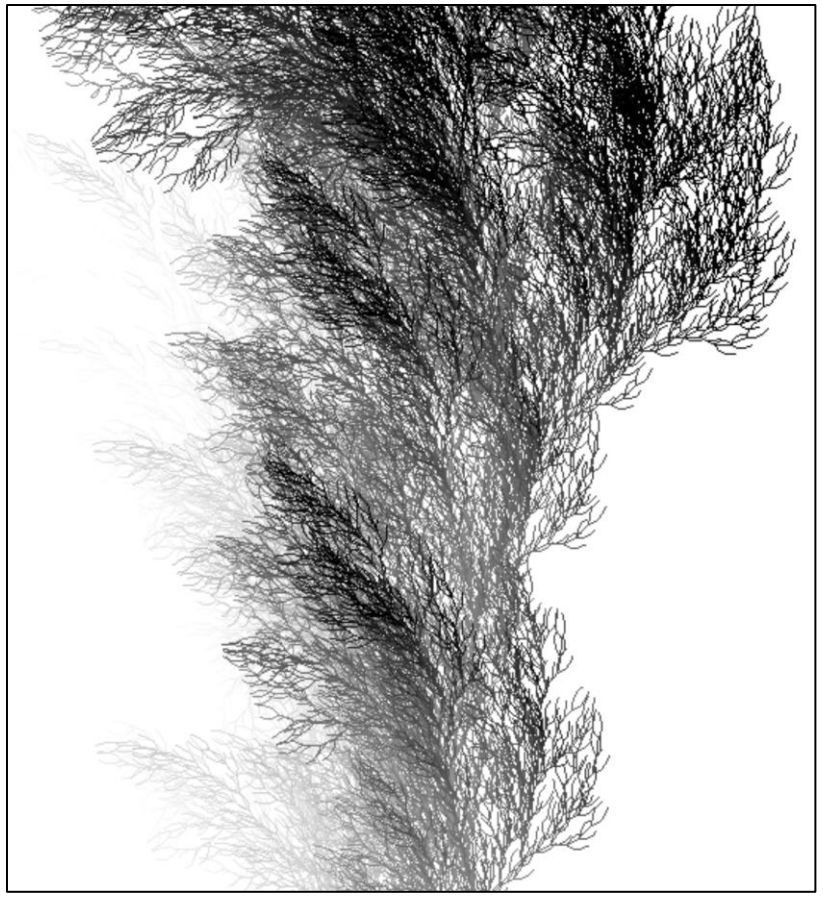
“F → FF+[+F-<F->F]-[-F+^F+vF]”



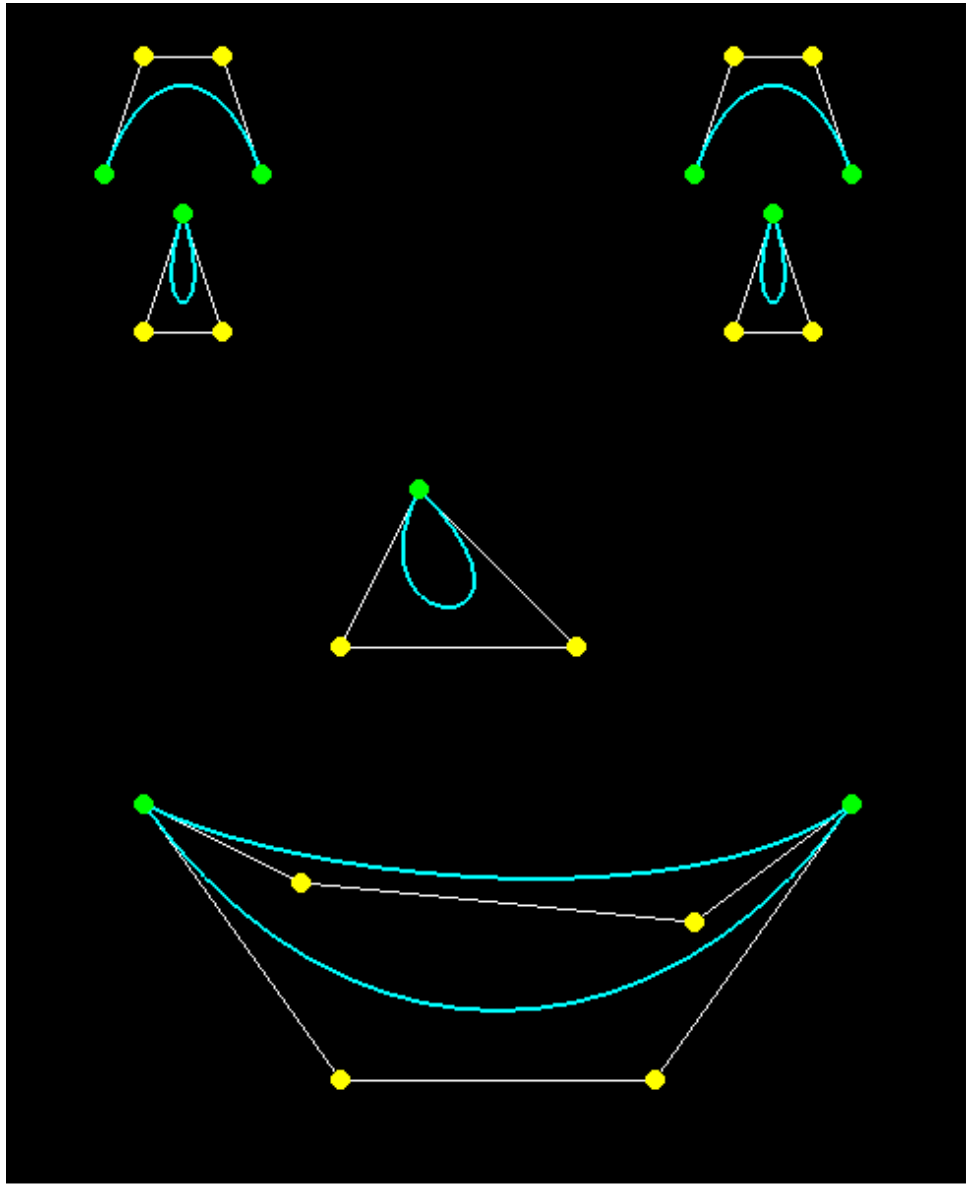
lsystems.mp4

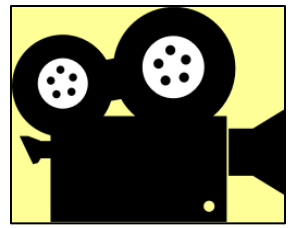
F	move forward one step
+	rotate + about Z
-	rotate - about Z
<	rotate + about Y
>	rotate – about Y
v	rotate + about X
^	rotate – about X
[	save position
]	restore position





Modeling Smooth Curves: Curve Sculpting



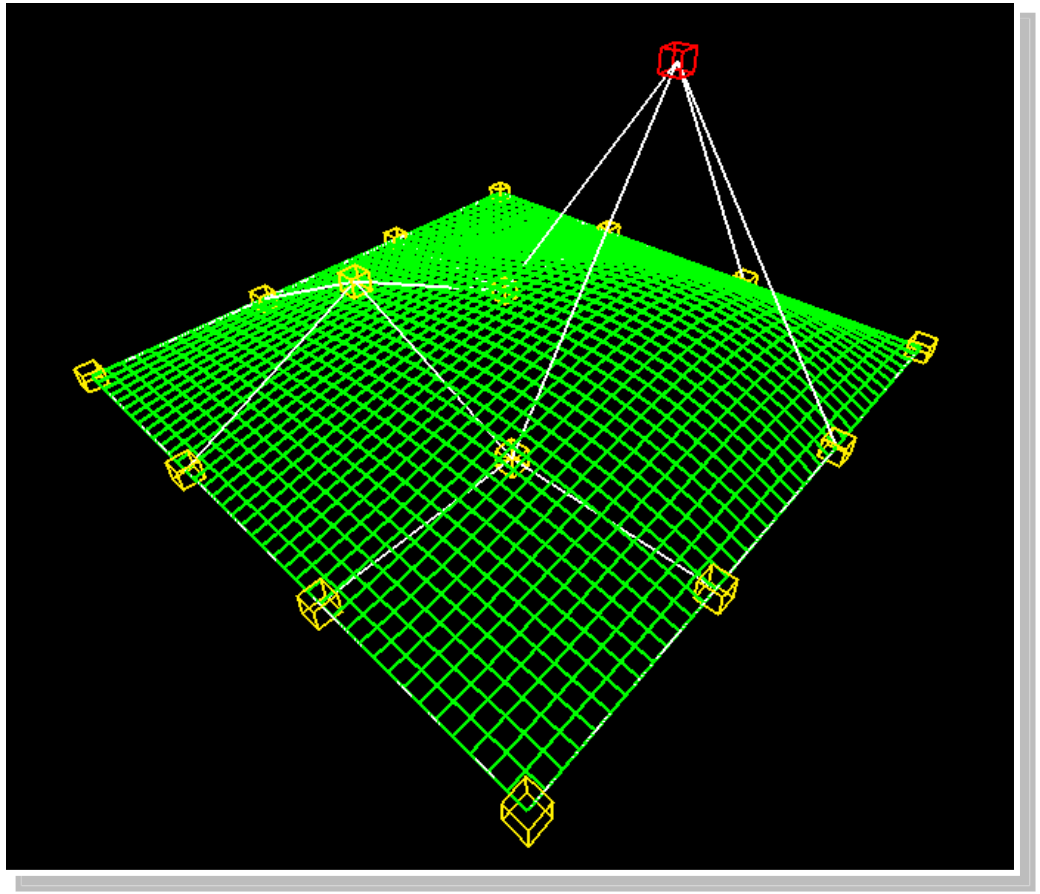


curves.mp4

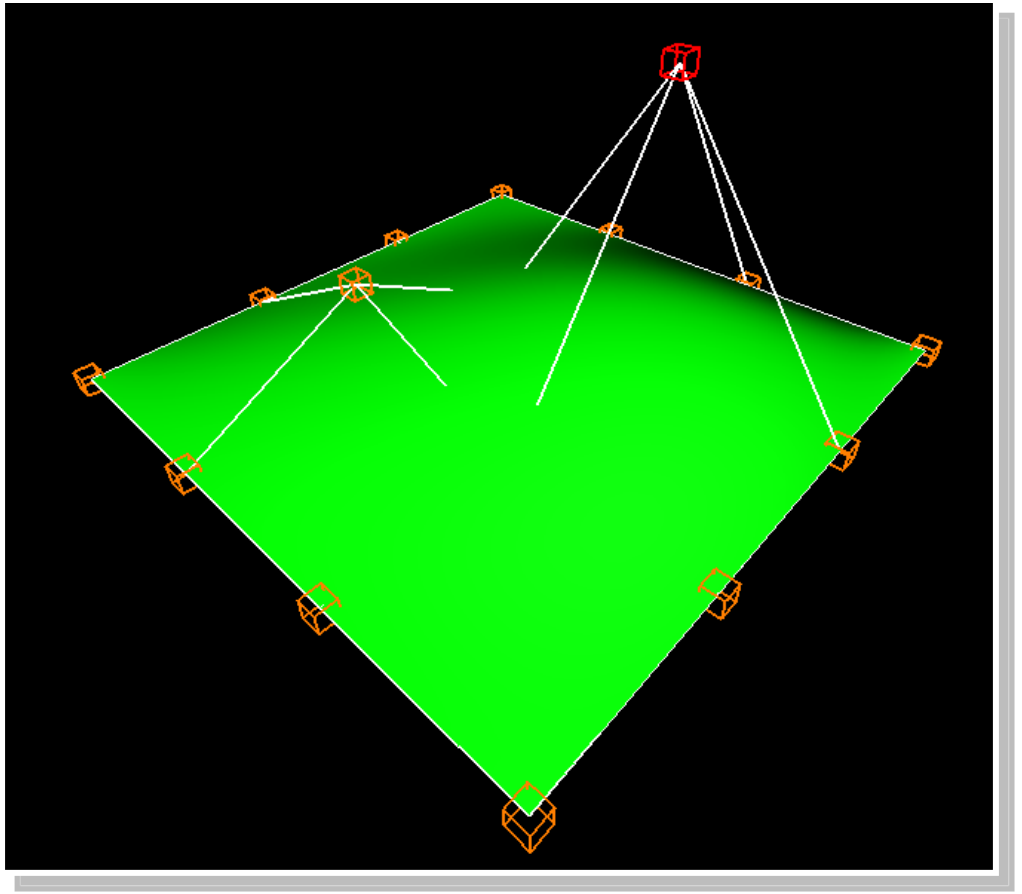
A *Small* Amount of Input Change
Results in a *Large* Amount of Output Change

Modeling Smooth Surfaces: Surface Sculpting

In general, these are referred to as ***Patches***. These, in particular, are Beziér patches. Non-uniform Rational B-spline Surfaces, or NURBS, are another popular type.



Wireframe

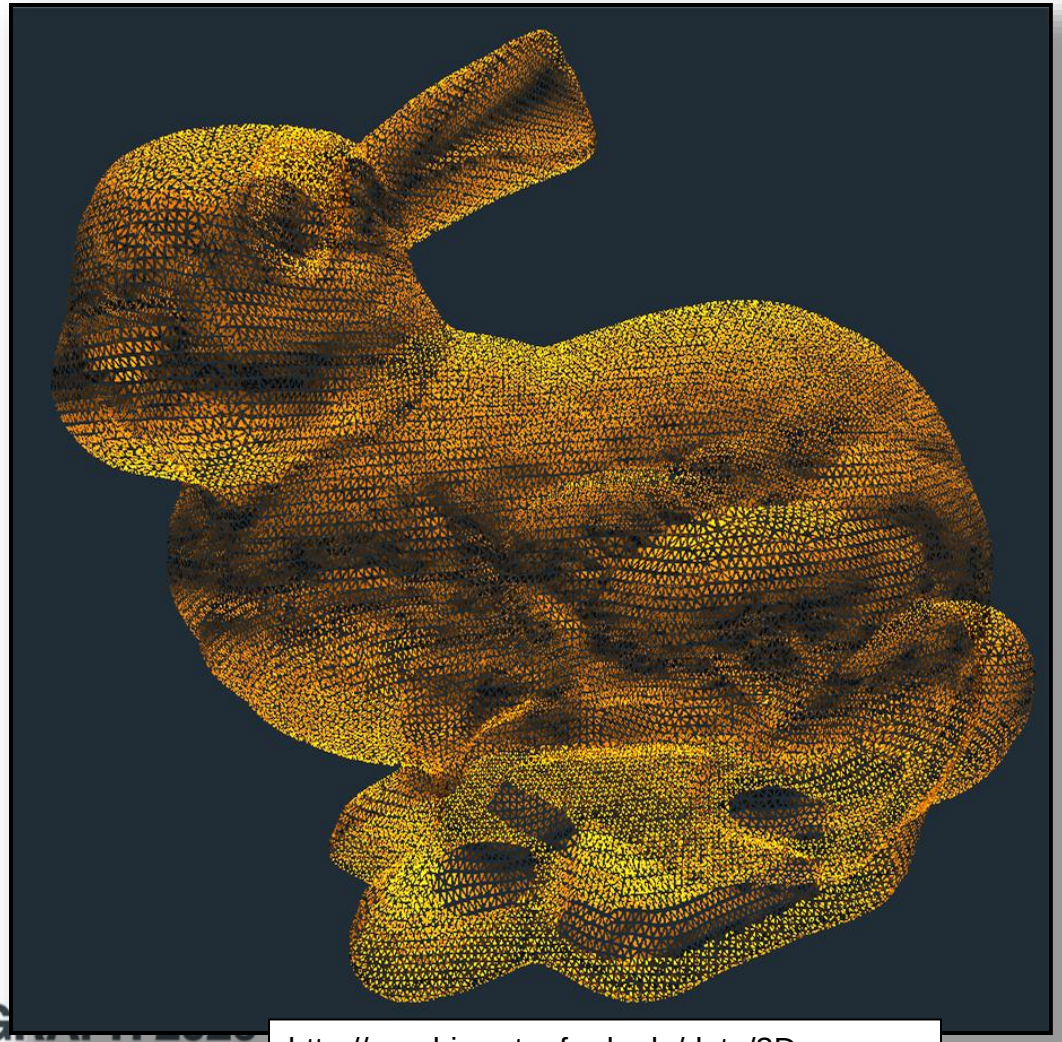


Surface

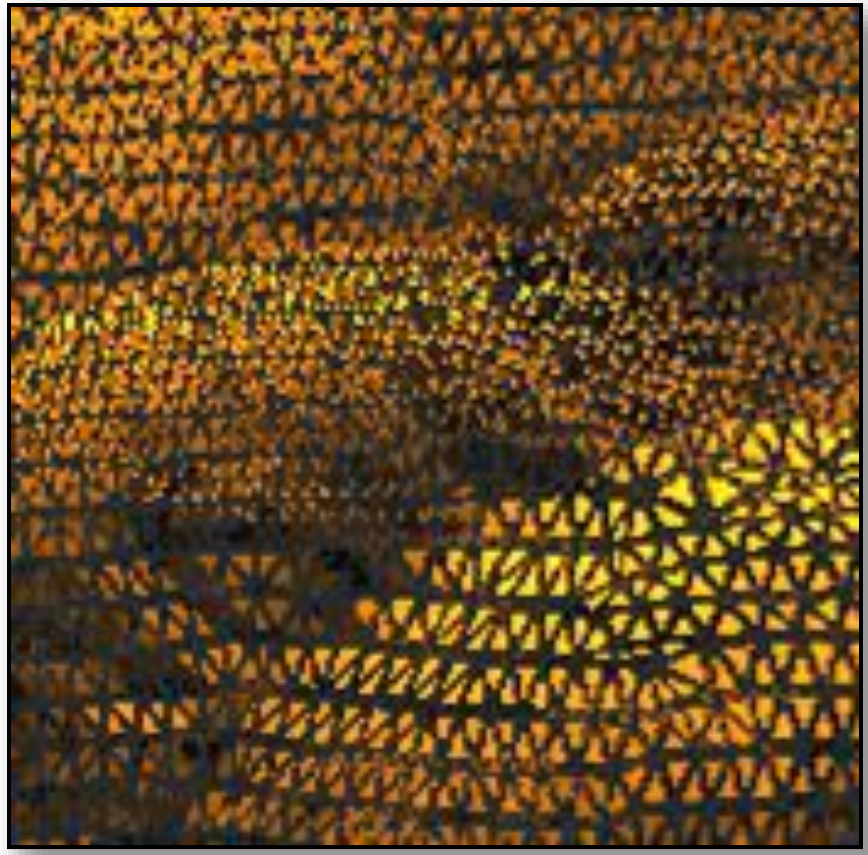
Like the curve sculpting, a *Small* Amount of Input Change Results in a *Large* Amount of Output Change

Explicitly Listing *Geometry* (3D Points) and *Topology* (How They Are Connected)

Models defined this way can consist of thousands of vertices and faces – we need some way to describe them

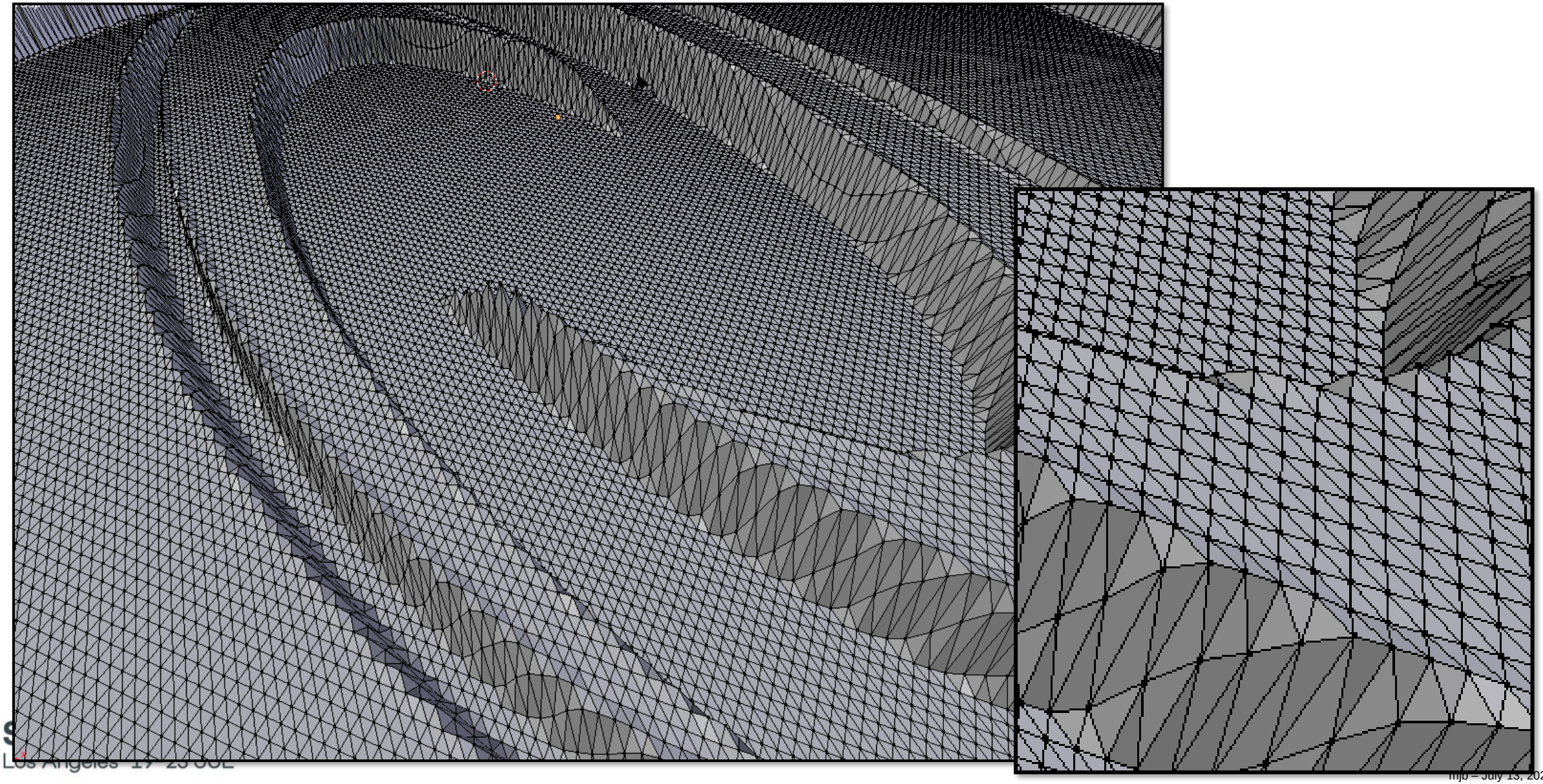


http://graphics.stanford.edu/data/3Dscanrep

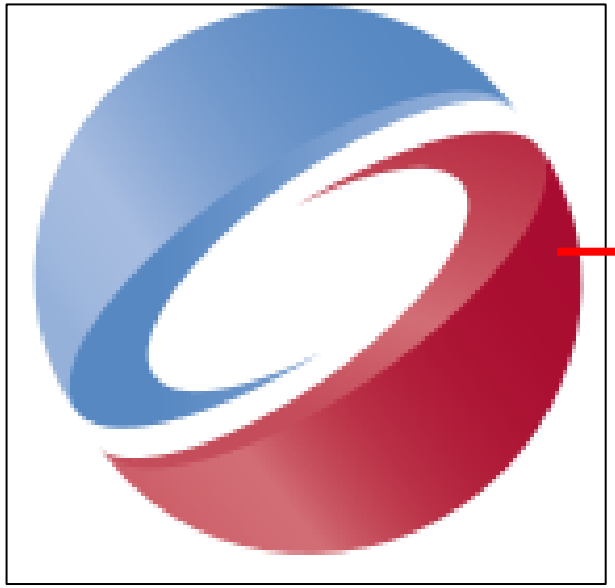


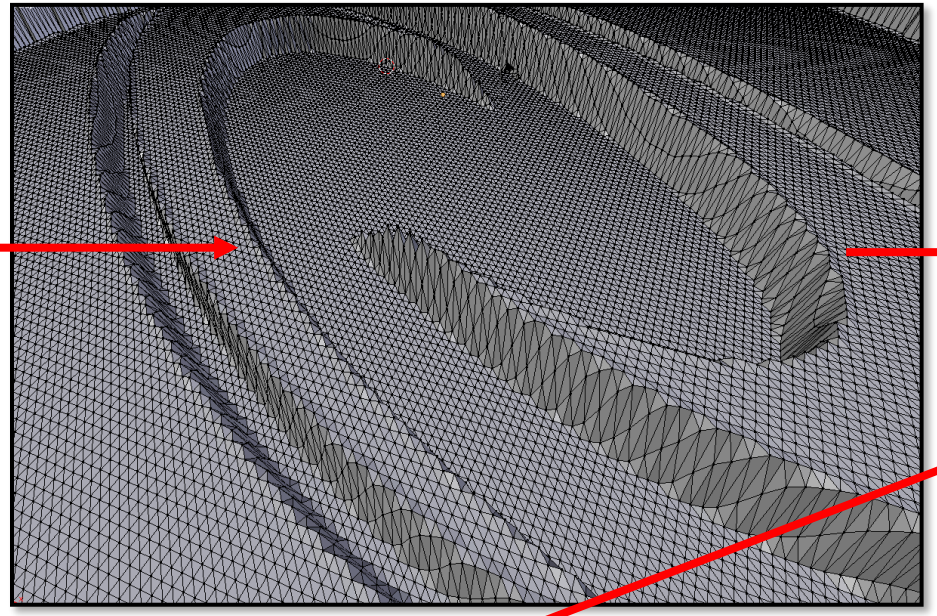
This is often called a **Mesh**, or sometimes a **Triangular Irregular Network** (TIN)**.**

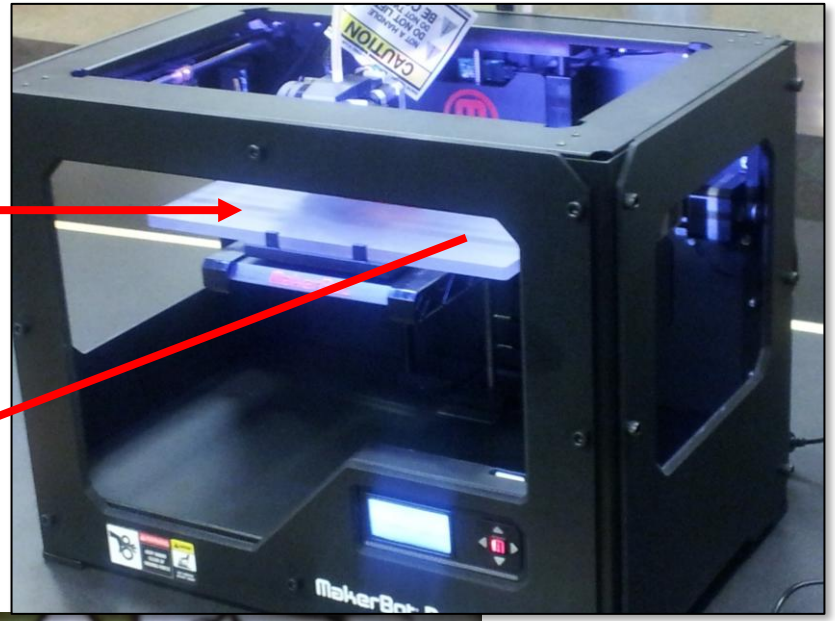
Triangular Meshes are Super Important These Days Because *3D Printing* Requires a Triangular Mesh Data Format

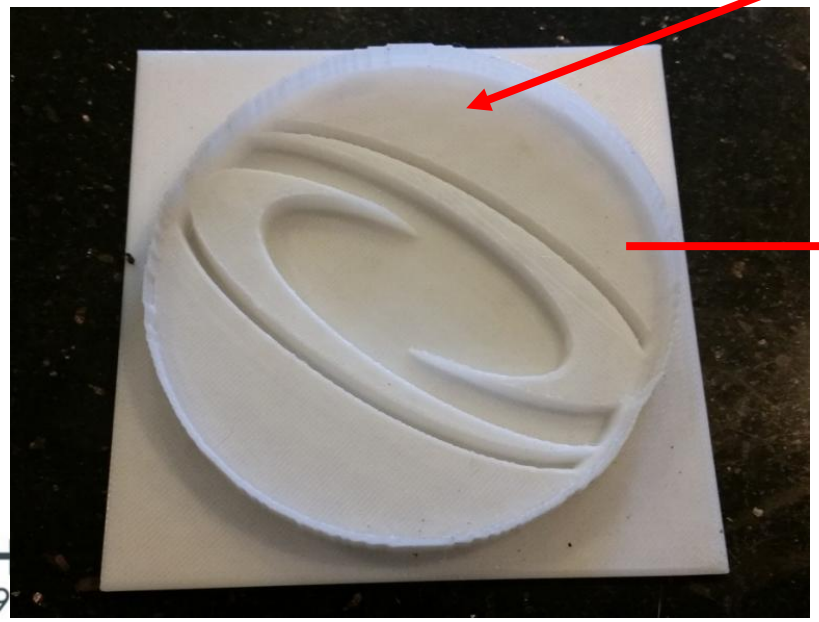


3D geometric modeling at its very best -- mmmm… :-)

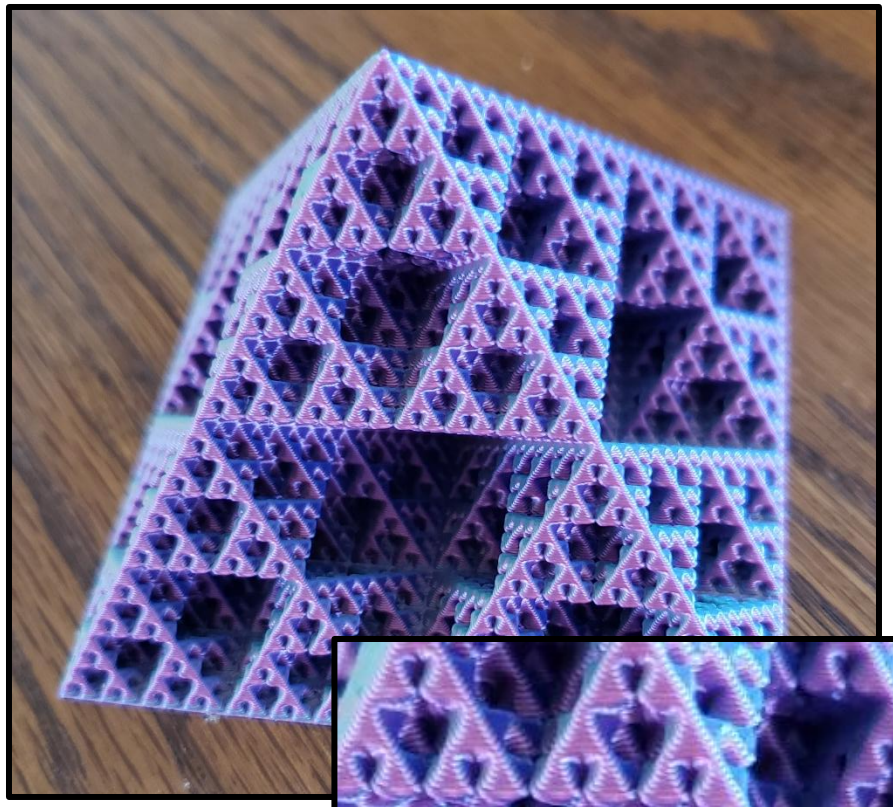


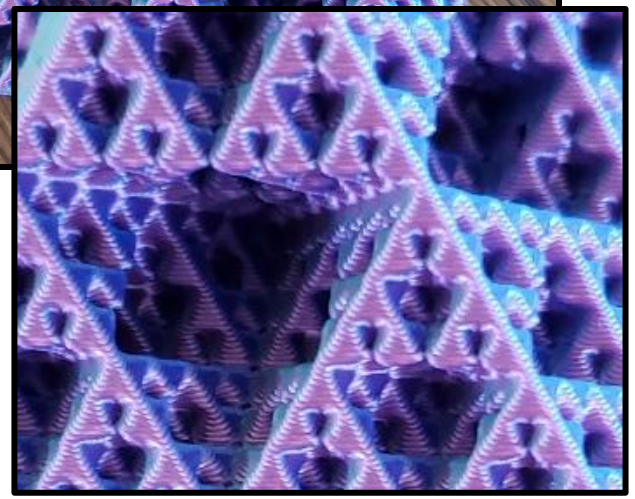






3D Printing Meshes Don't Always Look Very Mesh-ish Anymore – But They Are

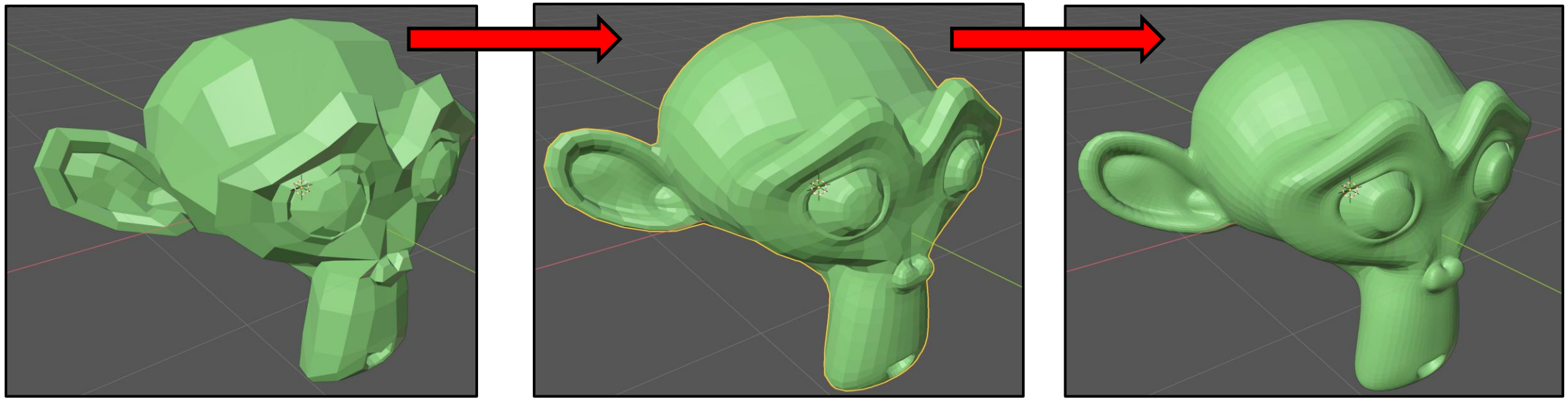




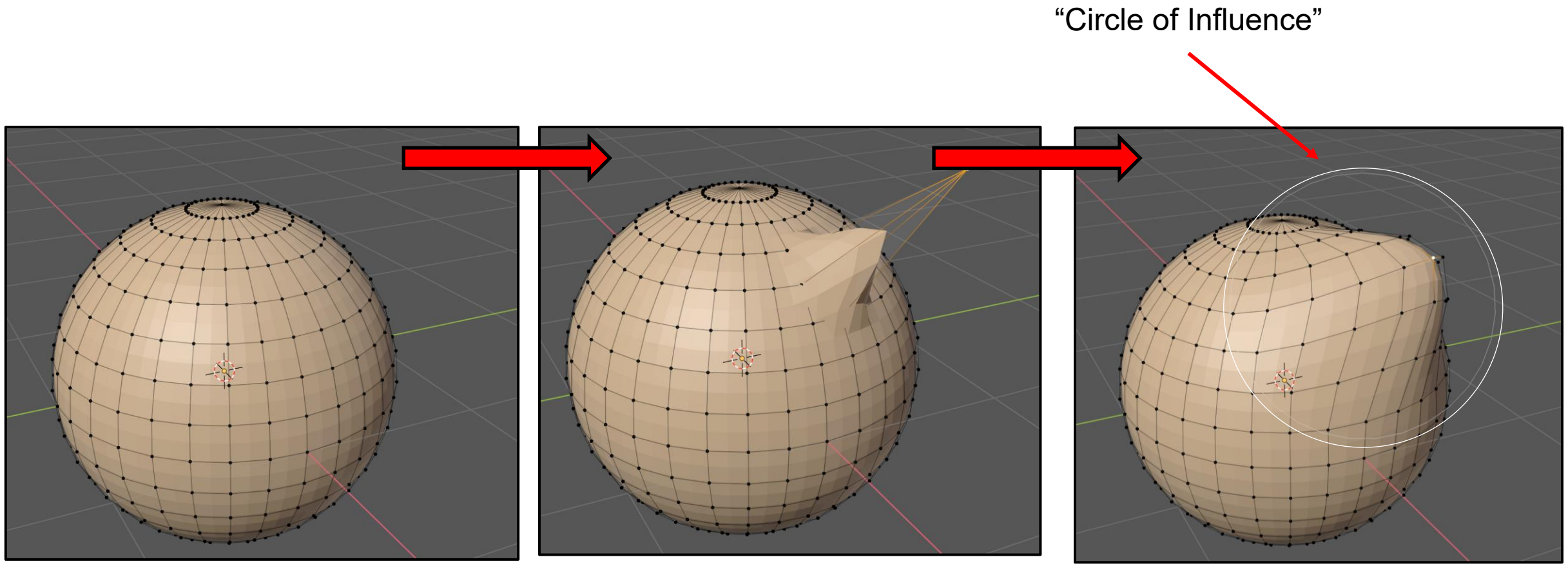
Models are from https://www.printables.com

