Lecture 1:

Course Introduction:
Welcome to Computer Graphics!

Interactive Computer Graphics
Stanford CS248, Winter 2020

Tunes

Gift of Gab
“Dreamin”
(Escape to Mars)

“Think of all the things you will be able to create with a bit of graphics knowledge.”
- Timothy Jerome Parker

Hi!

Kayvon Fatahalian

David
Yao

Yinchen
Xu

Wen
Zhou

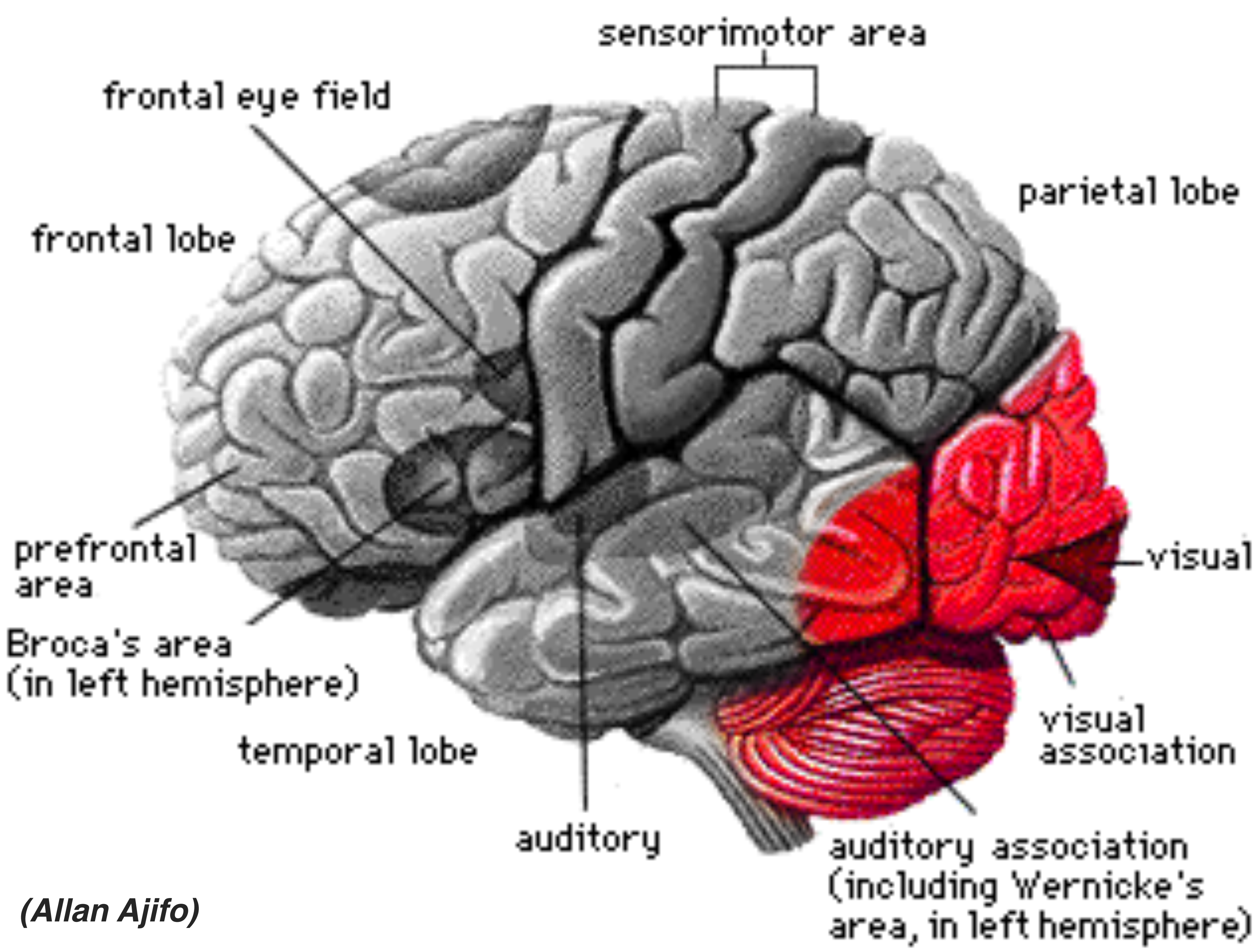
Discussion:
Why study computer graphics?

What is computer graphics?

com•put•er graph•ics /kəmˈpyo͞odər ˈgrafiks/ *n.*
The use of computers to synthesize and manipulate visual information.

Why *visual* information?

About 30% of brain dedicated to visual processing...

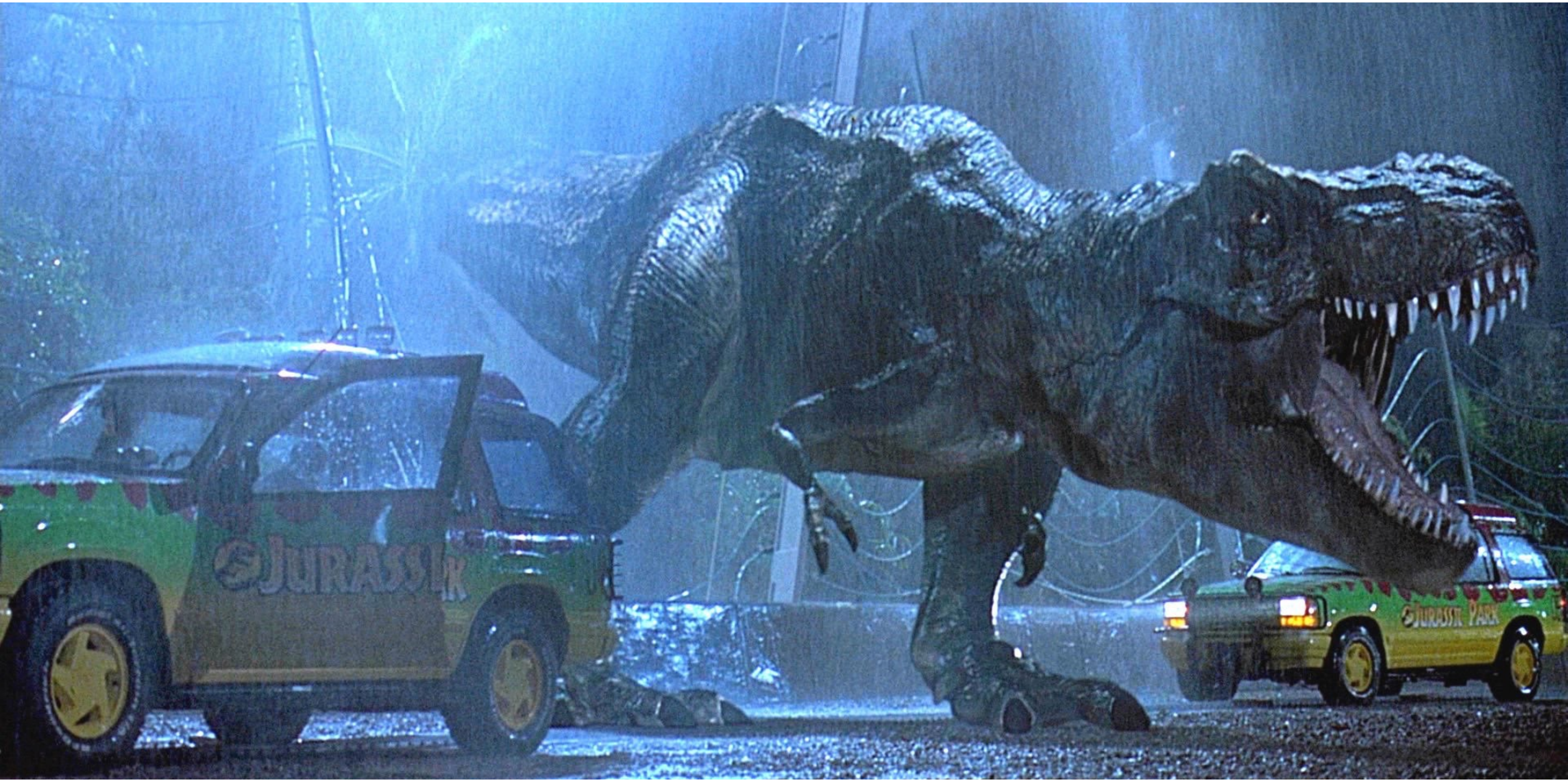


(Allan Ajifo)

(Petar Milošević)

...eyes are highest-bandwidth port into the head!

Movies



Jurassic Park (1993)

Movies

The Matrix (1999)

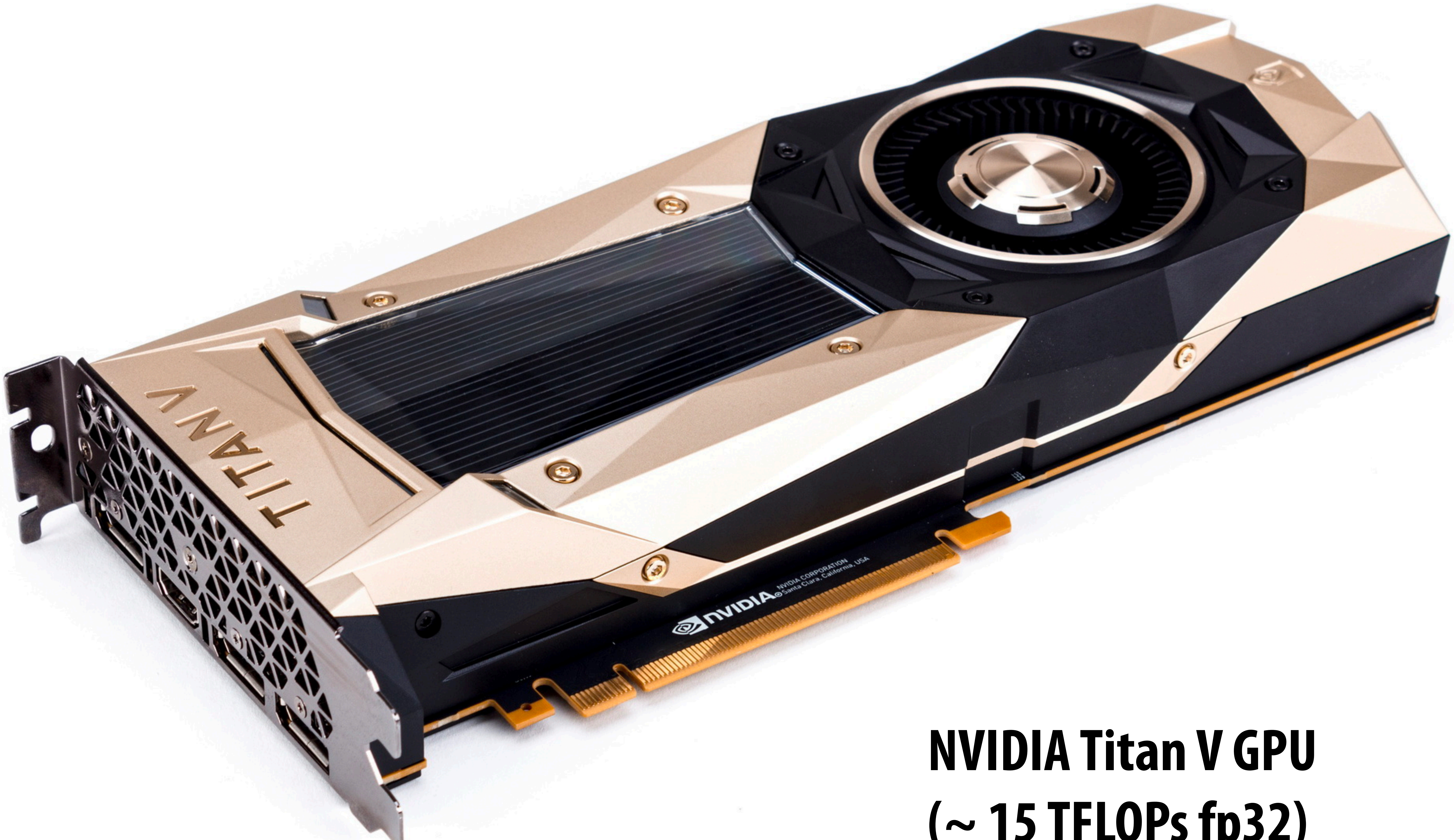
Computer games

This image is rendered in real-time on a modern GPU

Computer games

Assassin's Creed Odyssey (Ubisoft 2018)

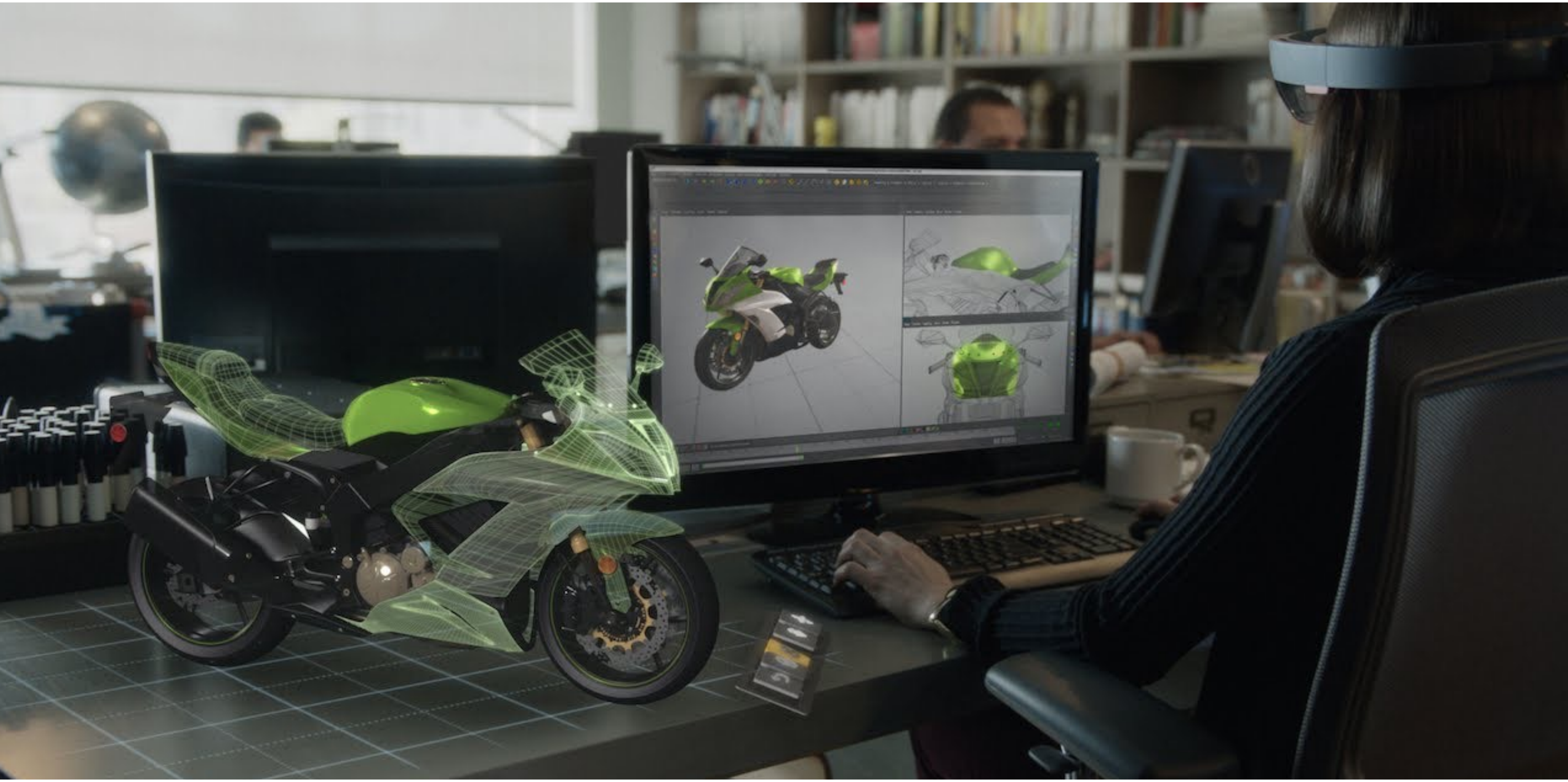
Supercomputing for games



NVIDIA Titan V GPU
(~ 15 TFLOPs fp32)