3D Printed by Ryan Bailey
Images used by permission

Meshes Can Be Smoothed



Meshes Can Be Edited

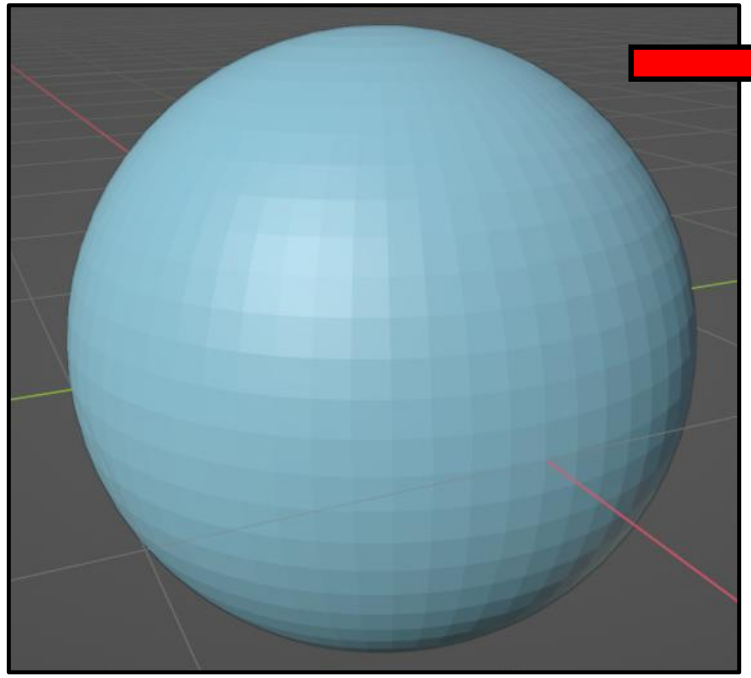


Original

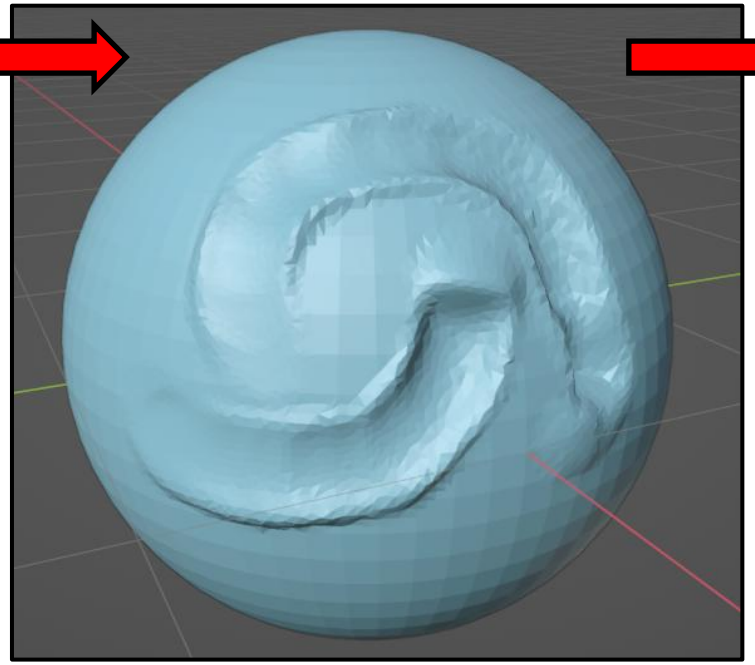
Pulling on a single Vertex

Pulling on a Vertex with *Proportional Editing* Turned On

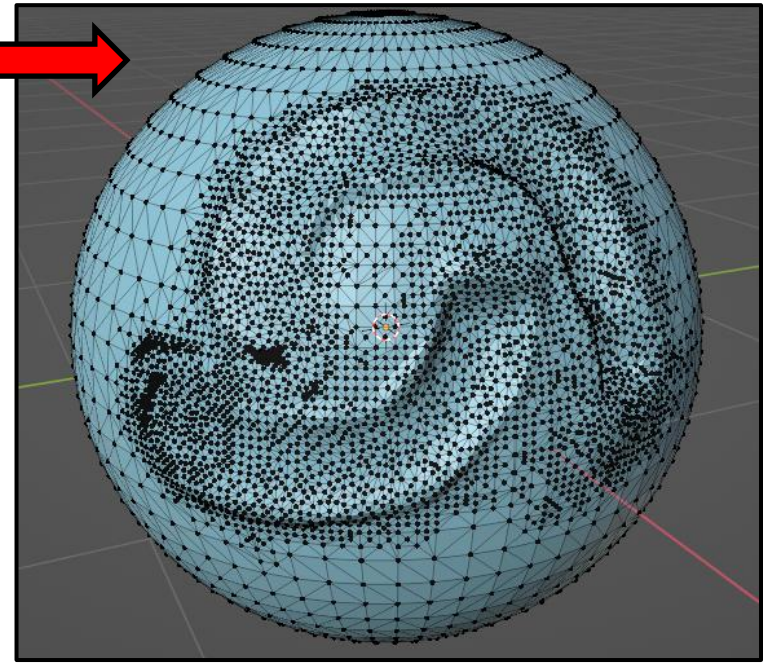
Meshes Can Be Sculpted



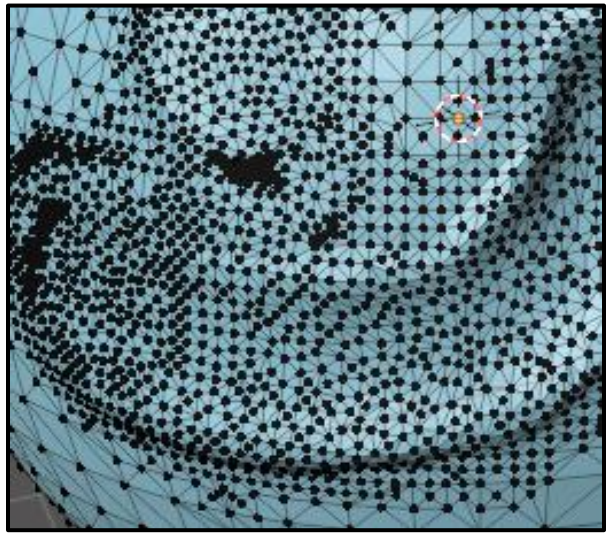
Original



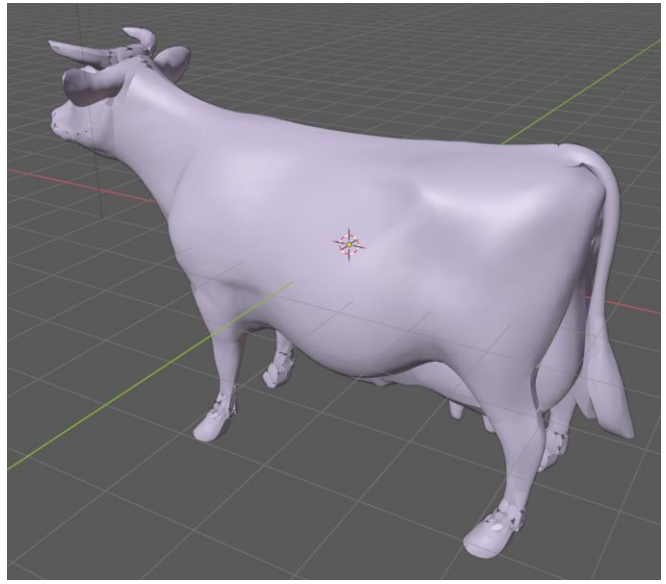
“Clay Thumb” Sculpting



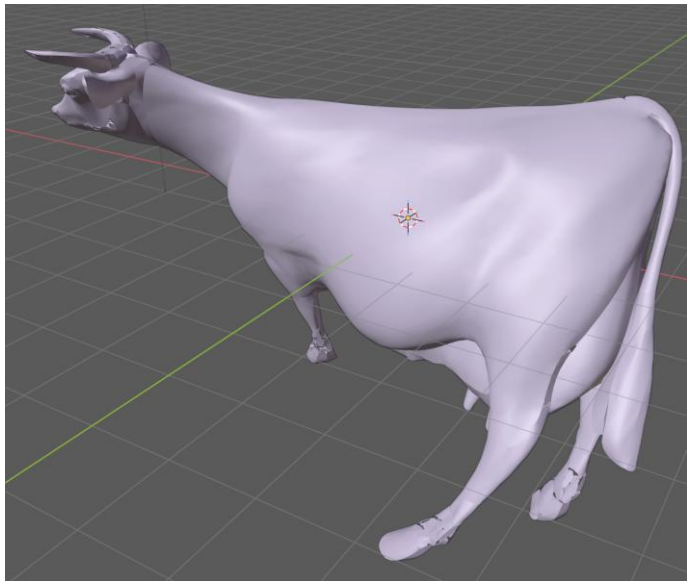
Sculpting Can Produce Additional Mesh Vertices



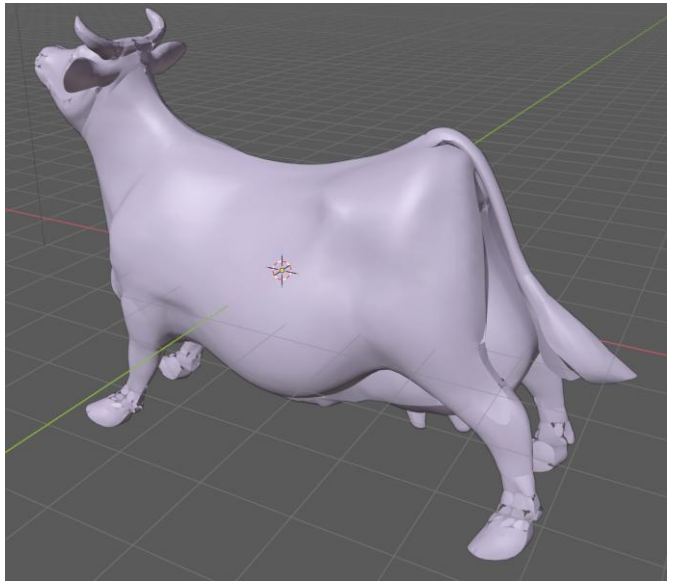
Meshes Can Be Deformed



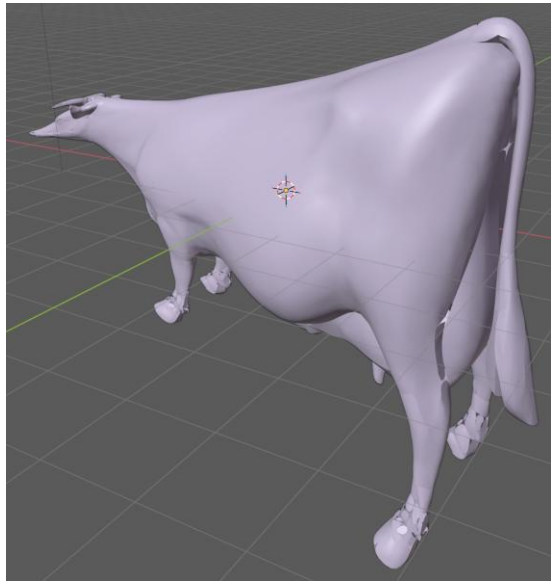
Original



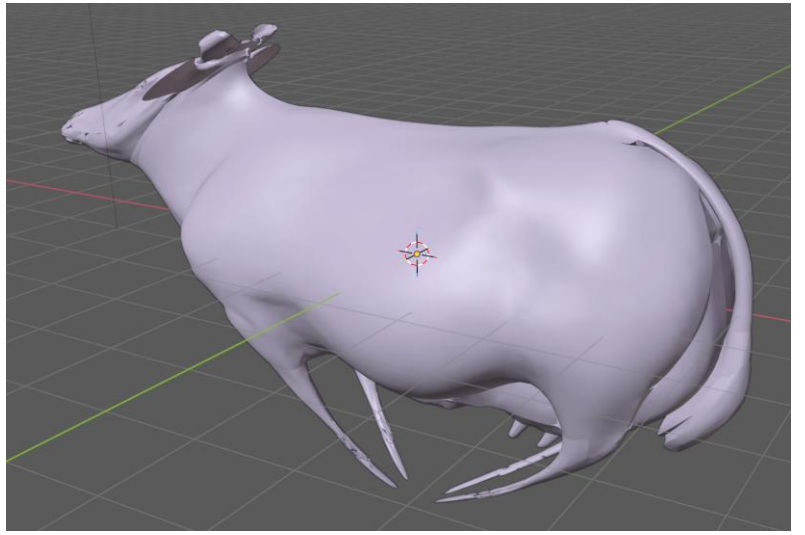
Twist



Bend

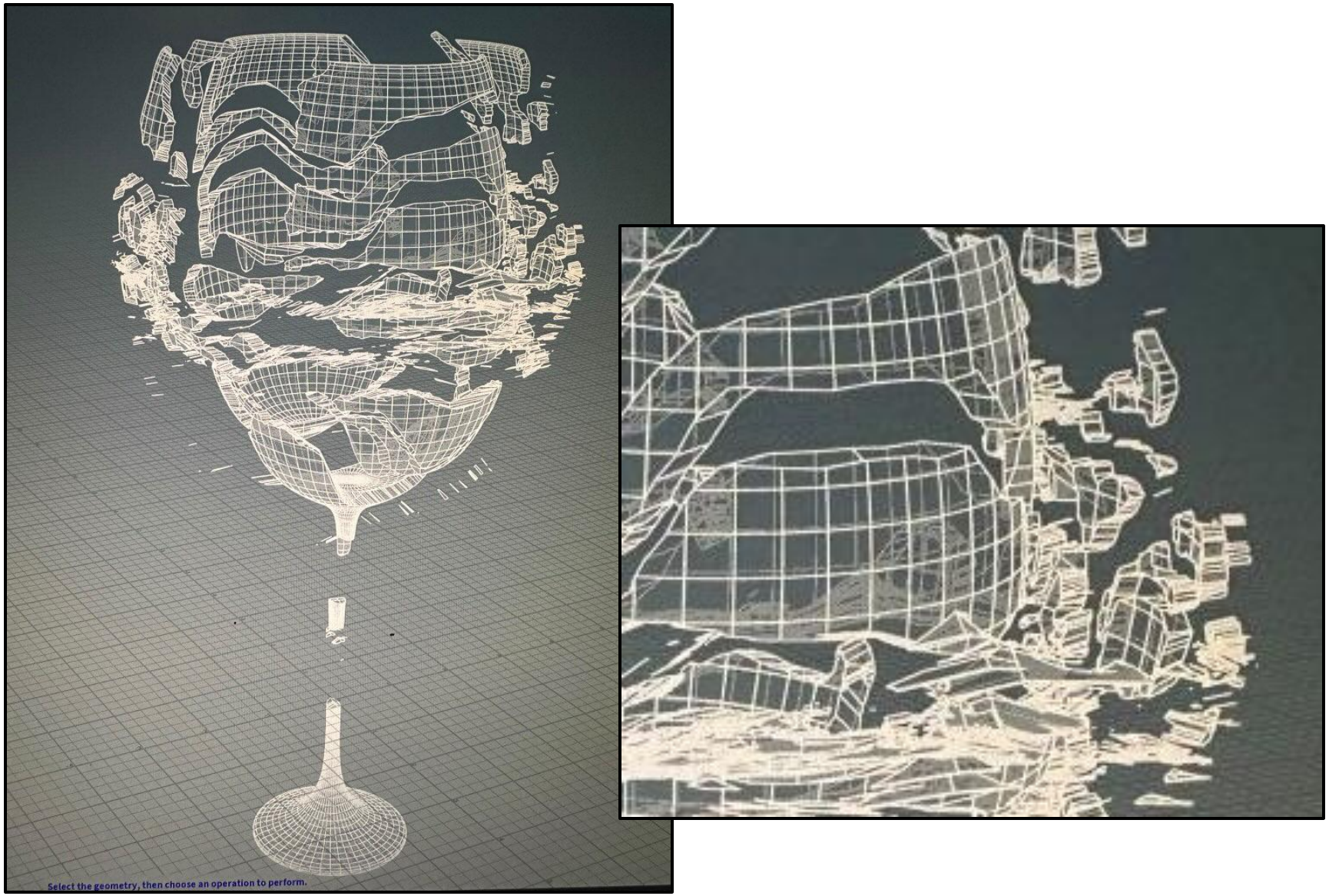


Taper



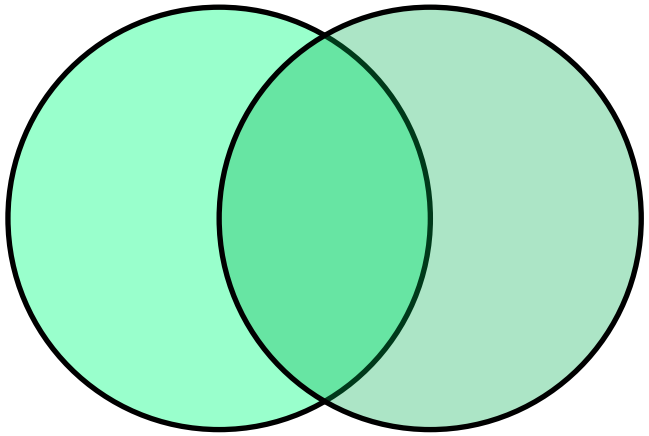
Stretch

Meshes Can Be Used to Compute Physics

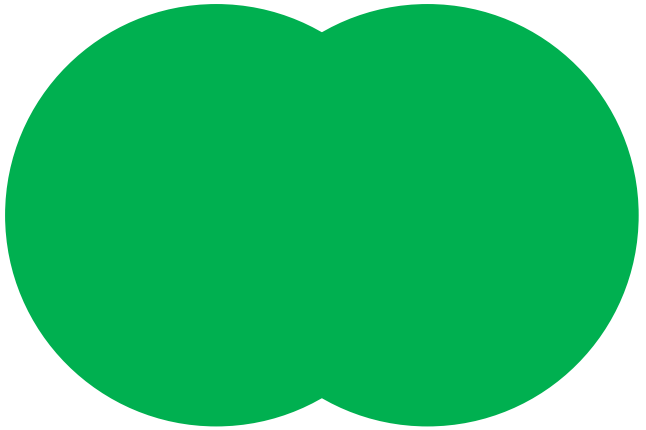


Natasha Anisimova, used by permission

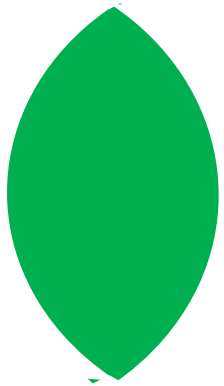
Another Way to Model in 3D:
Remember Venn Diagrams (2D Boolean Operators) from High School?



Two Overlapping Shapes

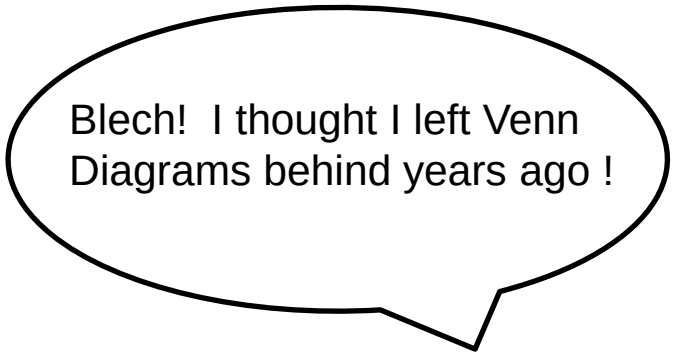


Their Union

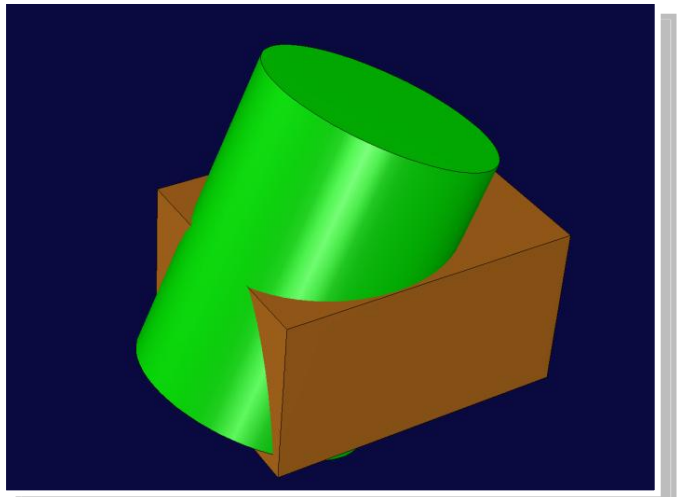


Their Intersection

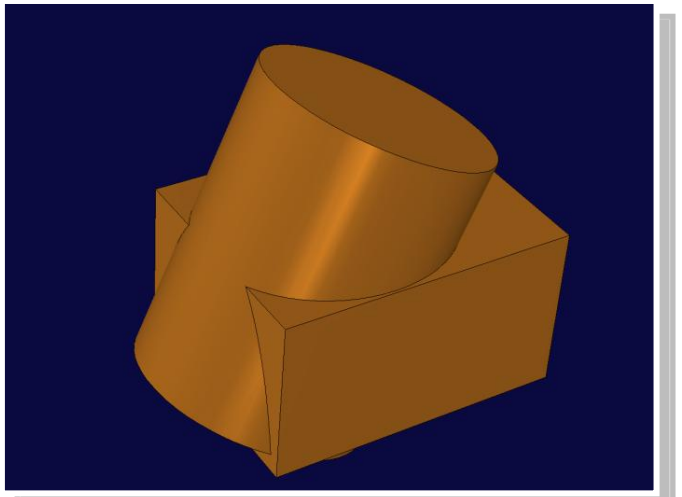
Their Difference



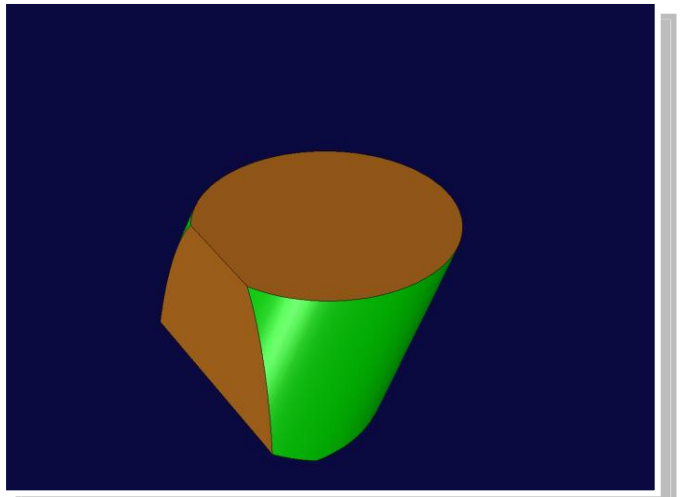
Solid Modeling Using 3D Boolean Operators



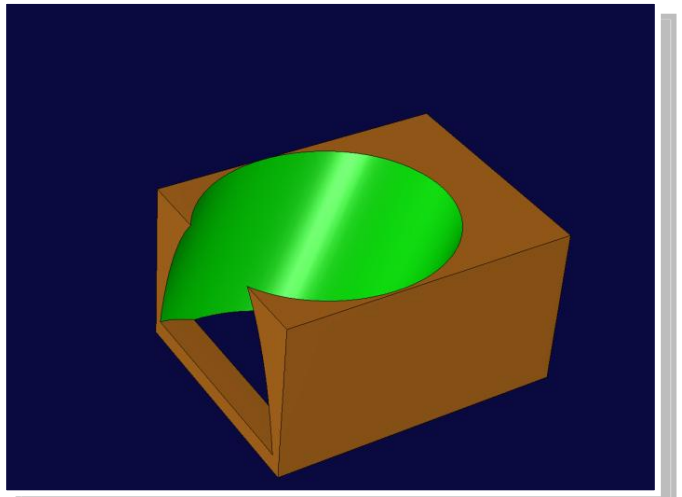
Two Overlapping Solids



Their Union



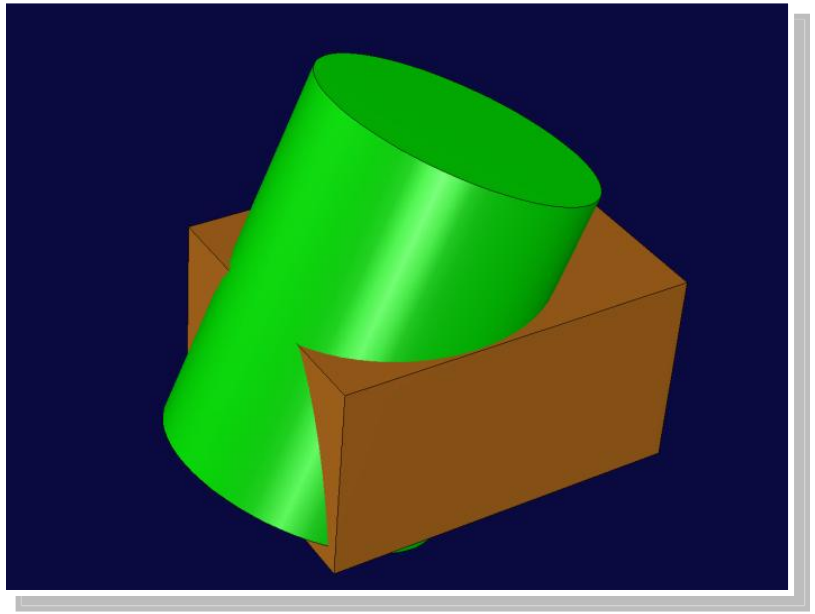
Their Intersection



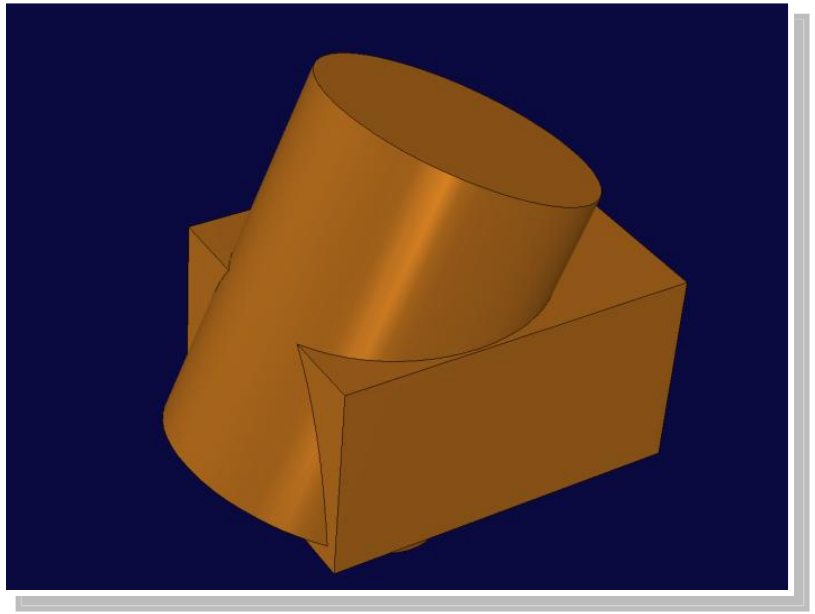
Their Difference

This is sometimes called *Constructive Solid Geometry*, or *CSG*

3D Boolean Operators are Important in 3D Printing as well as General Modeling



Two Overlapping Solids –
They Cannot Be 3D Printed



Two Overlapping Solids that have been Union'ed – Now They *Can* Be 3D Printed

Unioned, Intersected, and Differenced Solids can all be 3D Printed

Want to experiment with 3D Booleans for free?
Want some notes to help you get started?
http://cs.oregonstate.edu/~mjb/blender
http://cs.oregonstate.edu/~mjb/tinkercad

3D Gaussian Splatting (3DGS)

Splatting started life in the CG community as a 2D volumetric rendering technique in the 1990s. At the time, it was described as treating data points as snowballs and splatting them up against a window. ☺ Around 2023, it re-emerged as a spectacular 3D modeling/rendering technique.

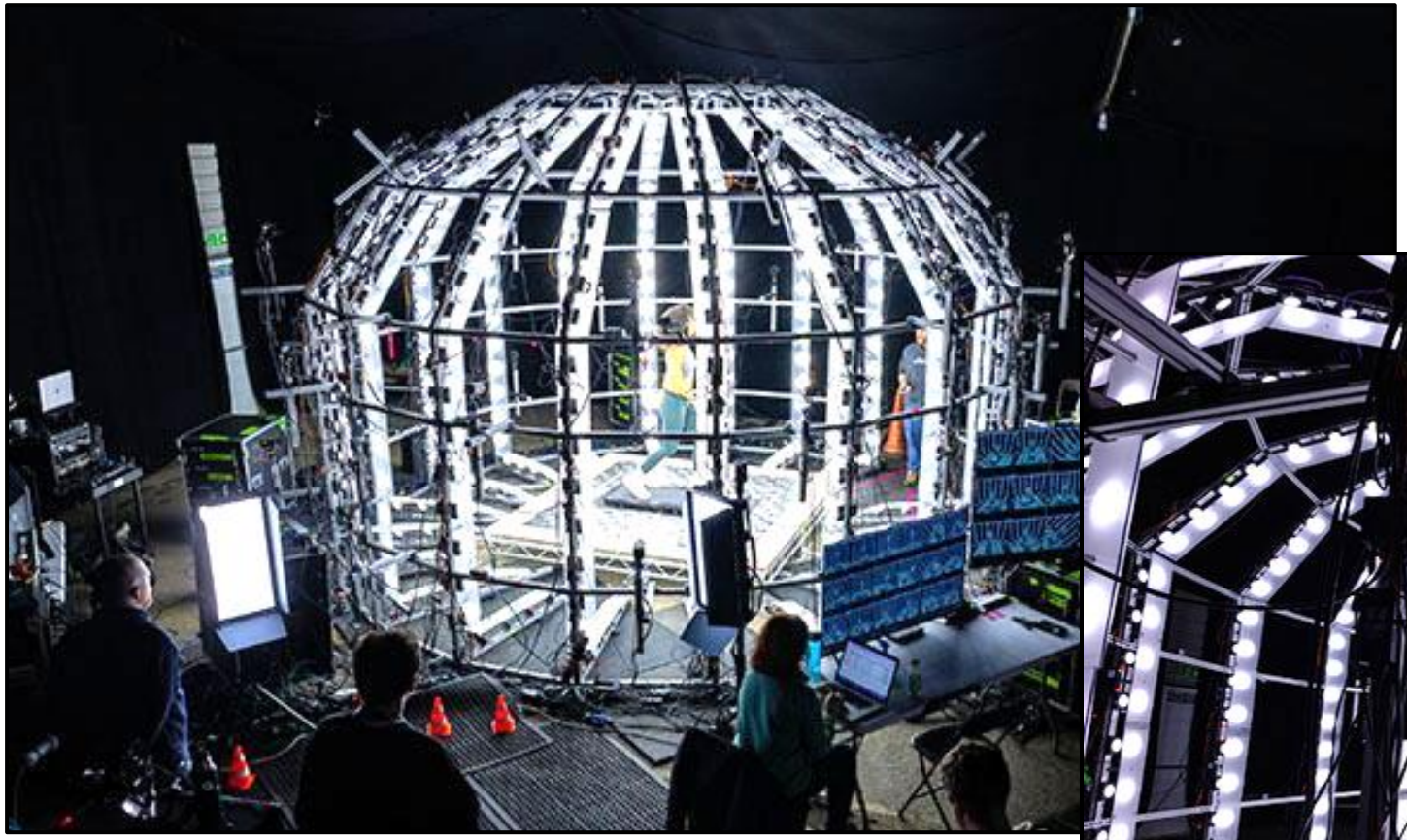
The general idea is to collect multiple camera views of a particular scene, typically from drone footage or specialized rigs (see the next slide!). Because the camera views are all different, essentially this process turns each images' pixels into 3D colored point clouds.

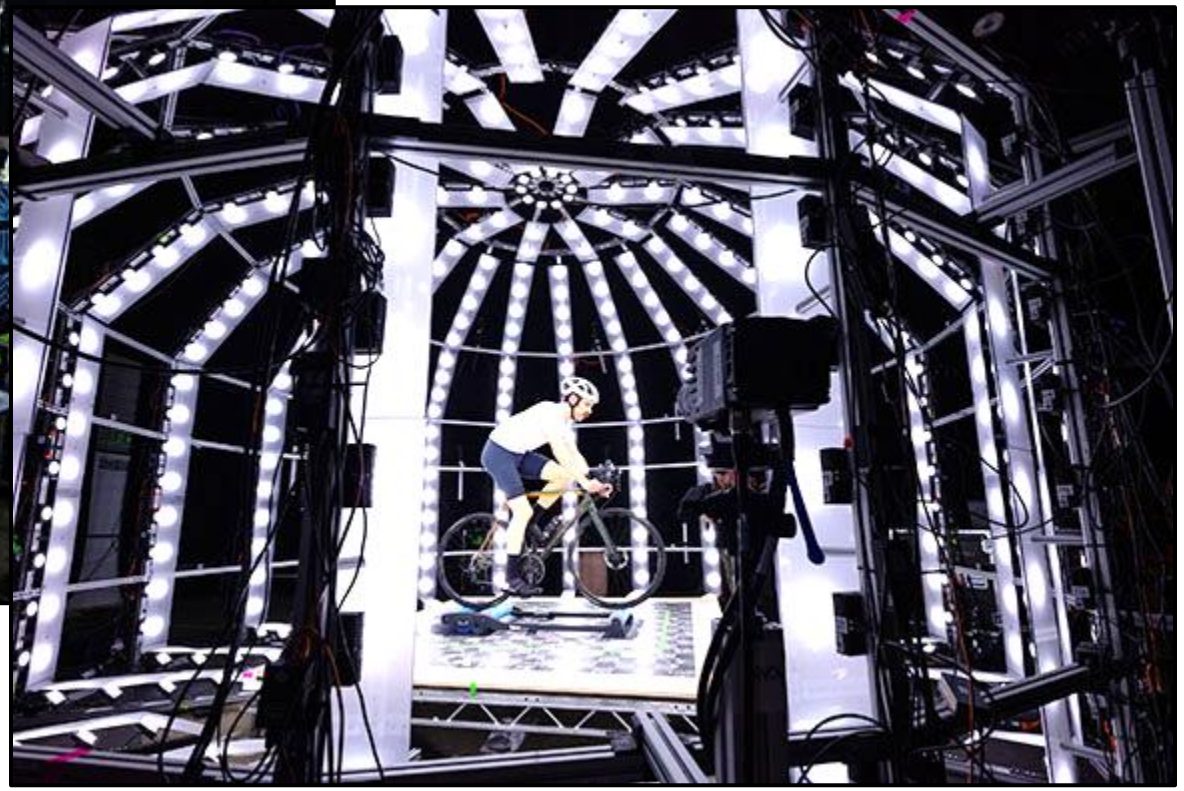
The algorithm uses a machine learning gradient descent algorithm to produce a data set where all scene elements have appropriate 3D positions, colors, surface normals, etc. assigned. This is the lengthy *training* part of the algorithm. Once this is completed, the resulting dataset can be rendered in interactive time.