Virtual reality experiences

Augmented reality



Microsoft Hololens augmented reality headset concept

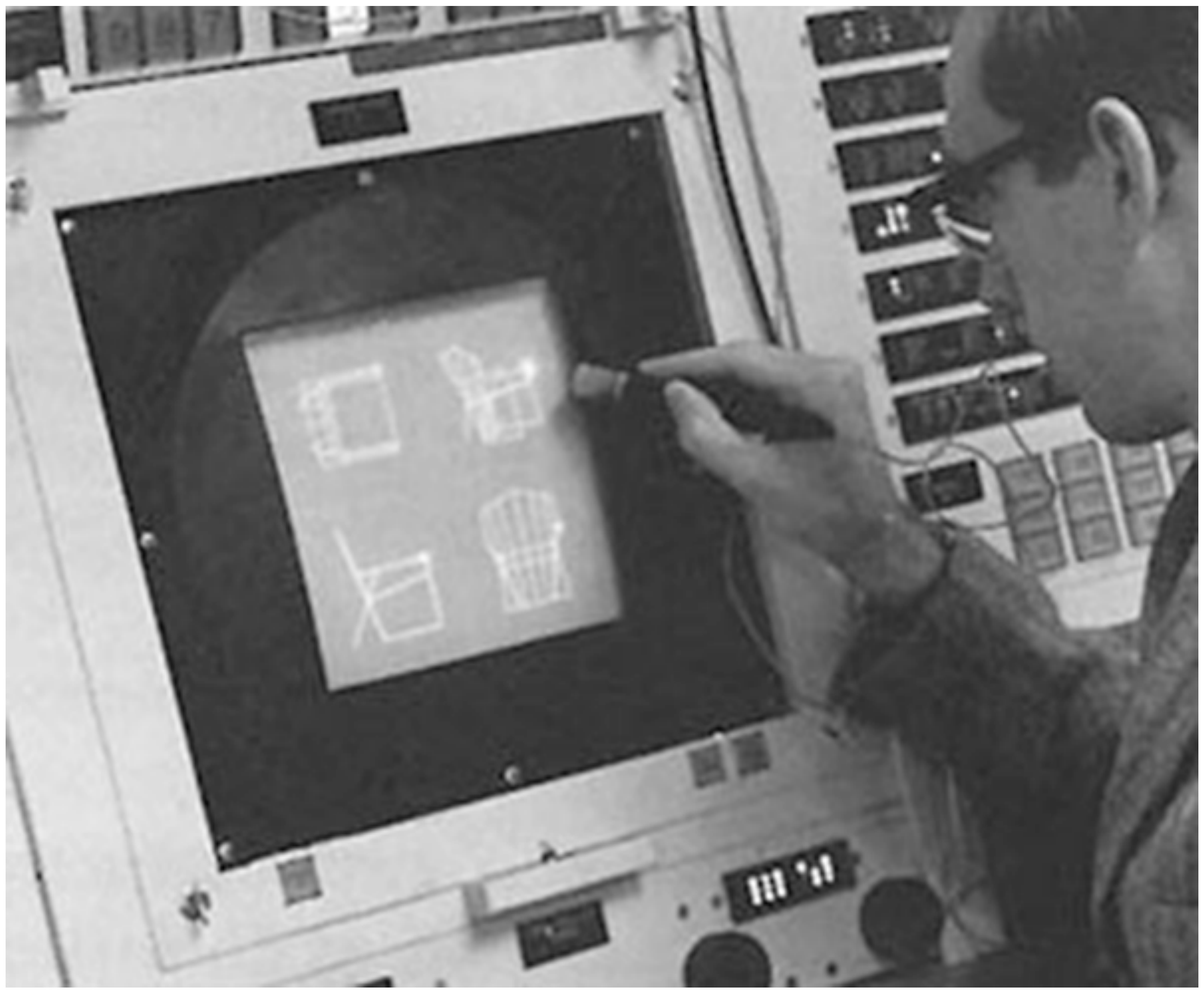
Illustration

Indonesian cave painting (~38,000 BCE)

Digital illustration

Meike Hakkart
http://maquenda.deviantart.com/art/Lion-done-in-illustrator-327715059

Graphical user interfaces



Ivan Sutherland, “Sketchpad” (1963)

Doug Engelbart
Mouse

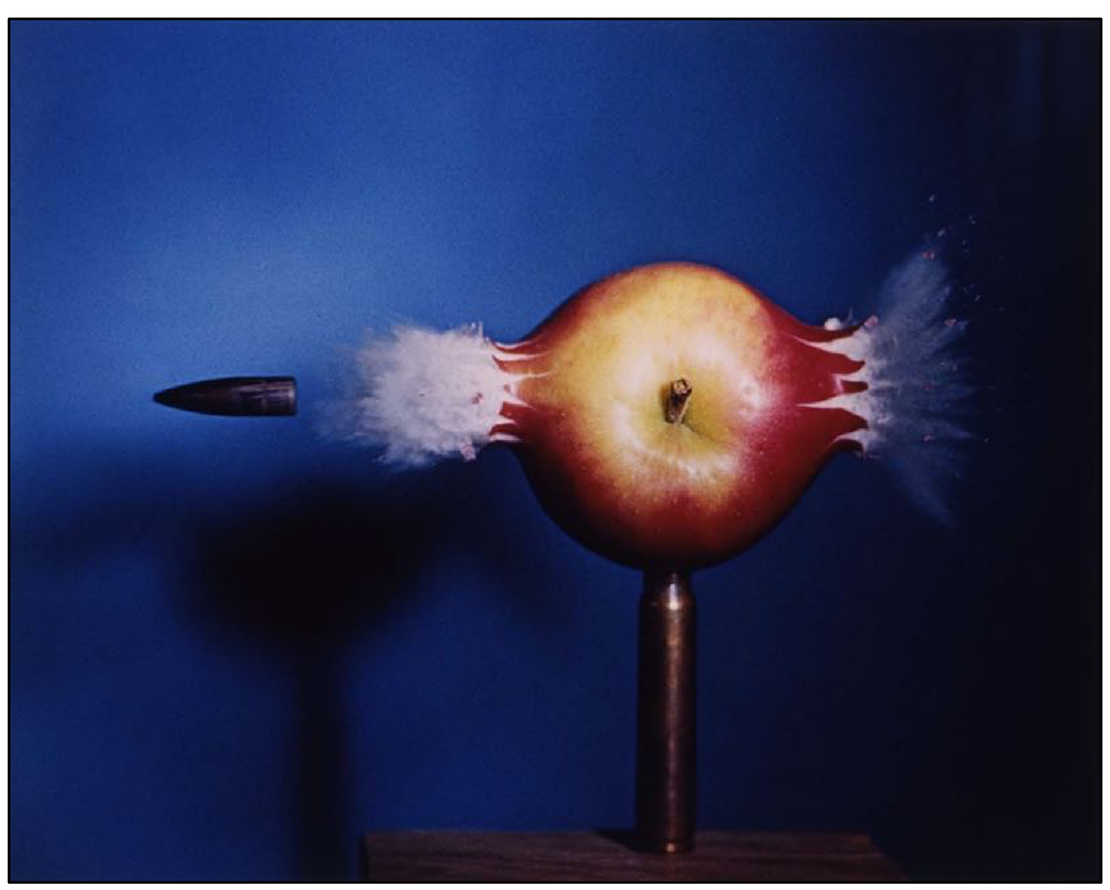
Modern graphical user interfaces

2D drawing and animation are ubiquitous in computing.

Typography, icons, images, transitions, transparency, …

(all rendered at high frame rate for rich experience)

Digital photography

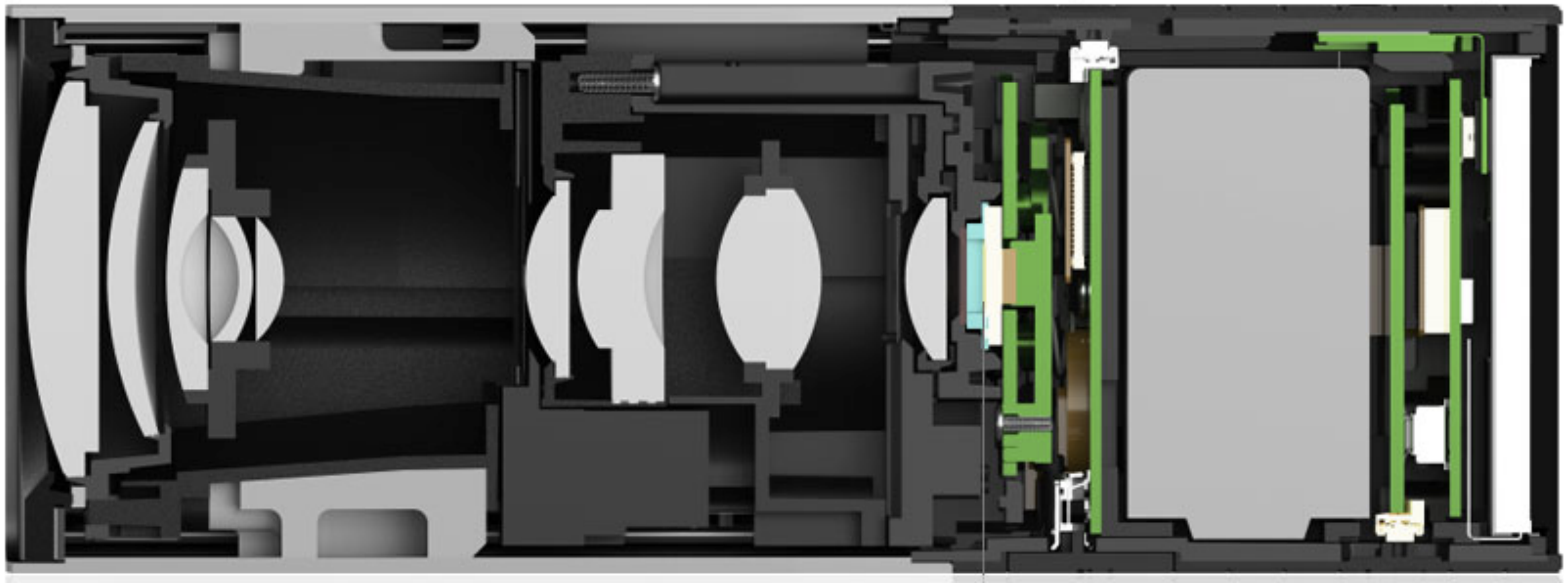


NASA | Walter Iooss | Steve McCurry
Harold Edgerton | NASA | National Geographic

Ubiquitous imaging

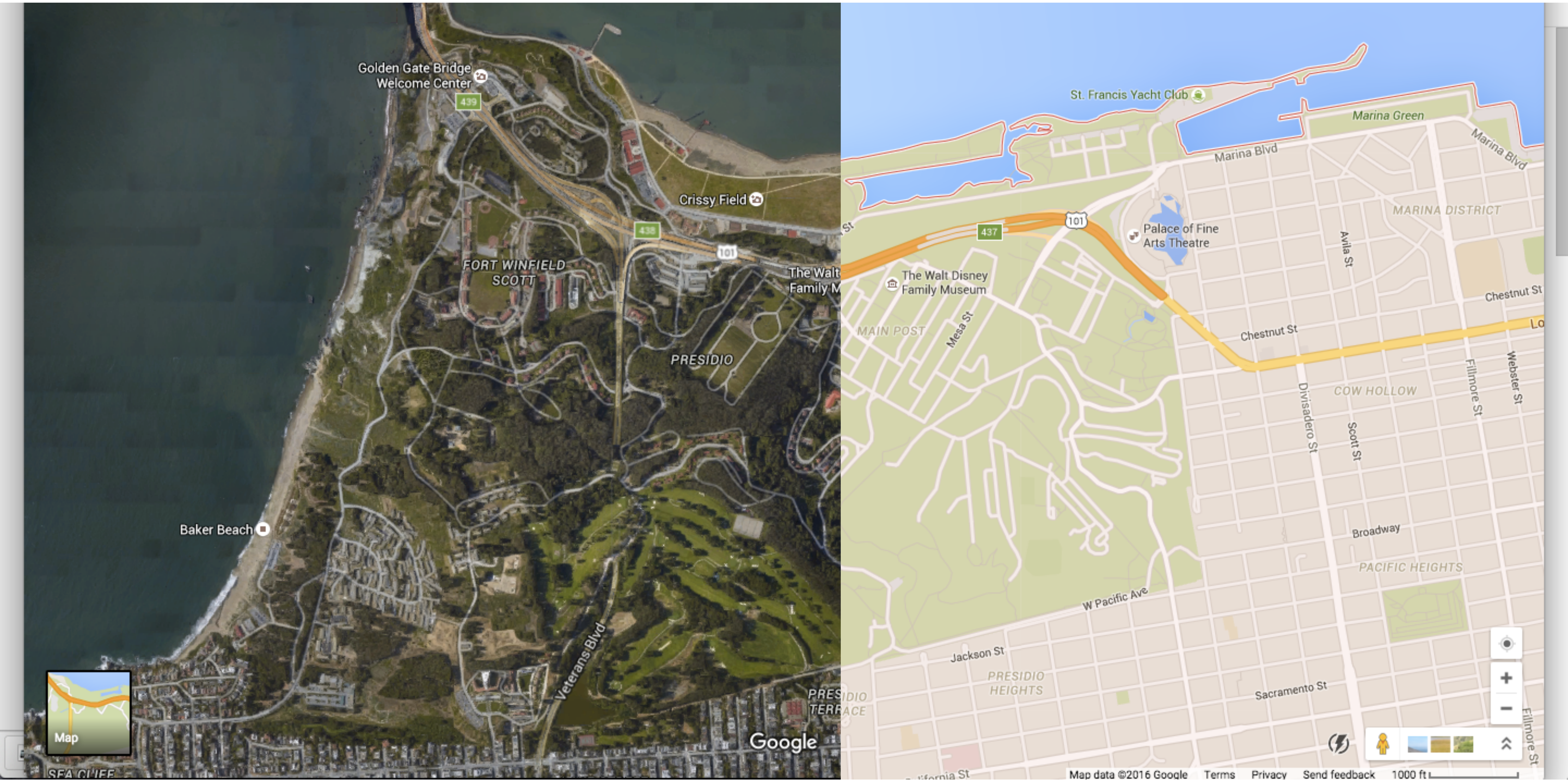
Cameras everywhere

Computational cameras



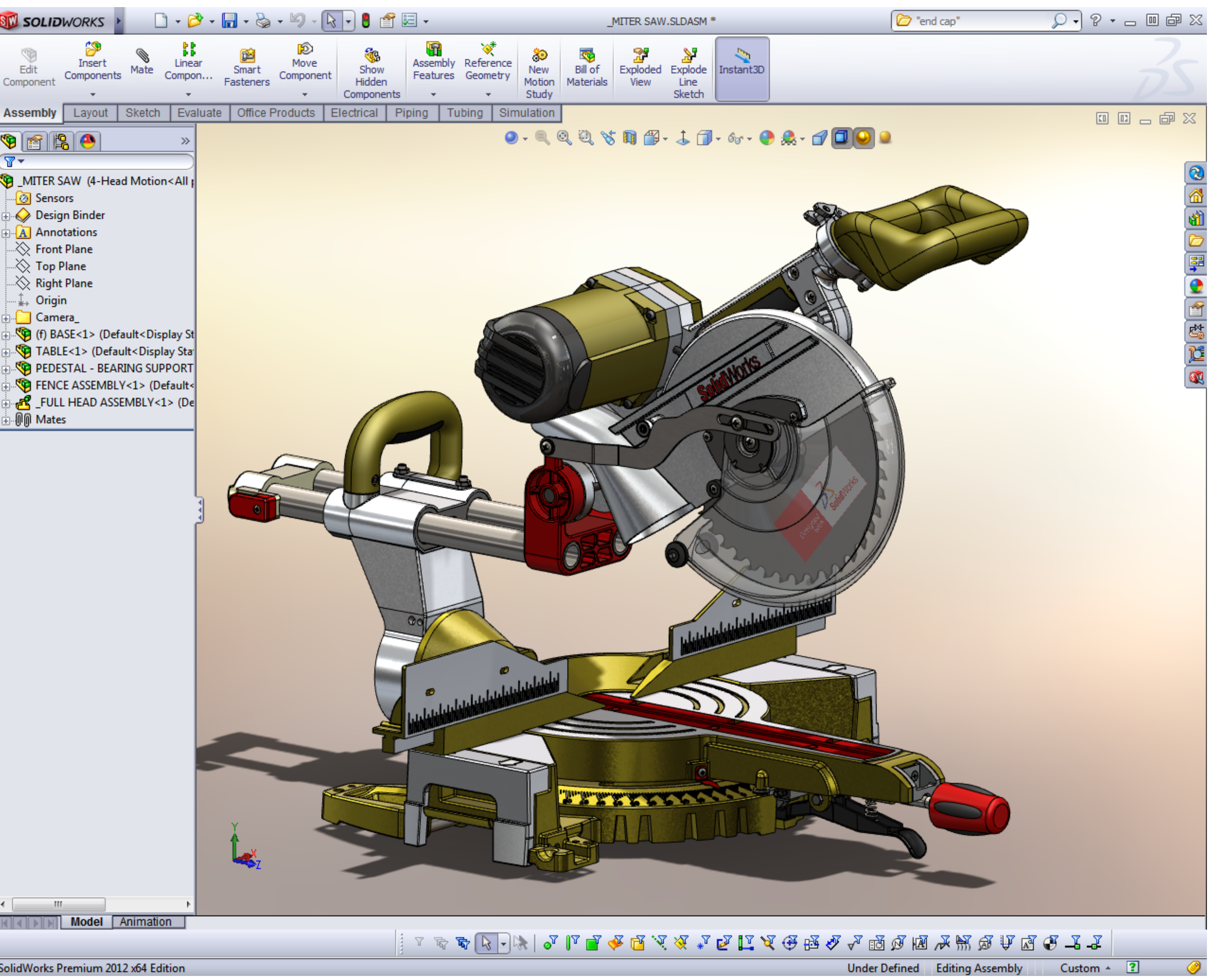
Panaromic stitching, HDR photos, light field cameras, …

Imaging for mapping

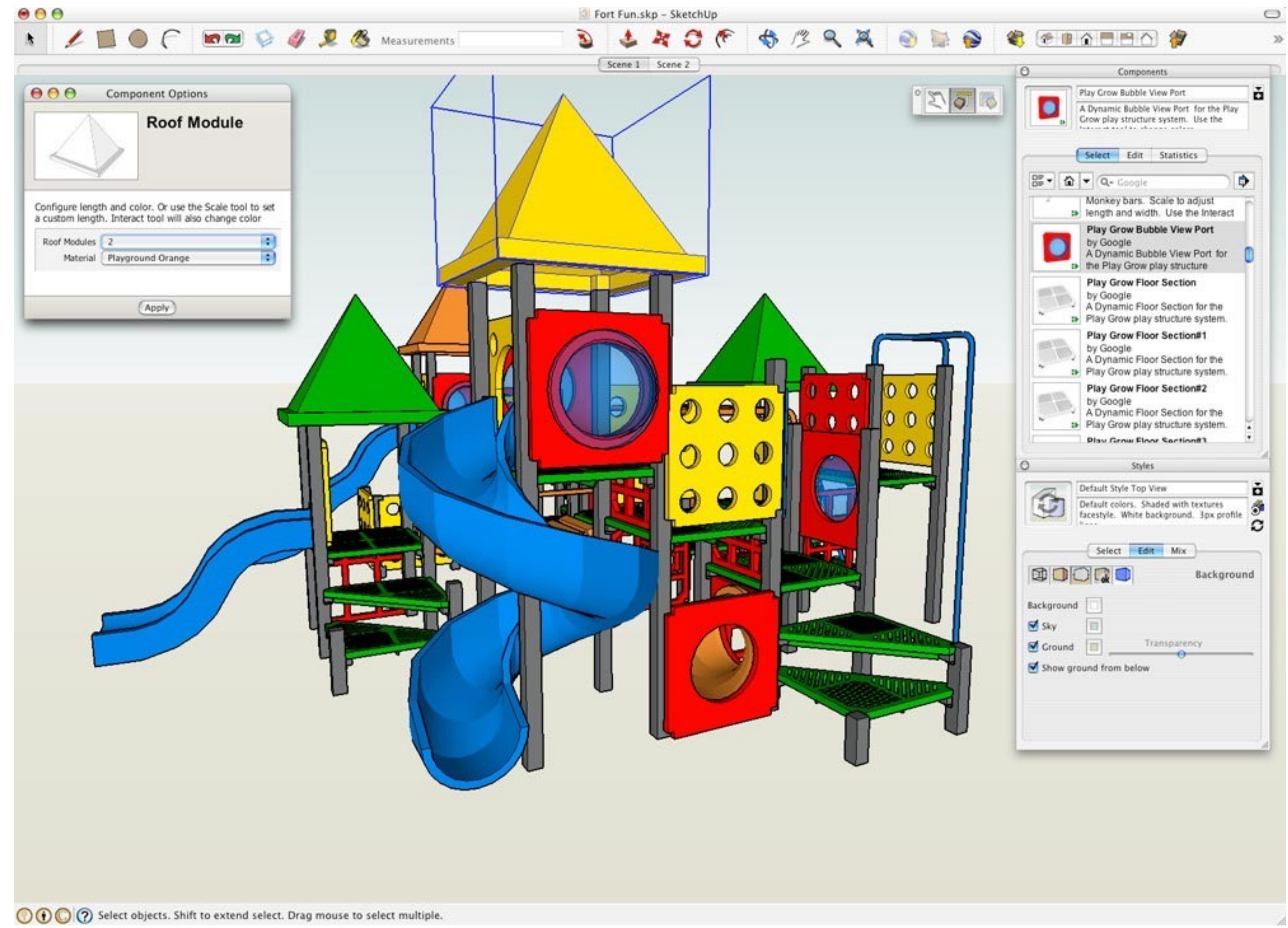


Maps, satellite imagery, street-level imaging,...

Computer aided design



SolidWorks



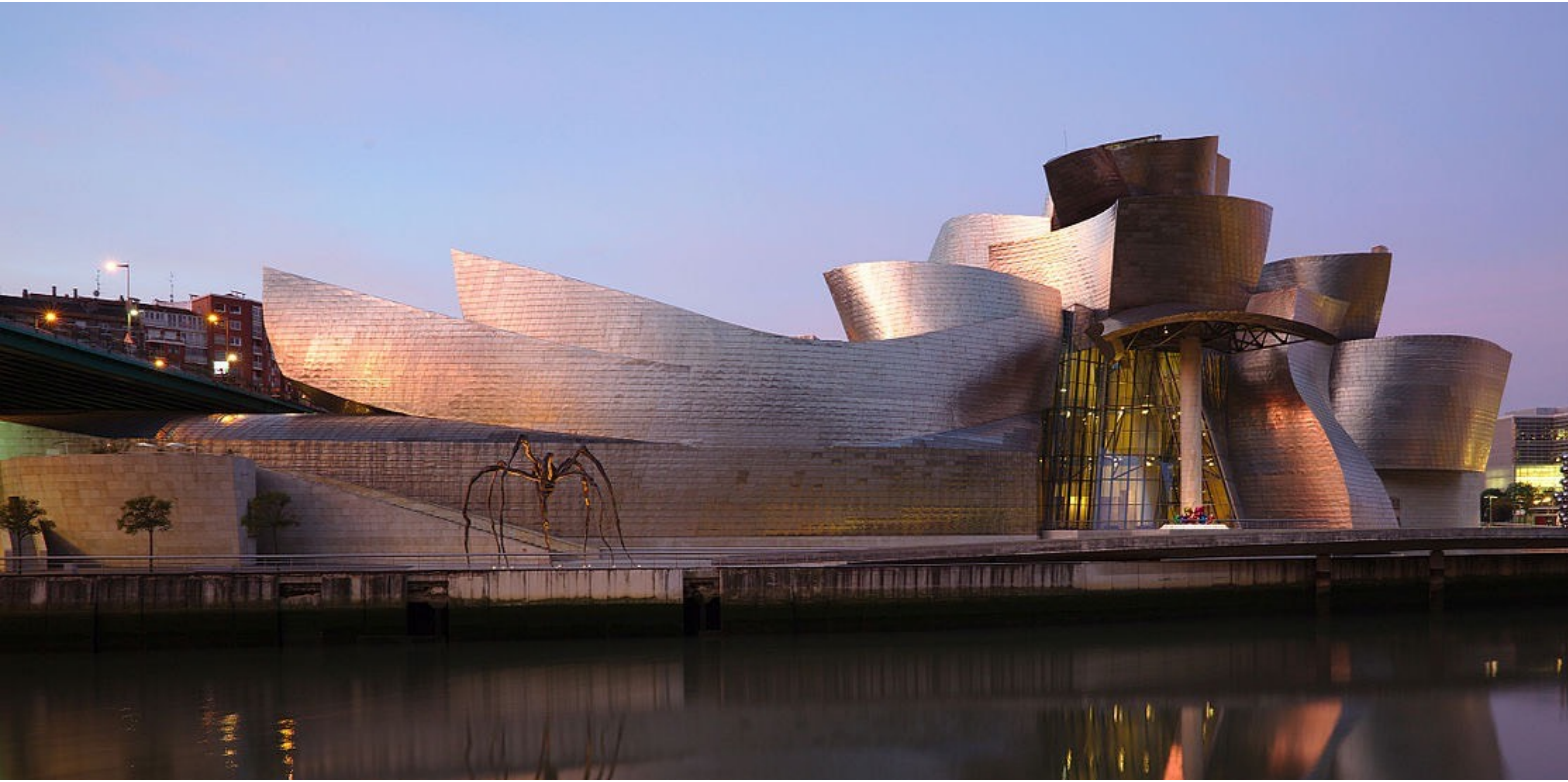
SketchUp

For mechanical, architectural, electronic, optical, …

Product design and visualization

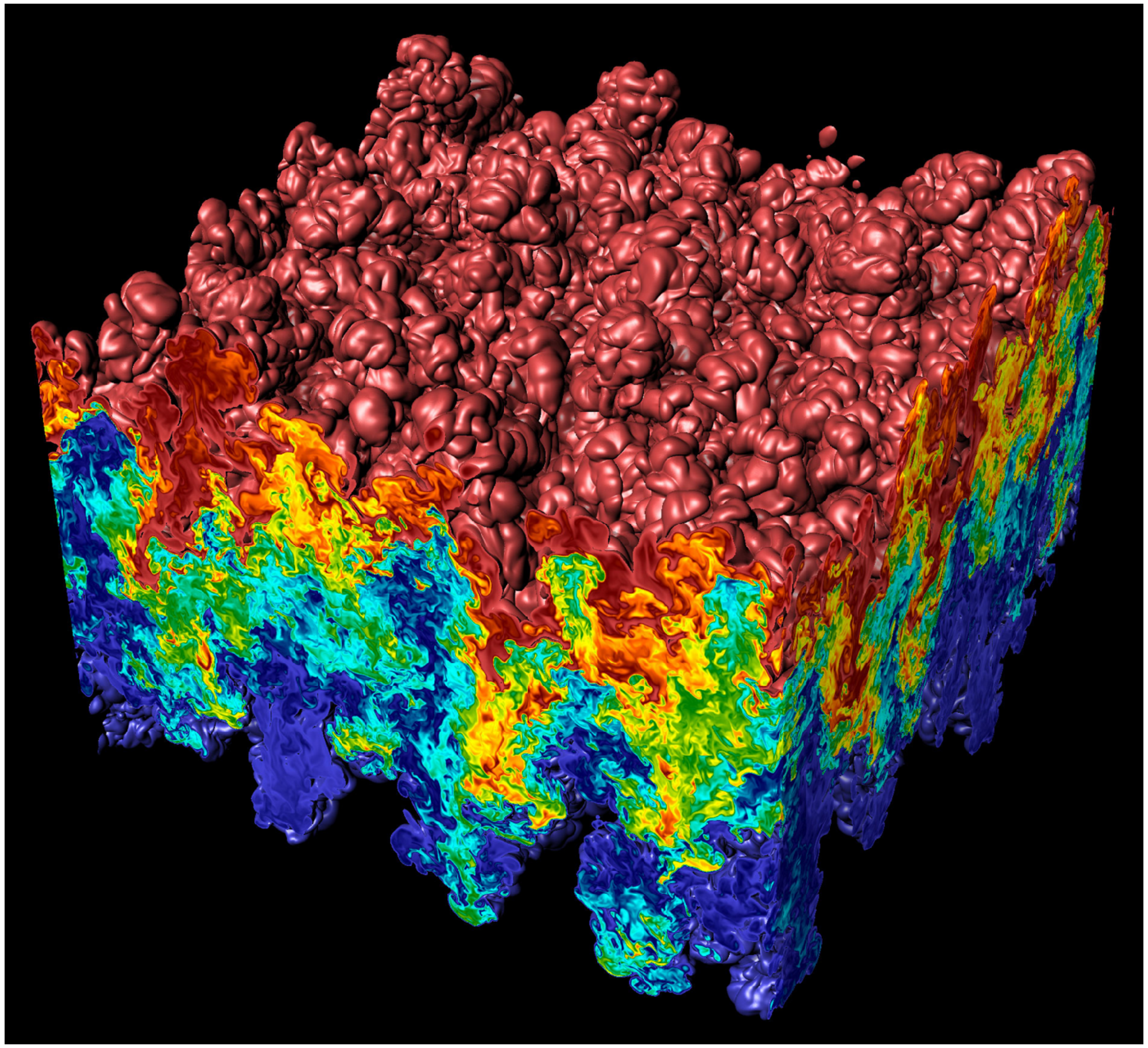
Ikea - 75% of catalog is rendered imagery

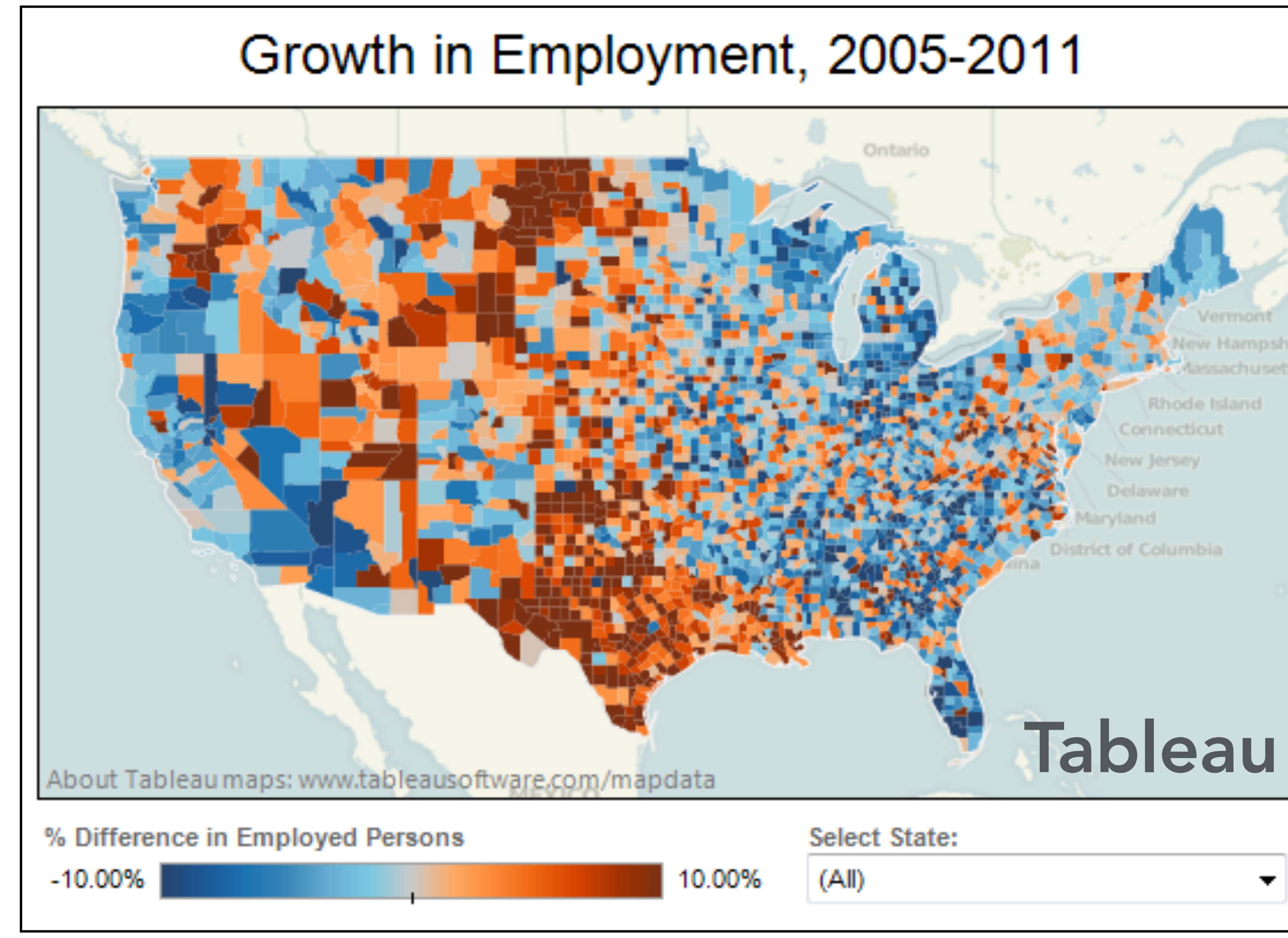
Architectural design



Bilbao Guggenheim, Frank Gehry

Visualization





Science, engineering, medicine, journalism, ...