Gaussian splatting model of a collapsed building taken from drone footage
https://en.wikipedia.org/wiki/Gaussian_splatting

How to Capture 360º Gaussian Splat Scene Data without a Drone

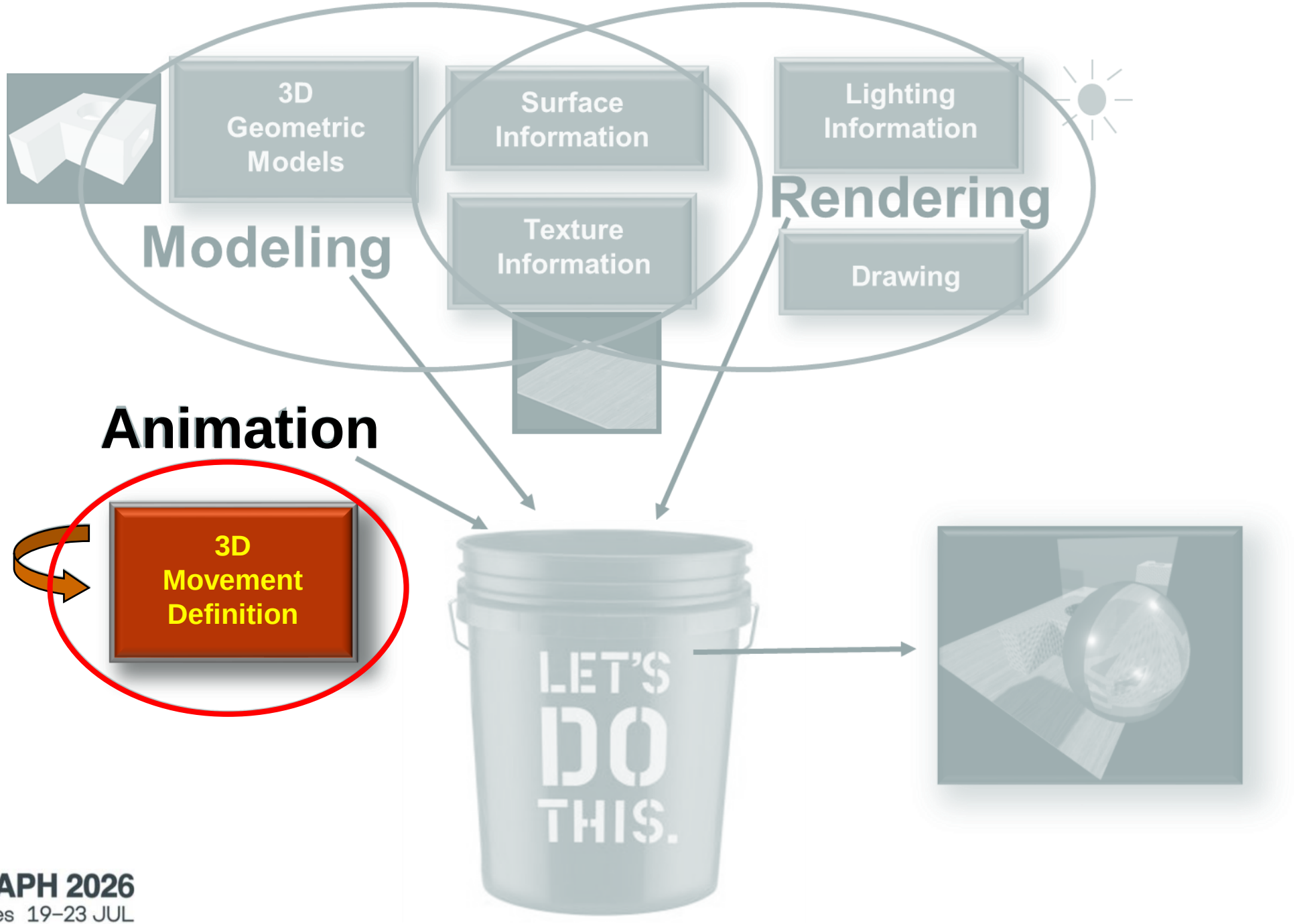
Clear Angle Studio's Volumetric Capture Rig:





https://www.clearanglestudios.co.uk/

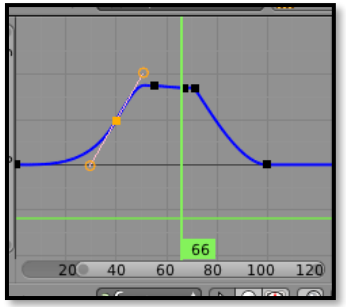
Creating the motion you want

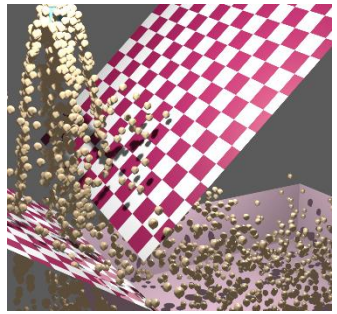

3D Geometric Models
Surface Information
Lighting Information
Modeling
Rendering
Texture Information
Drawing
Animation
3D Movement Definition
LET'S DO THIS.

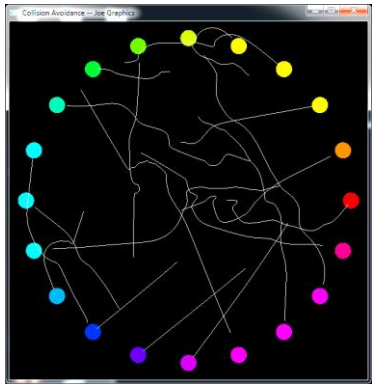
Animation

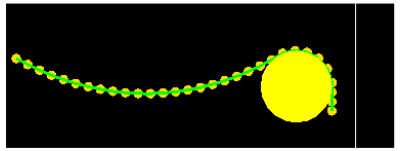
Rendering is the process of giving motion to your geometric modes. Again, there are questions you need to ask first:

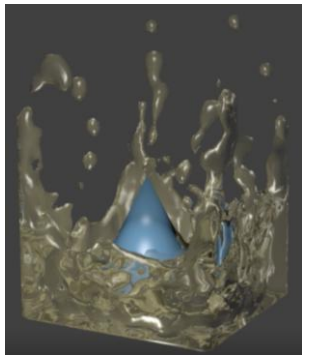
- Why am I doing this?
- Do I want the animation to obey the real laws of physics?
- Am I willing to “fake” the physics to get the objects to *want* to move in a way that I tell them?
- Do I have specific key positions I want the objects to pass through no matter what?
- Do I want to simply record the motion of a real person, animal, etc., and then play it back?

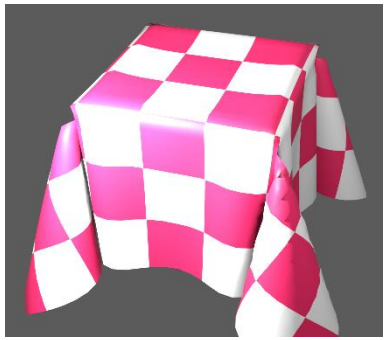


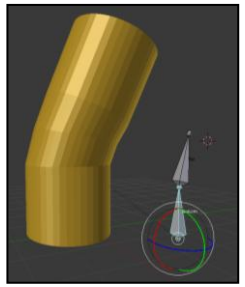


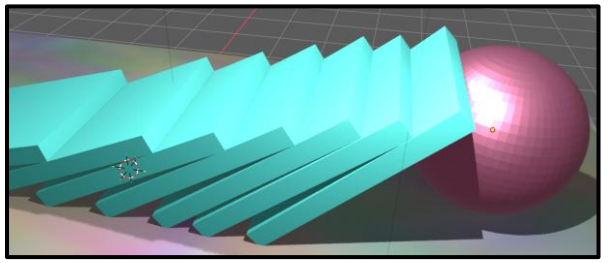






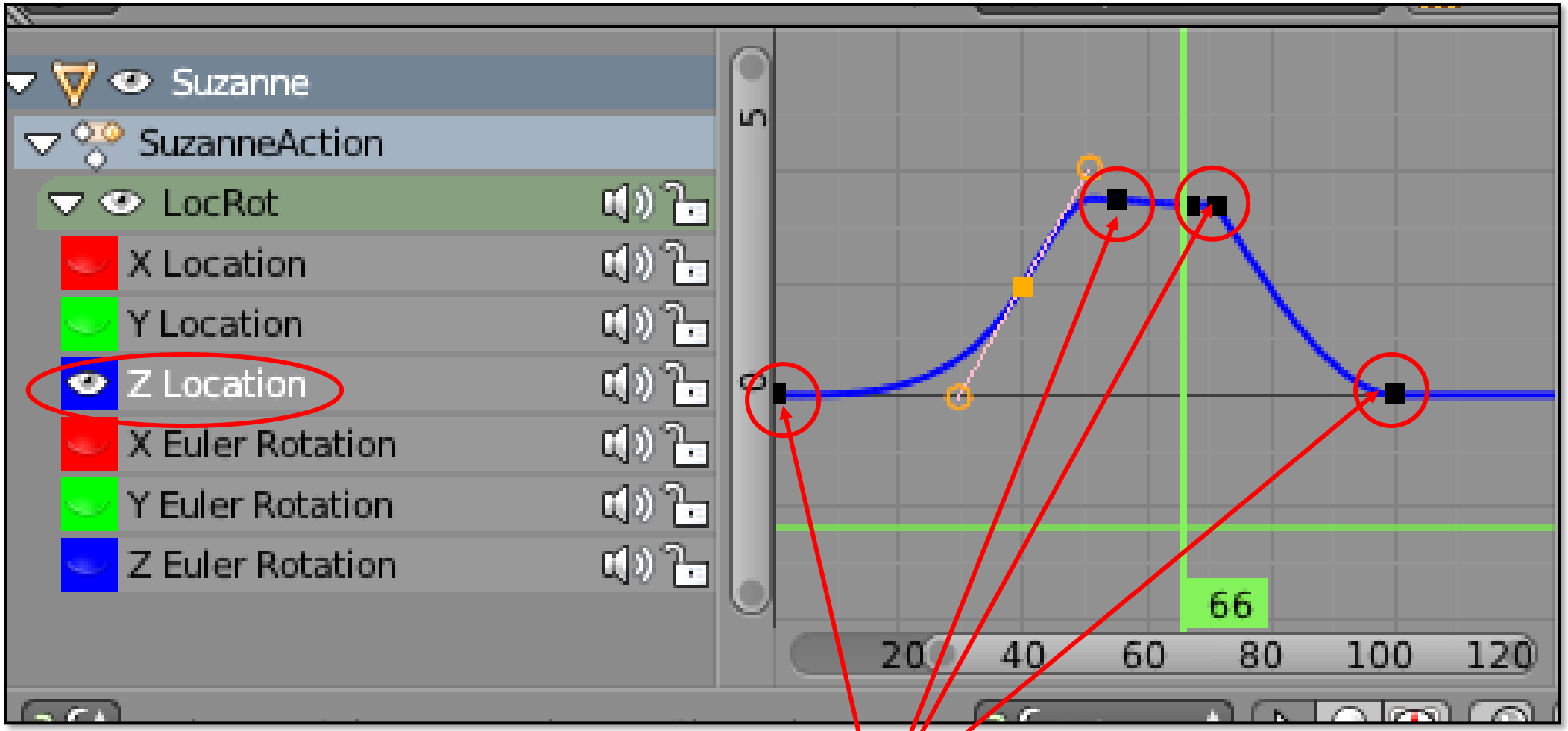




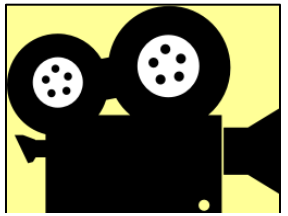


Want to experiment with free animation programs?
Want some notes to help you get started?
http://cs.oregonstate.edu/~mjb/blender
http://cs.oregonstate.edu/~mjb/tinkercad

Keyframe Animation – Adjusting Interpolation Curves



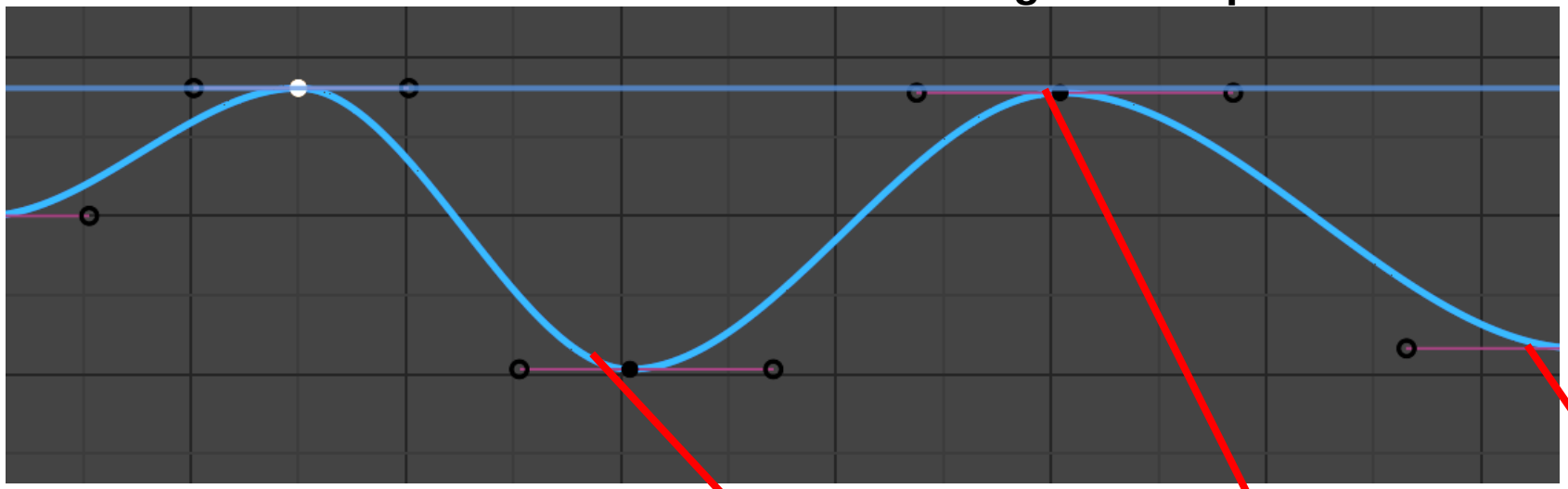
Z Locations at key frames of the animation



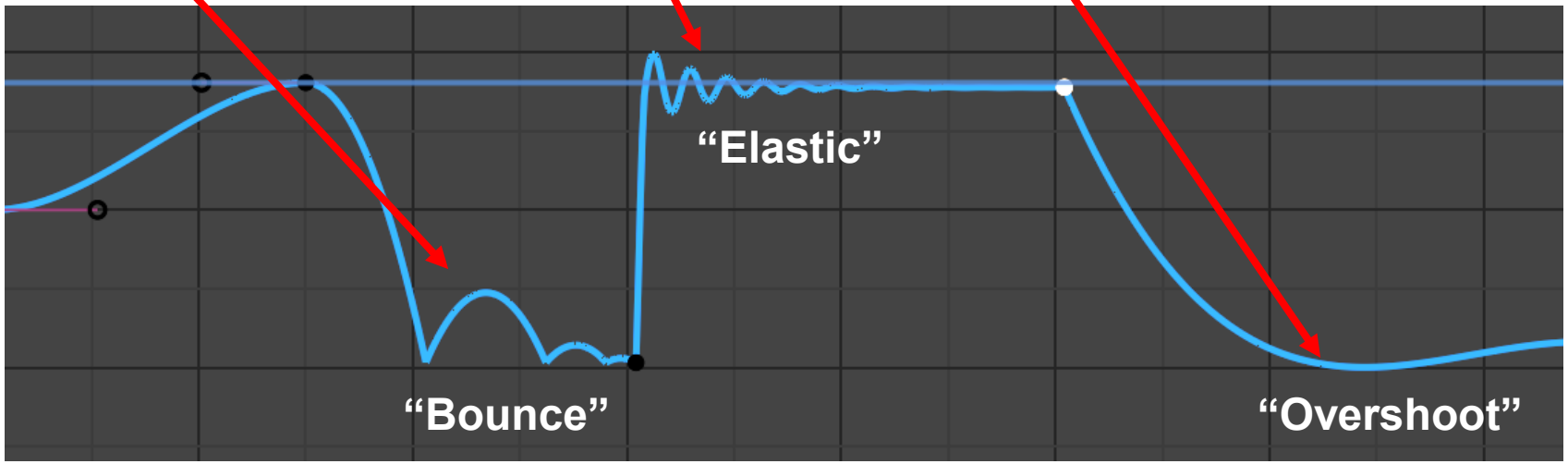
anim2.mp4

Keyframe Animation – Adding Special Effects to the Interpolation Curves

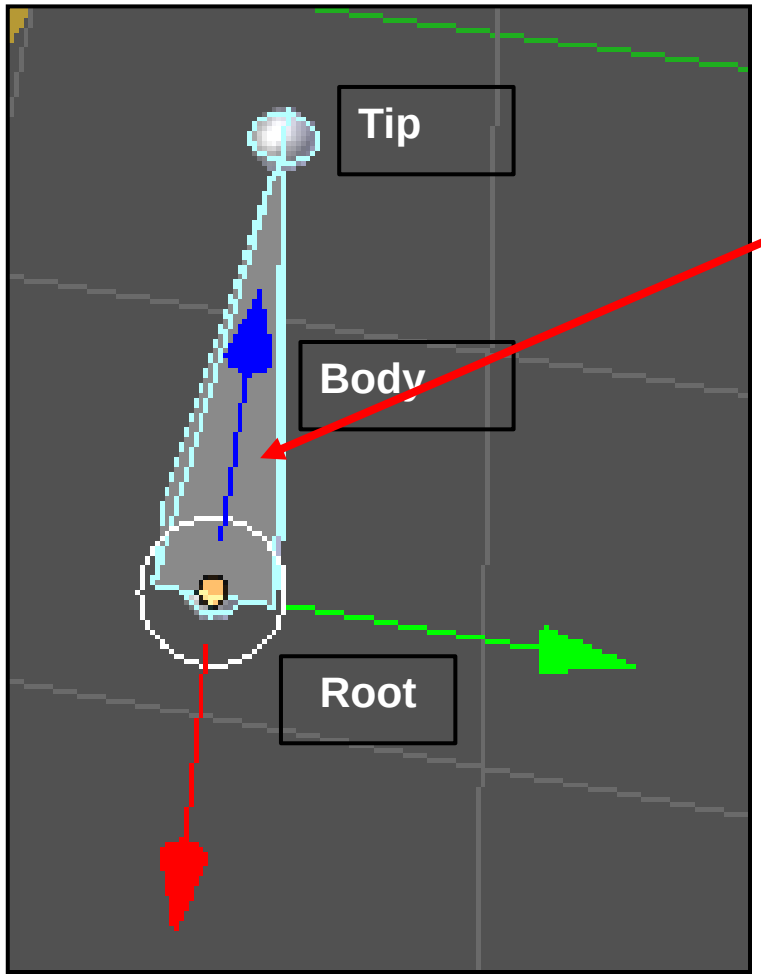
Original Interpolation Curves



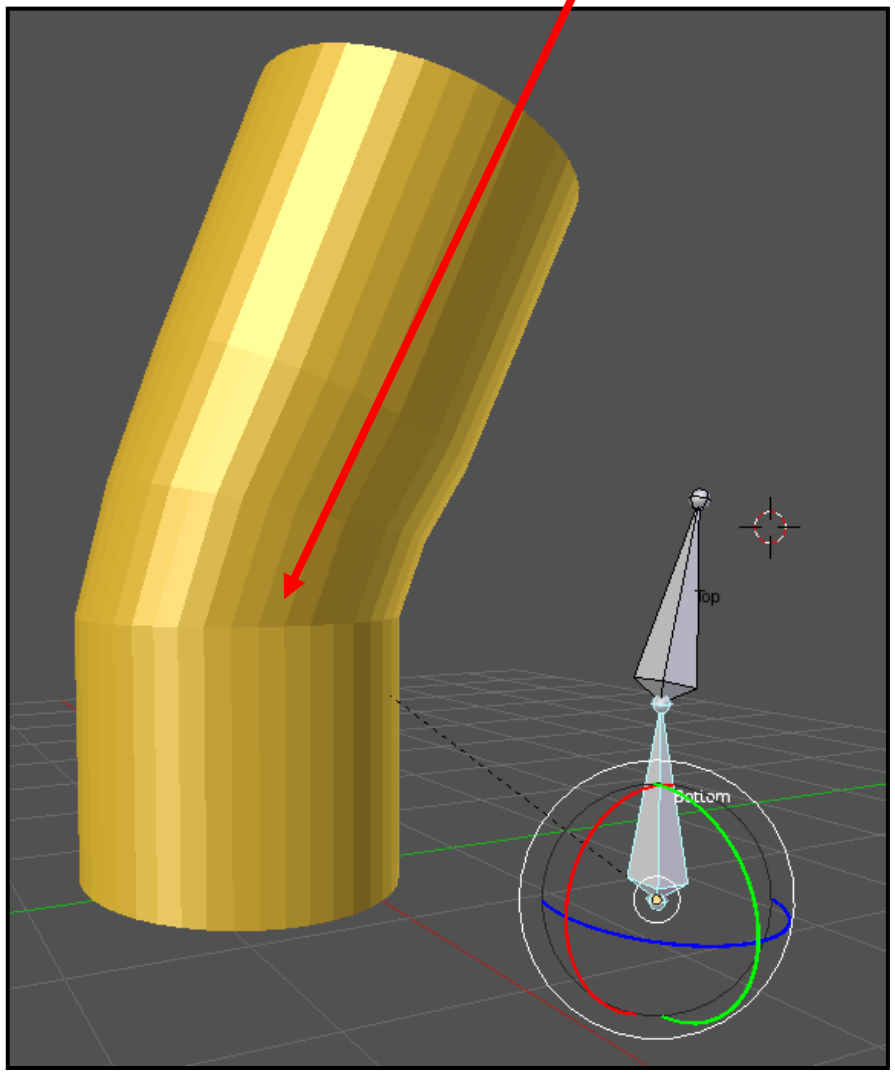
Modified Interpolation Curves



Rigging Animation



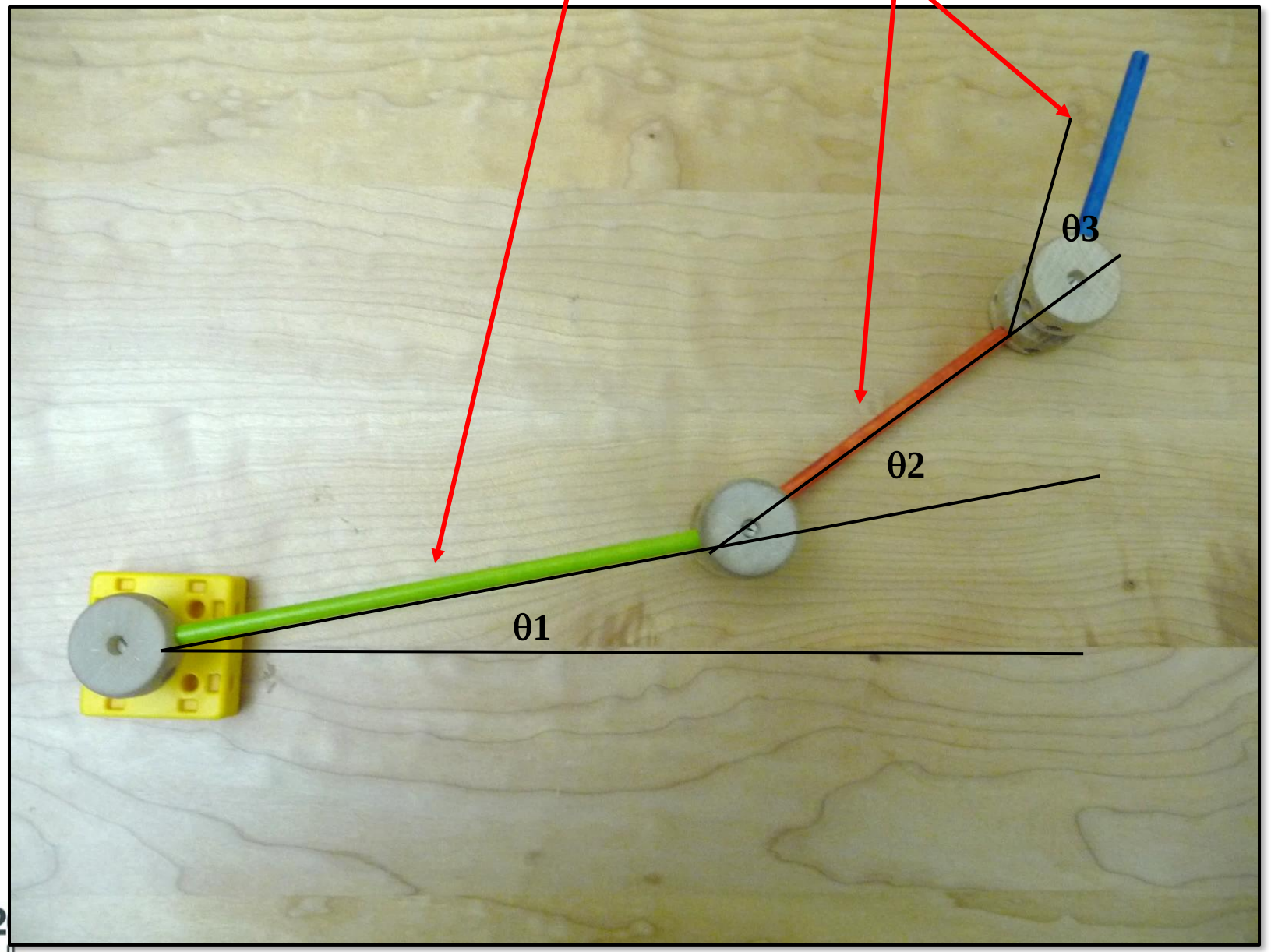
Use an ***armature*** to control the movement of groups of vertices



Forward Kinematics (FK):

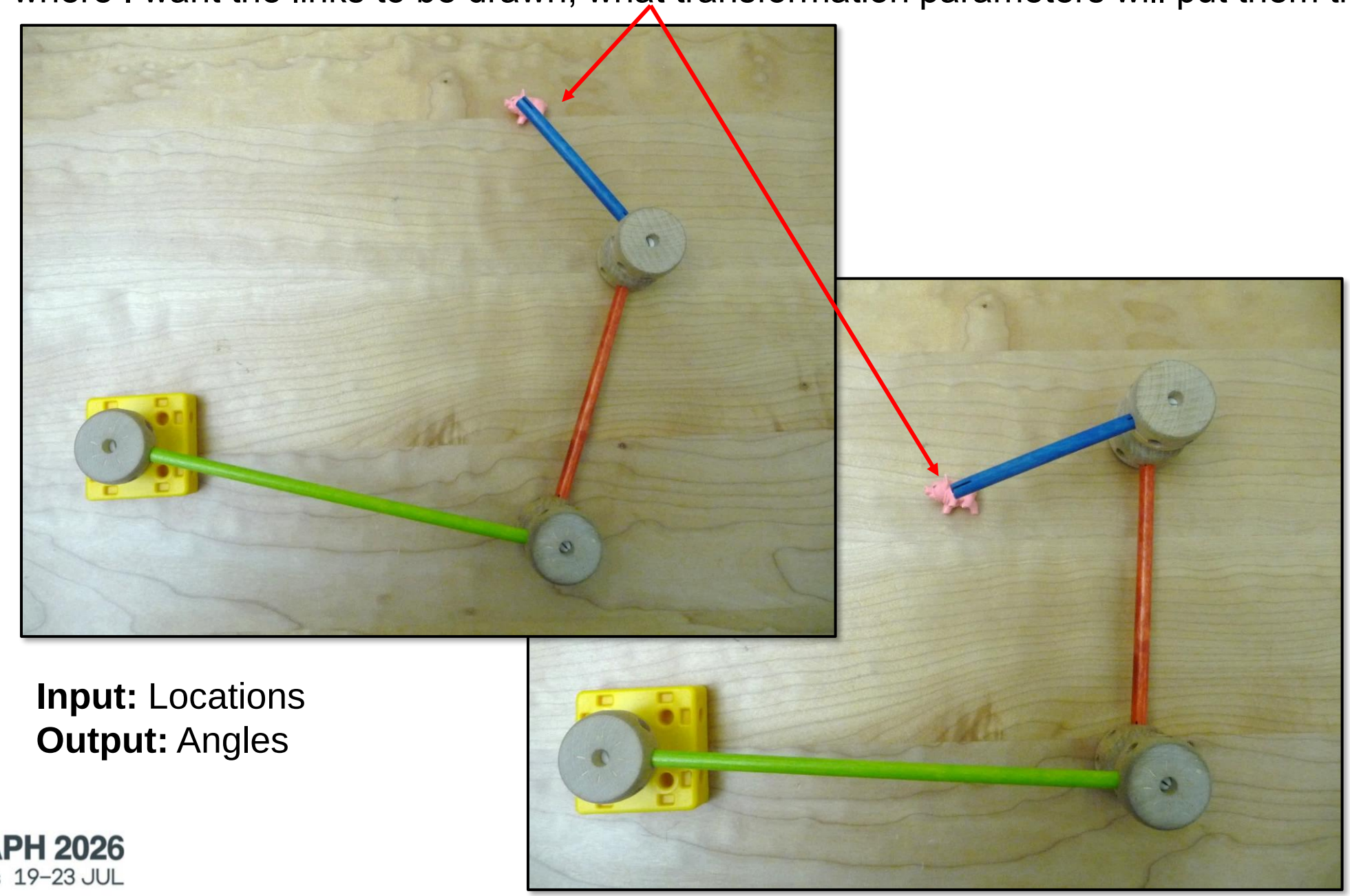
"If I know the link transformation parameters (e.g., angles), where do I draw the links?".

Input: Angles
Output: Locations



Inverse Kinematics (IK):

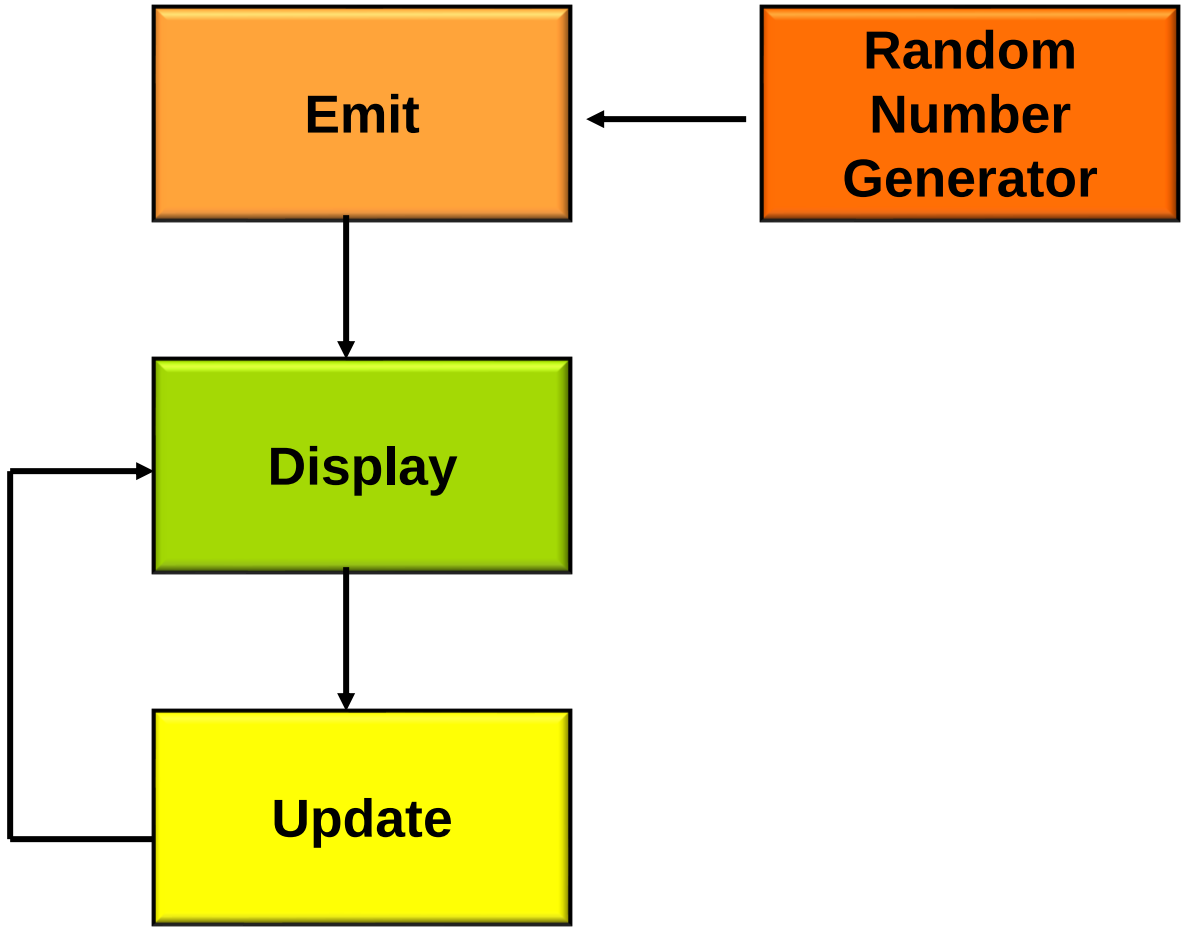
"If I know where I want the links to be drawn, what transformation parameters will put them there?"



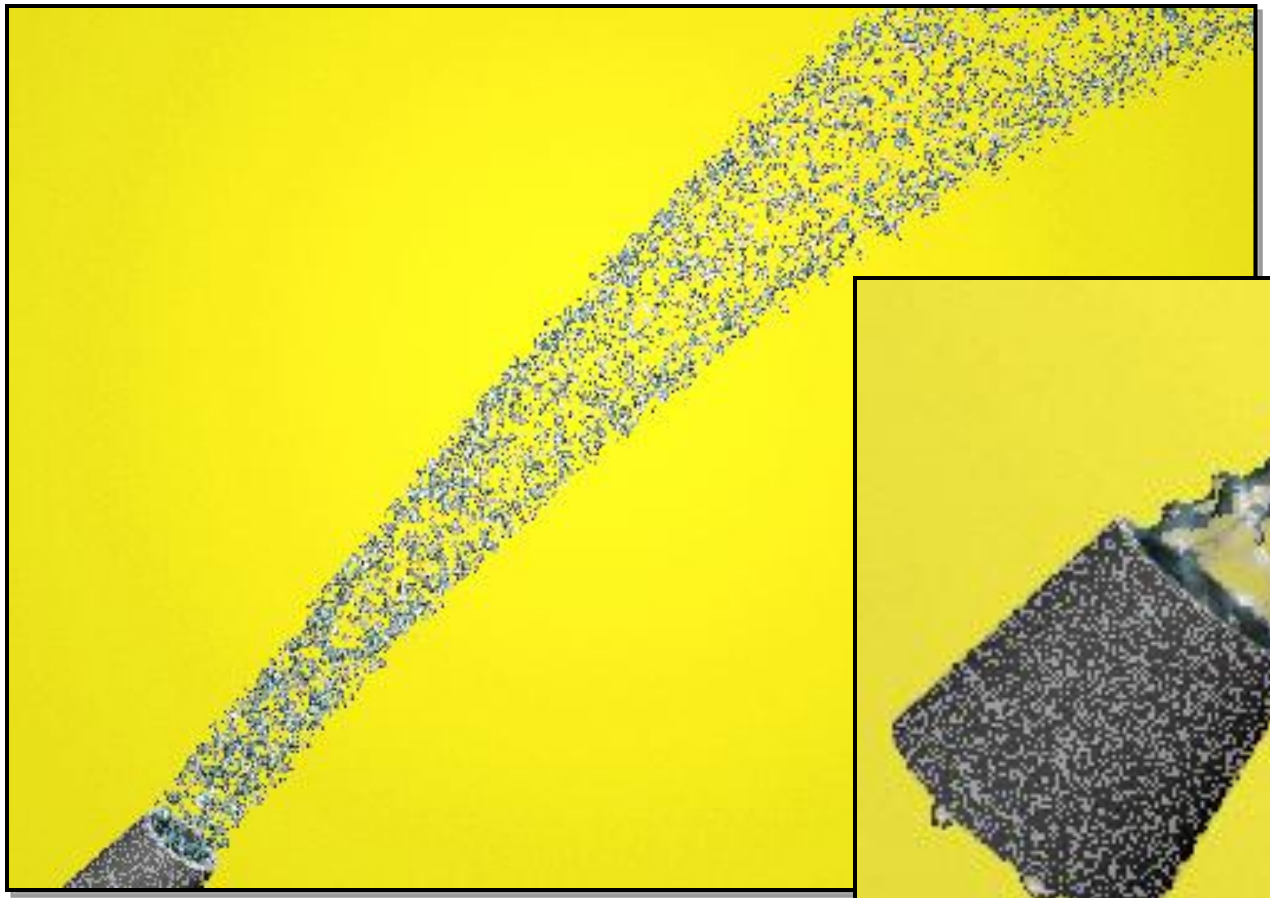
Input: Locations
Output: Angles

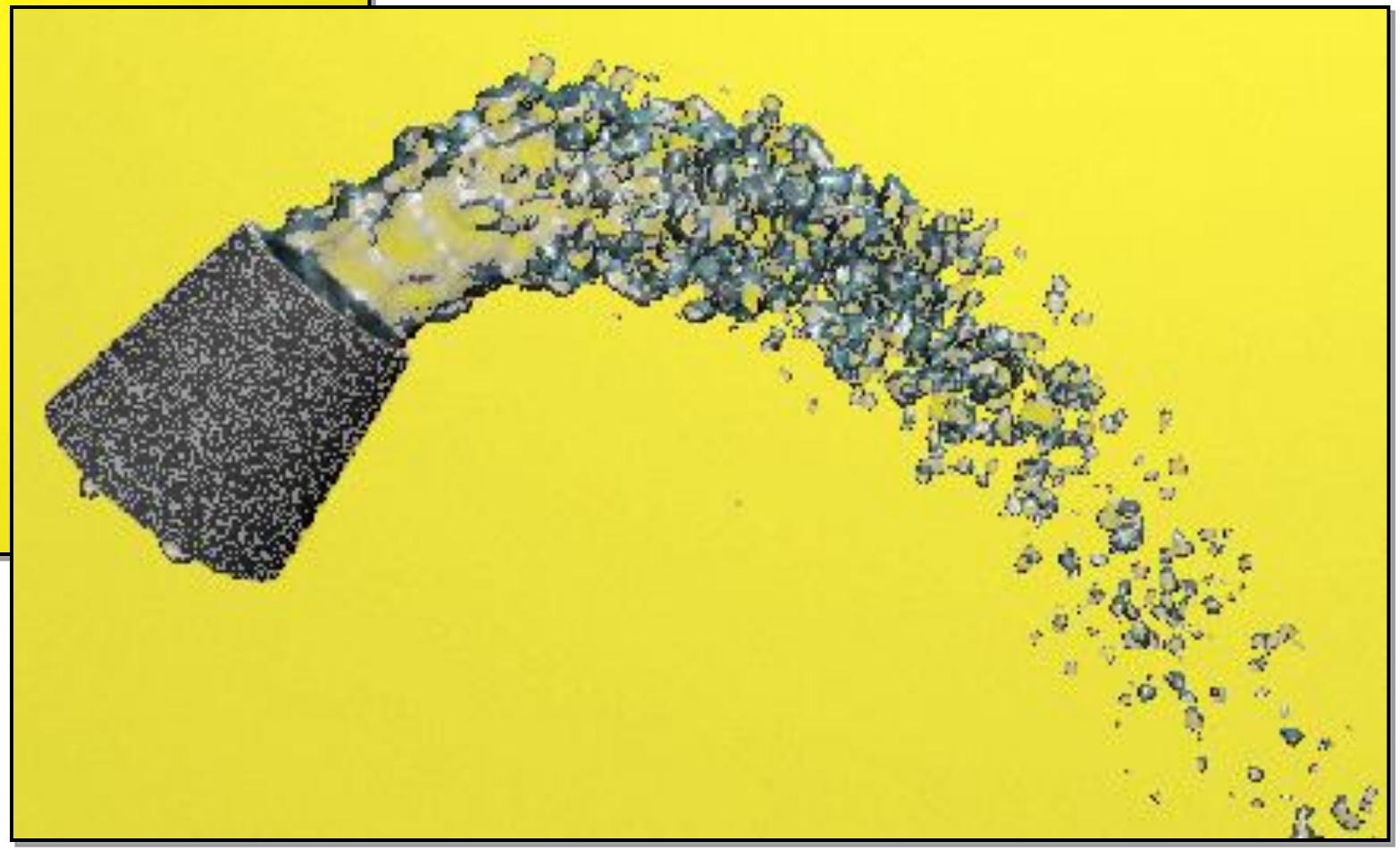
Particle Systems:
A Cross Between Modeling and Animation?