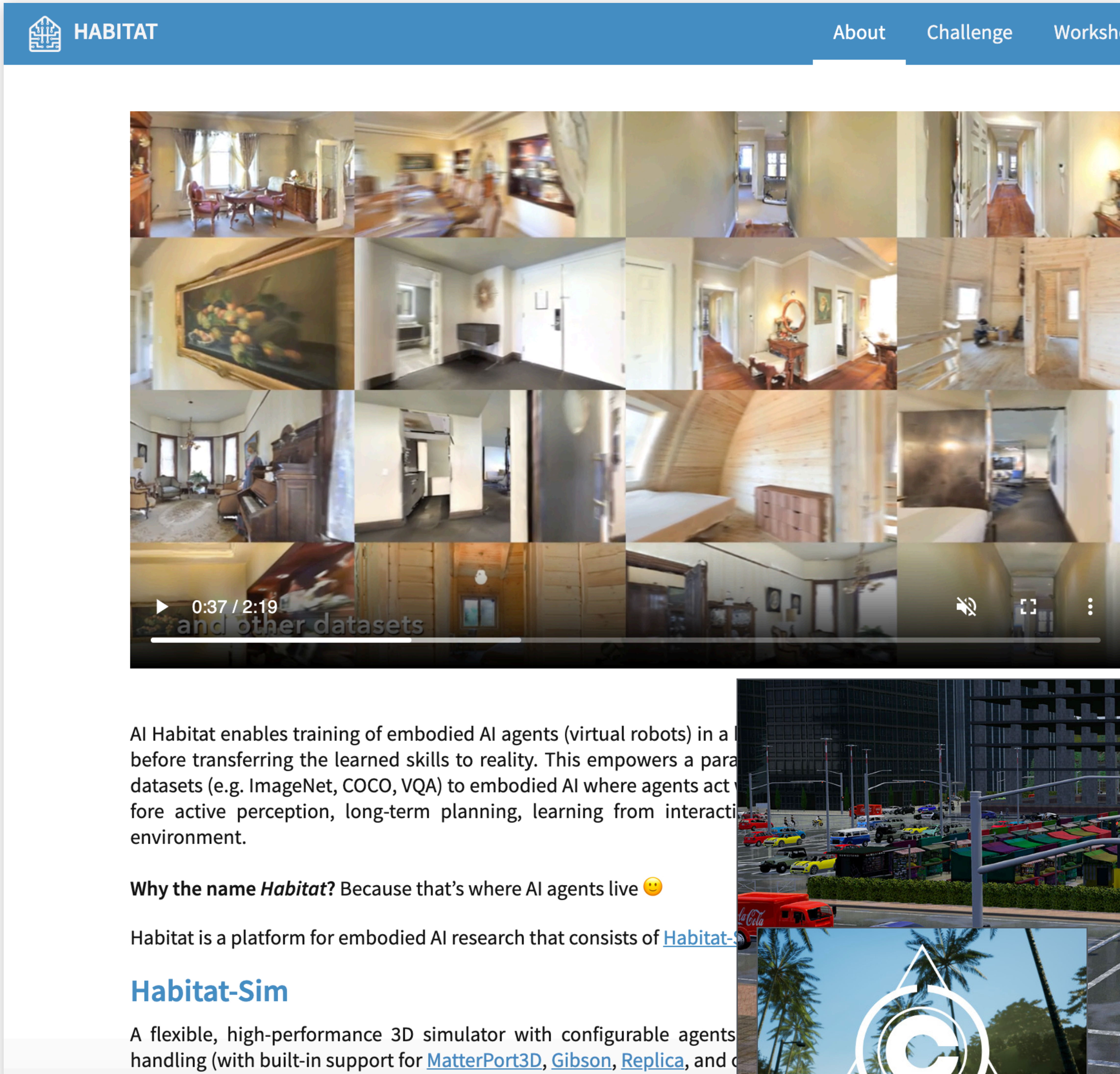
Simulation

Driving simulator
Toyota Higashifuji Technical Center

da Vinci surgical robot
Intuitive Surgical

Flight simulator, driving simulator, surgical simulator, . . .

Simulation for training models

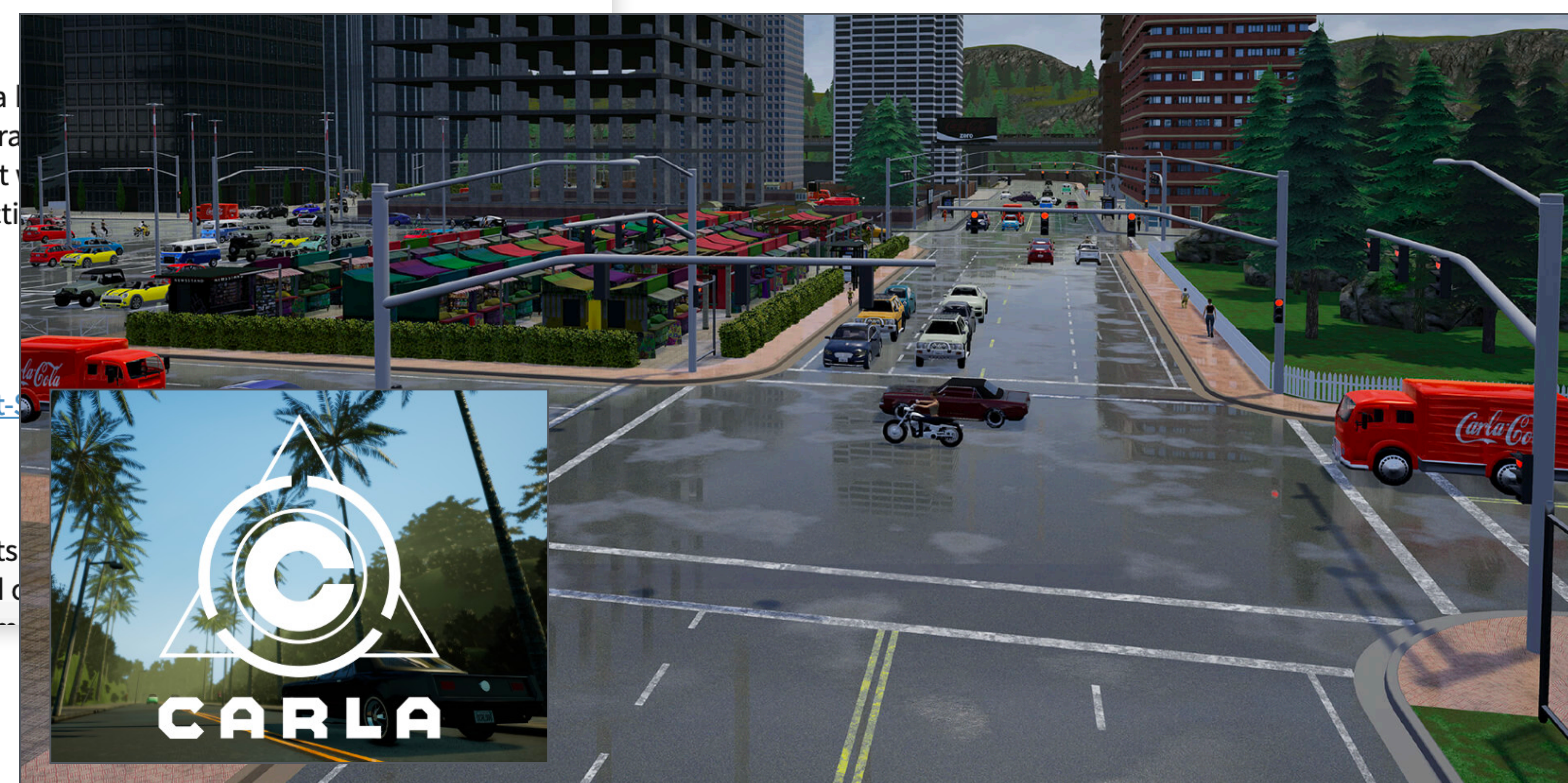


AI Habitat:

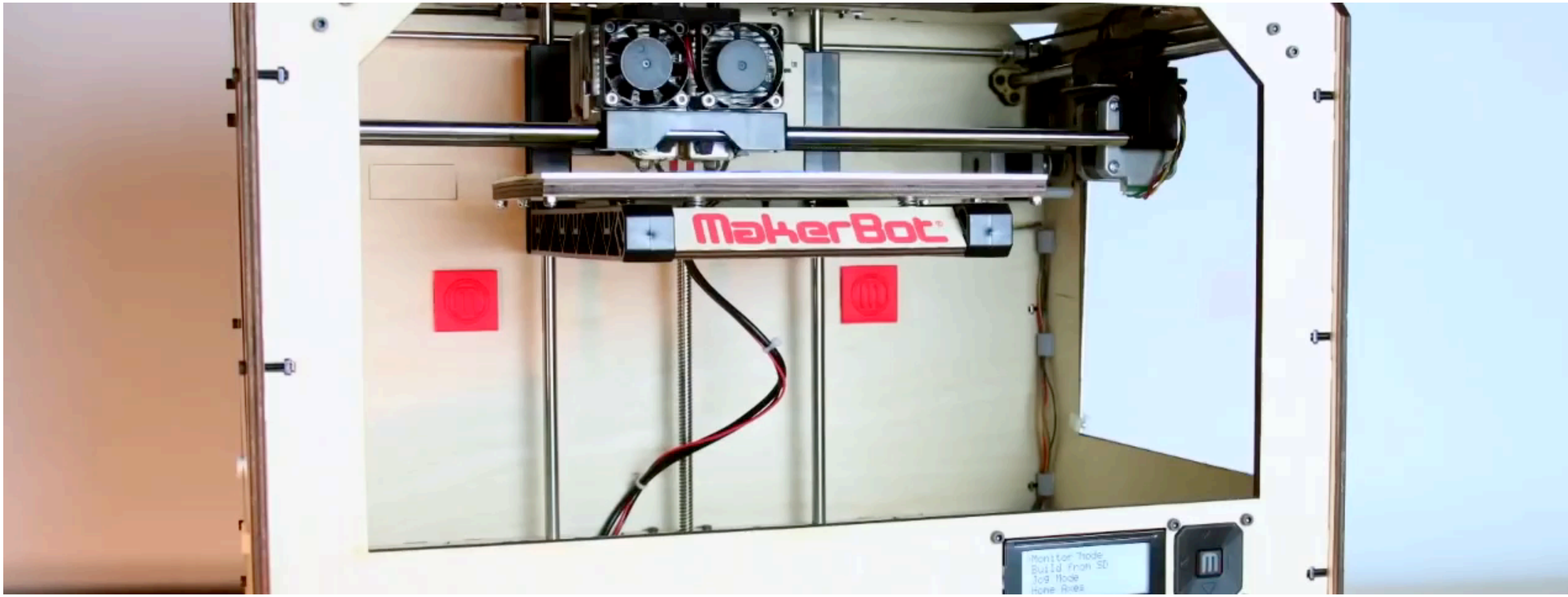
simulator for training AI agents

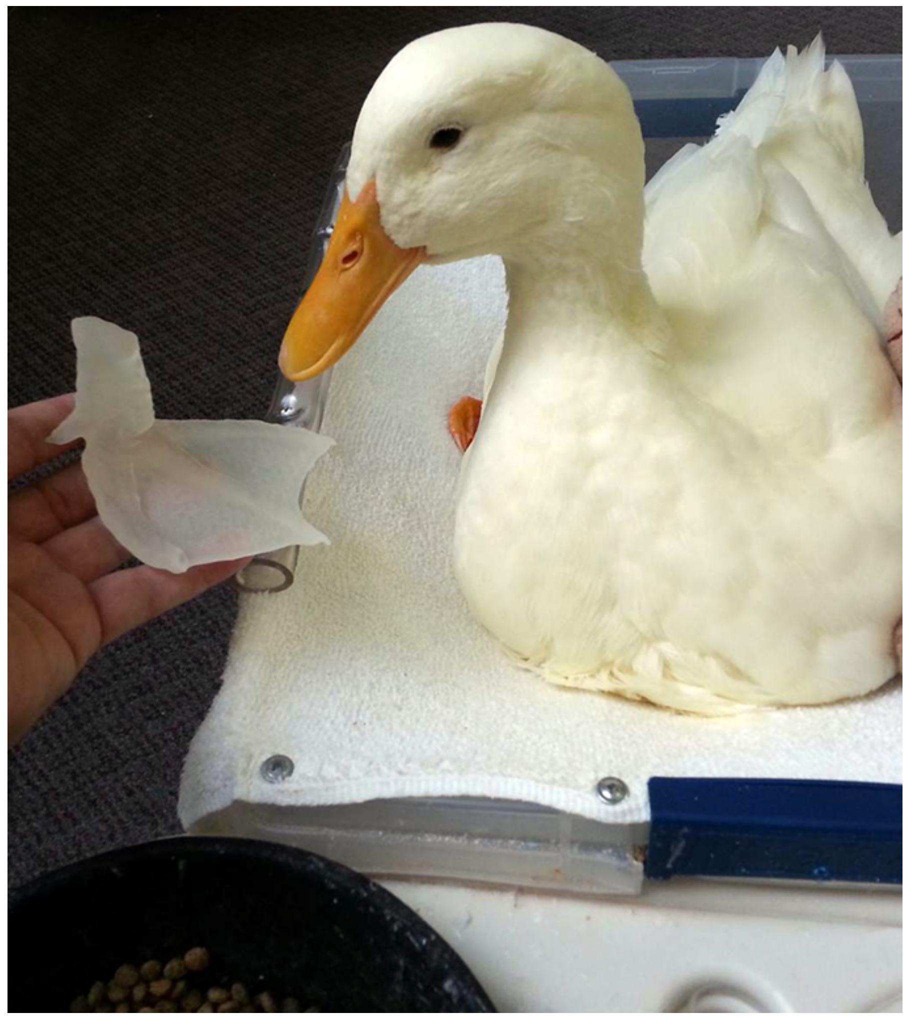
Carla:

autonomous driving simulator



3D fabrication





Foundations of computer graphics

- All these applications demand *sophisticated* theory and systems
- Science and mathematics
  - Physics of light, color, optics
  - Math of curves, surfaces, geometry, perspective, …
  - Sampling
- Systems
  - Parallel, heterogeneous processing
  - Graphics-specific programming systems
  - Input/output devices
- Art and psychology
  - Perception: color, stereo, motion, image quality, …
  - Art and design: composition, form, lighting, ...