The basic process is:



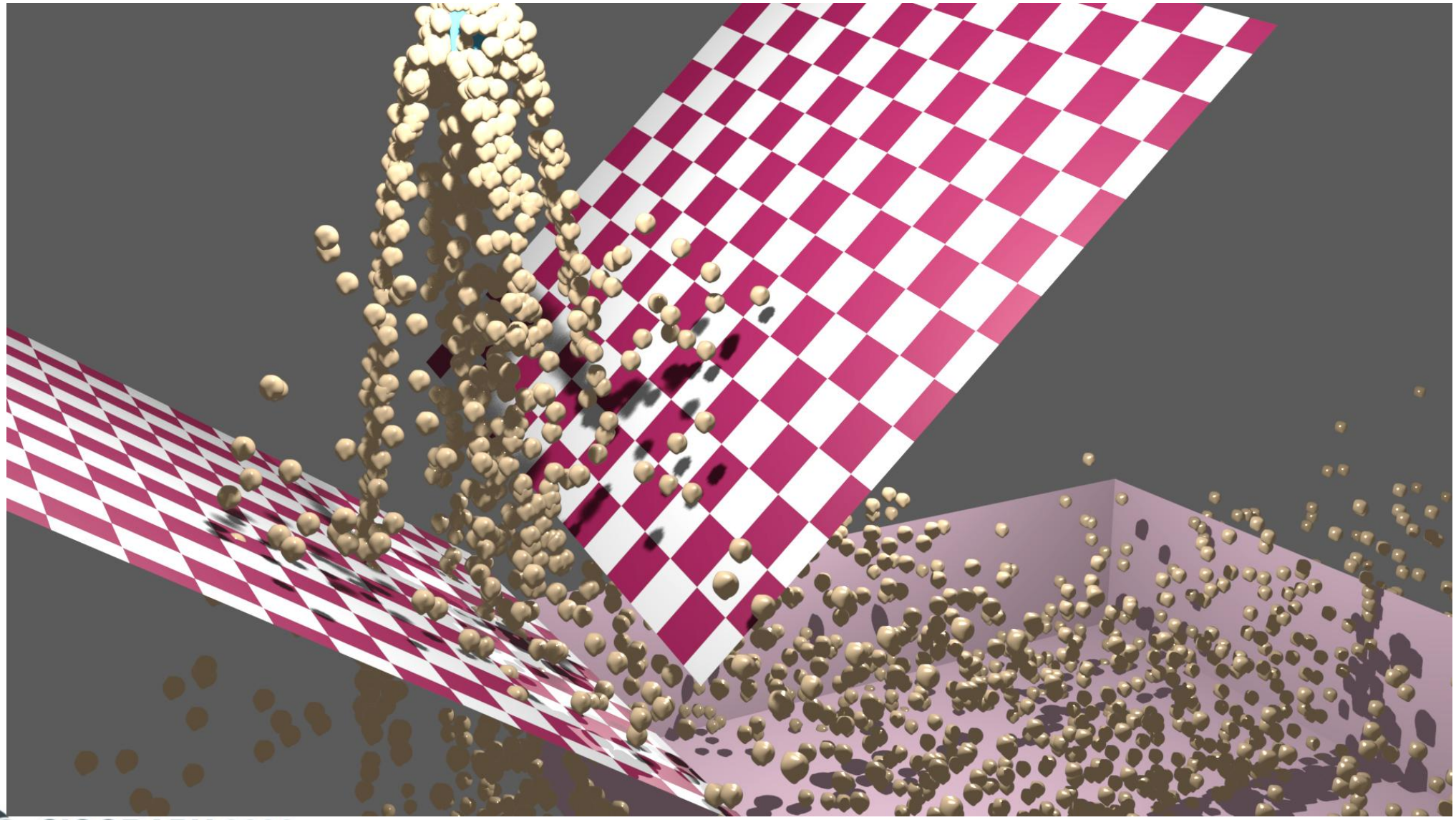
Particle Systems Examples

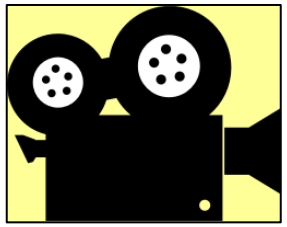




Images: Chuck Evans, used by permission

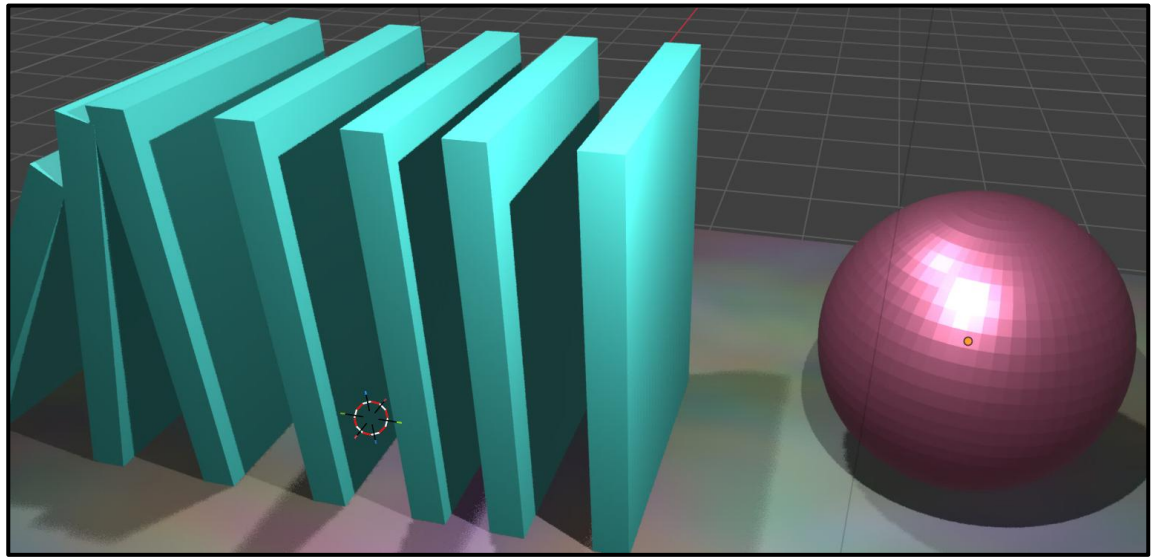
Particle Systems Examples

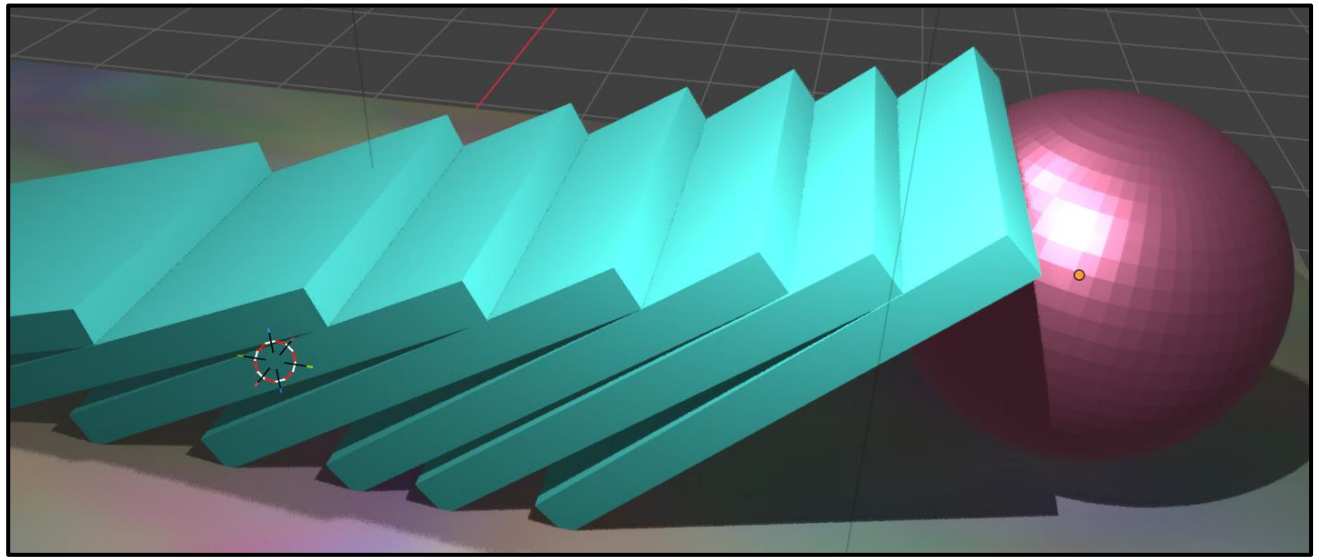




particles.mp4

Animating using Rigid-body Physics





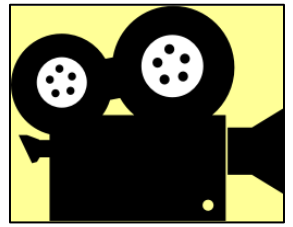
Newton's second law:

force = mass * acceleration

or:

$$\text{acceleration} = \{\ddot{x}\} = \frac{force}{mass}$$

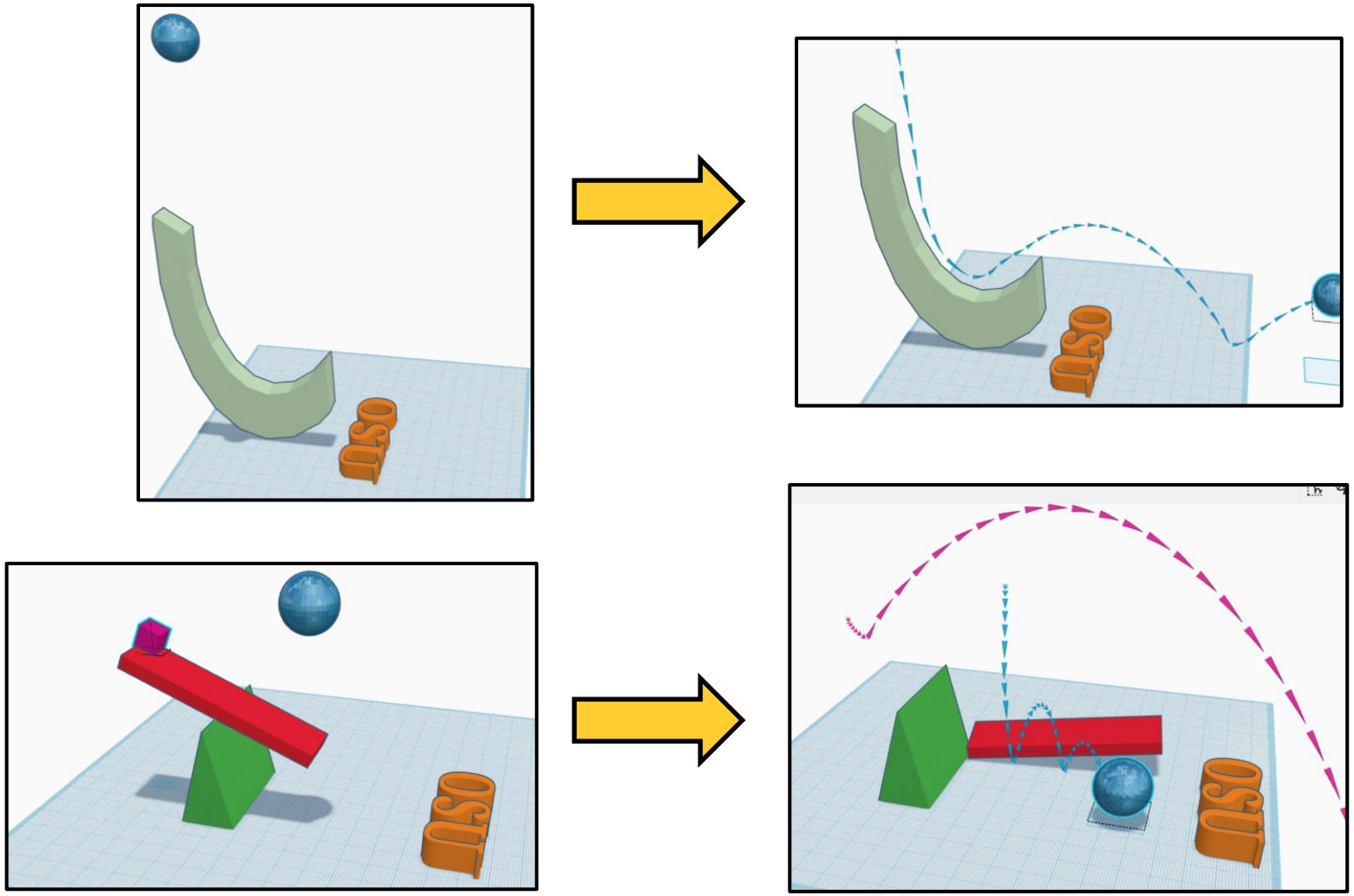
Newton's Second Law



dominos2.mp4

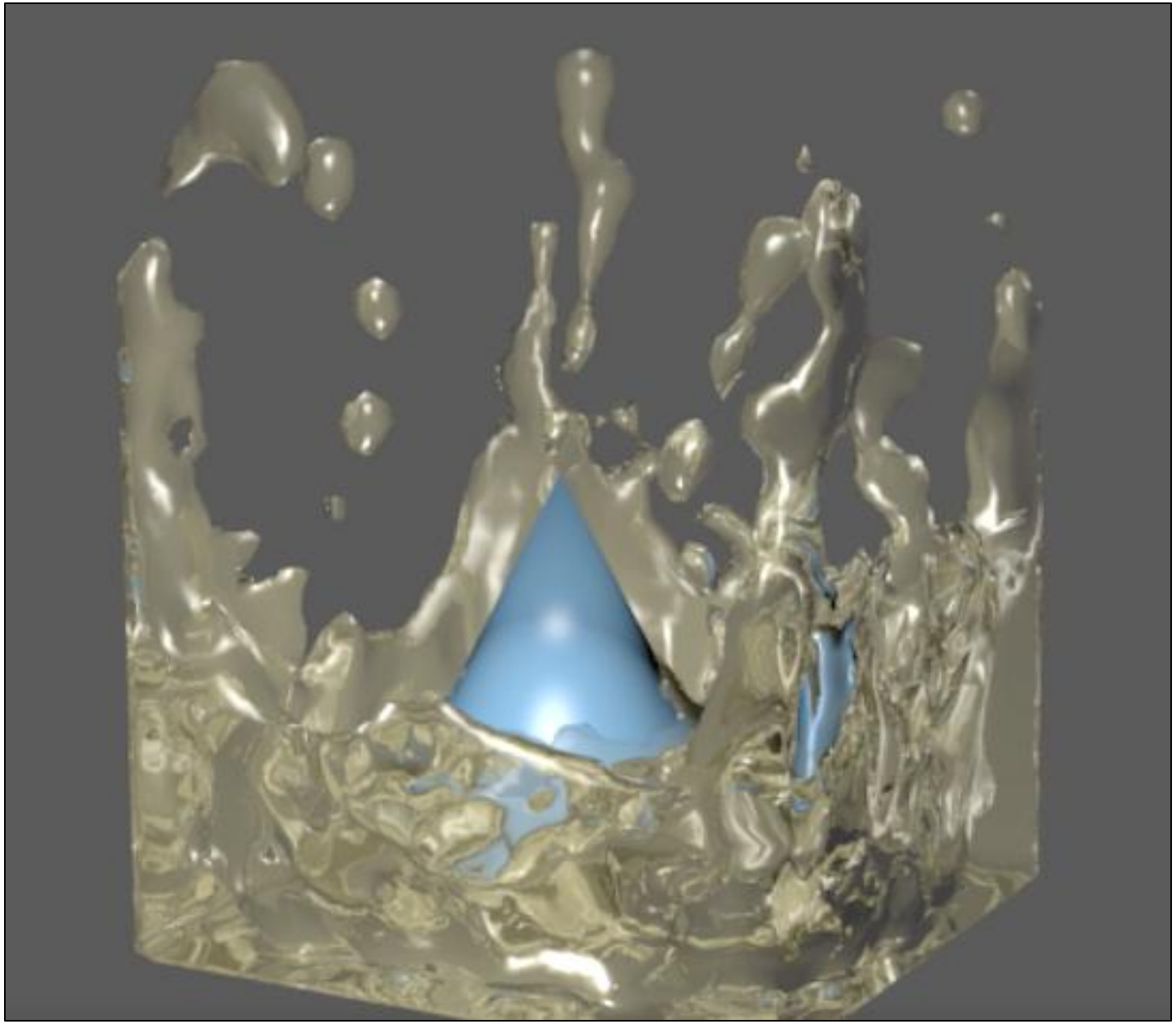
To make this work, you need to supply physical properties such as mass, center of mass, moment of inertia, coefficients of friction, coefficients of restitution, etc.

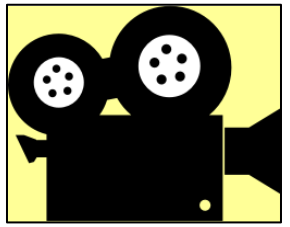
Even TinkerCad Now Has Rigid Body Physics Animation!



Want to experiment with TinkerCad physics animation for free?
Want some notes to help you get started?
http://cs.oregonstate.edu/~mjb/tinkercad

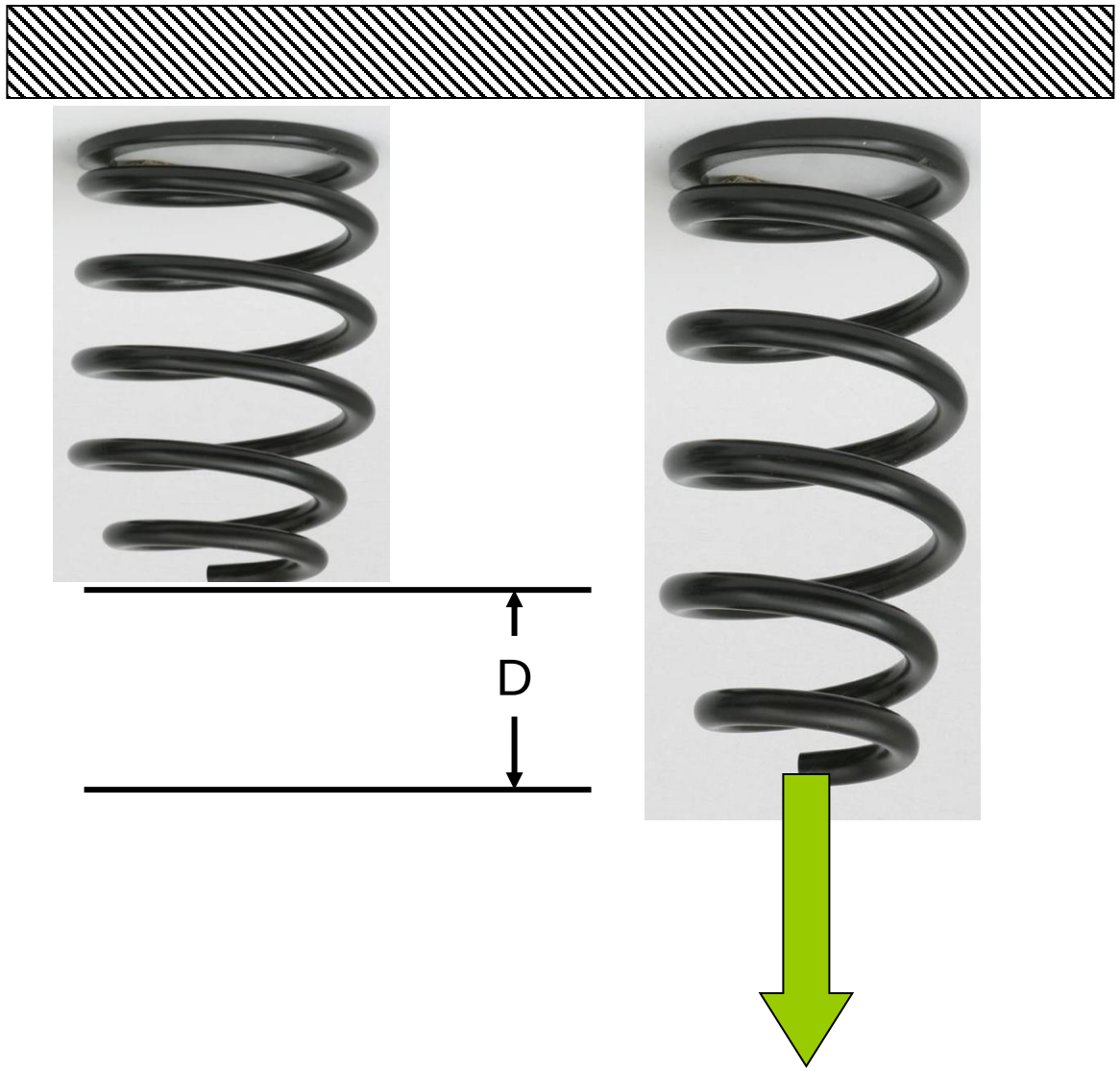
Animating using Fluid Physics





fluid.avi

Animating using Spring Physics



k = ***spring stiffness*** in newtons/cm or pounds/inch

If you know the force, the distance the spring stretches or compresses will be:

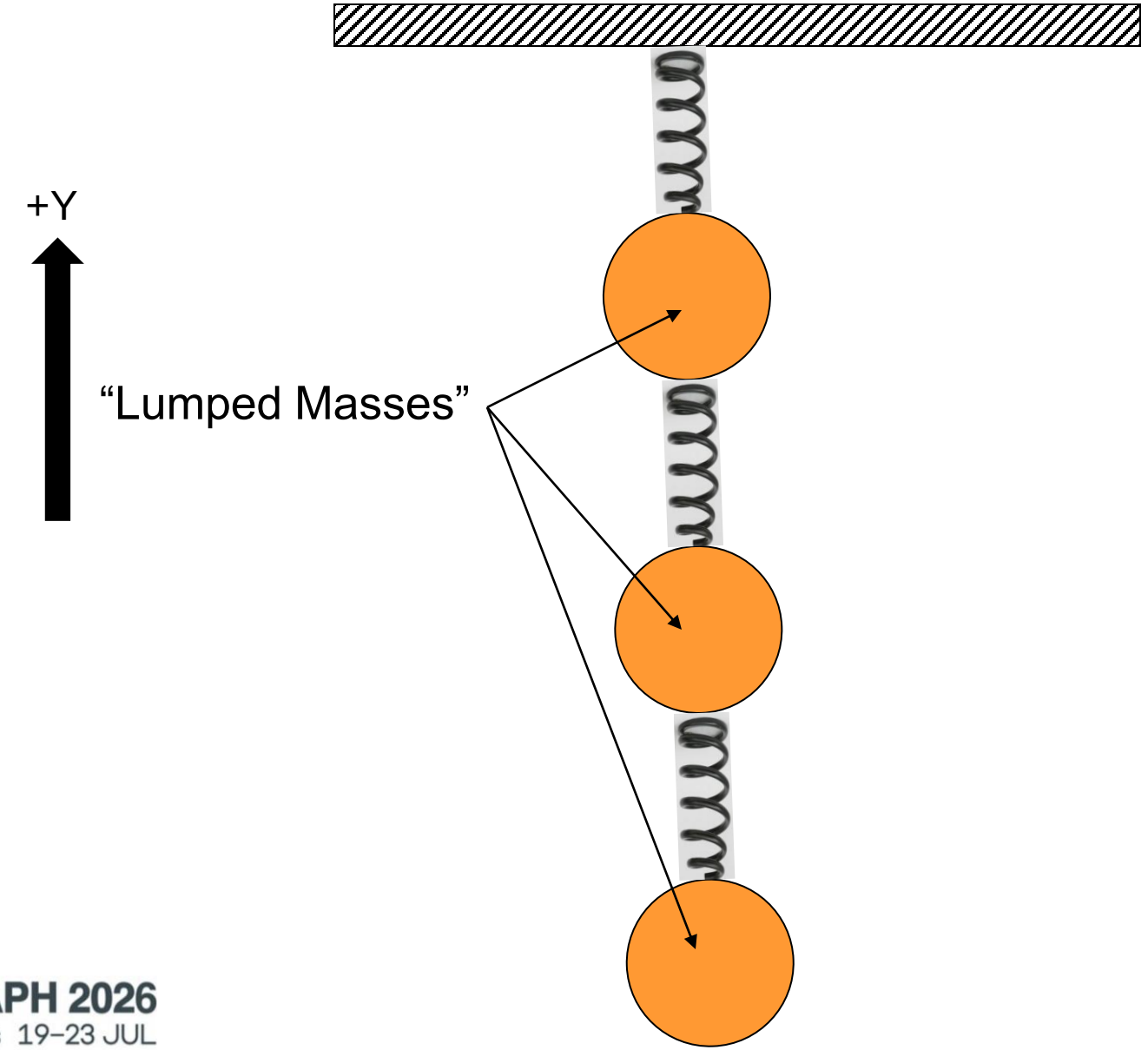
$$D = \frac{F}{k}$$

Or, if you know the distance the spring stretches or compresses, the force exerted by the spring will be:

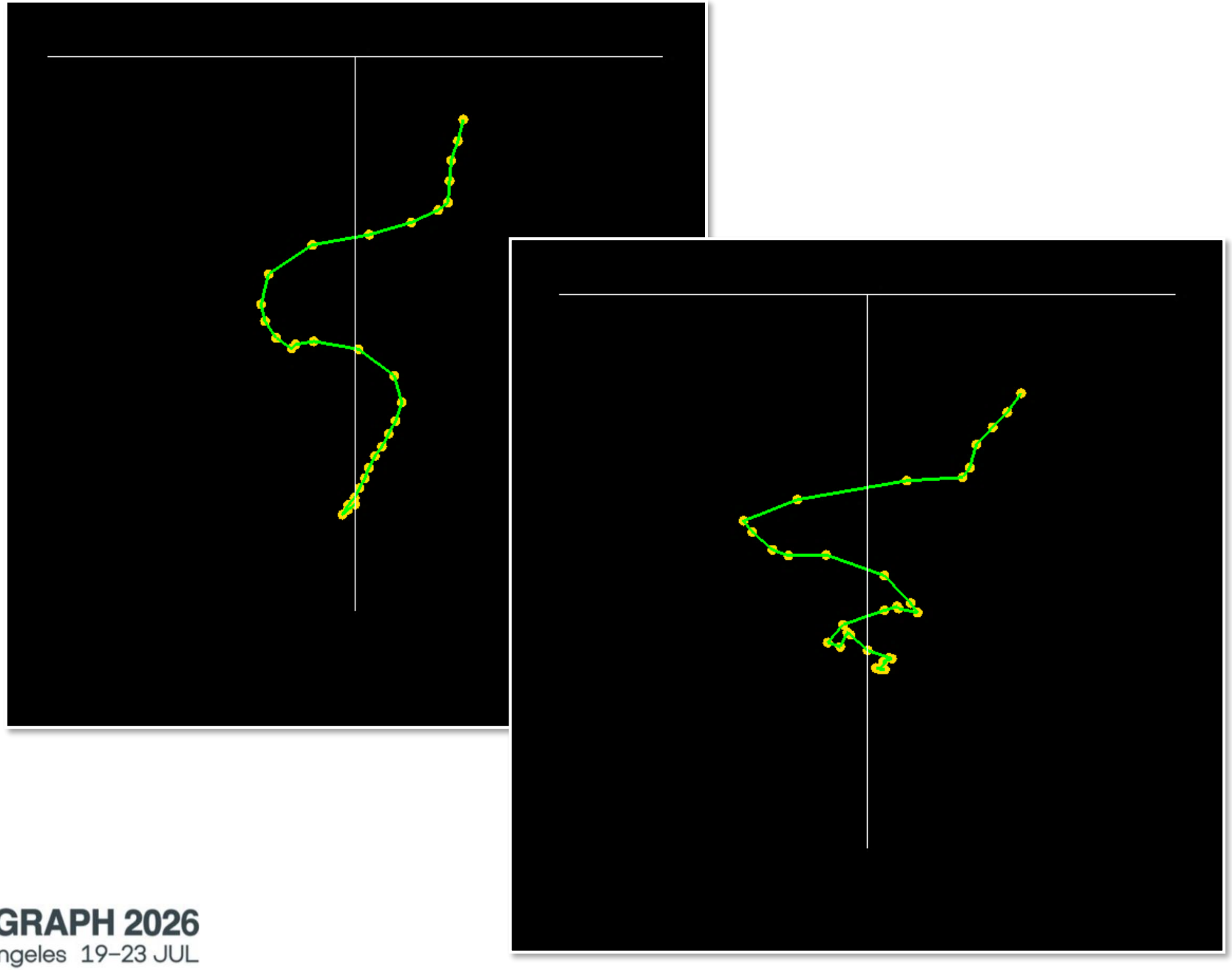
$$F = kD$$

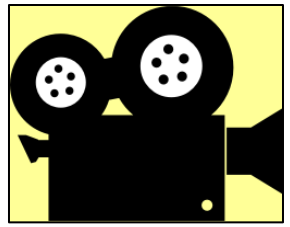
This is known as *Hooke's Law*

Animating using the Physics of a Mesh of Springs



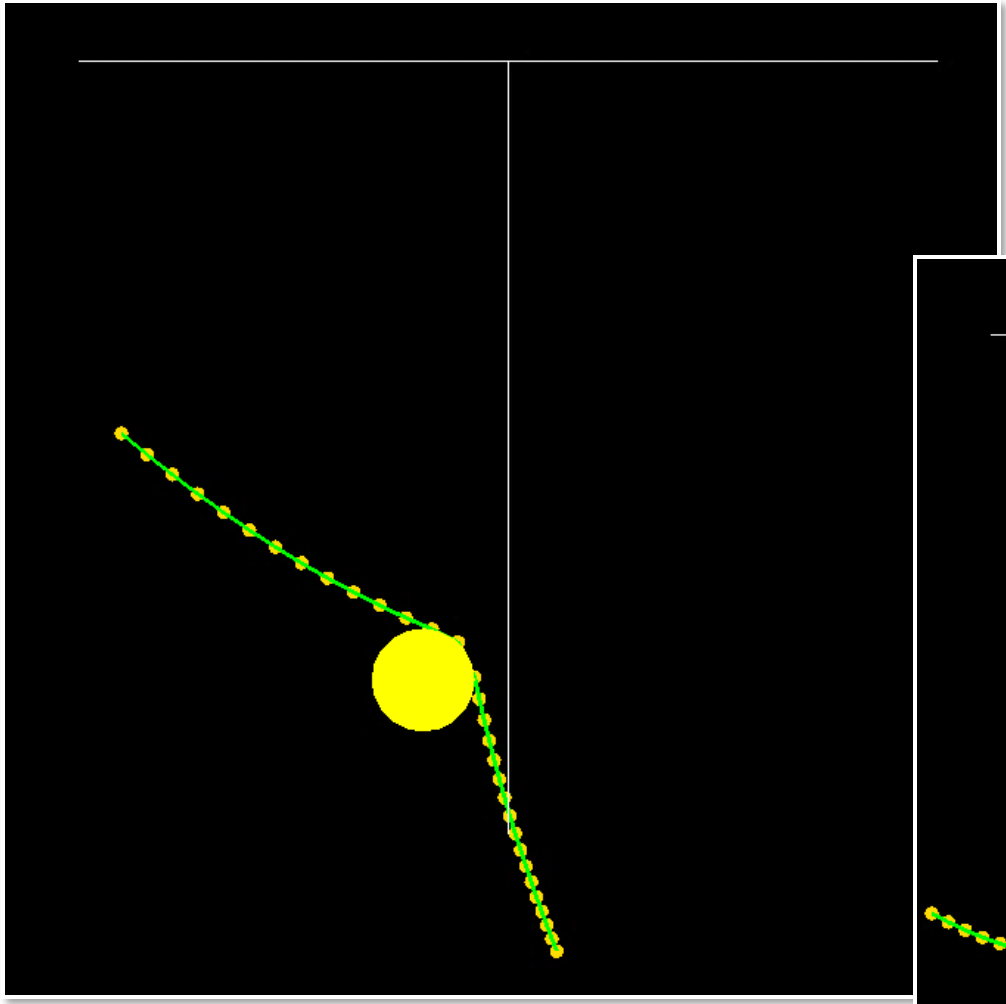
Simulating a Bouncy Chain

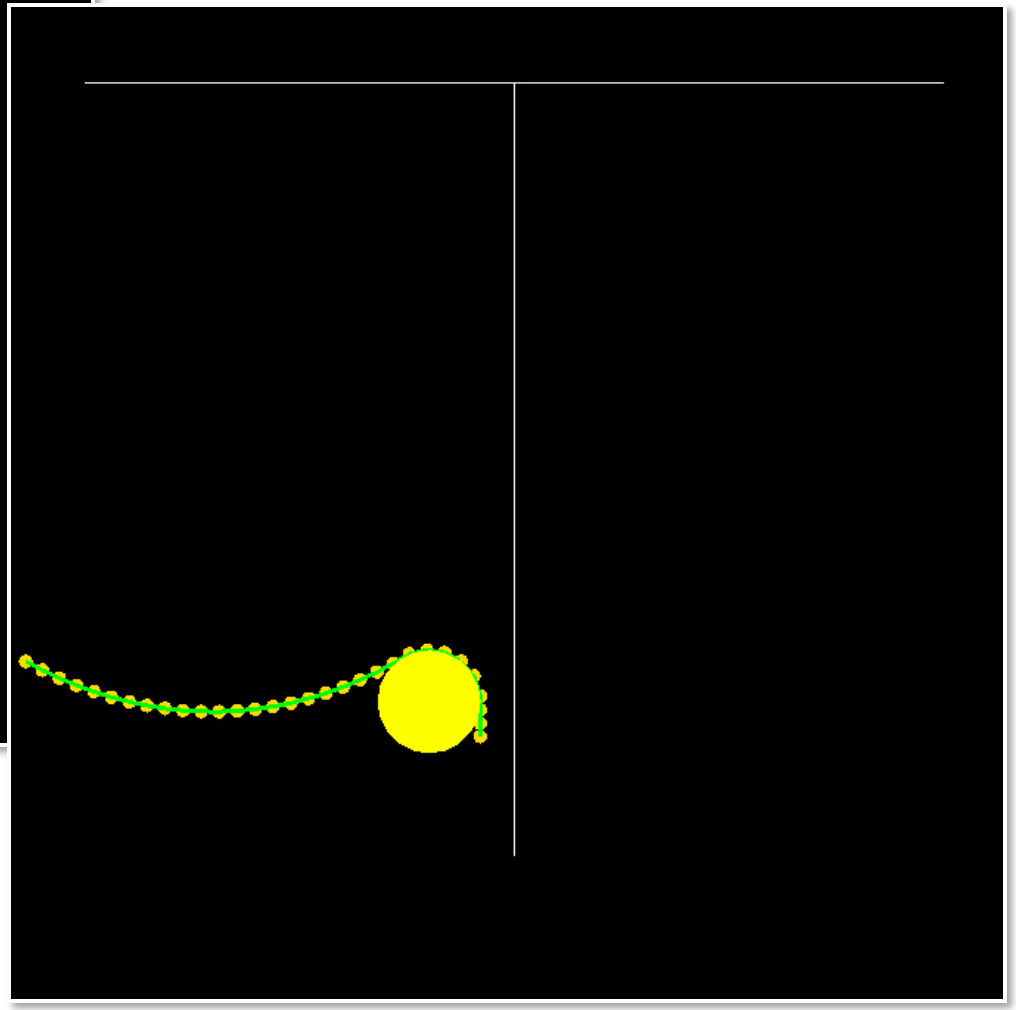




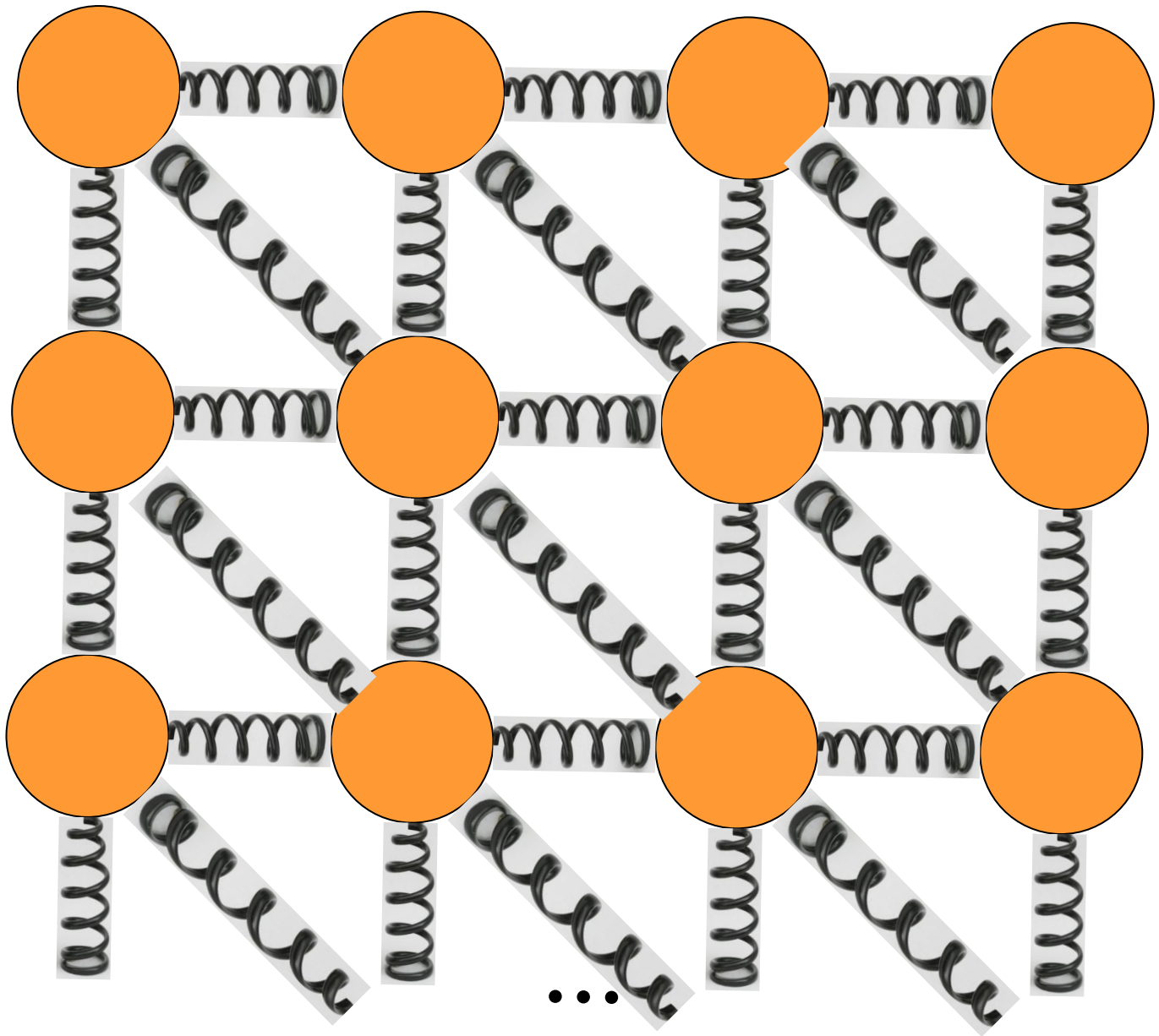
chain.mp4

Placing a Physical Barrier in the Scene

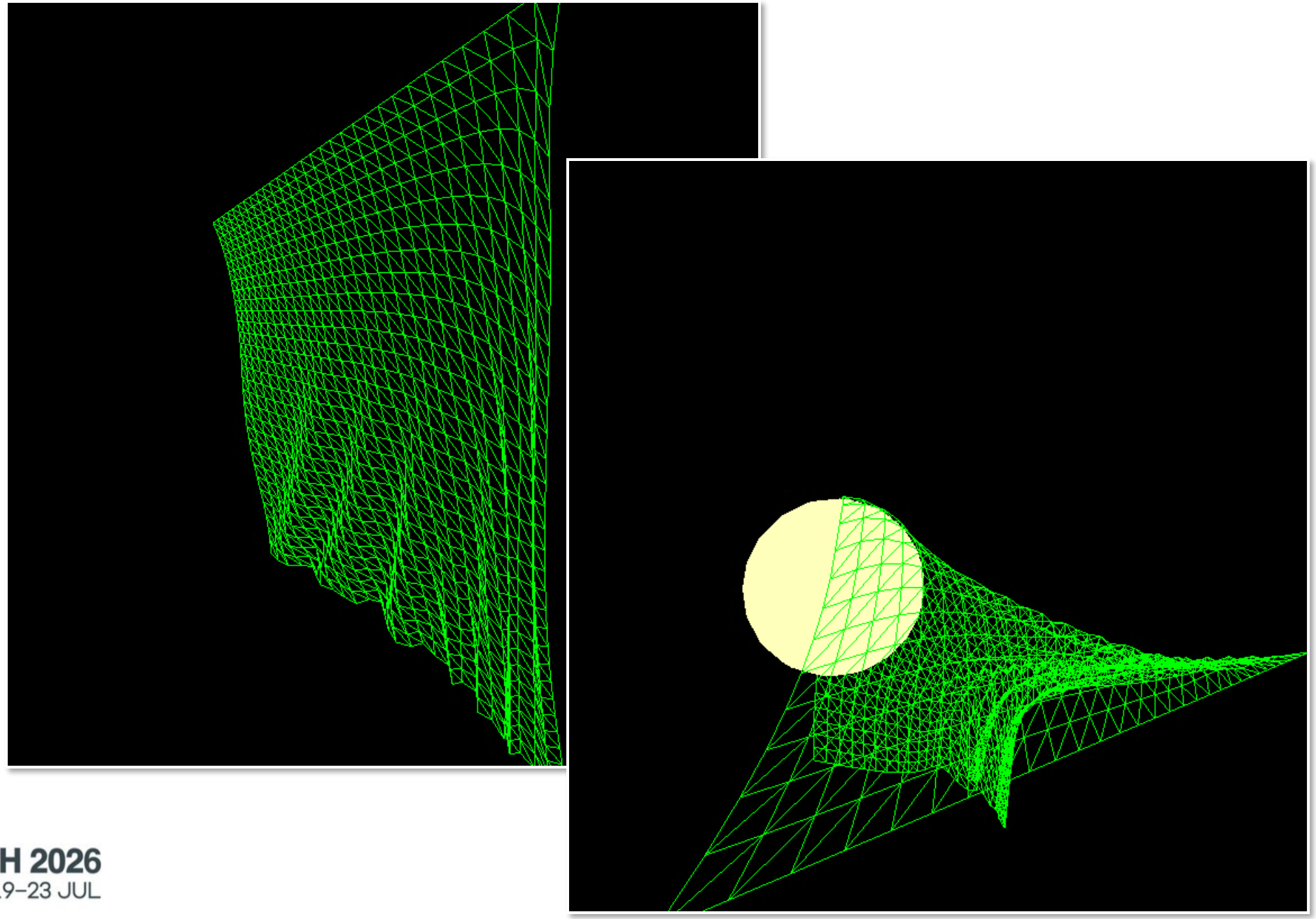




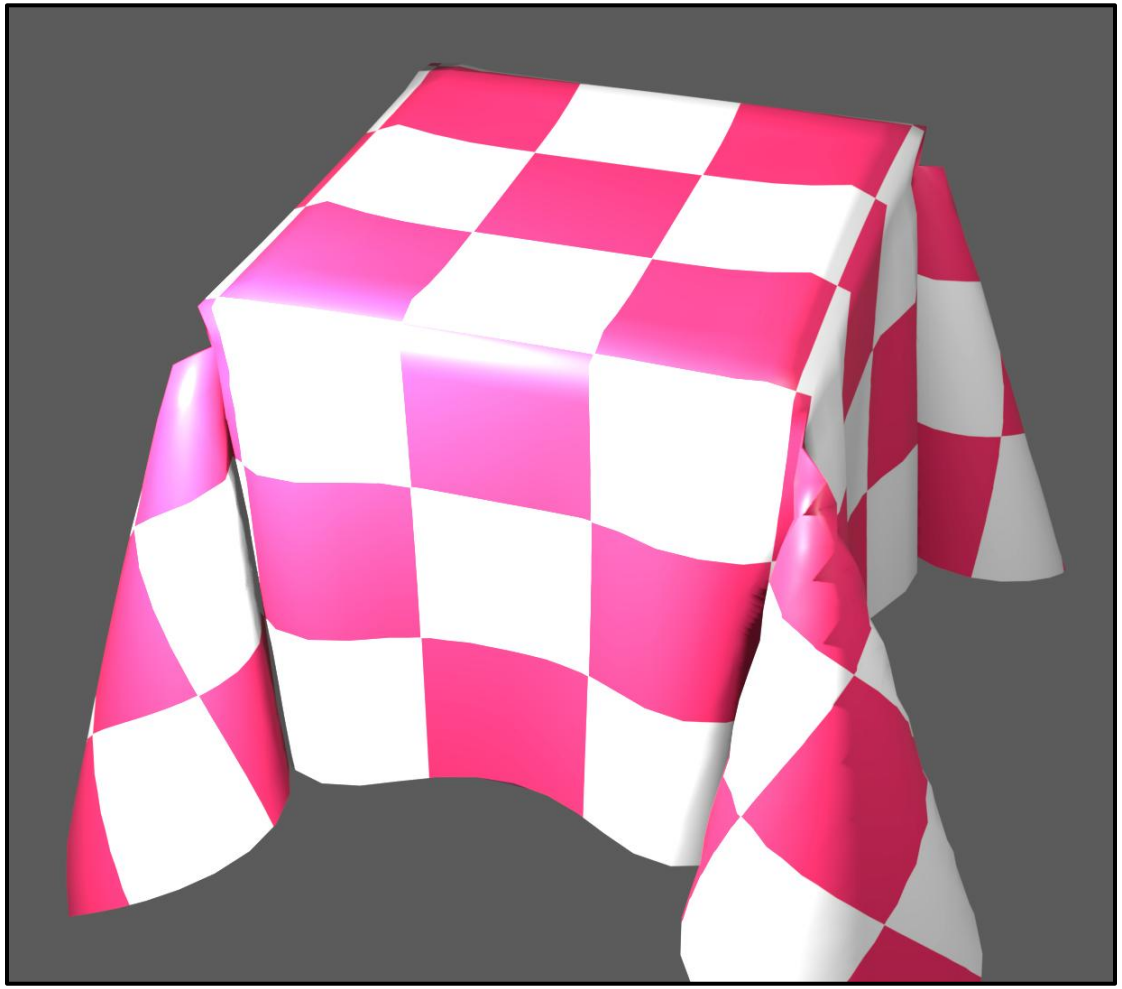
Animating Cloth



Cloth Example



Cloth Example



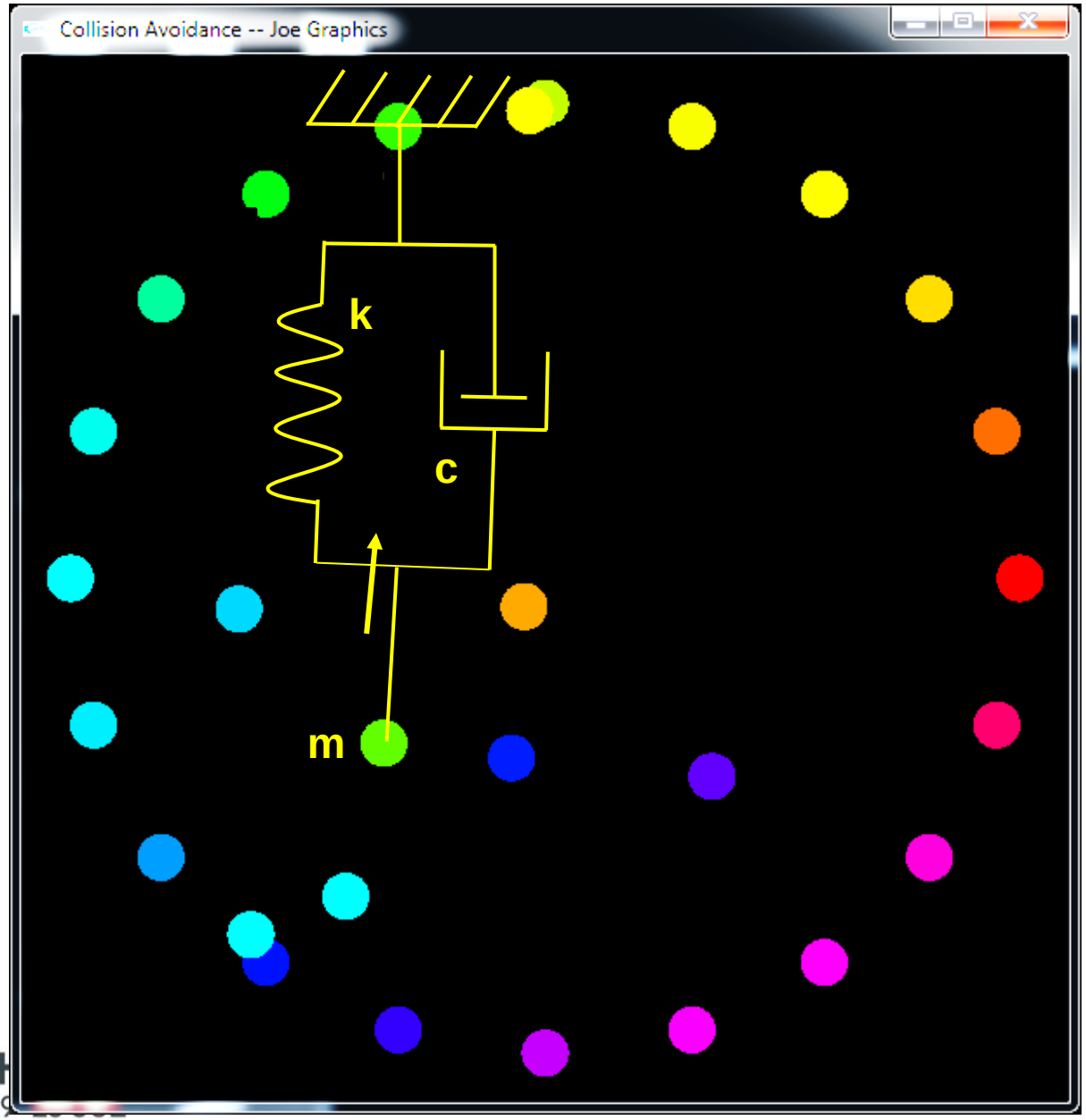
SiggraphFlag.blend

Want to experiment with a free cloth animation program?
Want some notes to help you get started?
http://cs.oregonstate.edu/~mjb/blender

Functional Animation – “Fake Physics”

The Challenge: animate a collection of objects, each trying to move to a target, but without colliding with each other.

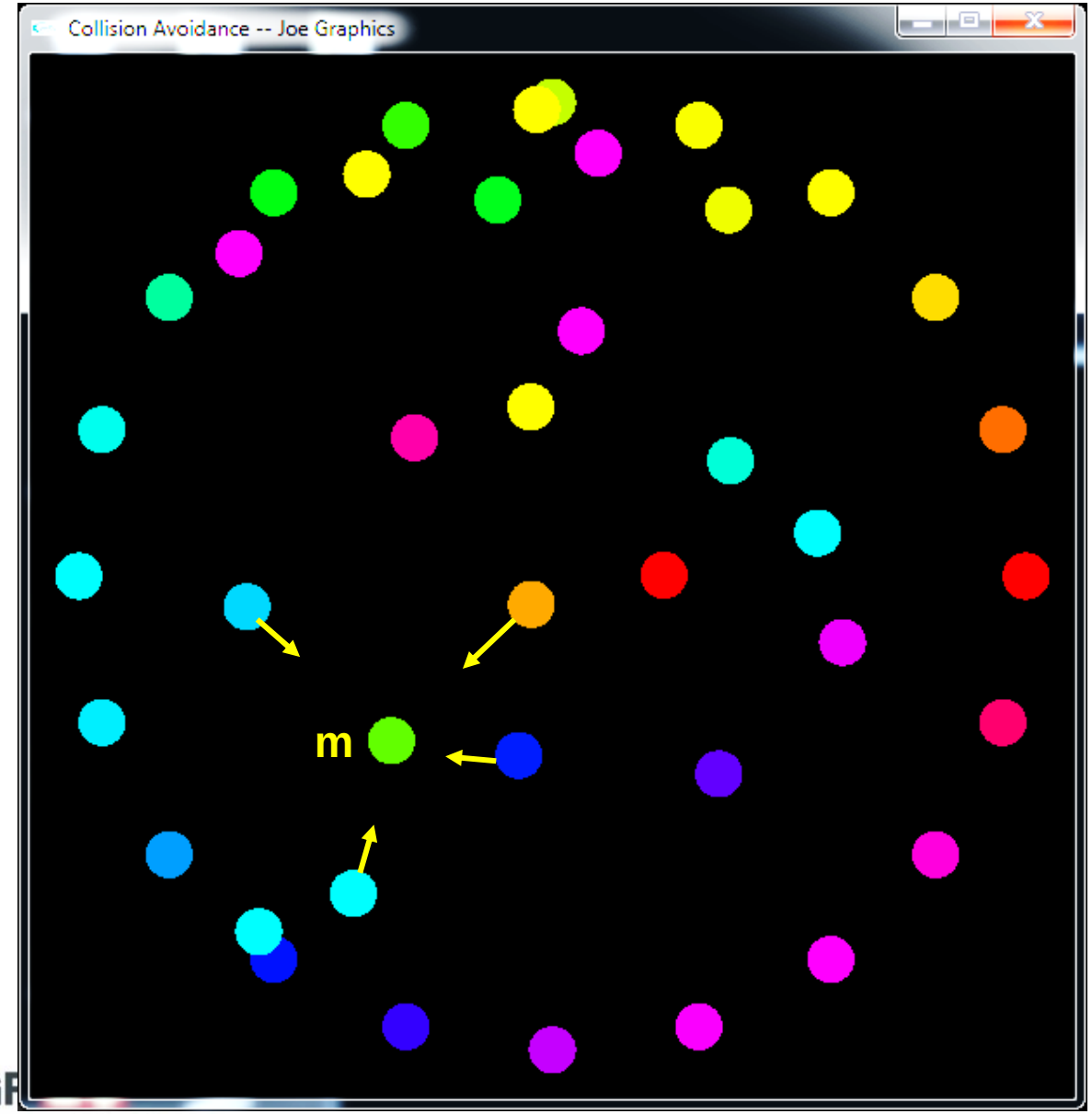
Functional Animation:
Make the Object *Want* to Move Towards a Goal Position . . .



$$m\ddot{x}+c\dot{x}+kx=0$$

Equation of motion for a spring-mass-damper system

Functional Animation:
. . . While Making it *Want* to Keep Away from all other Objects



$$m\ddot{x} = \sum F_{repulsive}$$

Repulsion Coefficient

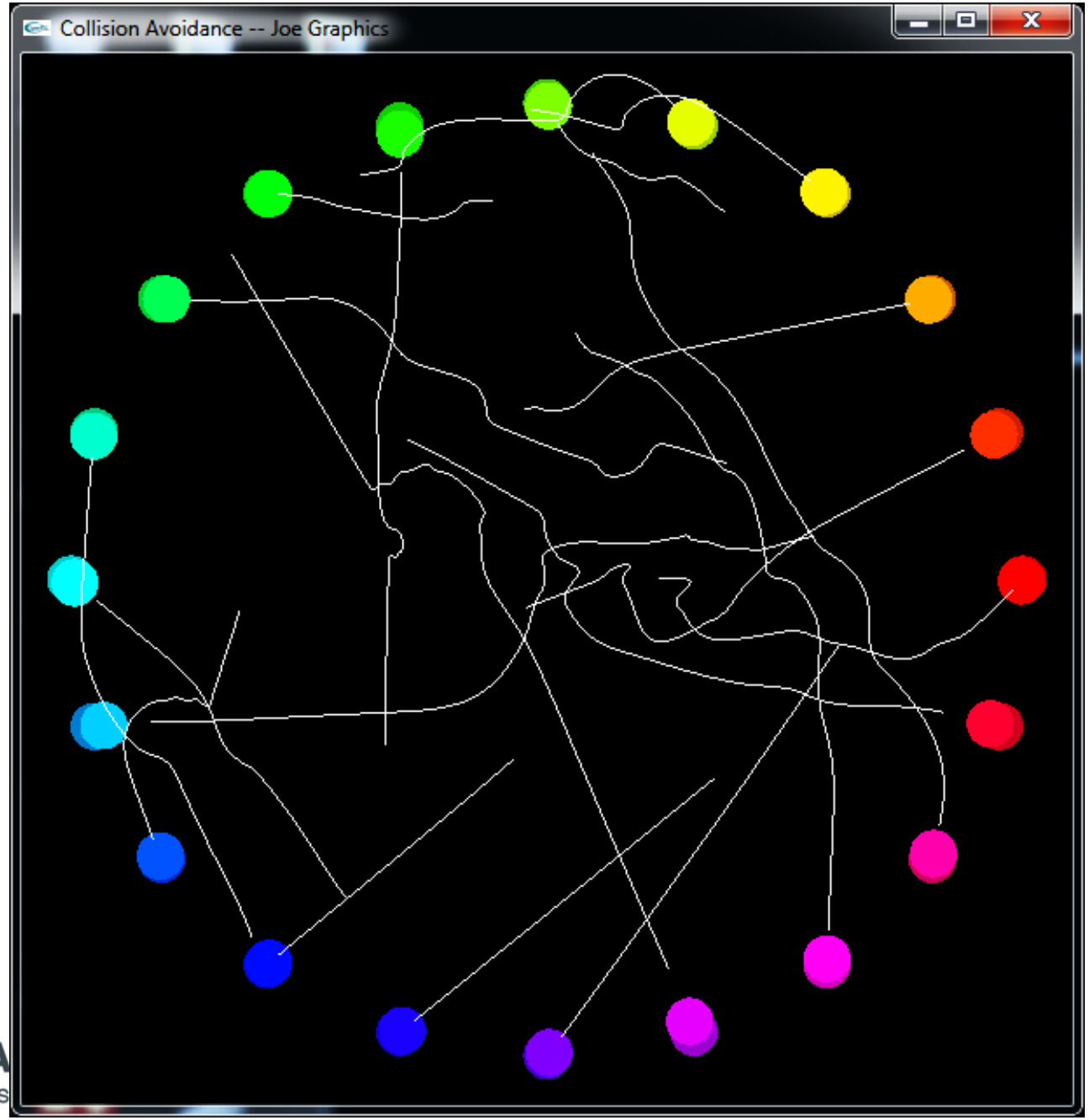
$$F_{repulsive} = \frac{C_{repulse}}{d^{Power}}$$

Distance between the boundaries of the 2 bodies

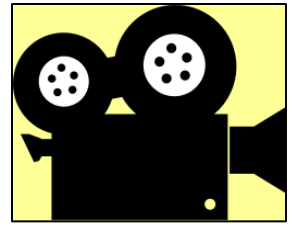
Repulsion Exponent

Fake equation of motion for two masses trying to push each other away – I just made this up…

Functional Animation

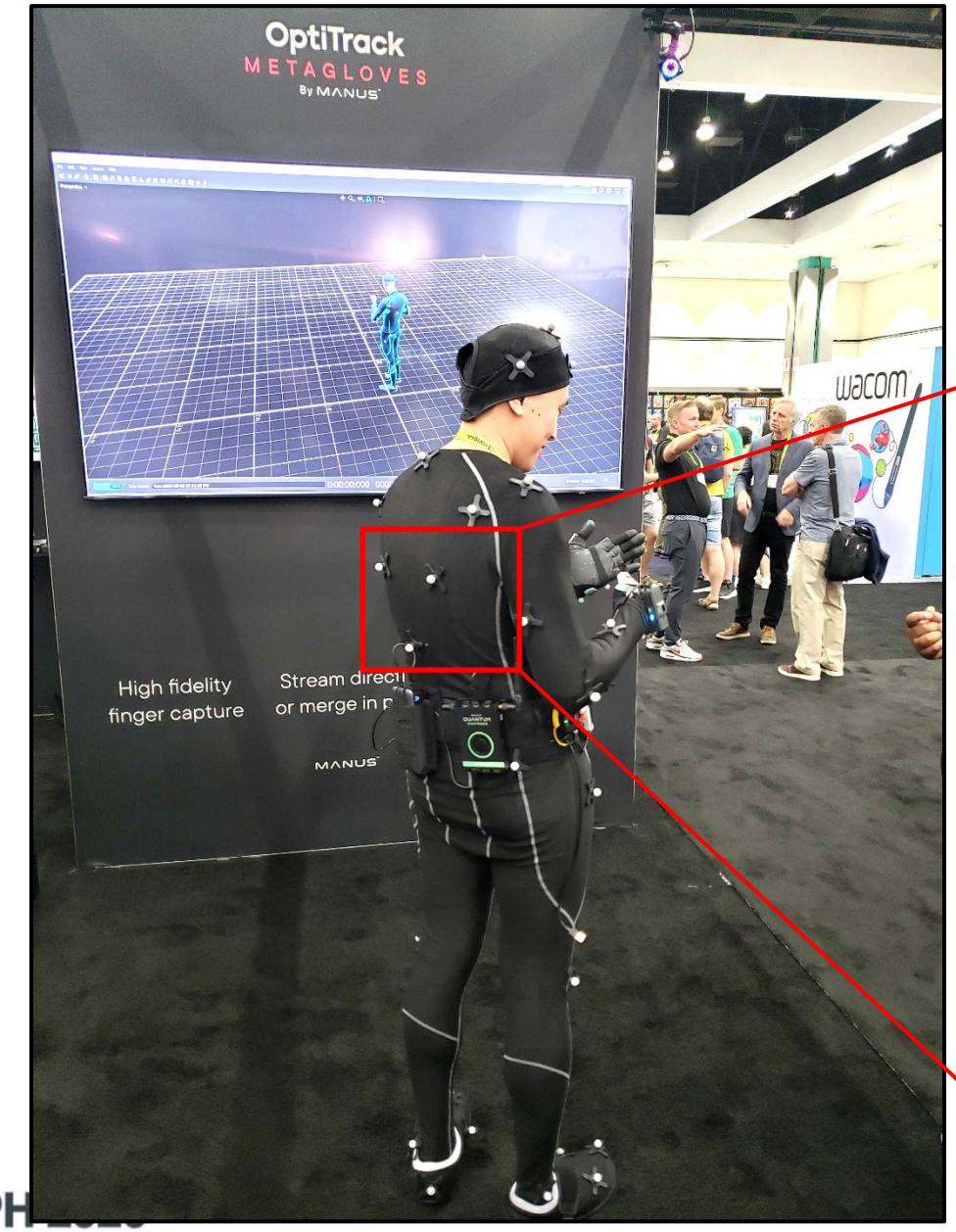


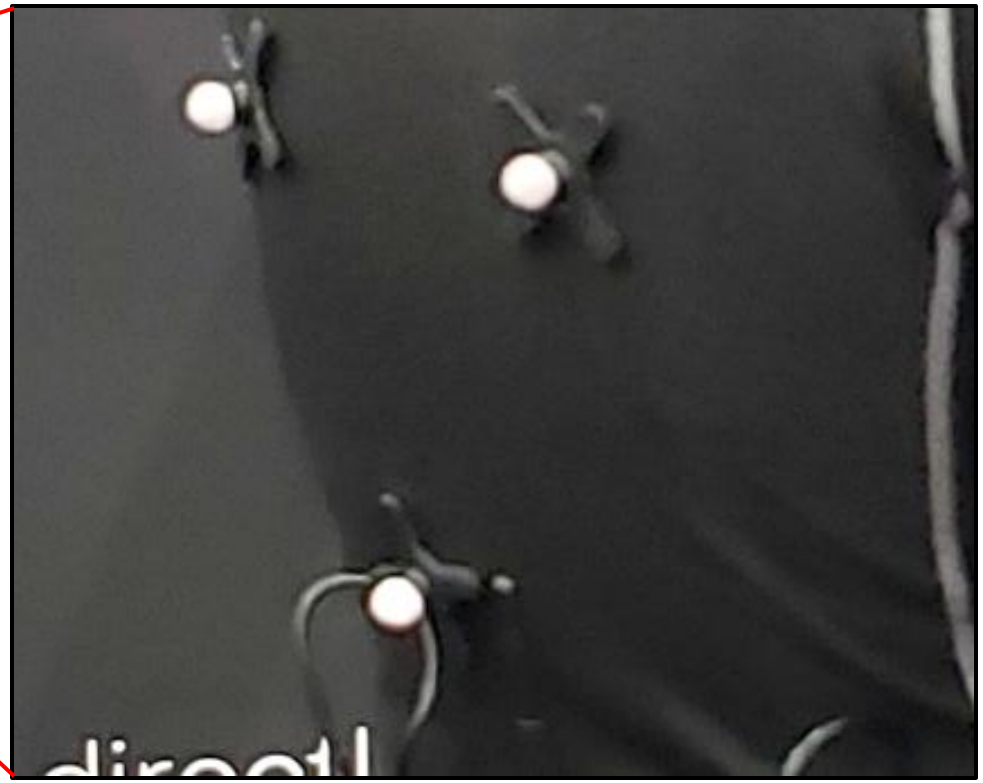
We get a collection of objects, each trying to move to a target, but without colliding with each other.



avoid.mp4

Motion Capture (“MoCap”) as an Input for Animation





Natural Point, used by permission

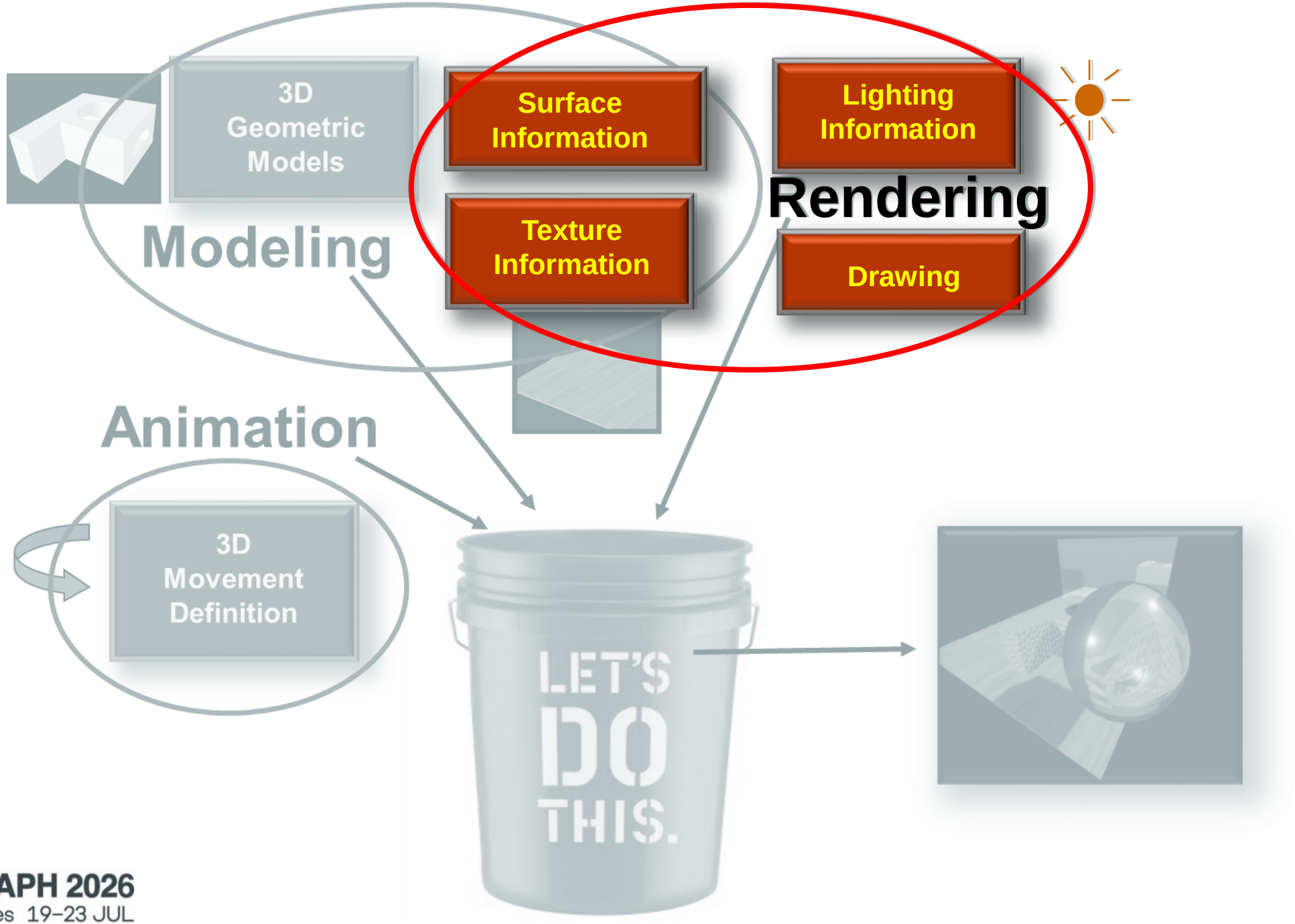
Photos: Mike Bailey

Even Animals can be MoCapped (if you dare…)

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

Photo courtesy of: DIGIC Services' Mocap Studio, used by permission

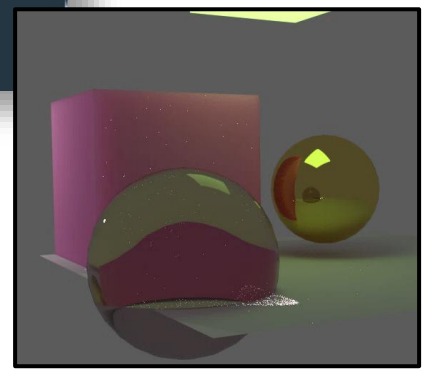
Creating an image of your scene

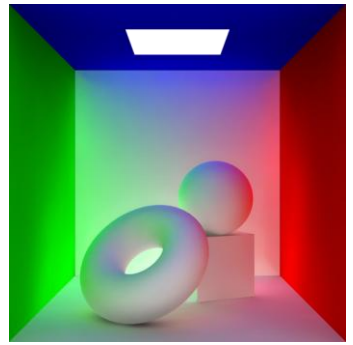

3D
Geometric
Models
Modeling
Surface
Information
Texture
Information
Lighting
Information
Rendering
Drawing
Animation
3D
Movement
Definition
LET'S
DO
THIS.

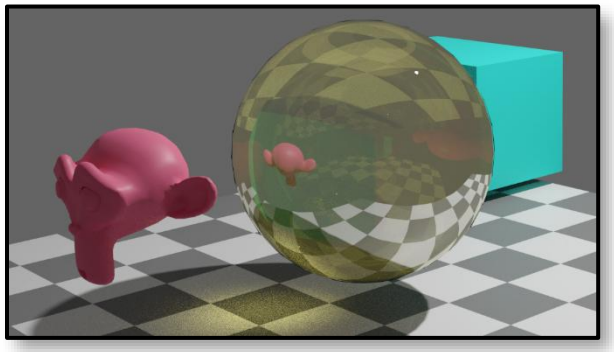
Rendering

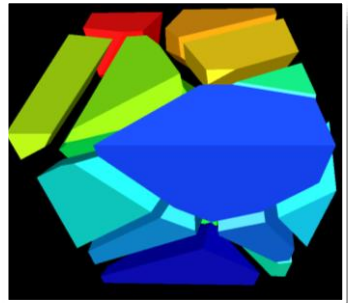
Rendering is the process of creating an image of geometric models. Again, there are questions you need to ask first:

- Why am I doing this?
- How realistic do I want this image to be?
- How much compute time do I want this to take?
- Do I need to take lighting into account?
- Does the illumination need to be global or will local do?
- Do I need to create shadows?
- Do I need to create reflections and refractions?







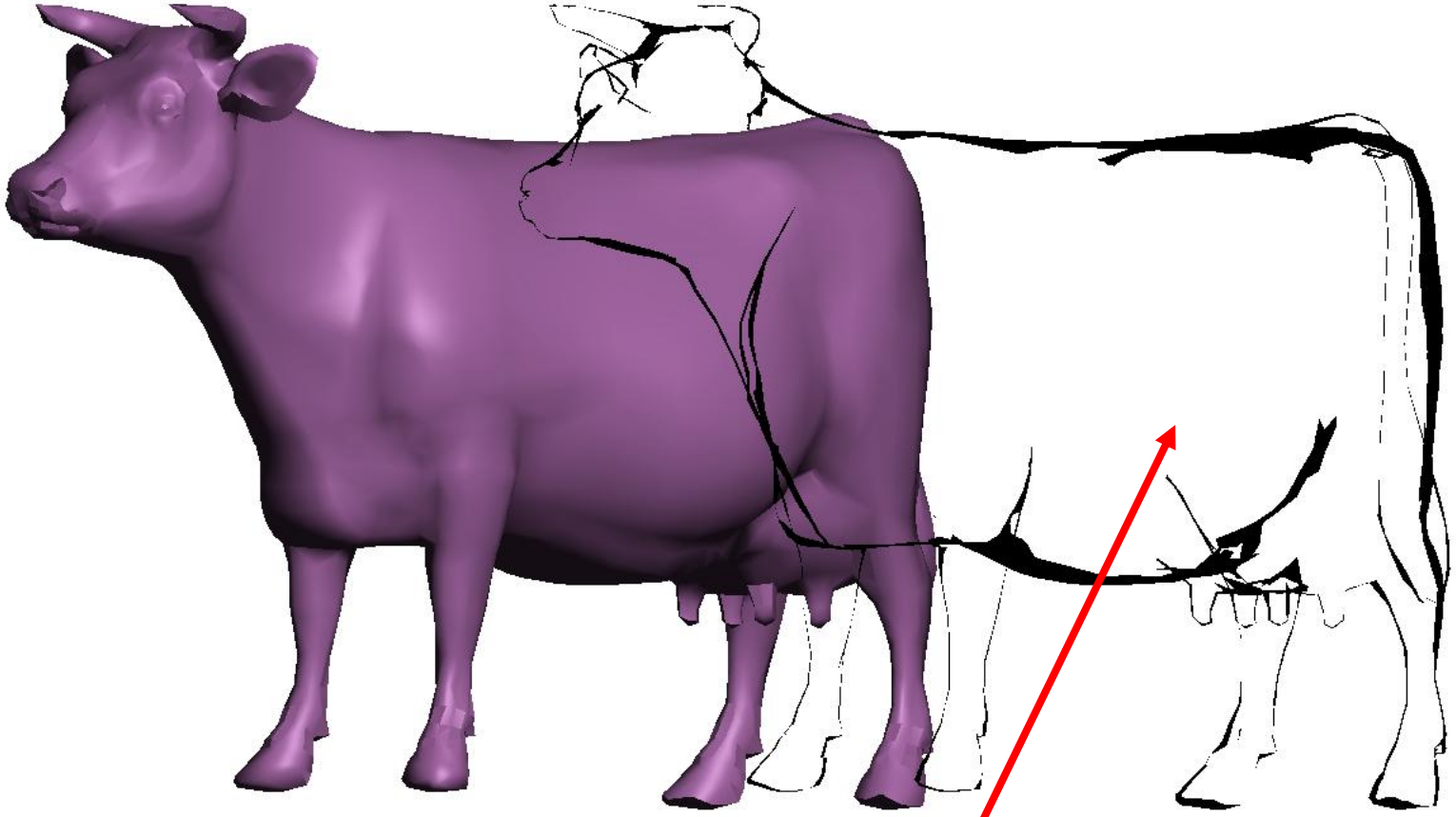


Want to experiment with a free rendering program?
Want some notes to help you get started?
http://cs.oregonstate.edu/~mjb/blender

Non-Photorealistic Rendering: Toon Shading

Photo by Steve Cunningham, used with permission

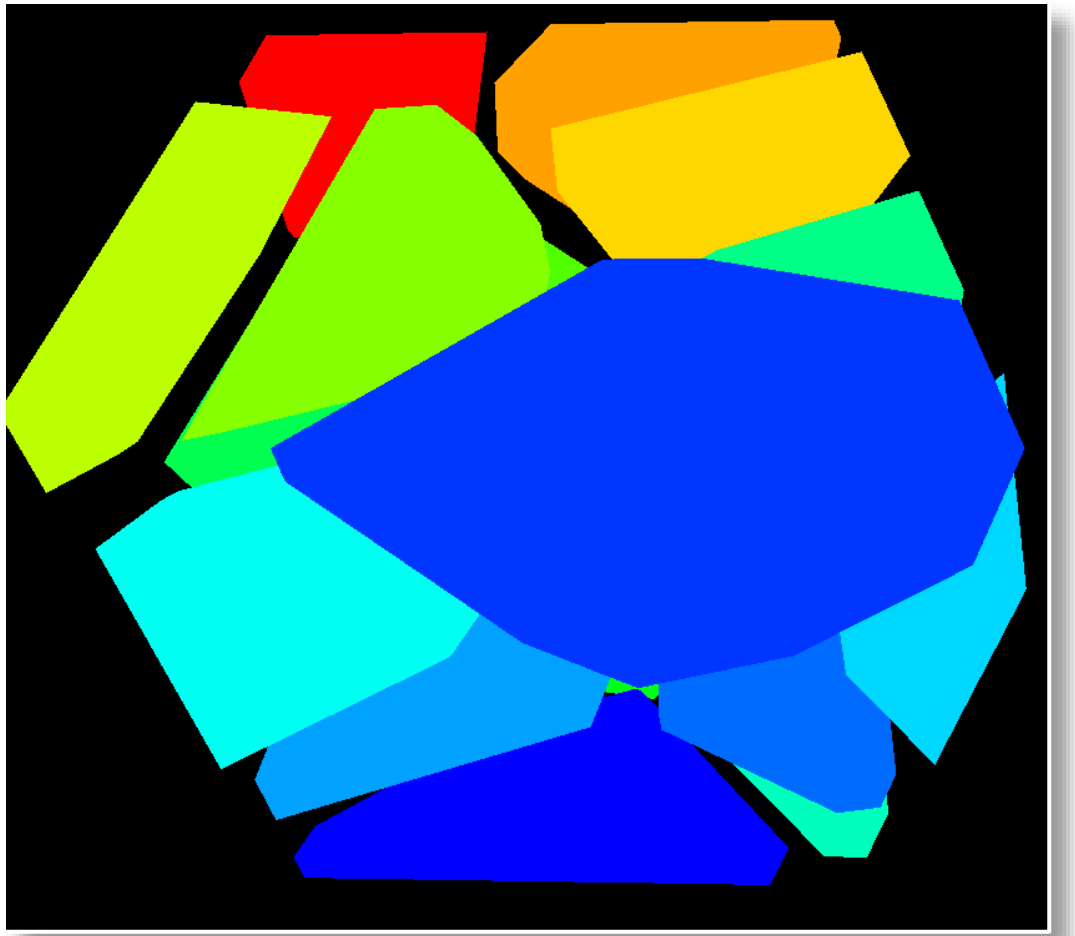
Non-Photorealistic Rendering: Silhouettes using Shaders



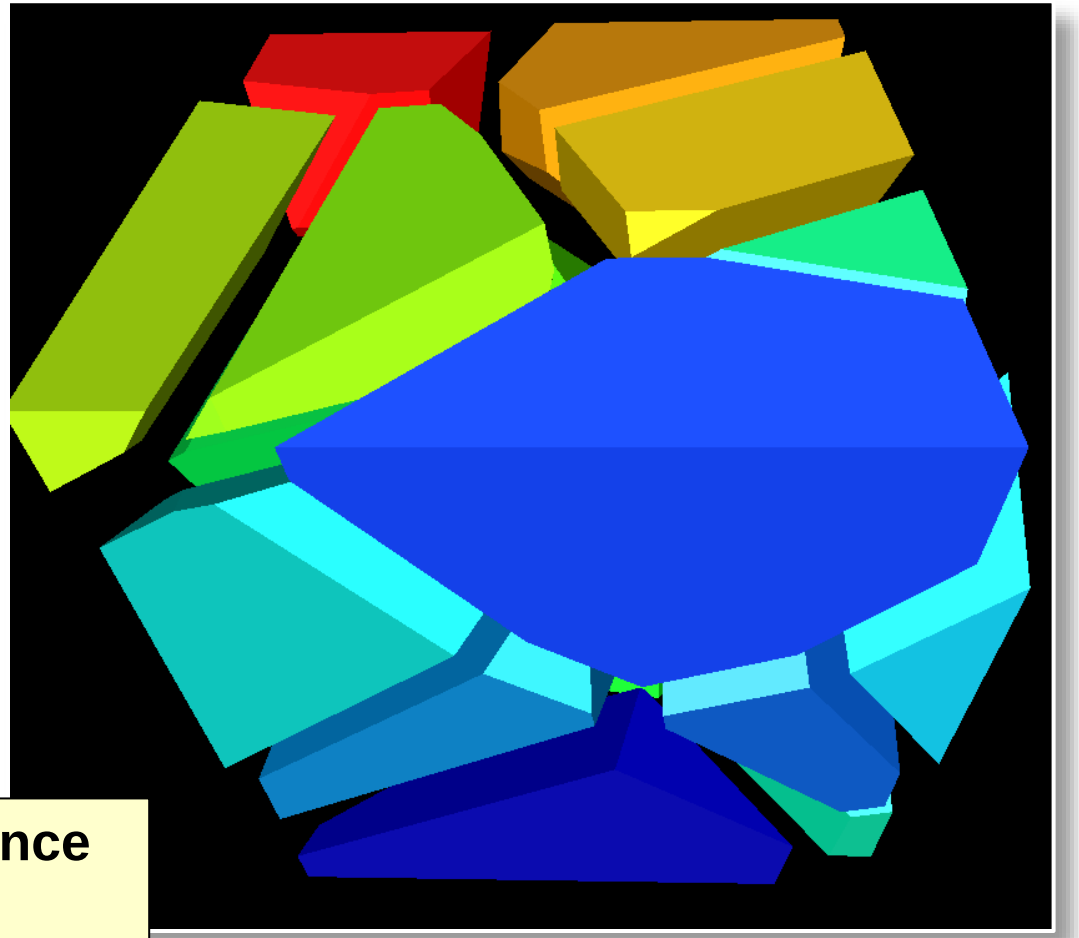
```
if( abs( dot(Eye,Normal) ) > uTol )
        discard;
else
        gl_FragColor = vec4( SILHCOLOR, 1. );
```

Why Do We Care About Lighting?

No lighting



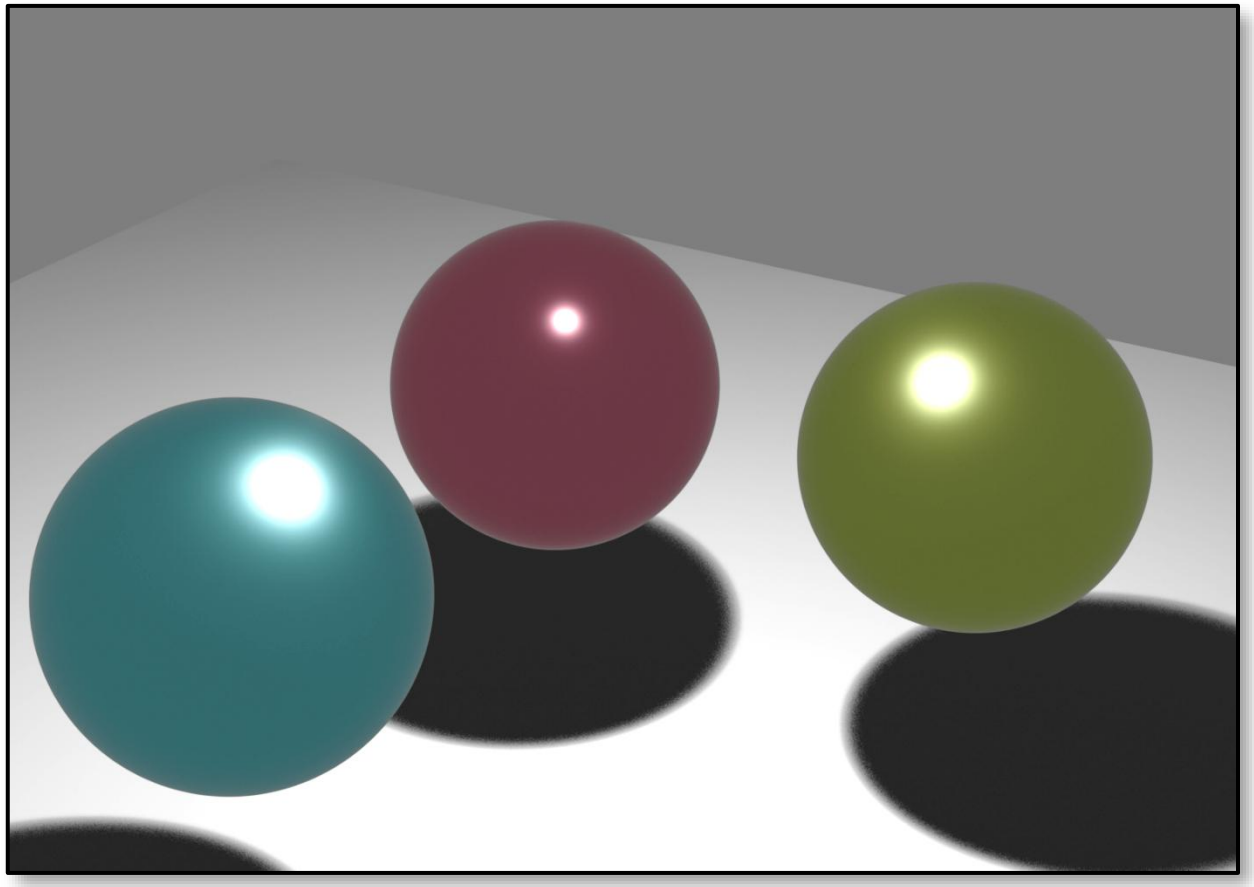
Lighting



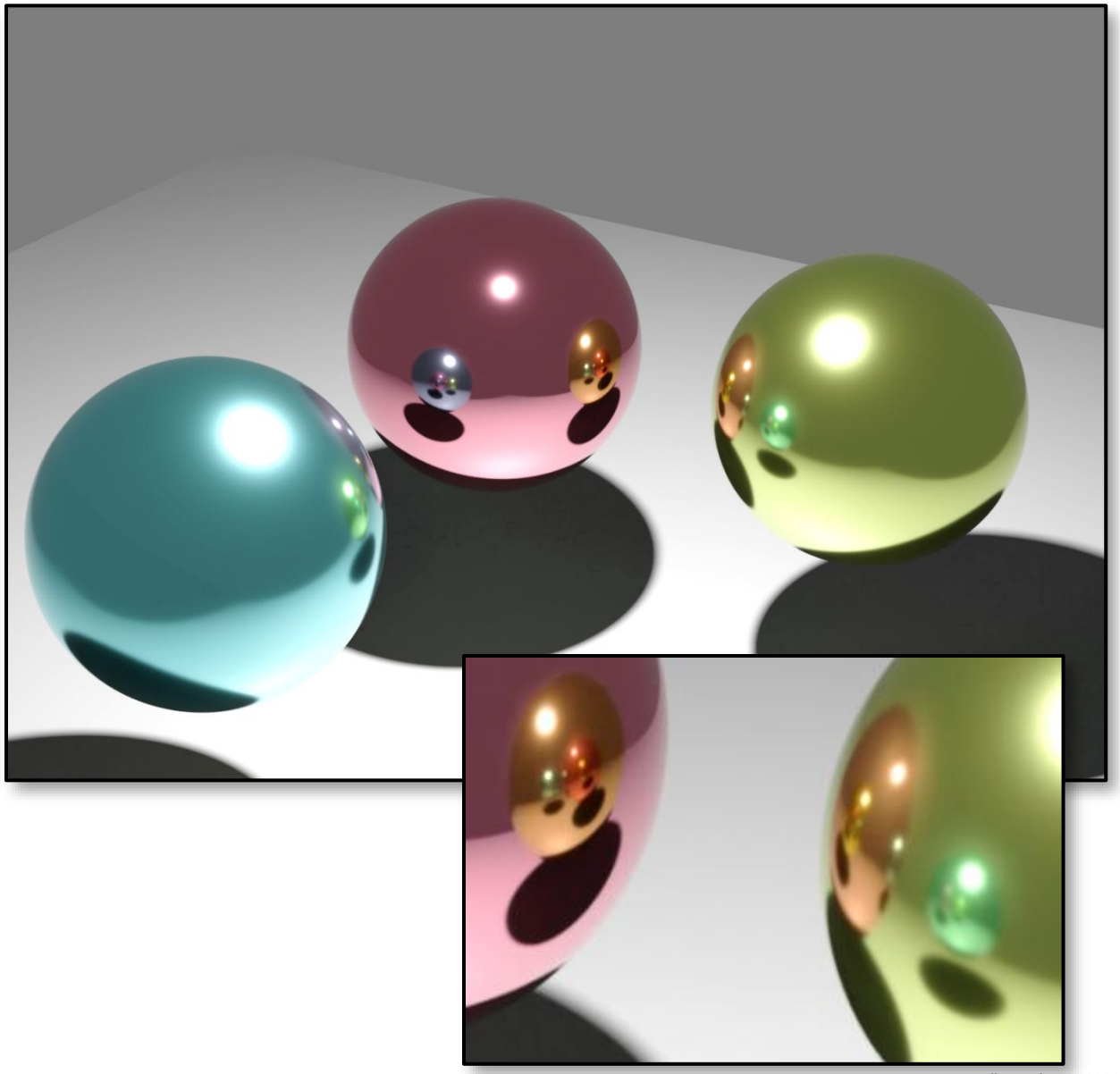
Lighting makes it possible to tell the difference between surfaces or parts of surfaces

Local vs. Global Illumination

Local Illumination



Global Illumination

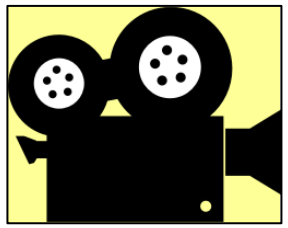


A Common type of *Local* Illumination: Ambient-Diffuse-Specular (ADS)

+

+

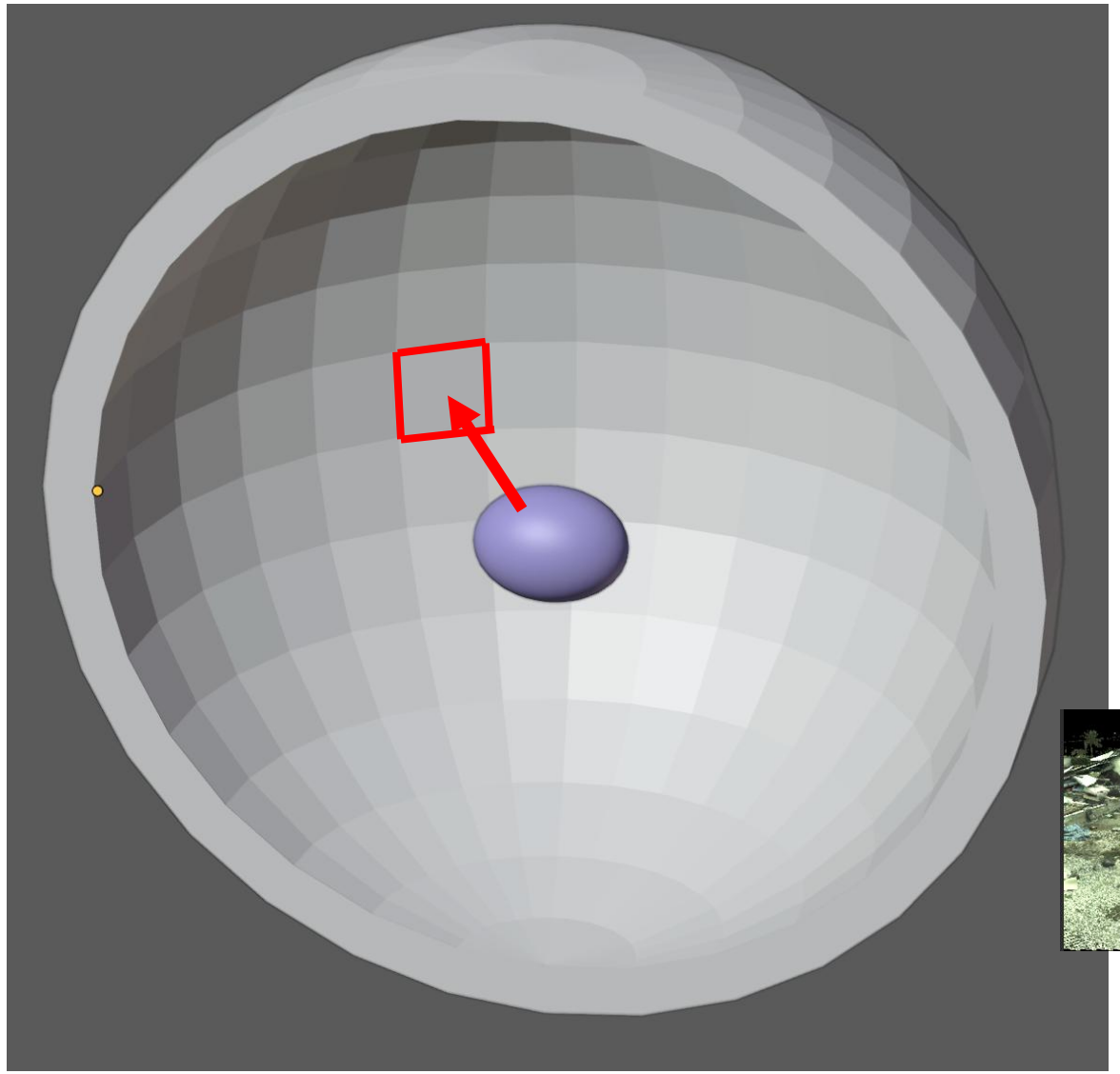
=



lighting.mp4

Putting them all together!

Spherical Rendering



Goal: turn a full 360º 3D scene into an image that can then be "wrapped" around the viewer

Strategy:

1. Surround the entire scene with an imaginary sphere.
2. Render the scene a tile at a time.
3. Combine the rendered tiles into a single texture image.
4. View the scene in VR goggles by re-mapping the texture image to a sphere.

Lidar Data from Mike Olsen, Oregon State University

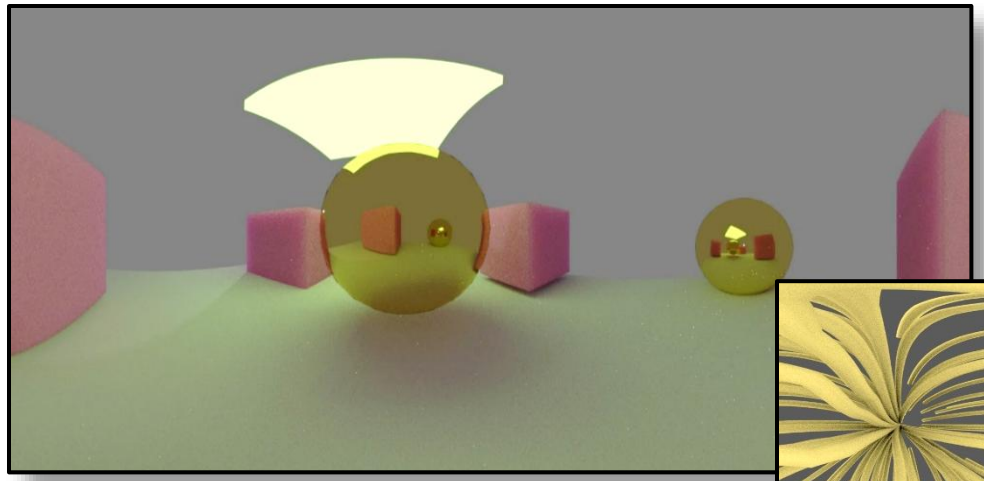
Spherical Rendering

Lidar Data from Mike Olsen, Oregon State University

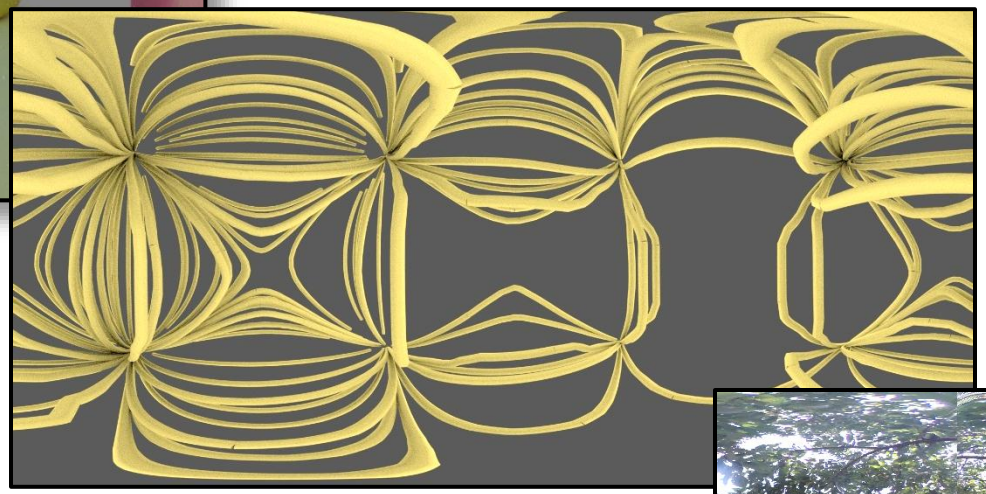
Strategy:

1. Surround the entire scene with an imaginary sphere.
2. Render the scene a tile at a time.
3. Combine the rendered tiles into a single texture image.
4. View the scene in VR goggles by re-mapping the texture image to a sphere.

Other Examples of Spherical Rendering



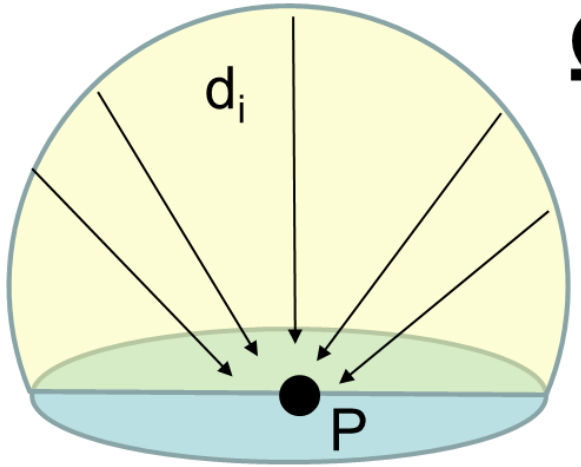
Blender Scene



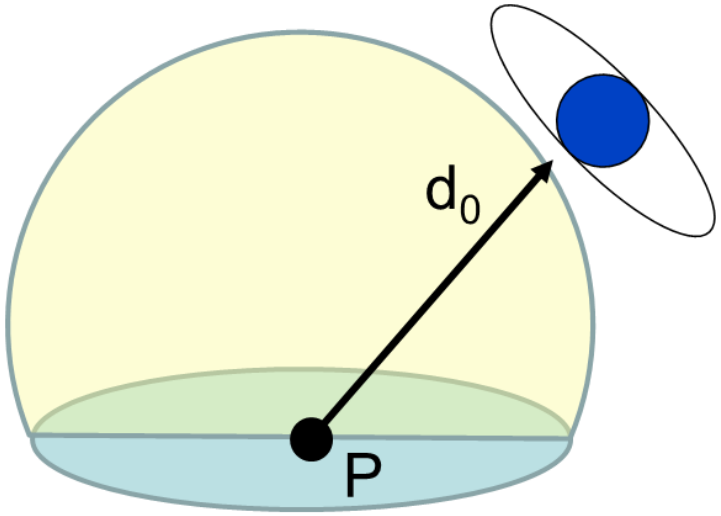
Fluid Flow

My Side Yard

Global Illumination: The Rendering Equation



Light arriving at Point P from everywhere



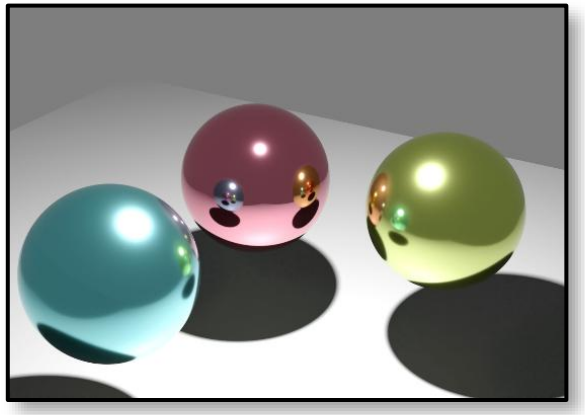
Light departing from Point P in the direction that we are viewing the scene from

Insert messy calculus equation here… ☺

In plain language, the messy calculus describes a “light balance”:

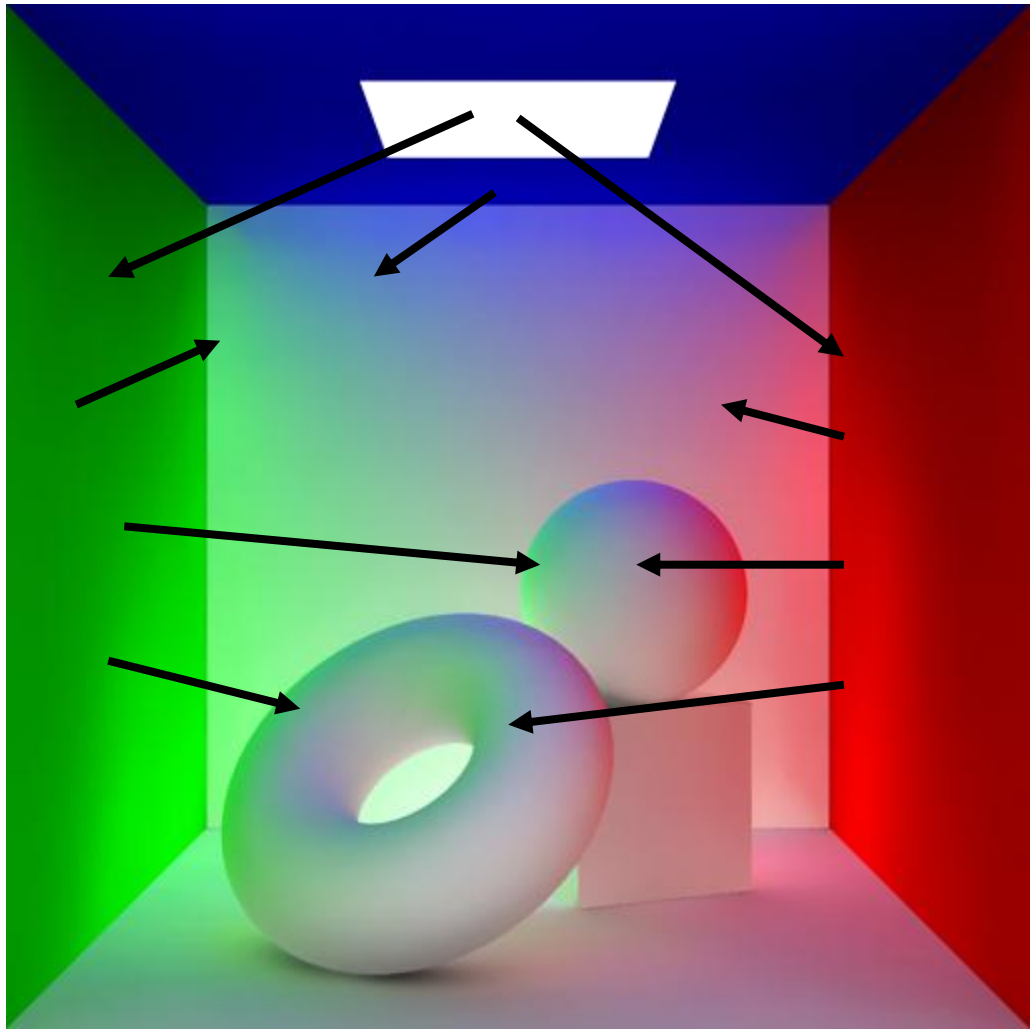
“The light shining from the point P towards your eye is the reflection of all the incoming light directed at the point P from all of the other objects in the scene.”

The Global Lighting Equation at Work



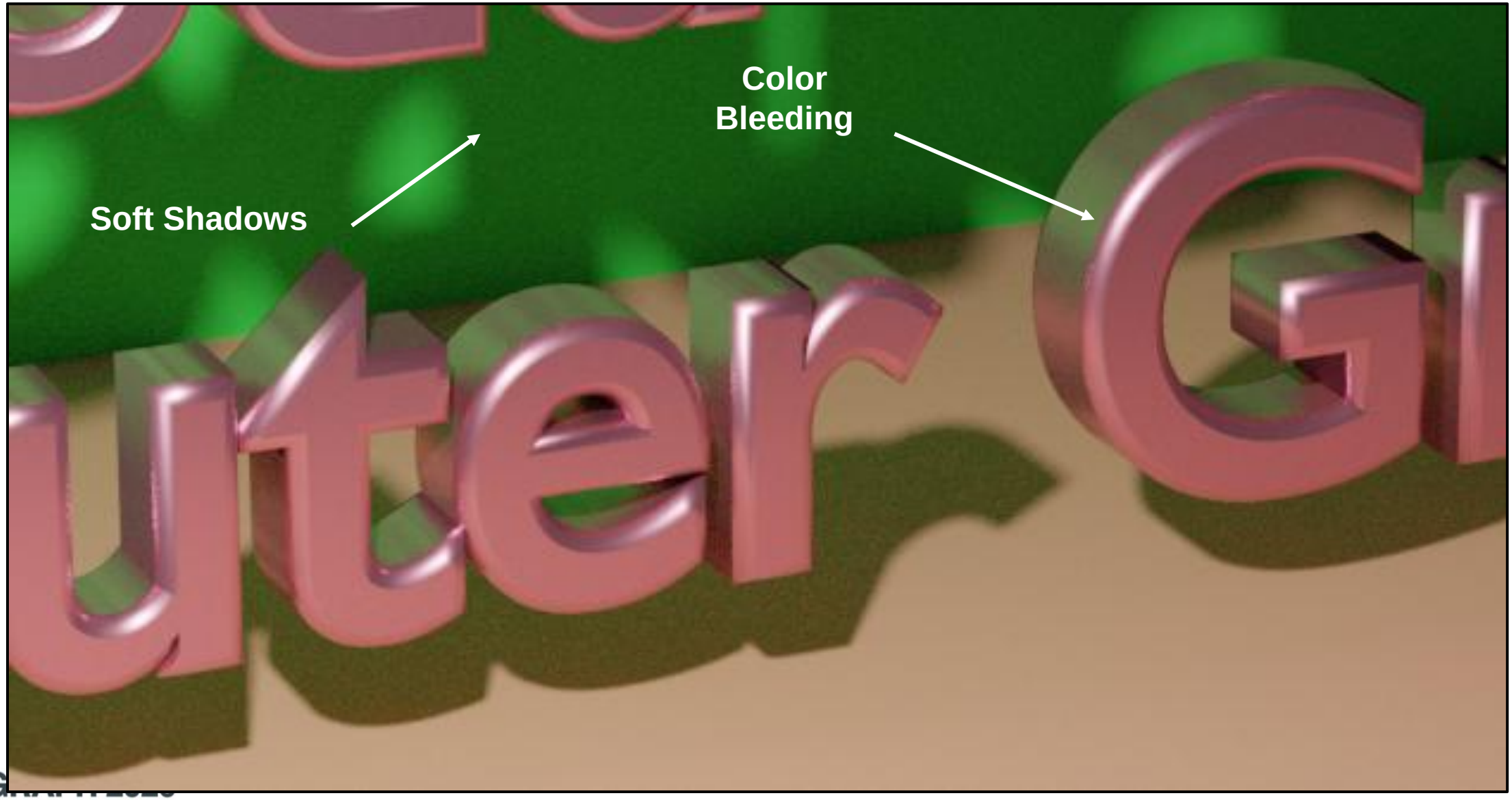
- The left wall is green.
- The right wall is red.
- The back wall is white.
- The ceiling is blue
- The ceiling has a light source in the middle of it.
- The objects sitting on the floor are white.