ACTIVITY: modeling and drawing a cube

- Goal: generate a realistic drawing of a cube
- Key questions:
  - *Modeling:* how do we describe the cube?
  - *Rendering:* how do we then visualize this model?

ACTIVITY: modeling the cube

- **Suppose our cube is...**
  - **centered at the origin (0,0,0)**
  - **has dimensions 2 x 2 x 2**
- **QUESTION: What are the coordinates of the cube vertices?**

```
A: ( 1, 1, 1 )   E: ( 1, 1,-1 )
B: (-1, 1, 1 )   F: (-1, 1,-1 )
C: ( 1,-1, 1 )   G: ( 1,-1,-1 )
D: (-1,-1, 1 )   H: (-1,-1,-1 )
```

- **QUESTION: What about the edges?**

```
AB, CD, EF, GH,
AC, BD, EG, FH,
AE, CG, BF, DH
```

ACTIVITY: drawing the cube

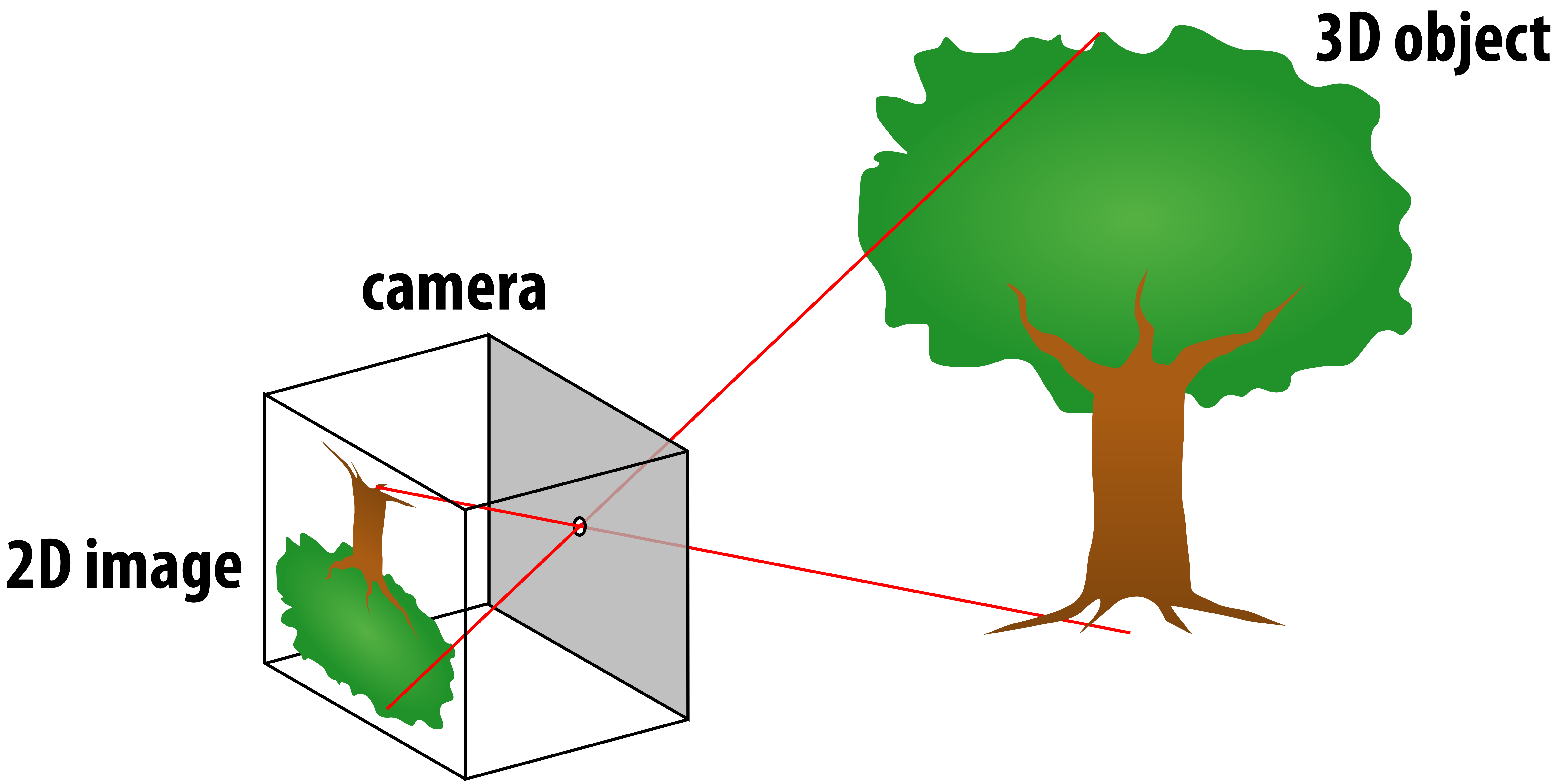
- **Now have a digital description of the cube:**

```
VERTICES                                      EDGES
A: ( 1, 1, 1 )   E: ( 1, 1,-1 )
B: (-1, 1, 1 )   F: (-1, 1,-1 )               AB, CD, EF, GH,
C: ( 1,-1, 1 )   G: ( 1,-1,-1 )               AC, BD, EG, FH,
D: (-1,-1, 1 )   H: (-1,-1,-1 )               AE, CG, BF, DH
```

- **How do we draw this 3D cube as a 2D (flat) image?**
- **Basic strategy:**
  1. **Project 3D vertices to 2D points in the image**
  2. **Connect 2D points with straight lines**
- **...Ok, but how?**

Perspective projection

- Objects look smaller as they get further away ("perspective")
- Why does this happen?
- Consider simple ("pinhole") model of a camera:

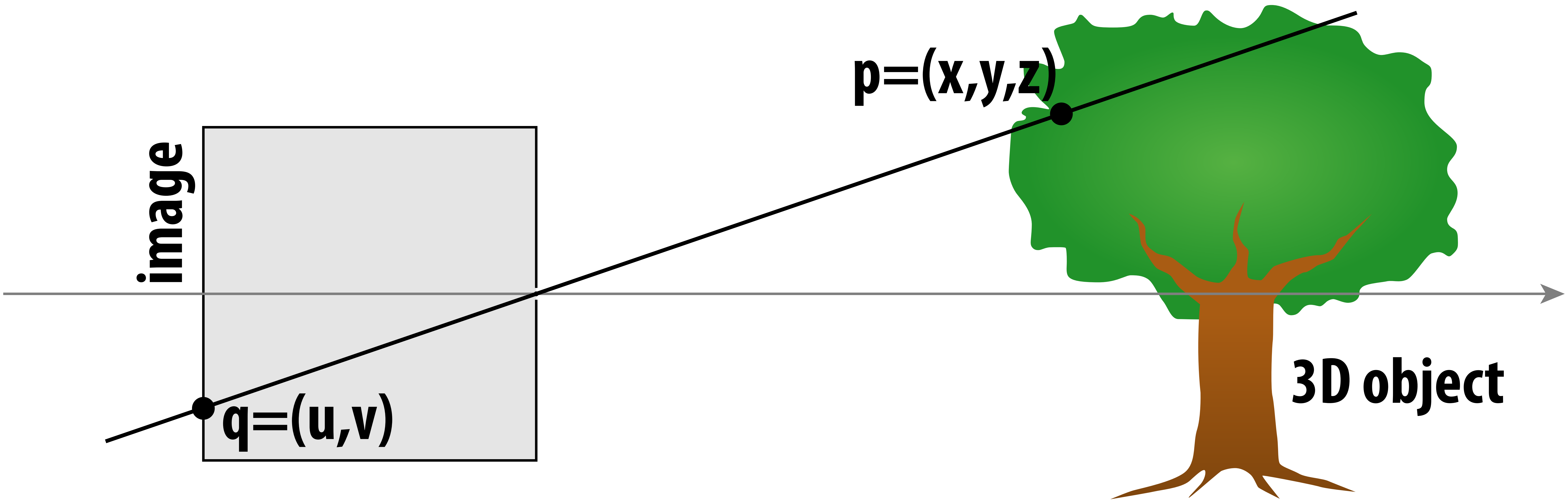


For those that didn't do this in grade school

http://jdaniel4smom.com/2017/06/pinhole-camera.html

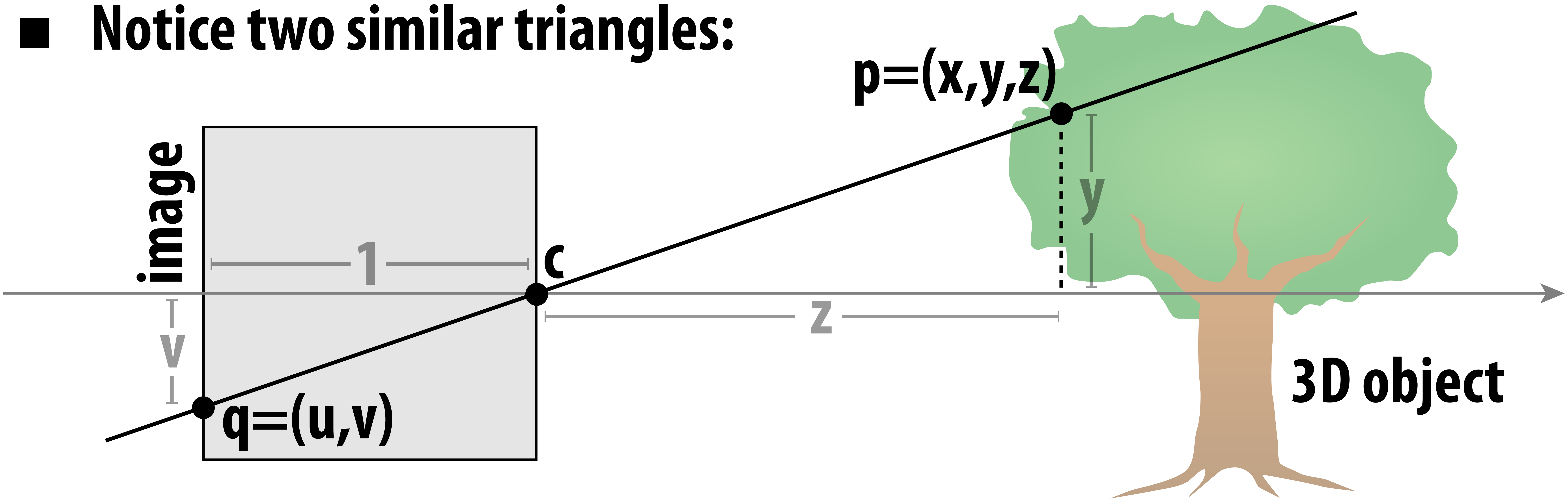
Perspective projection: side view

- Where exactly does a point p = (x,y,z) end up on the image?
- Let's call the image point q=(u,v)



Perspective projection: side view

- Where exactly does a point p = (x,y,z) end up on the image?
- Let's call the image point q=(u,v)
- Notice two similar triangles:



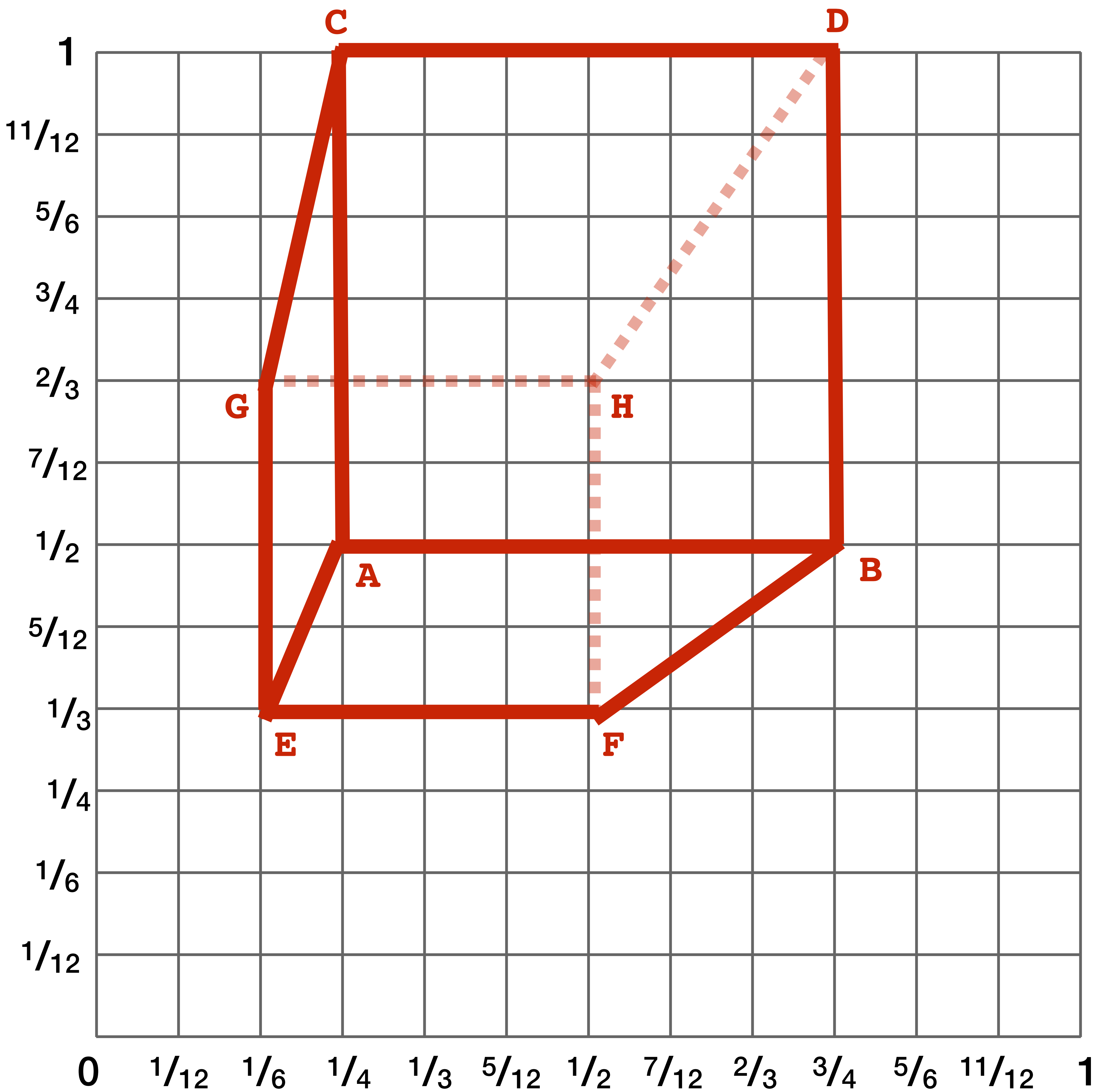
- Assume camera has unit size, coordinates relative to pinhole c
- Then v/1 = y/z, i.e., vertical coordinate is just the slope y/z
- Likewise, horizontal coordinate is u=x/z

ACTIVITY: now draw image made by pinhole camera

- **Need 12 volunteers**
  - each person will draw one cube edge
  - assume camera is at point c=(2,3,5)
  - convert (X,Y,Z) of both endpoints of edge to (u,v):
    1. subtract camera c from vertex (X,Y,Z) to get (x,y,z)
    2. divide x and y by z to get (u,v)—*write as a fraction*
  - draw line between (u1,v1) and (u2,v2)

```
VERTICES                                    EDGES
A: ( 1, 1, 1 )   E: ( 1, 1,-1 )
B: (-1, 1, 1 )   F: (-1, 1,-1 )             AB, CD, EF, GH,
C: ( 1,-1, 1 )   G: ( 1,-1,-1 )             AC, BD, EG, FH,
D: (-1,-1, 1 )   H: (-1,-1,-1 )             AE, CG, BF, DH
```