If the appearance of an object is affected by the appearances of other objects, then you are doing **Global Illumination.**



http://www.swardson.com/unm/tutorials/mentalRay3/

An Example from the Title Slide



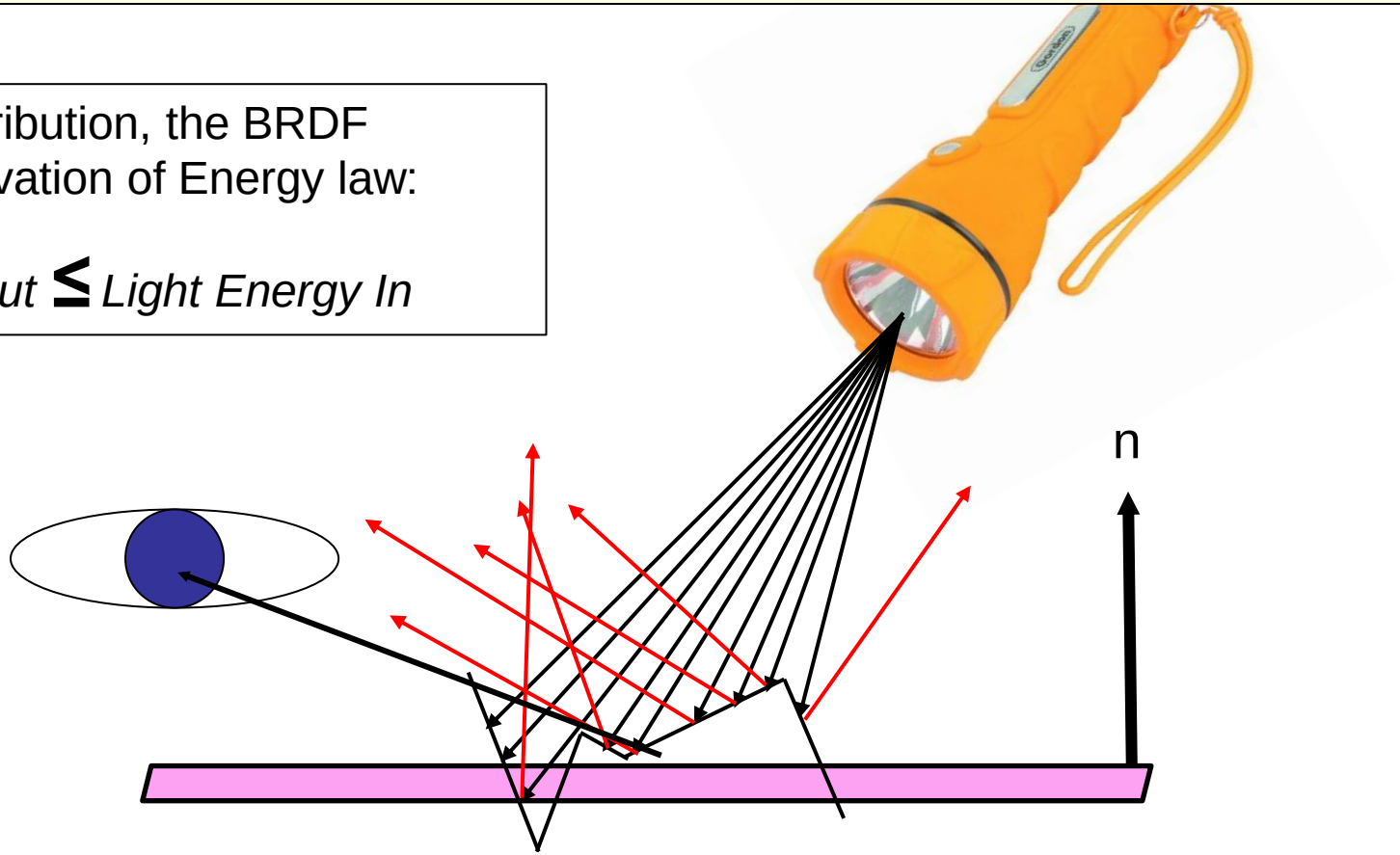
When light hits a surface, it bounces in particular ways depending on the angle and the material

This distribution of bounced light rays is called the *Bidirectional Reflectance Distribution Function*, or BRDF.

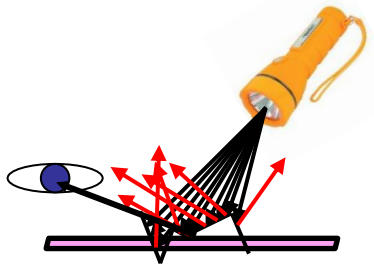
For a given material, the BRDF behavior of a light ray is a function of 4 variables: the 2 spherical coordinates of the incoming ray and the 2 spherical coordinates of the outgoing ray.

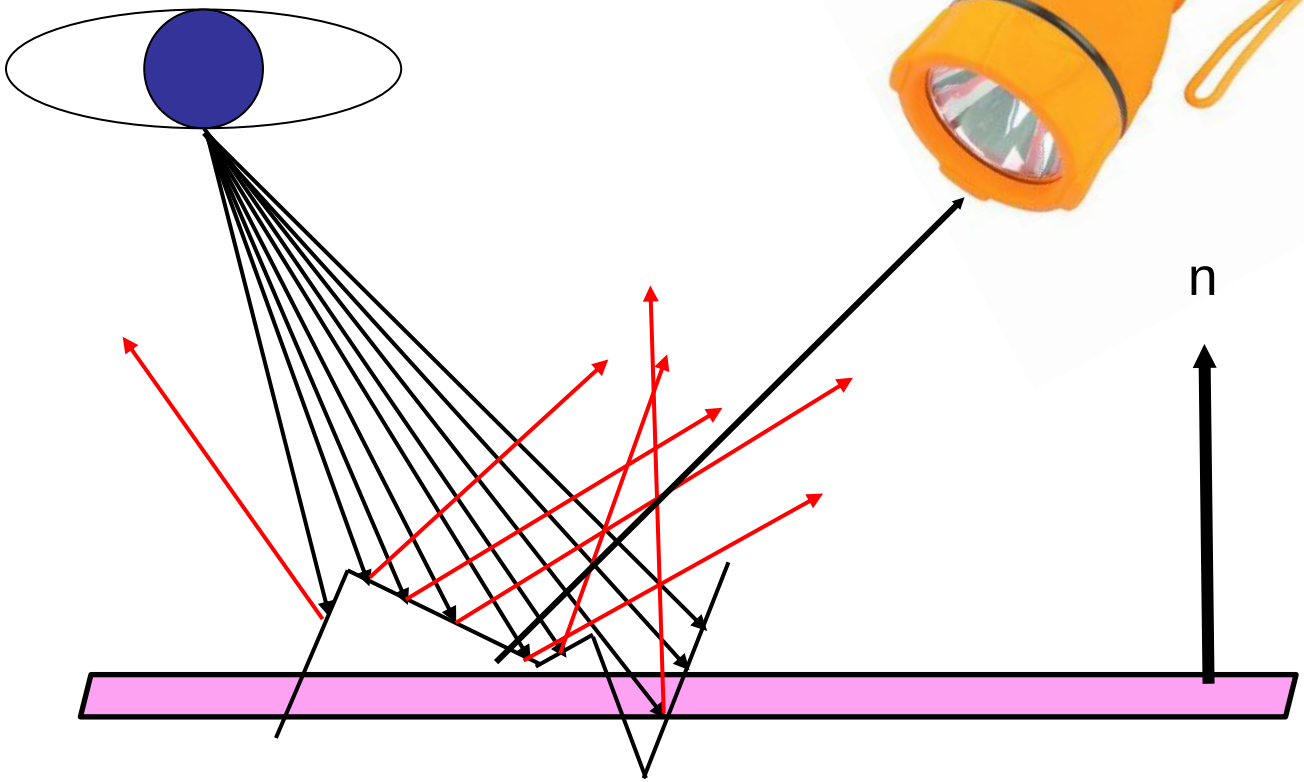
Besides being a distribution, the BRDF enforces the Conservation of Energy law:

Light Energy Out $\leq$ *Light Energy In*



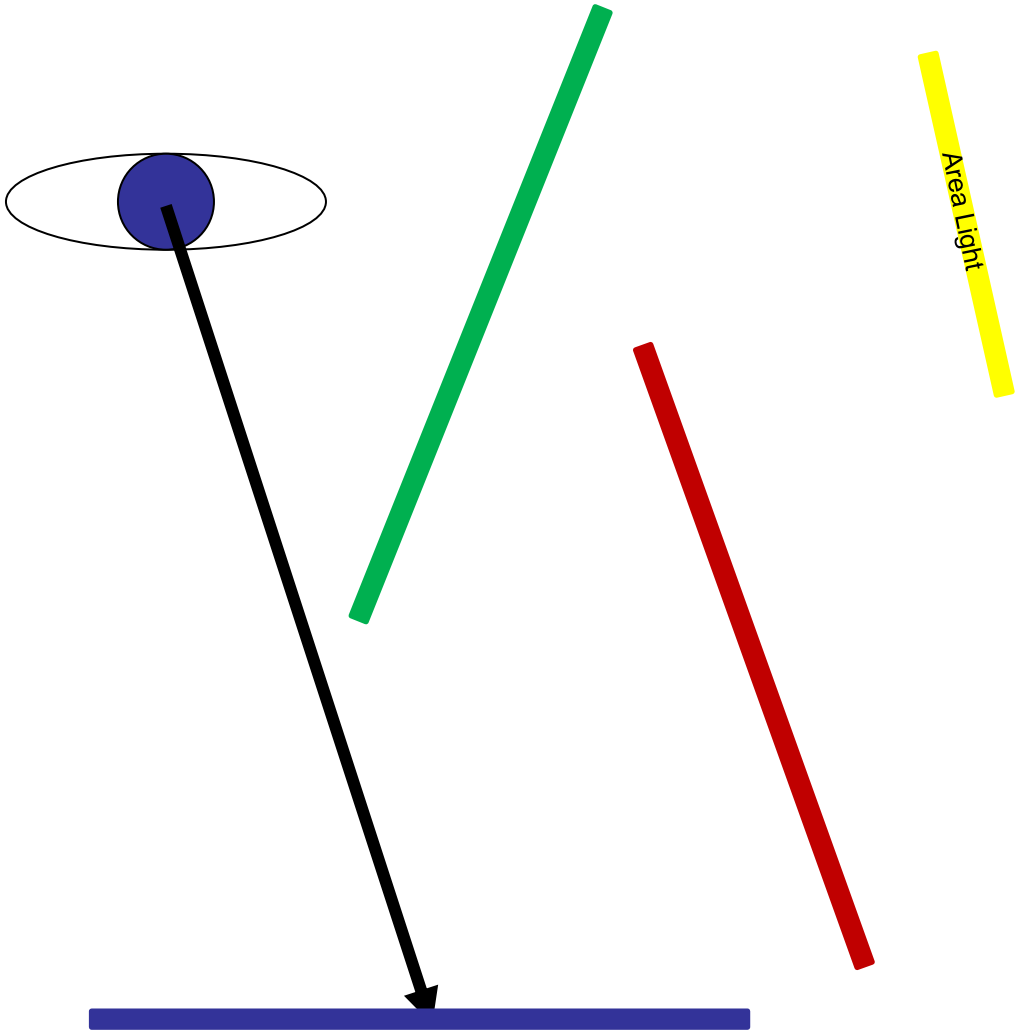
Usually, it is easier to trace from the eye





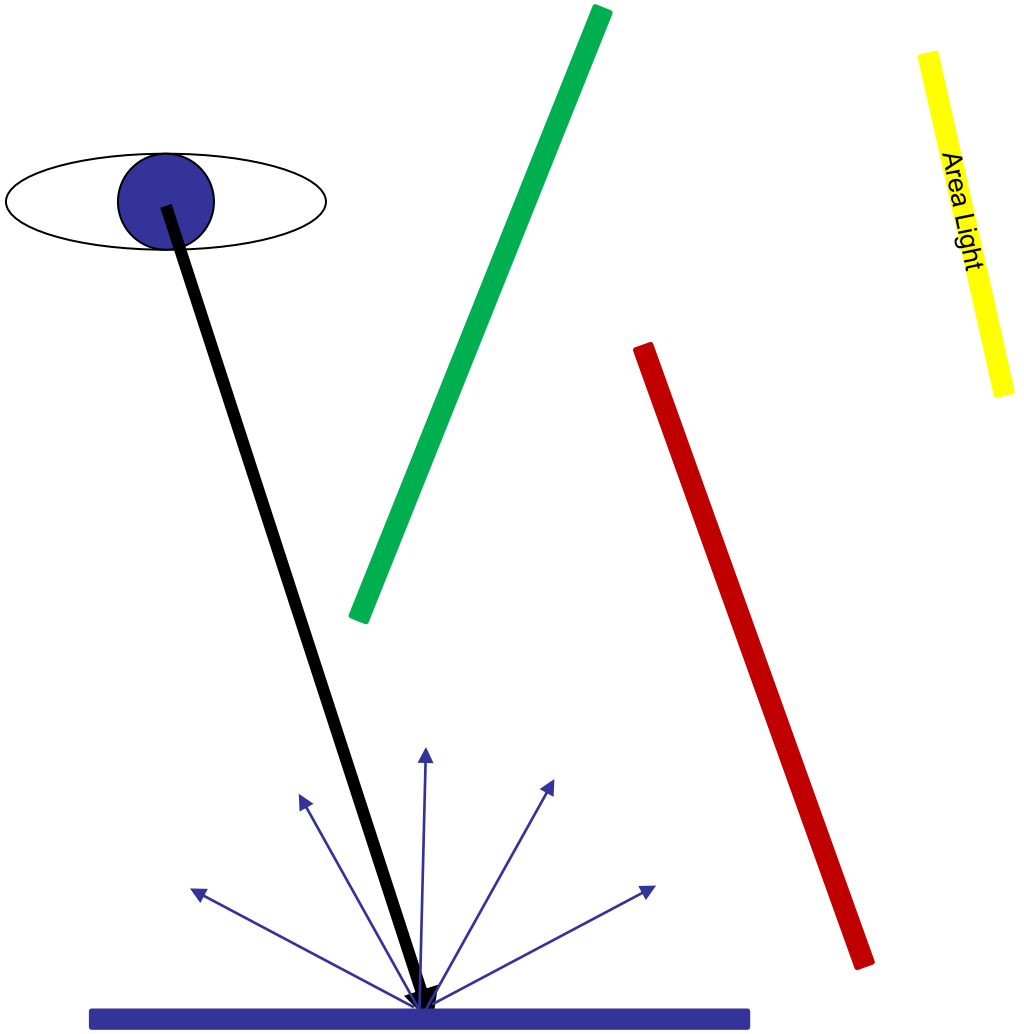
Physically-Based Rendering

Let light can bounce around the scene, depending on how the different materials behave.

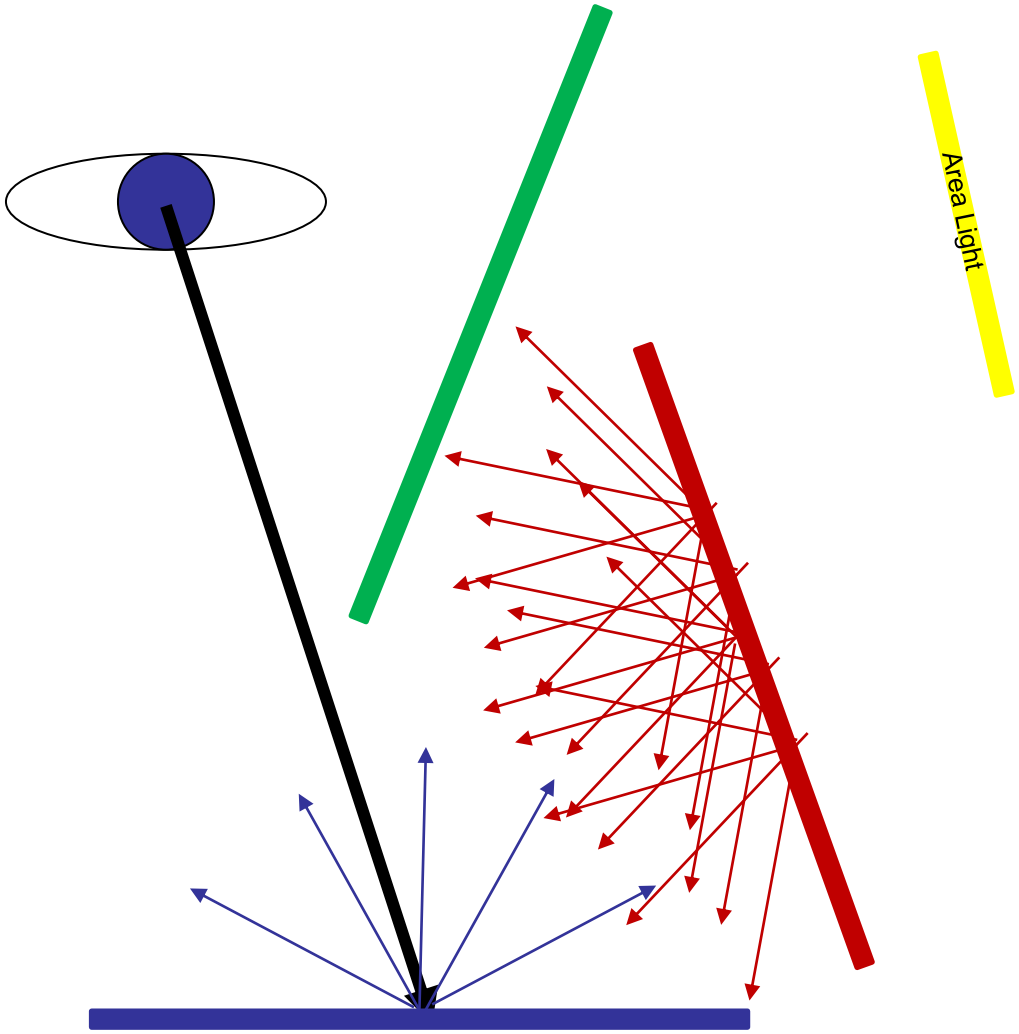


Physically-Based Rendering

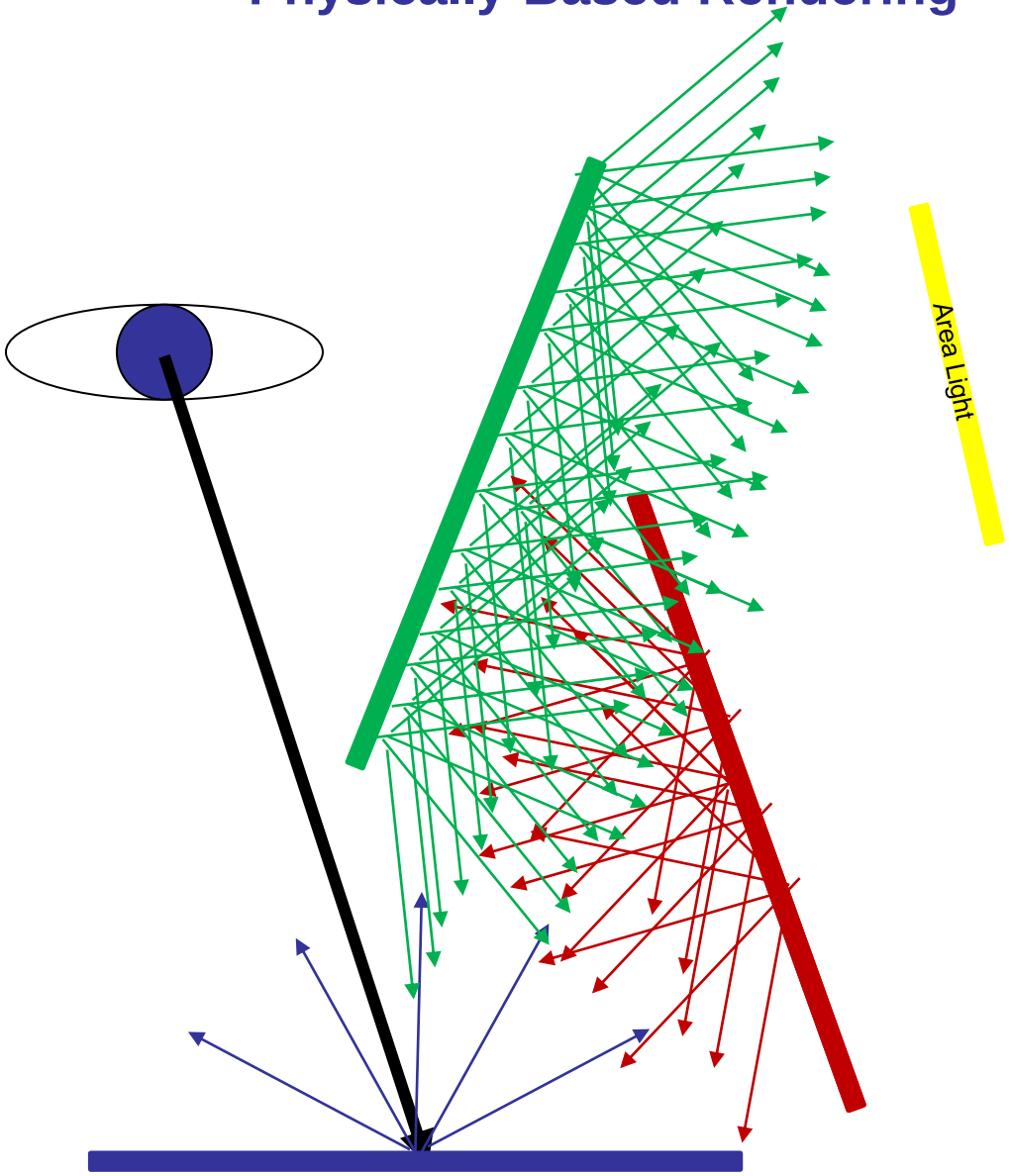
Let light can bounce around the scene, depending on how the different materials behave.



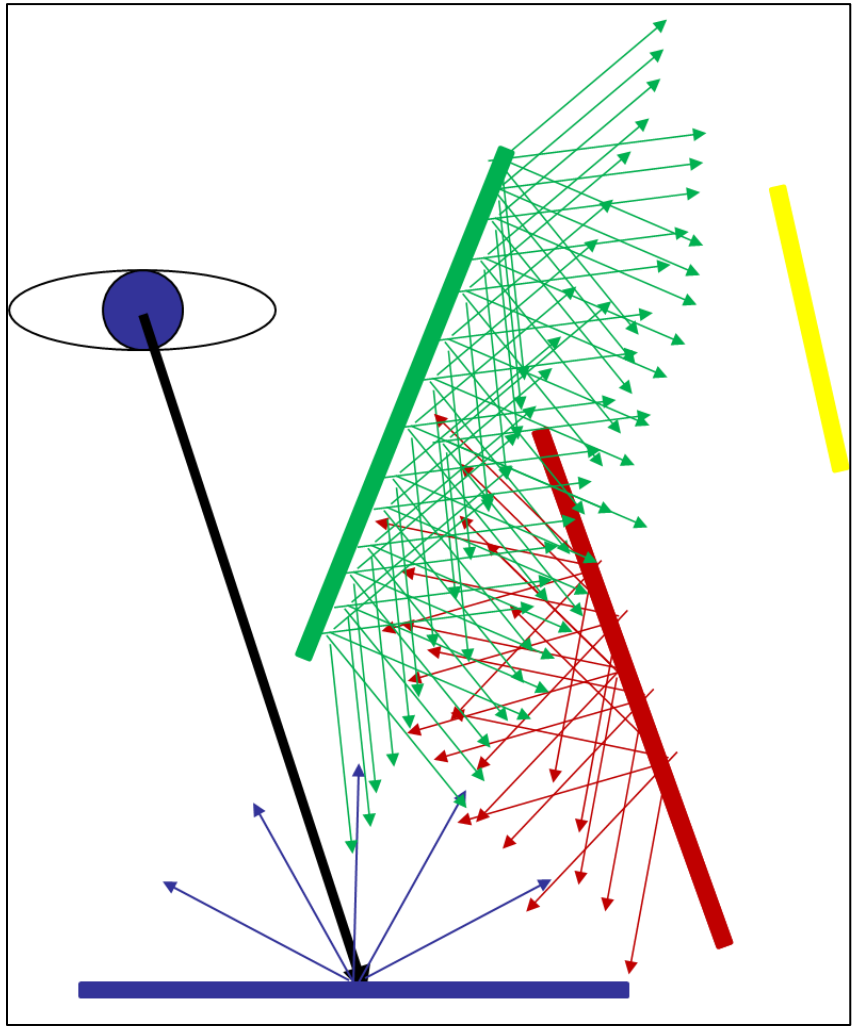
Physically-Based Rendering



Physically-Based Rendering



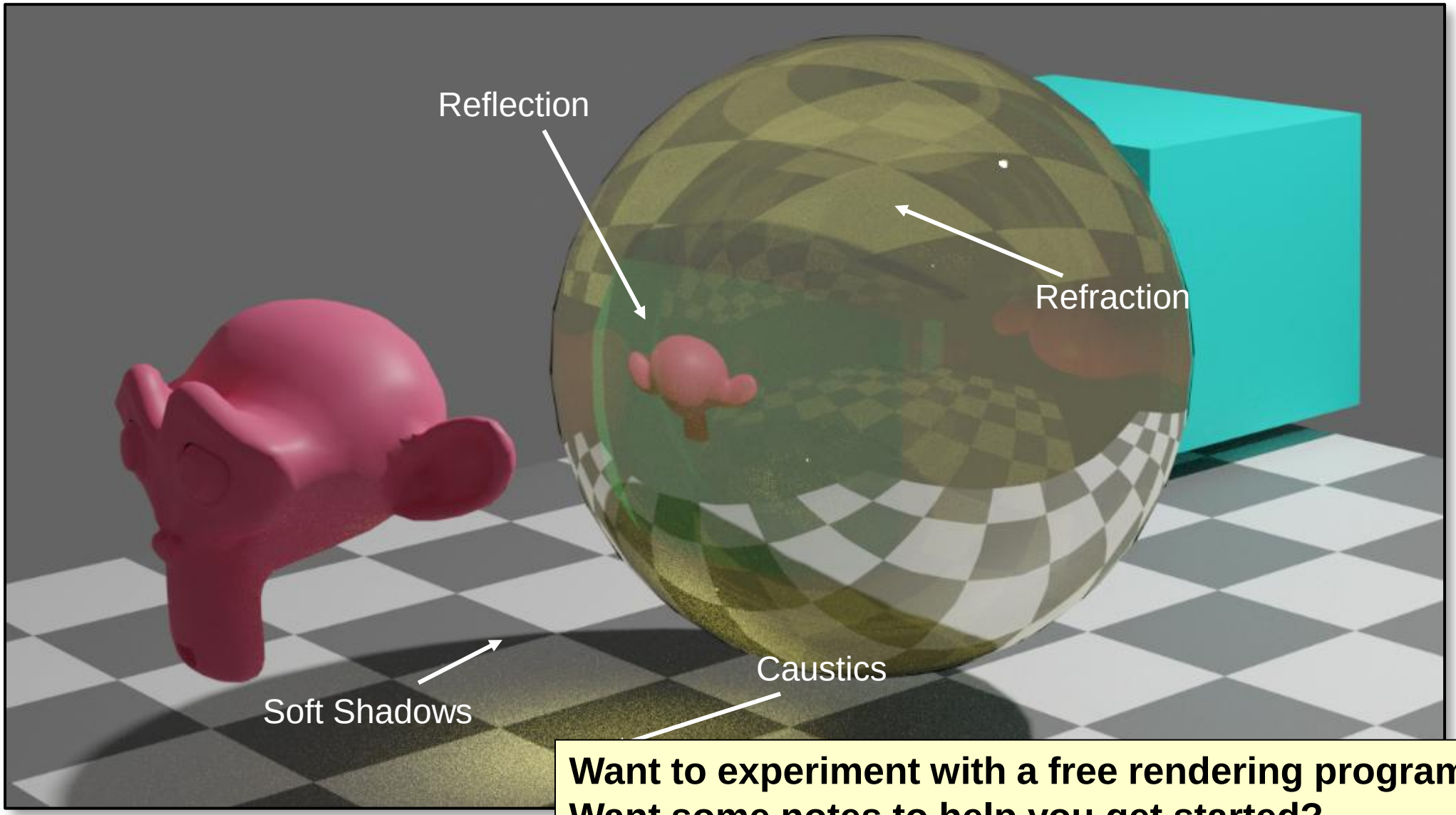
Physically-Based Rendering



Clearly this process can spawn an infinite number of rays. How do we handle this?

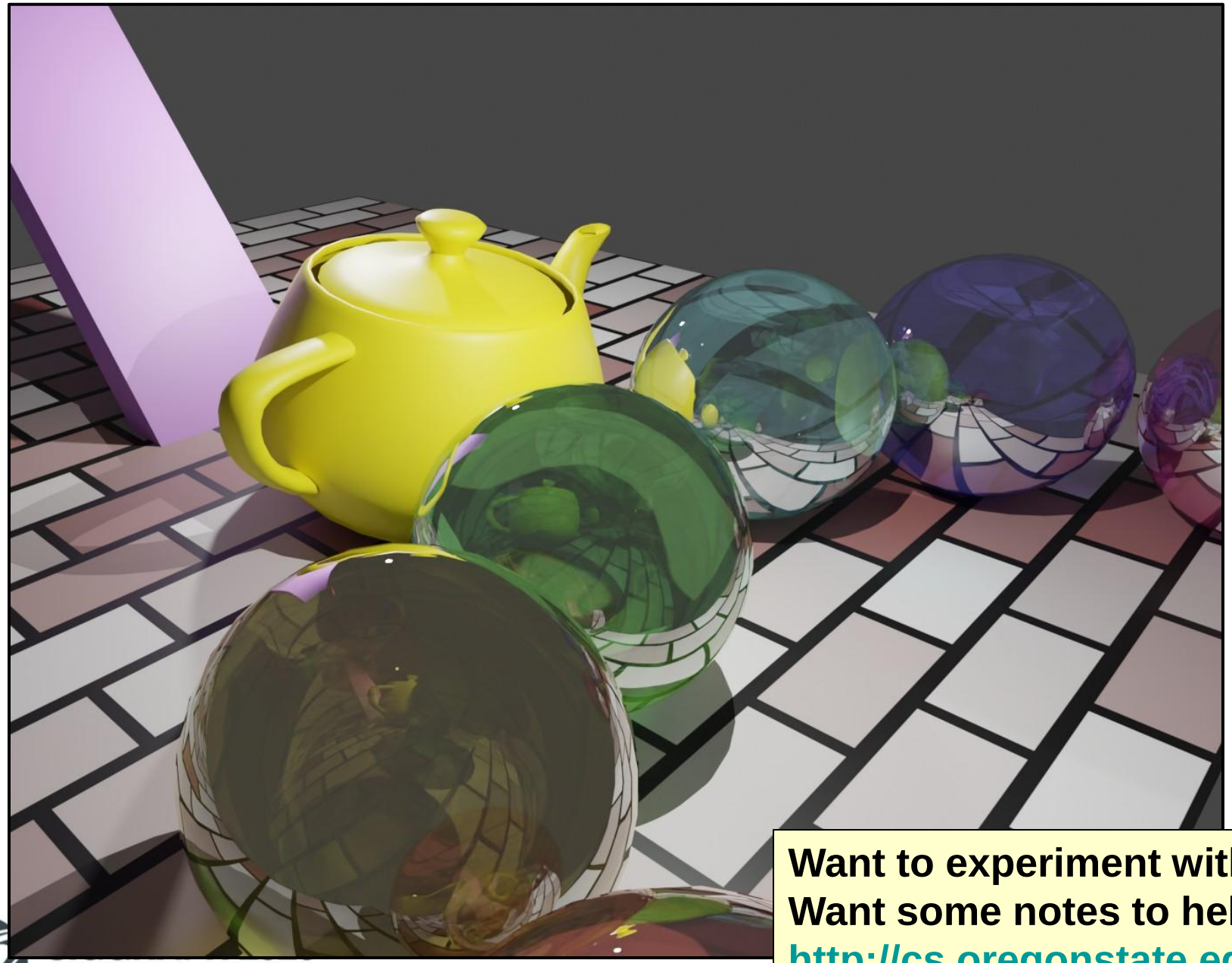
We can't use all rays that are possible, so we use a statistical subset of the rays.

This is known as **Monte Carlo** simulation.