ACTIVITY: how did we do? *



2D coordinates:

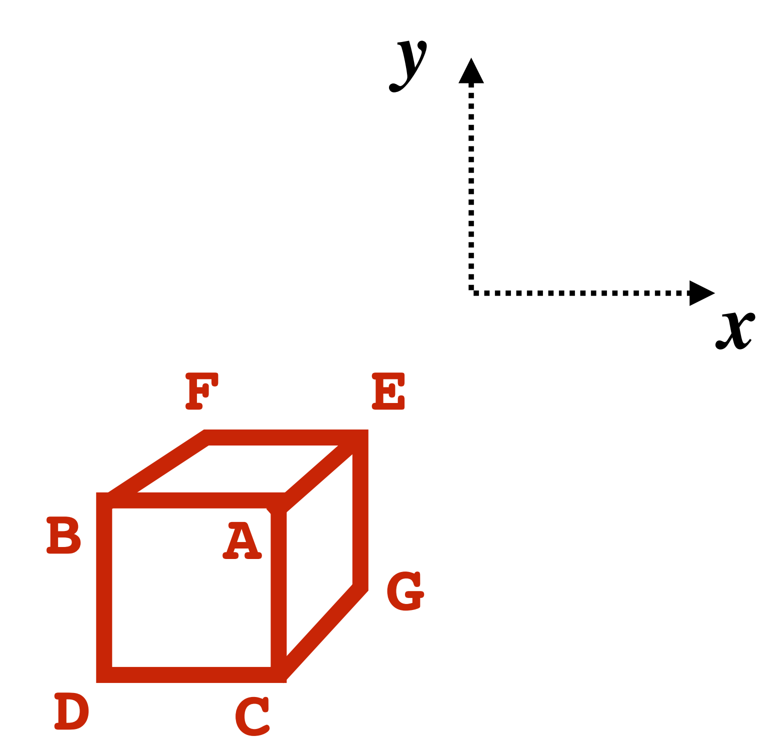
A: (1/4, 1/2)
B: (3/4, 1/2)

C: (1/4, 1)
D: (3/4, 1)

E: (1/6, 1/3)
F: (1/2, 1/3)

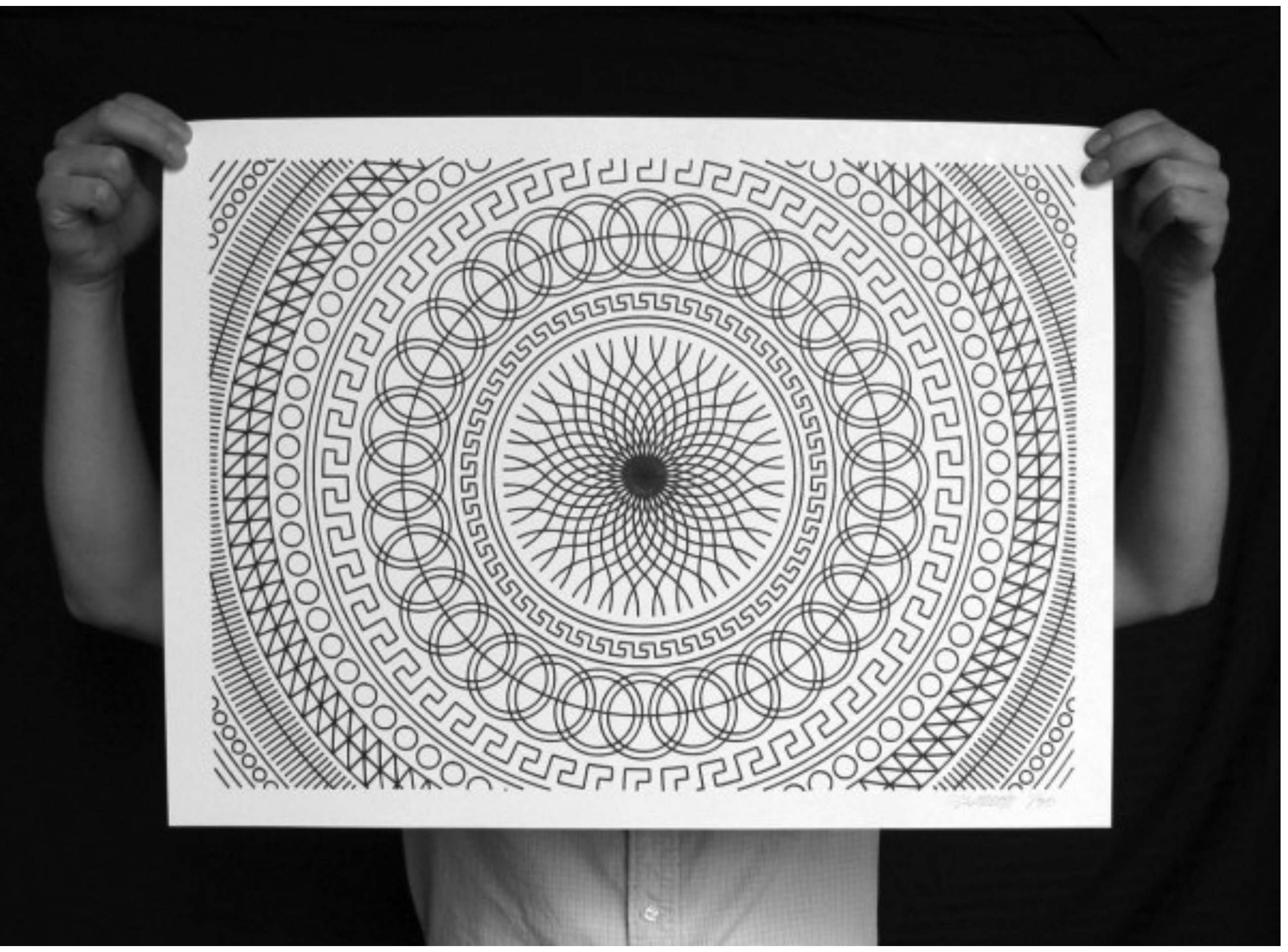
G: (1/6, 2/3)
H: (1/2, 2/3)

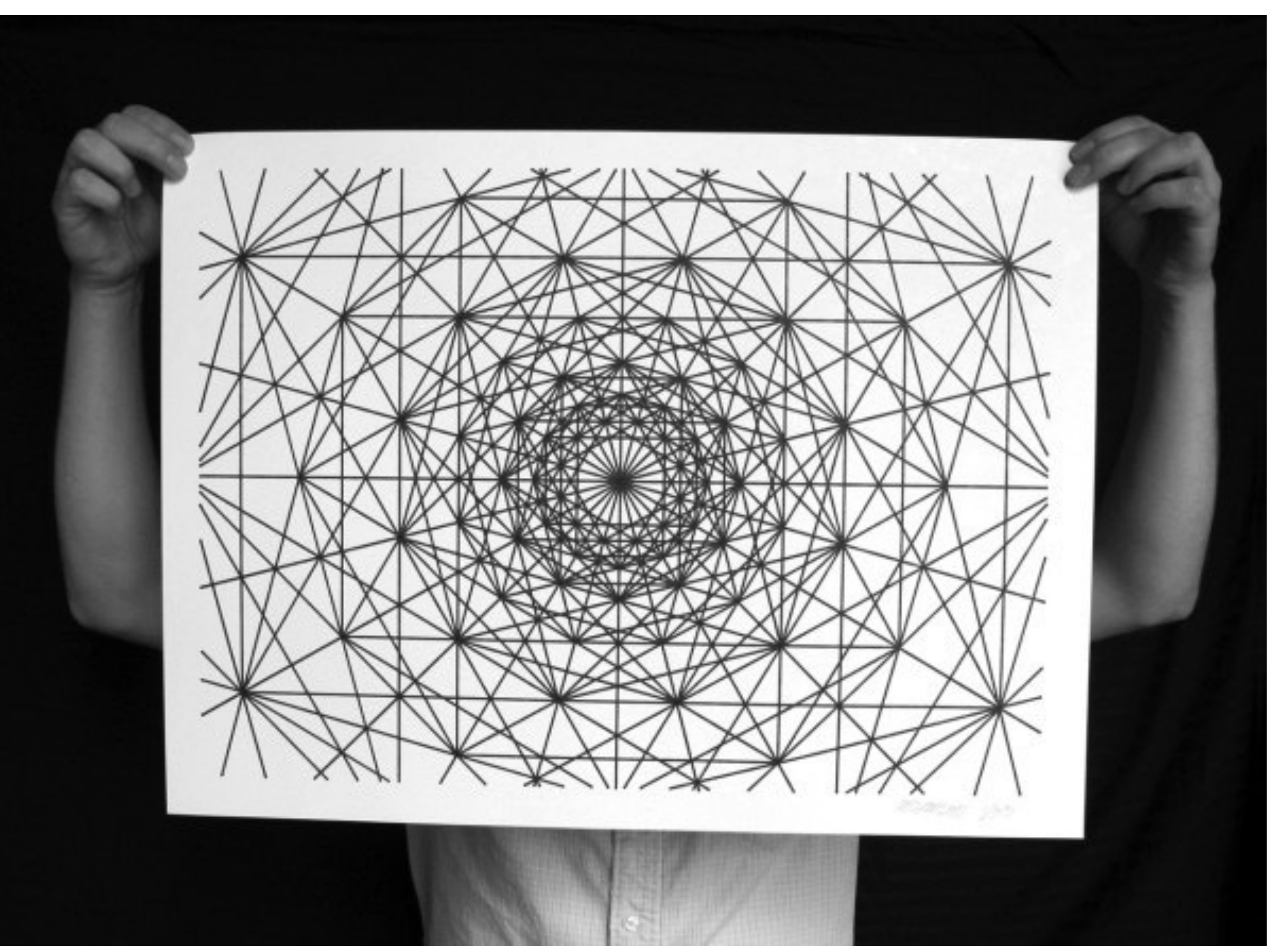
* keep in mind, this image is mirrored since it is a pinhole projection. Mirror the result and you get…



But wait…
How do we draw lines on a computer?

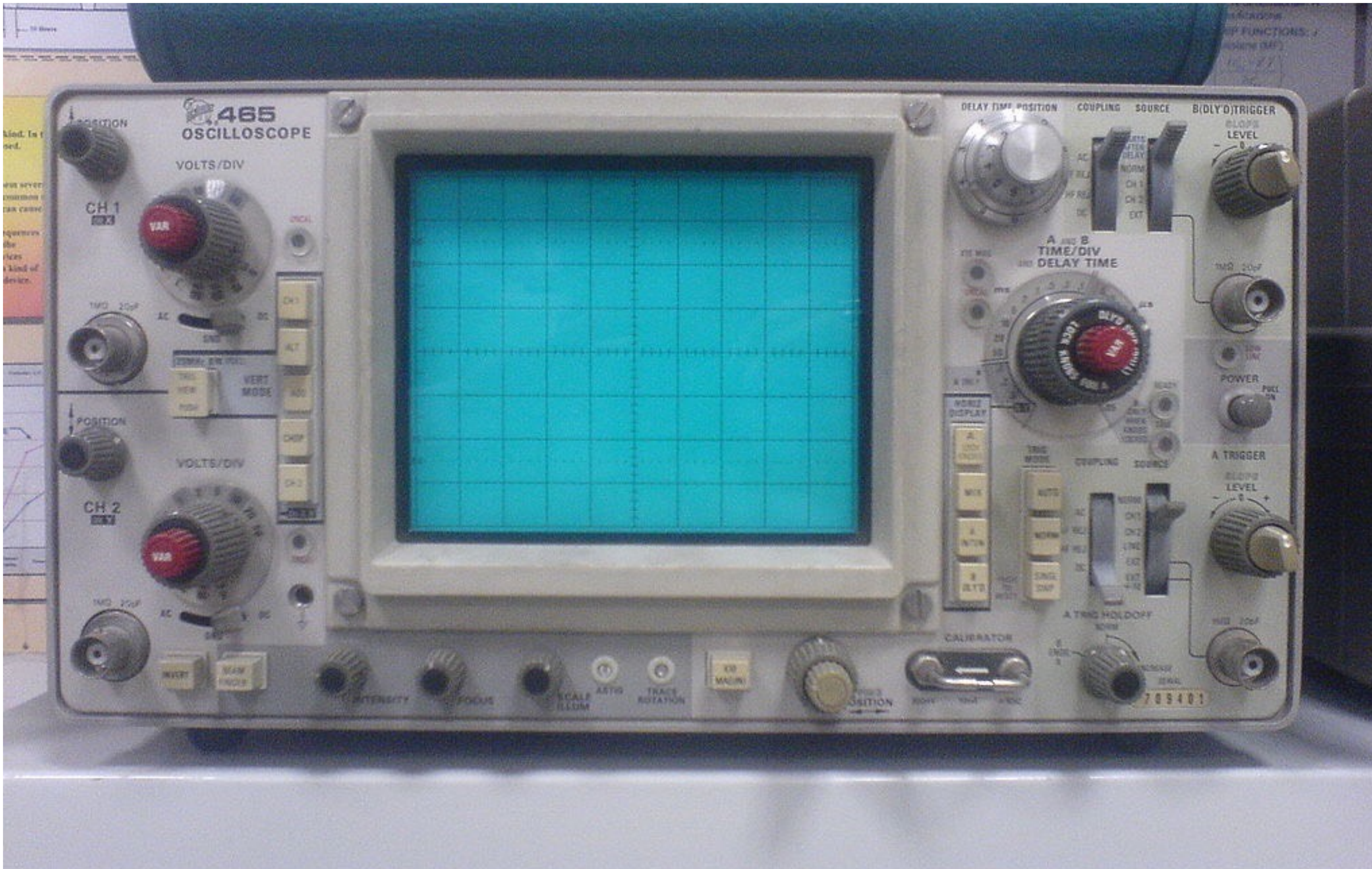
CNC sharpie drawing machine ;-)



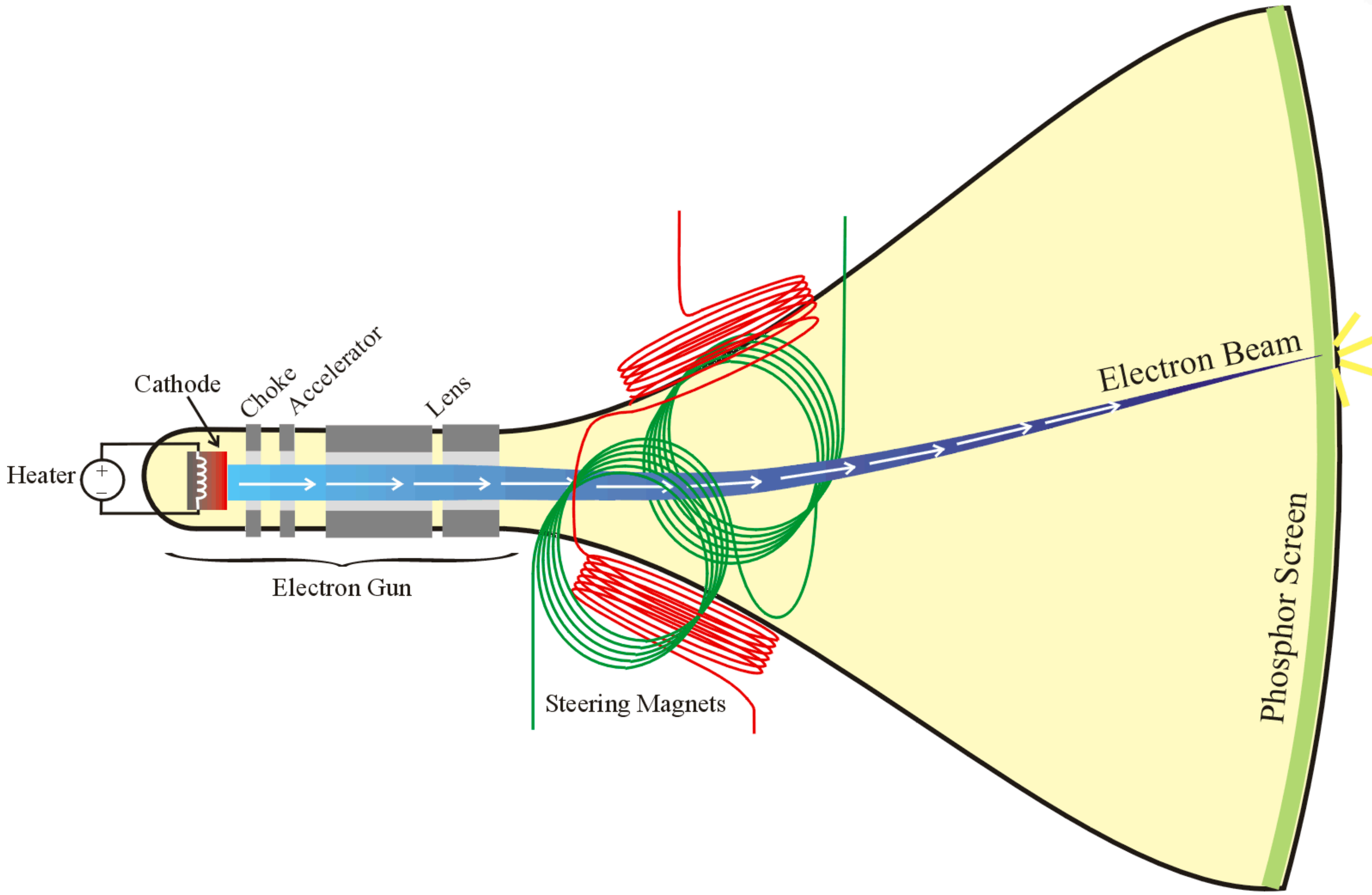


http://44rn.com/projects/numerically-controlled-poster-series-with-matt-w-moore/

Oscilloscope

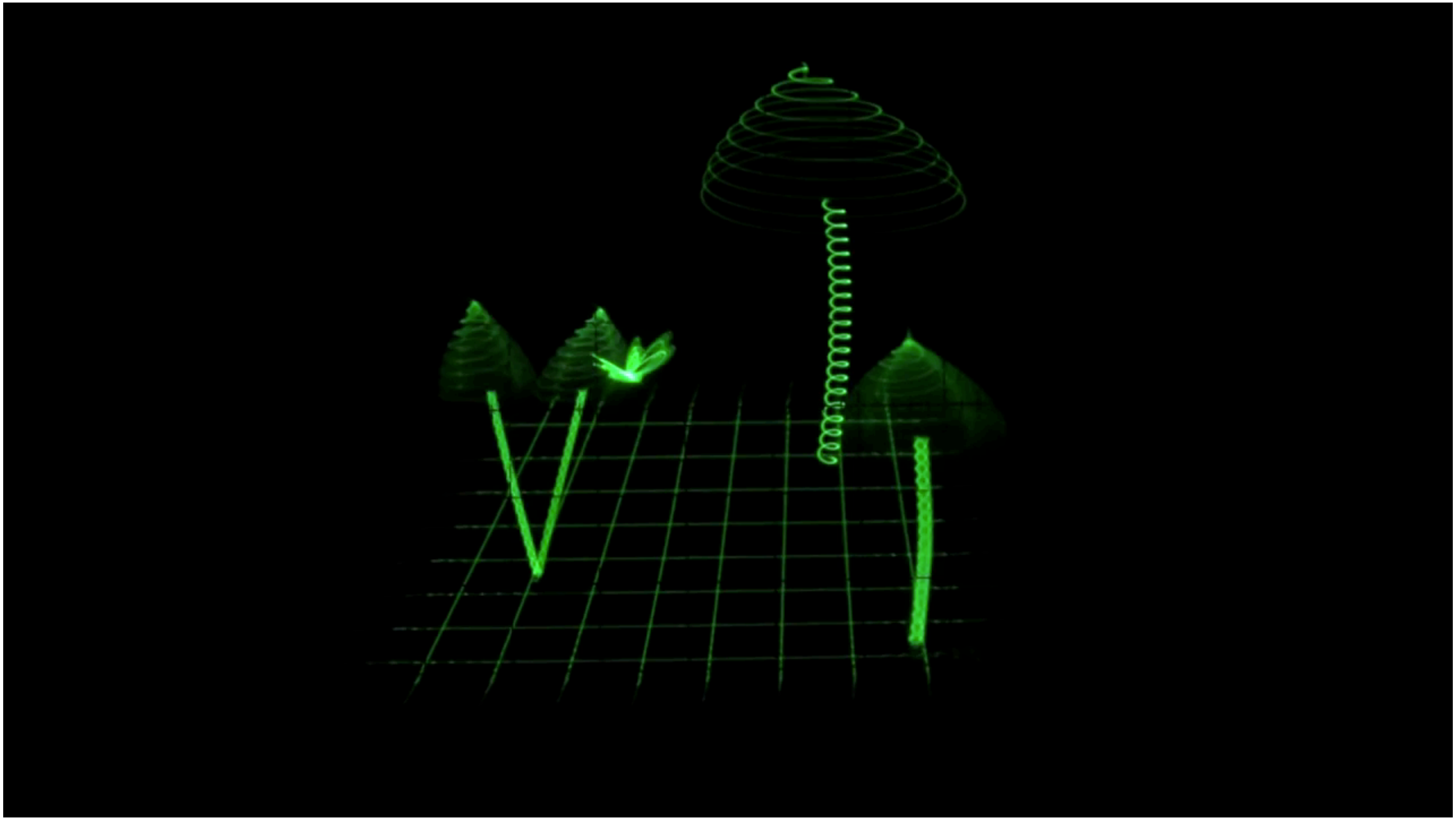


Cathode ray tube



[Credit: http://propagation.ece.gatech.edu/ECE3025/tutorials/CathodeRayTube/CRToverview.htm]

Oscilloscope art :-)



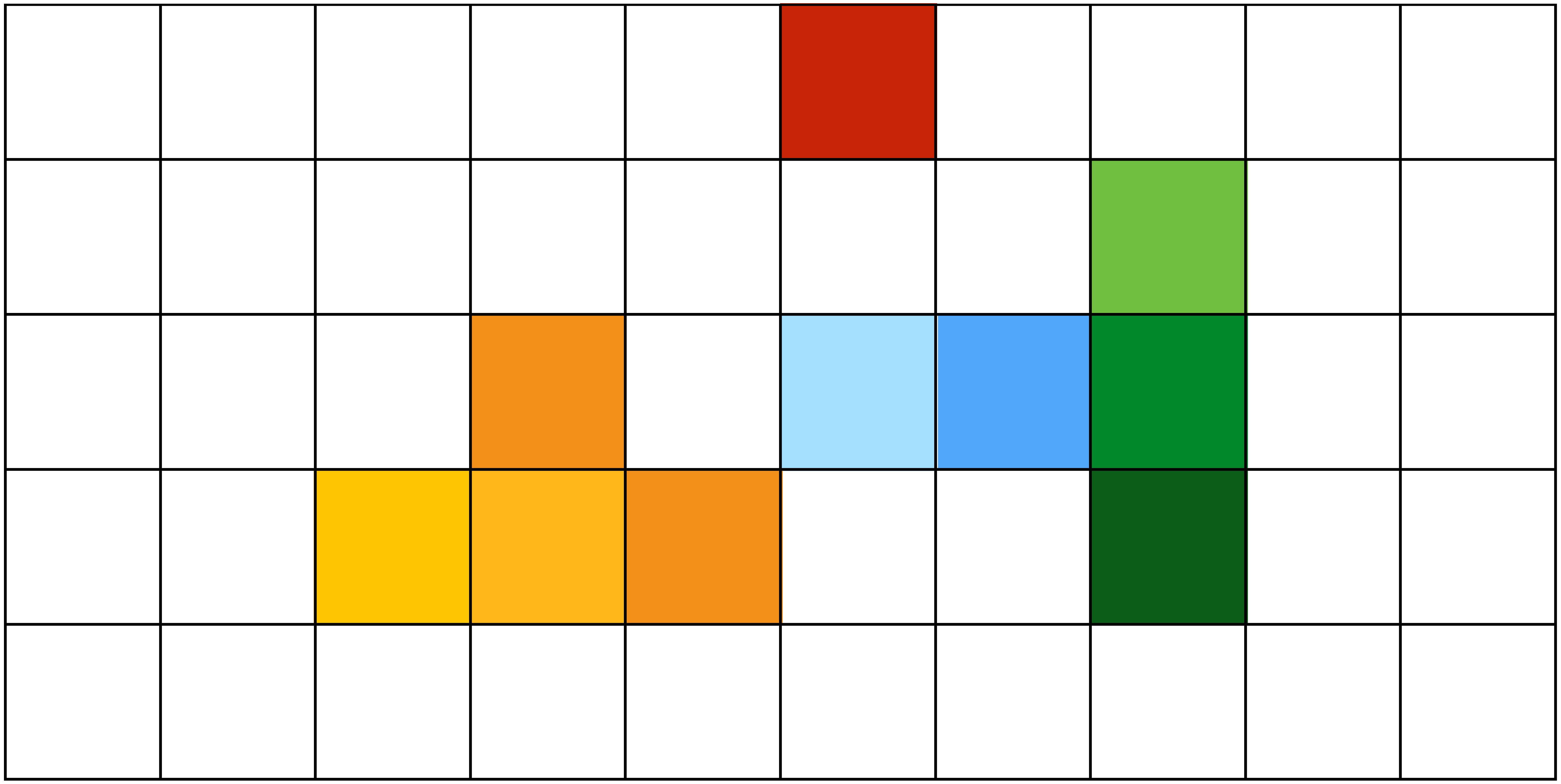
https://www.youtube.com/watch?v=rtR63-ecUNo

Frame buffer: memory for a raster display

image = "2D array of colors"

Output for a raster display

- **Common abstraction of a raster display:**
  - Image represented as a 2D grid of “pixels” (picture elements) **
  - Each pixel can can take on a unique color value



** Kayvon will strongly challenge this notion of a pixel “as a little square” next class. But let’s go with it for now. ;-)

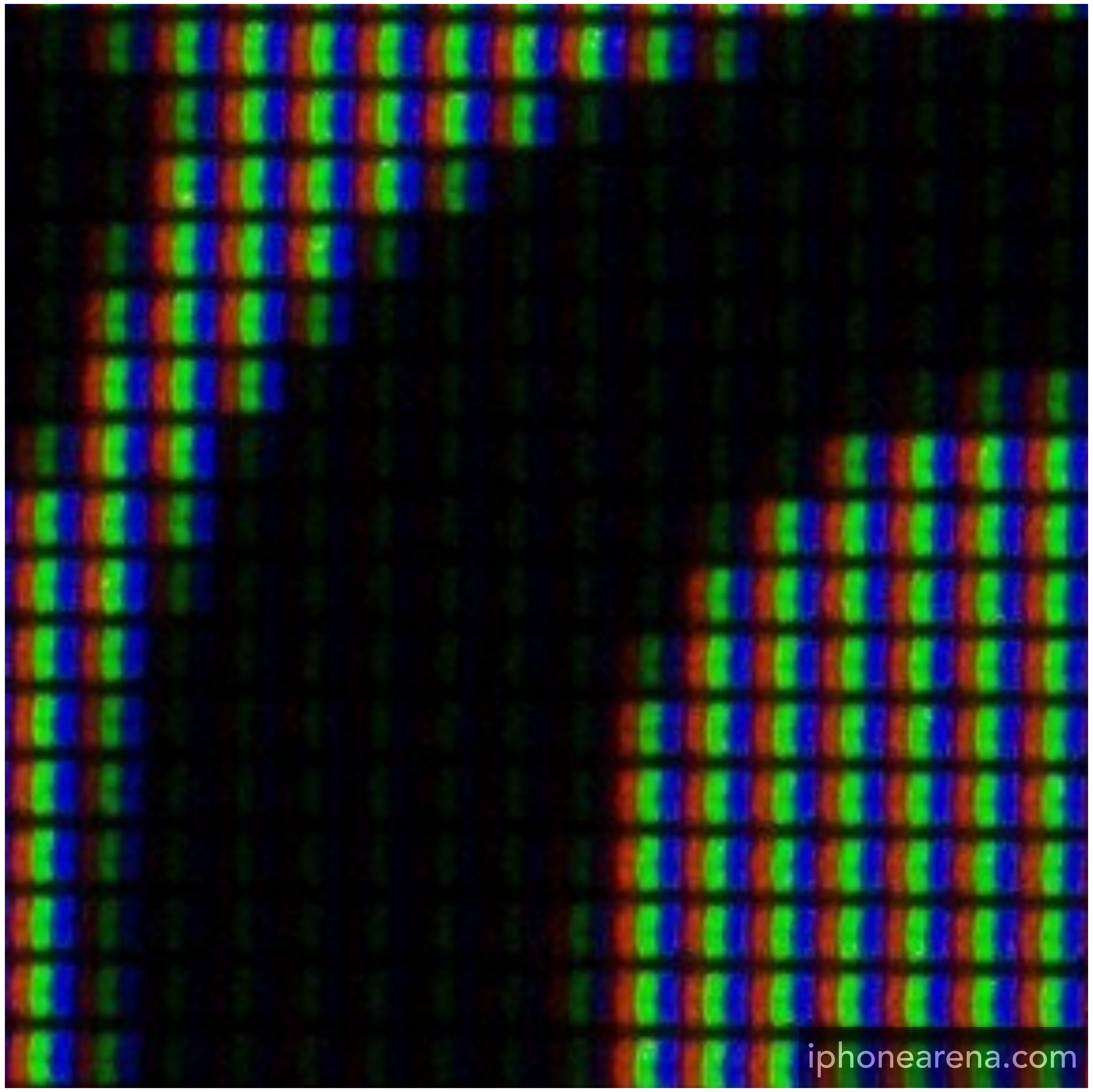
Flat panel displays

Low-Res LCD Display

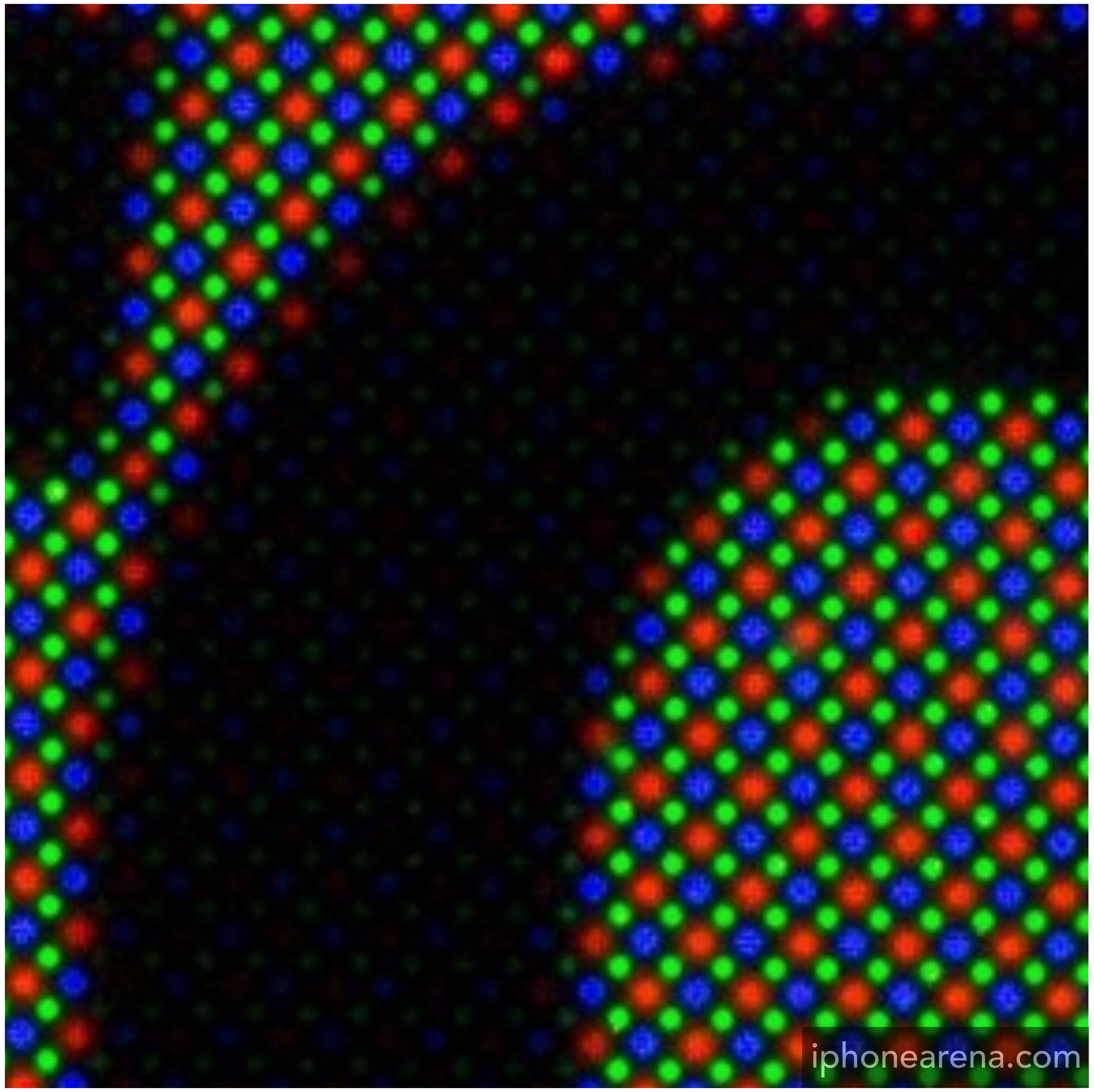
High resolution color LCD, OLED, …

Close up photo of pixels on a modern display

LCD screen pixels (closeup)