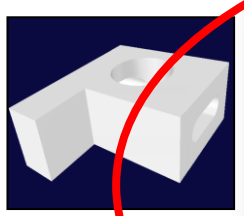
Want to experiment with a free rendering program?
Want some notes to help you get started?
http://cs.oregonstate.edu/~mjb/blender

More from the Blender *Cycles* Renderer



Want to experiment with a free rendering program?
Want some notes to help you get started?
http://cs.oregonstate.edu/~mjb/blender

Summary



3D Geometric Models

Surface Information

Texture Information

Modeling

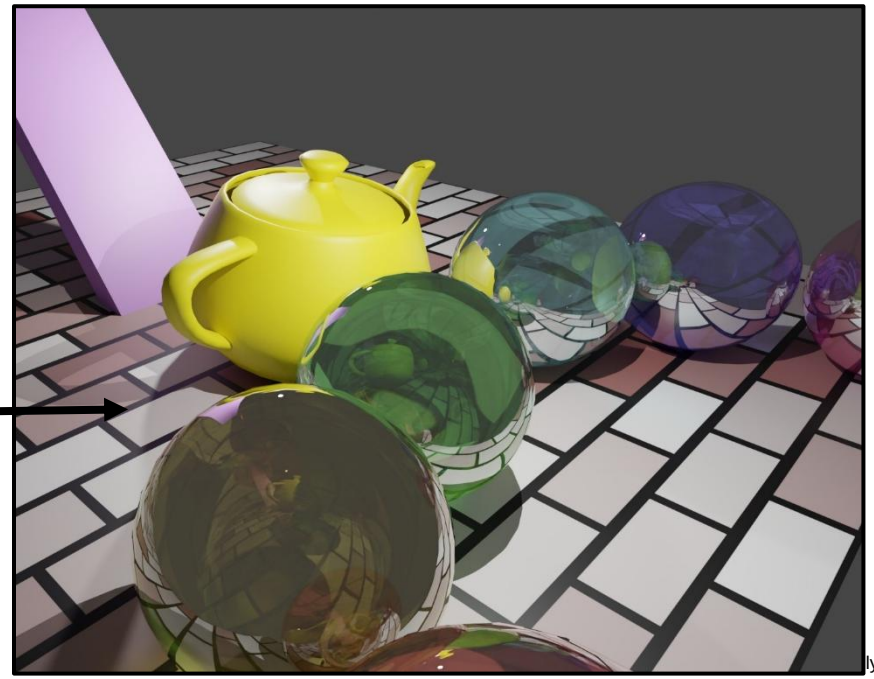
Lighting Information

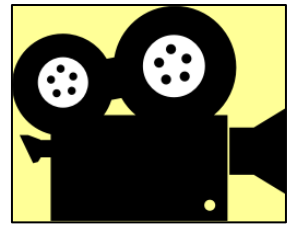
Rendering

Drawing

3D Movement Definition

Animation





SixSpheres.mp4

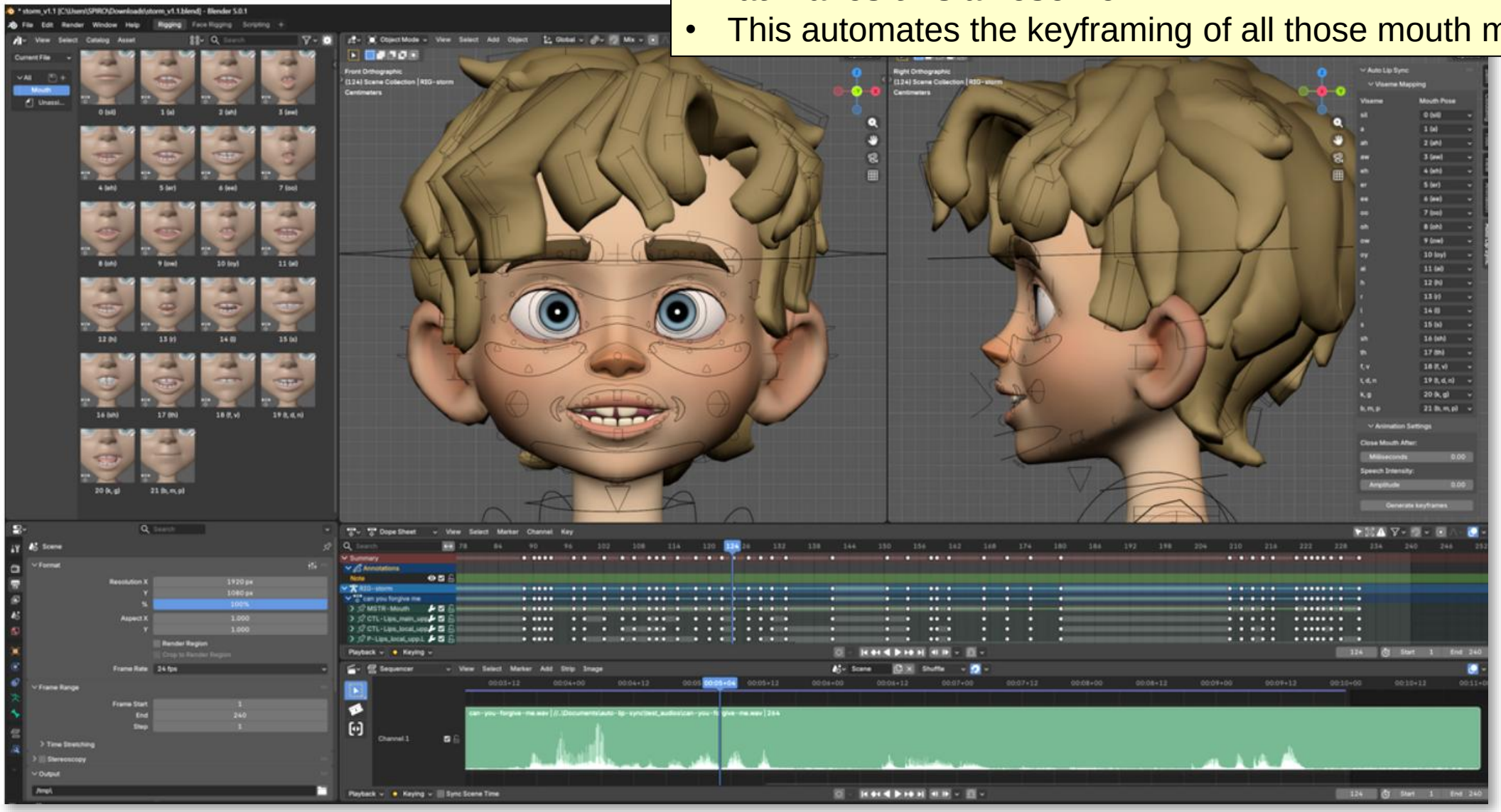
. . . and what it is that makes them awesome!

Cool Renderings: Blender Add-On to turn Audio into Lip Movements

Annette Tongsak, used by permission

What makes this awesome:

- This automates the keyframing of all those mouth movements

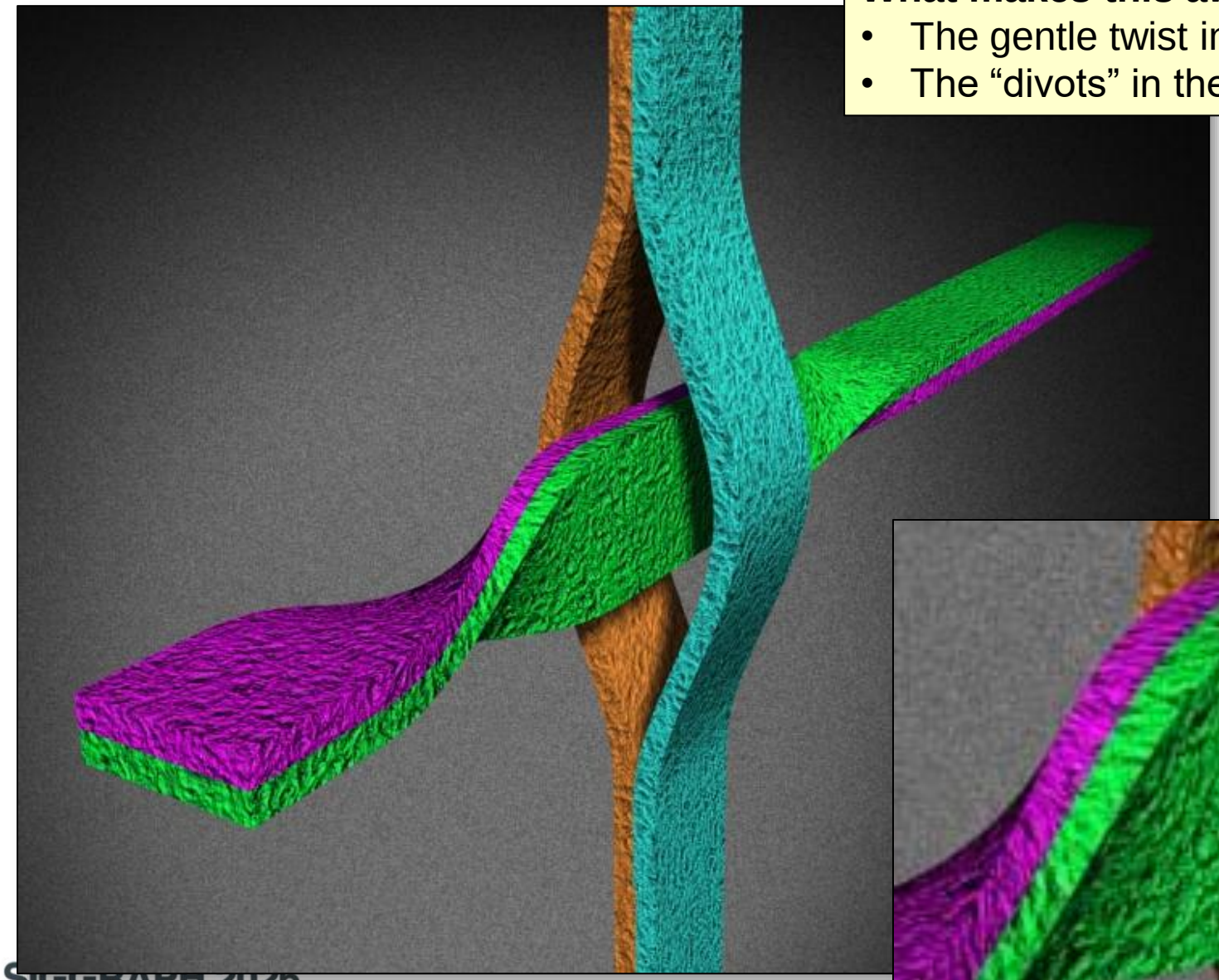


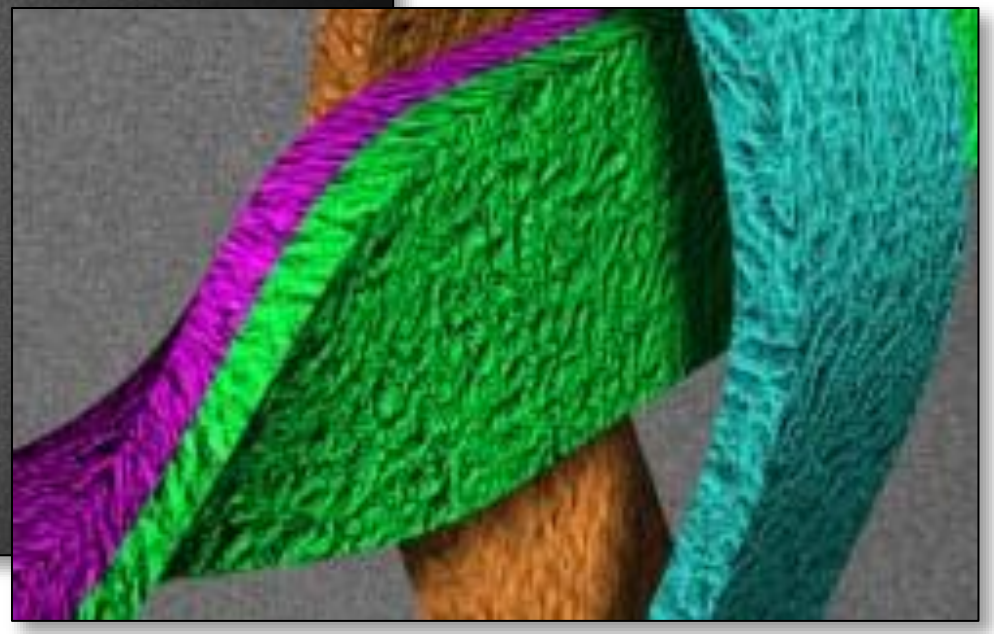
Cool Renderings: Bumpy Cloth (Celtics Knots)

Andrew Glassner, used by permission

What makes this awesome:

- The gentle twist in the cloth
- The “divots” in the cloth and how light interacts with them



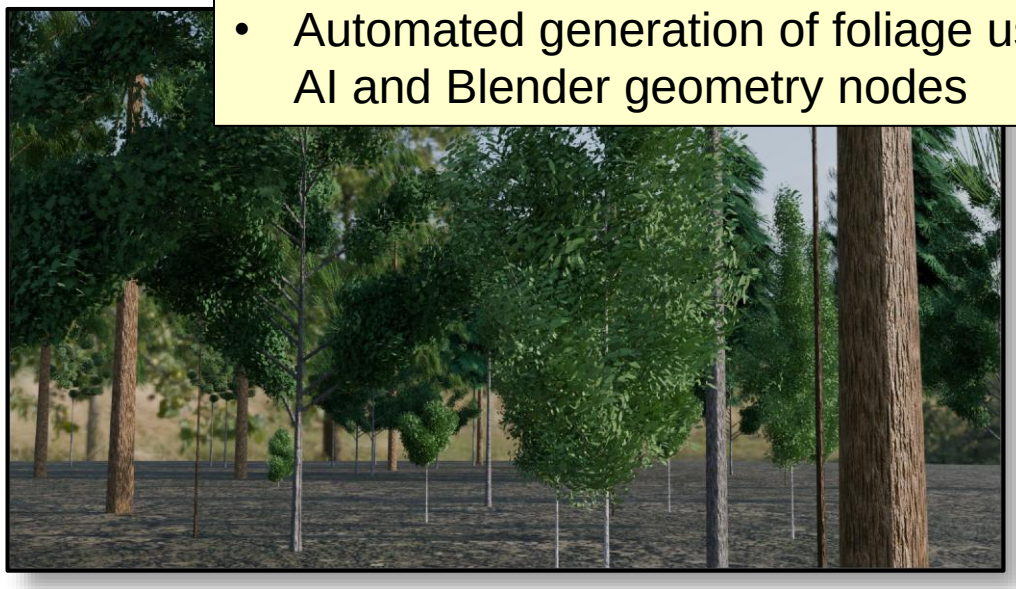


Cool Renderings: Foliage Rendering

Grace Todd, used by permission

What makes this awesome:

- Automated generation of foliage using generative AI and Blender geometry nodes

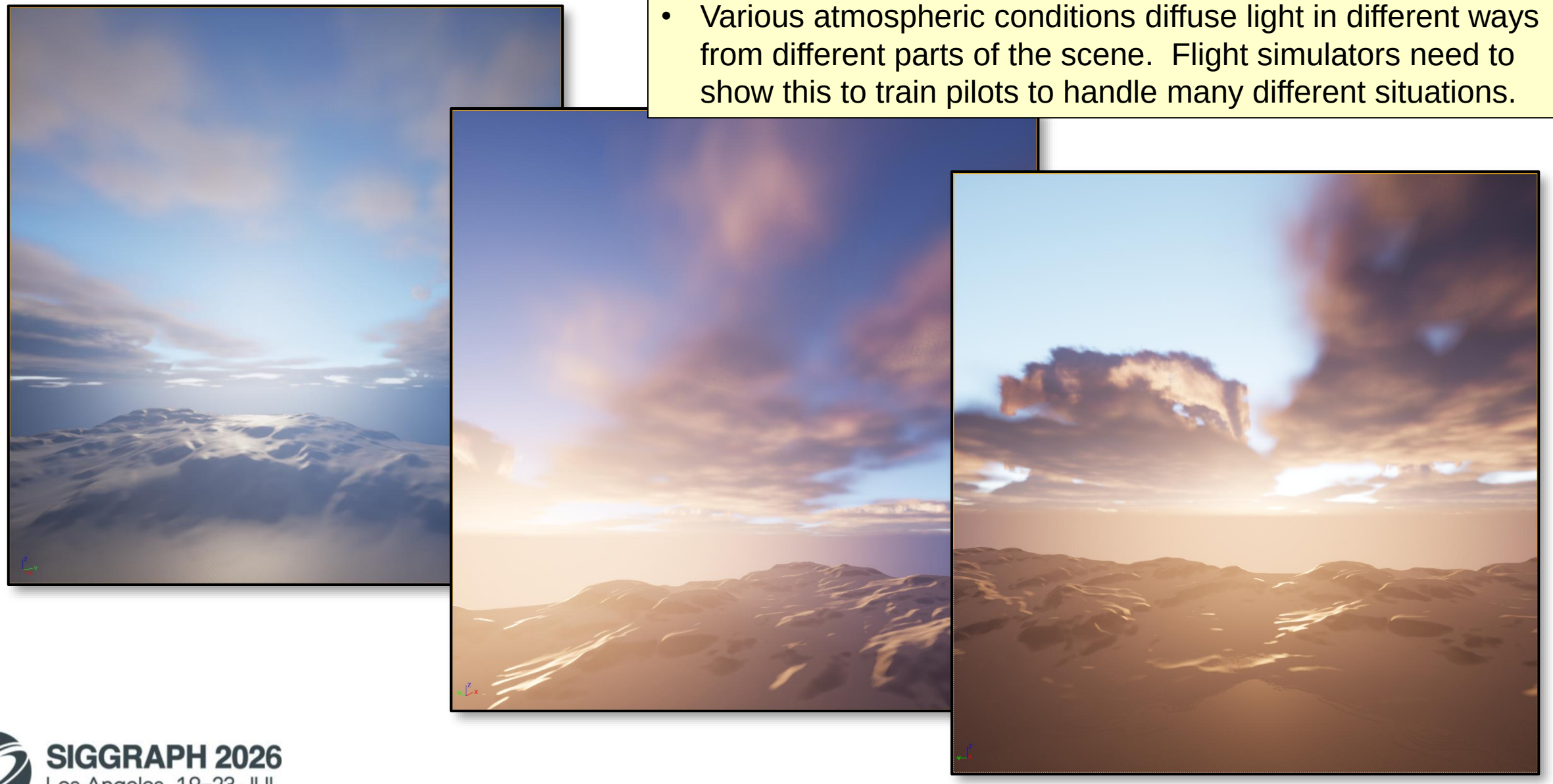


Cool Renderings: Atmospherics

Nathan DeStafeno, used by permission

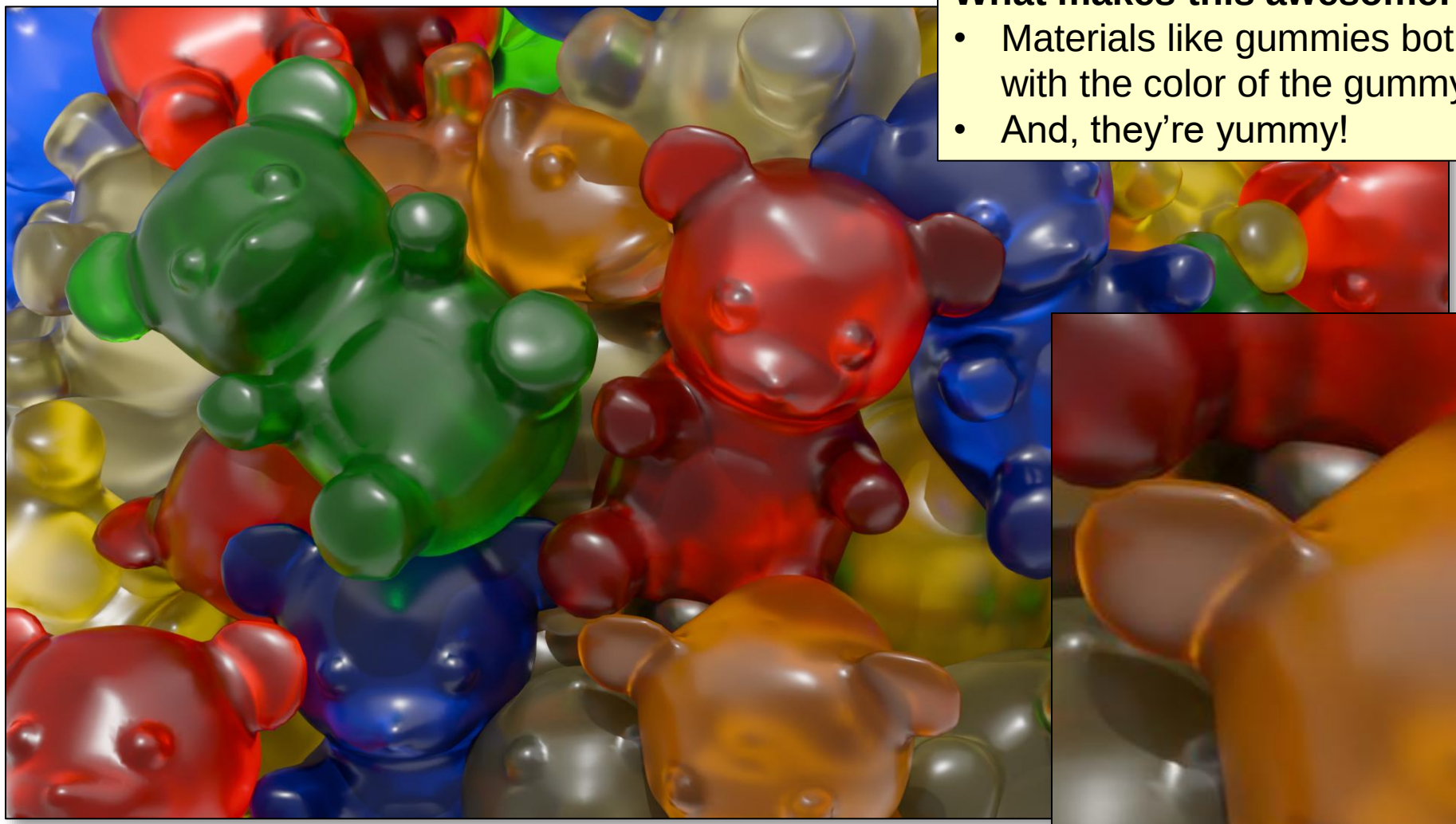
What makes this awesome:

- Various atmospheric conditions diffuse light in different ways from different parts of the scene. Flight simulators need to show this to train pilots to handle many different situations.



Cool Renderings: Mmmmm, Gummies!

Grace Todd, used by permission



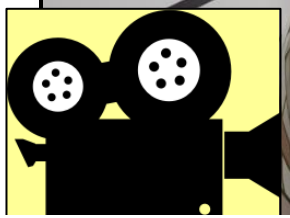
What makes this awesome:

- Materials like gummies both *reflect* and *refract* light, with the color of the gummy acting like a color filter
- And, they're yummy!

Cool Renderings: Hardware Ray-Tracing

What makes this awesome:

- The speed with which today's hardware can render ray-traced images



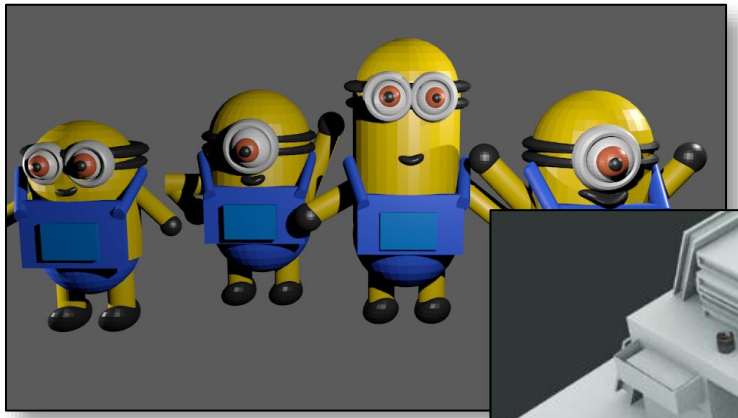
SixSpheres.mp4

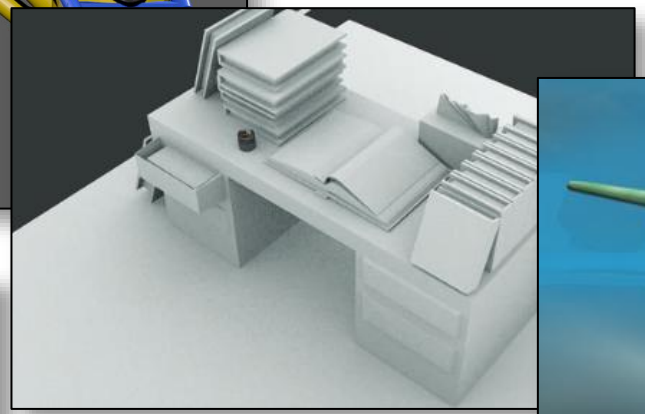
1920x1080 pixels rendered in ***2.68*** seconds using Blender on an Nvidia RTX A6000

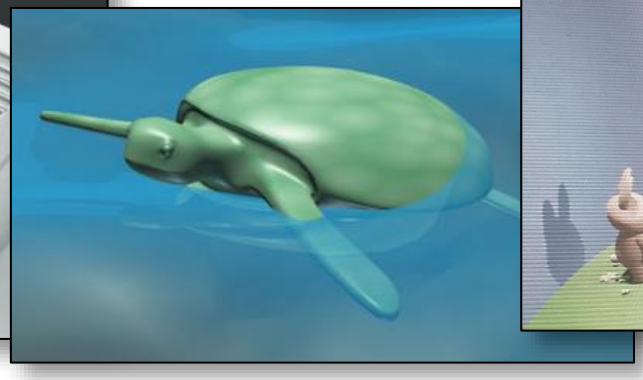
Cool Renderings: Grades 3-12 using TinkerCad and Blender

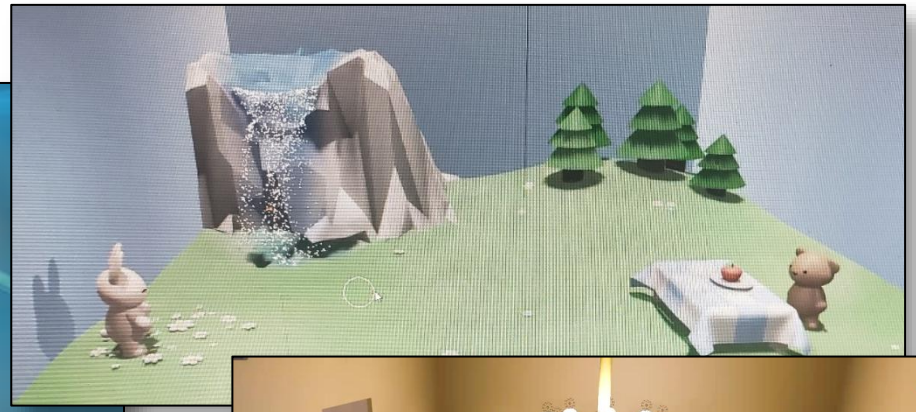
What makes this awesome:

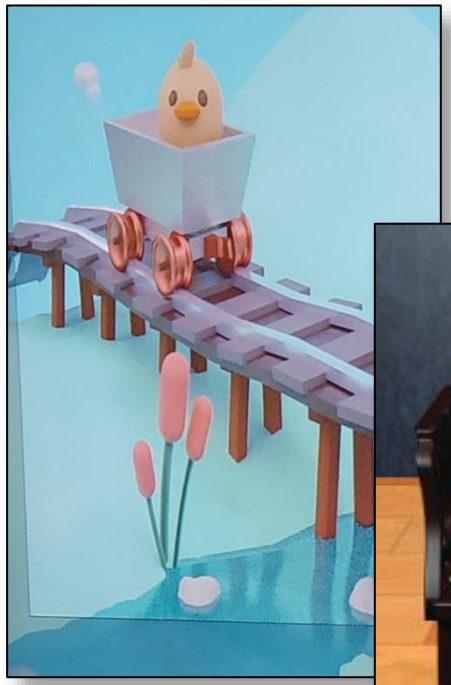
- With the right introduction, kids can become impressively good at this sort of thing. They will soon be at SIGGRAPH to kick our butts. ☺

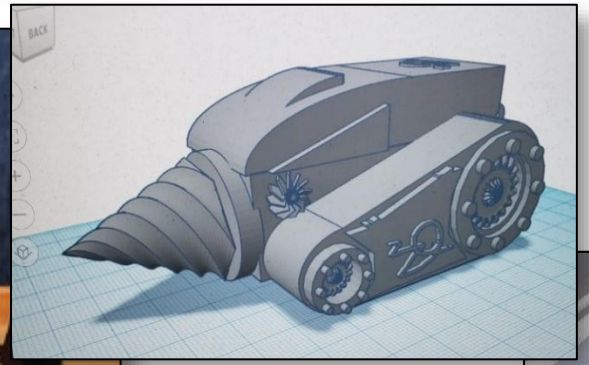












Some Advice

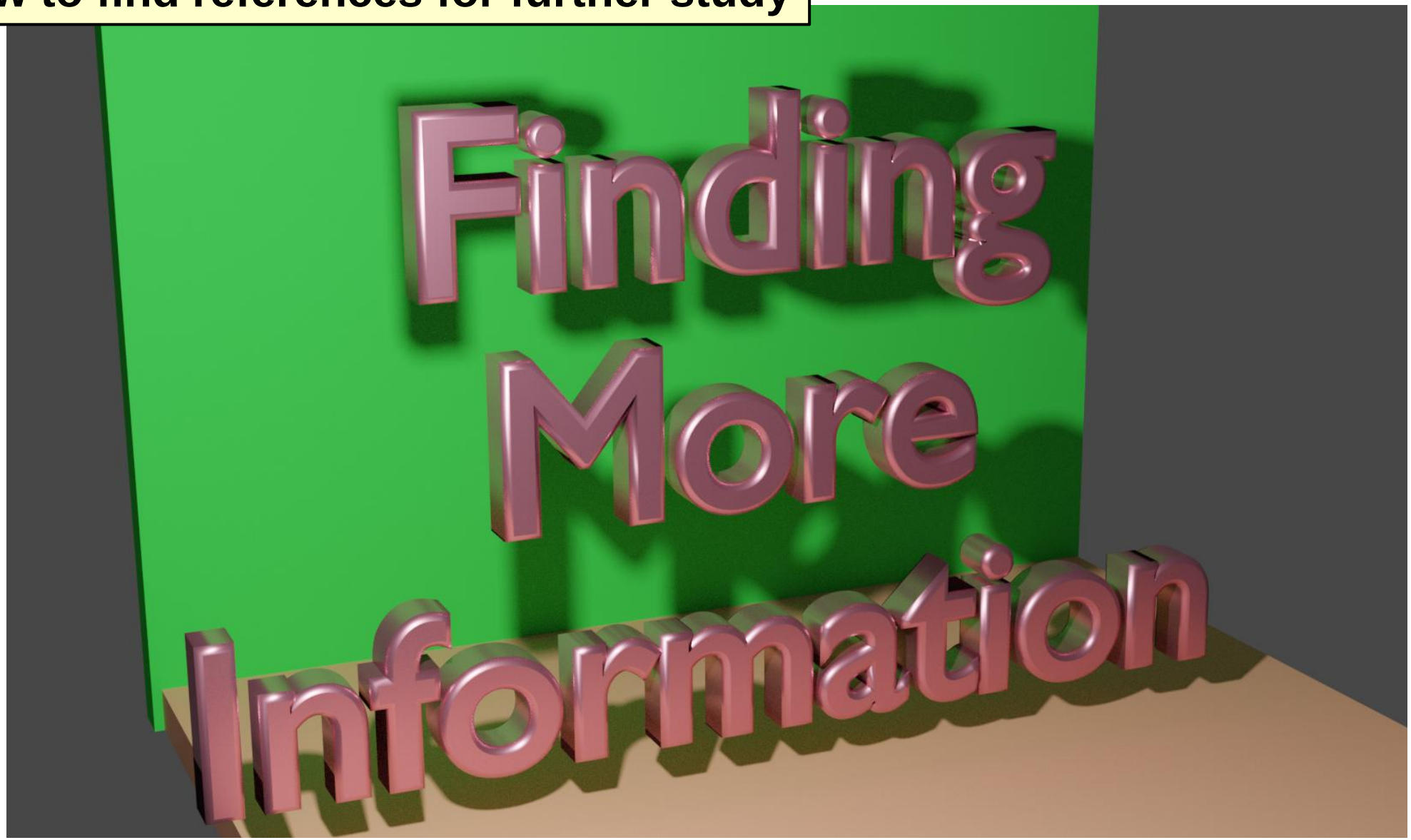
- SIGGRAPH moments will never come again. Well, this is usually true, but through the magic of the 2026 videos, they might reappear. *But be aware of what is going to be recorded and archived and what isn't.* And, if it is to be archived, how long will you have access to it? And where is it going to be?

- Especially take advantage of the not-to-be-archived or not-to-be-archived-for-very-long events because you cannot re-live them forever.

- Combine what you have just learned here with what else you learn this week at the conference and *relate them to your career and life goals*.

- Have fun doing it!

- **How to find references for further study**



http://cs.oregonstate.edu/~mjb/whirlwind

Check Out the *More Information* Document to be found at:

Where to Find More Information about Computer Graphics and Related Topics

Mike Bailey
Oregon State University

1. References

1.1 General Computer Graphics

SIGGRAPH Online Bibliography Database:
http://www.siggraph.org/learn/computer-graphics-bibliography-database

Edward Angel and Dave Shreiner, *Interactive Computer Graphics: A Top-down Approach with OpenGL*, 6th Edition, Addison-Wesley, 2011.

Francis Hill and Stephen Kelley, *Computer Graphics Using OpenGL*, 3rd Edition, Prentice Hall, 2006.

Steve Cunningham, *Computer Graphics: Programming in OpenGL for Visual Communication*, Prentice-Hall, 2007

Alan Watt, *3D*

Peter Shirley, *Fundamentals* of Computer Graphics, 2nd Edition, AK Peters, 2005.

Andrew Glassner, *Graphics Gems*, Academic Press, 1990.

http://cs.oregonstate.edu/~mjb/whirlwind

Check Out Other Sets of Free Notes at:

http://cs.oregonstate.edu/~mjb/cgeducation

University course notes:

- Introduction to Computer Graphics
- Computer Graphics Shaders
- CS Skills for Simulation and Game Programming
- Parallel Programming
- Scientific Visualization
- Vulkan

SIGGRAPH course notes:

- A Whirlwind Introduction to Computer Graphics

K-12 notes:

- Blender
- CodeBlocks
- Processing
- Scratch
- Scratch Jr.
- SimLab
- Tinkercad

These notes are all licensed under a **Creative Commons Attribution-NonCommercial-NoDerivatives 4.0** International License.

And Finally…

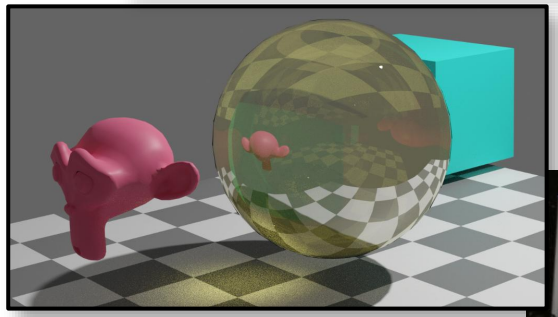
There is no merit badge for completing this Whirlwind course, but if there was, I would want it to look like this:*

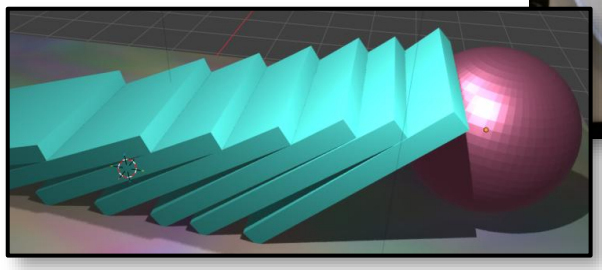
☺

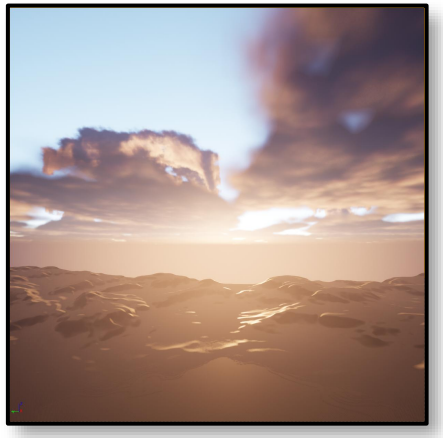
*Thanks to Microsoft CoPilot for generating this image and thanks to my good friend Andrew Glassner for suggesting the prompts to do it.

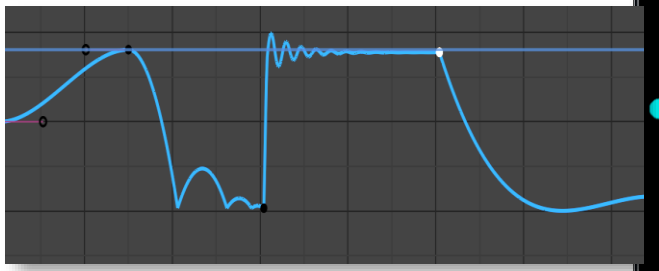
Thank You!

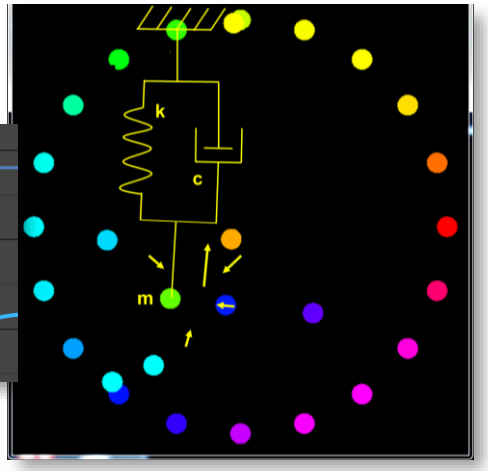
Mike Bailey
mjb@cs.oregonstate.edu

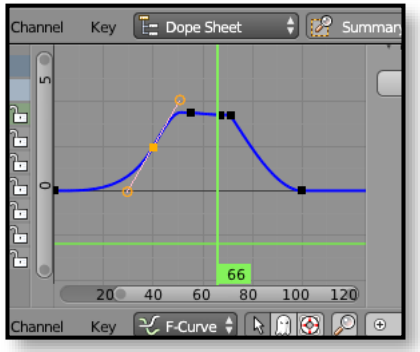


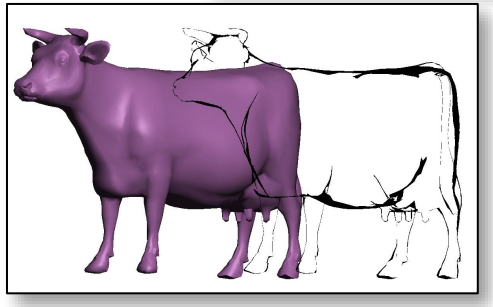












http://cs.oregonstate.edu/~mjb/whirlwind

- **(lsystems.exe)**
- **Curves.exe**
- **Mesh in Blender**
- **Deform in Blender**
- **Keyframe animation in Blender**
- **Particles.blend**
- **Dominos.blend**
- **FluidSimulation.blend**
- **chain.exe**
- **Avoid.exe**
- **global.blend**
- **reflrefrr6.blend**