iPhone 6S



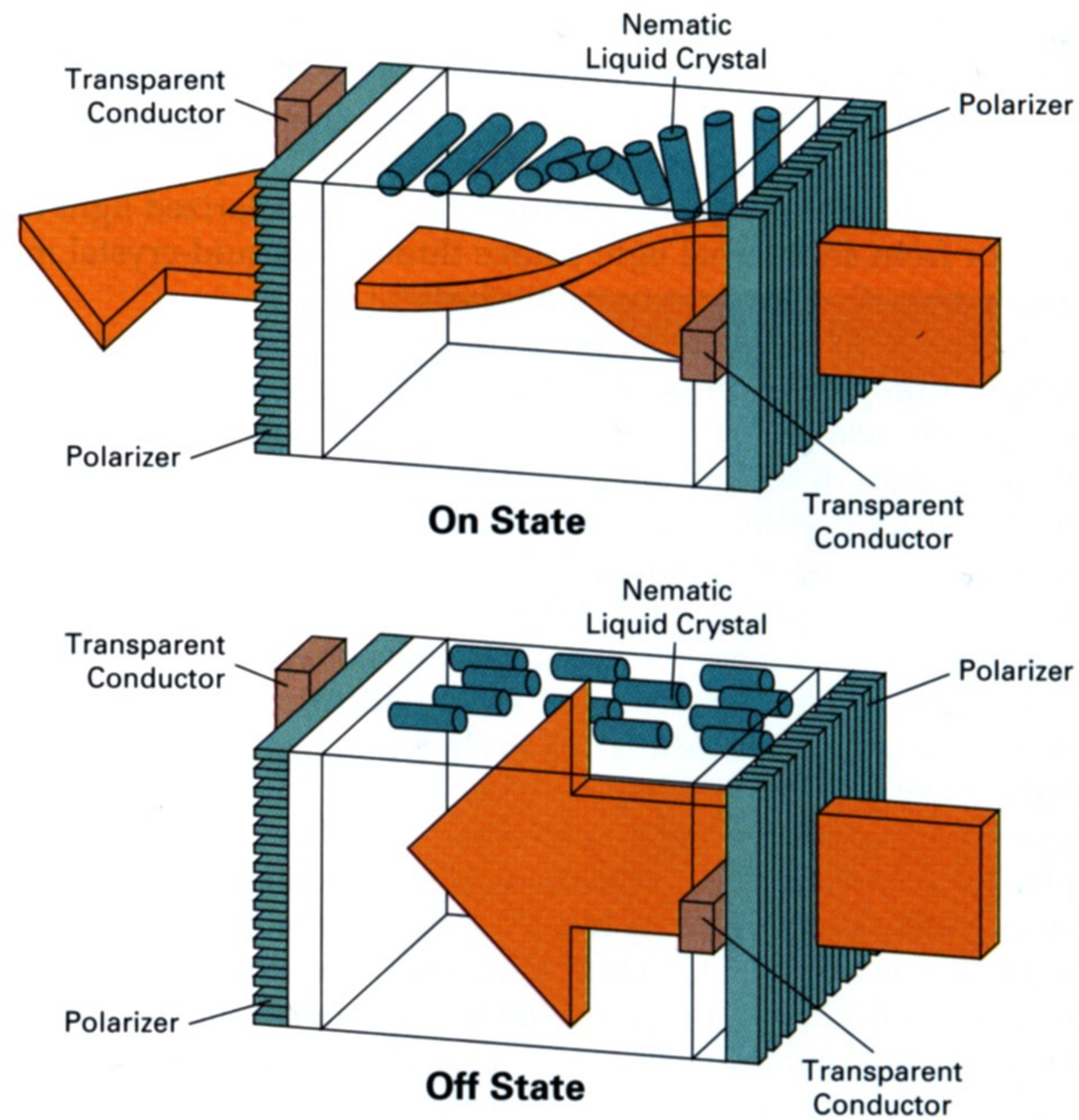
Galaxy S5

LCD (liquid crystal display) pixel

- Principle: block or transmit light by twisting polarization

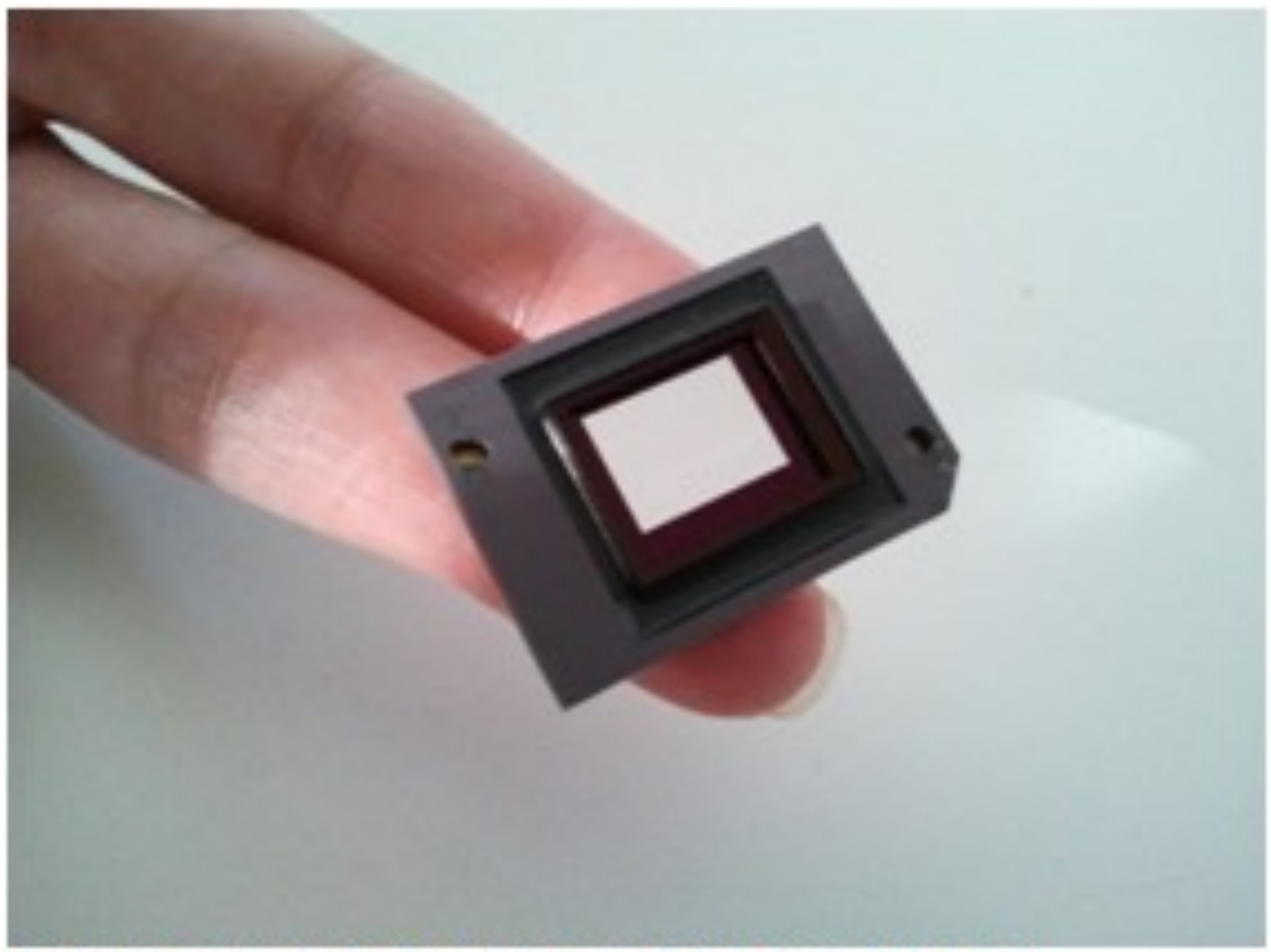
- Illumination from backlight (e.g. fluorescent or LED)

- Intermediate intensity levels by partial twist

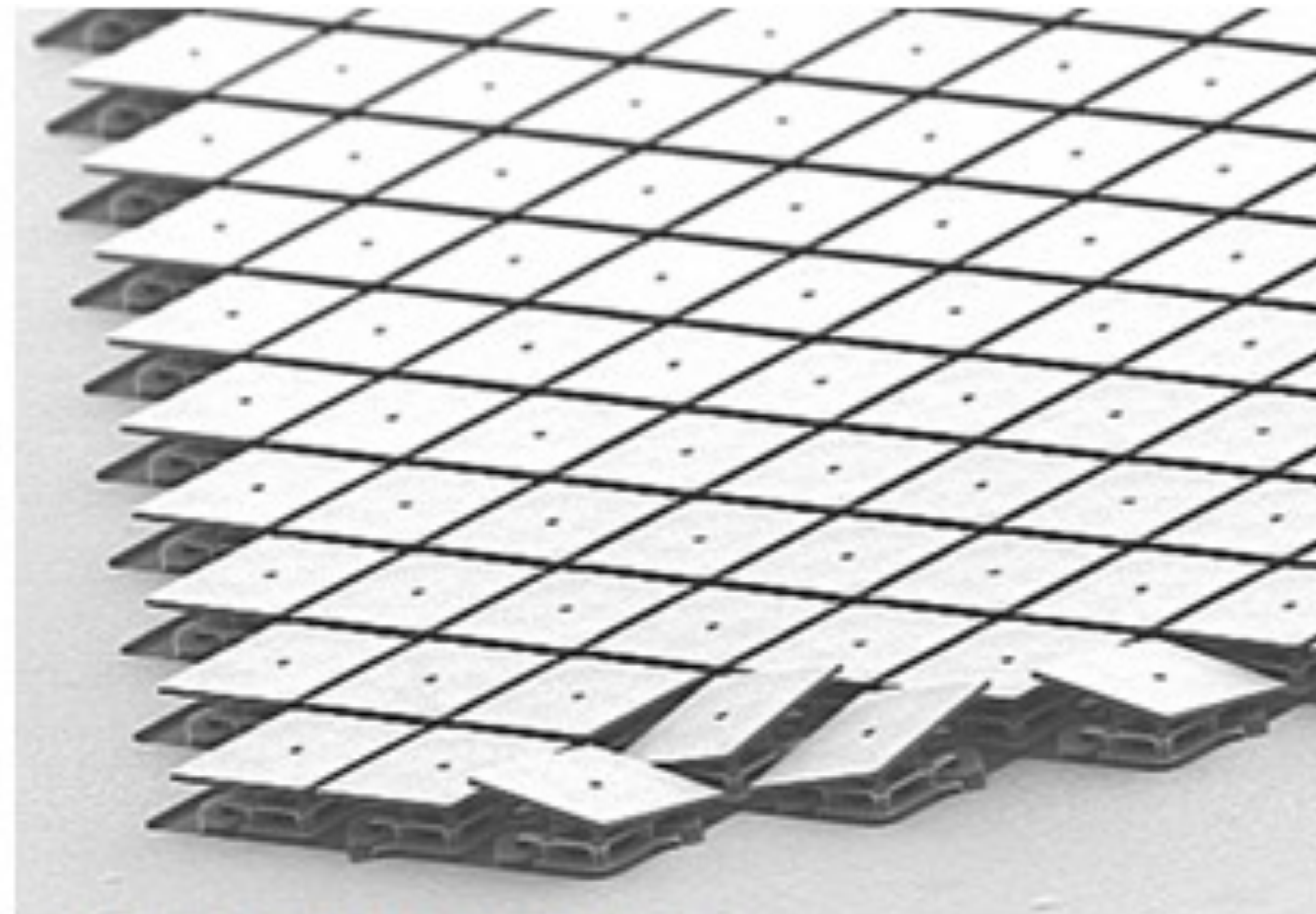


[Image credit: H&B fig. 2-16]

DMD projection display



DIGITAL MICRO MIRROR DEVICE **(DMD)**
(**SLM** - Spatial Light Modulator)



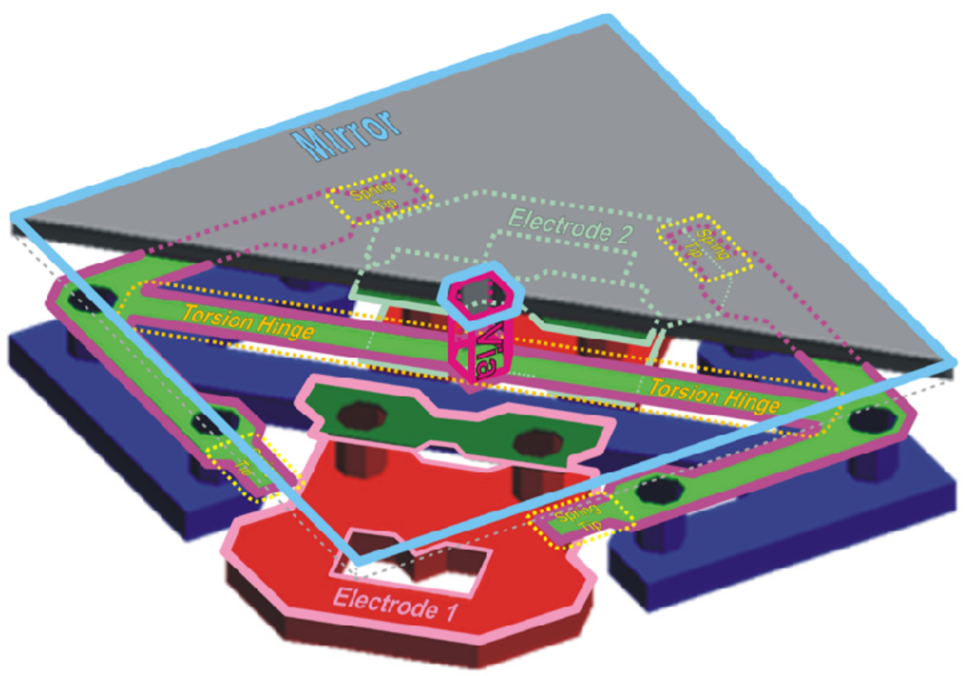
MICRO MIRRORS CLOSE UP

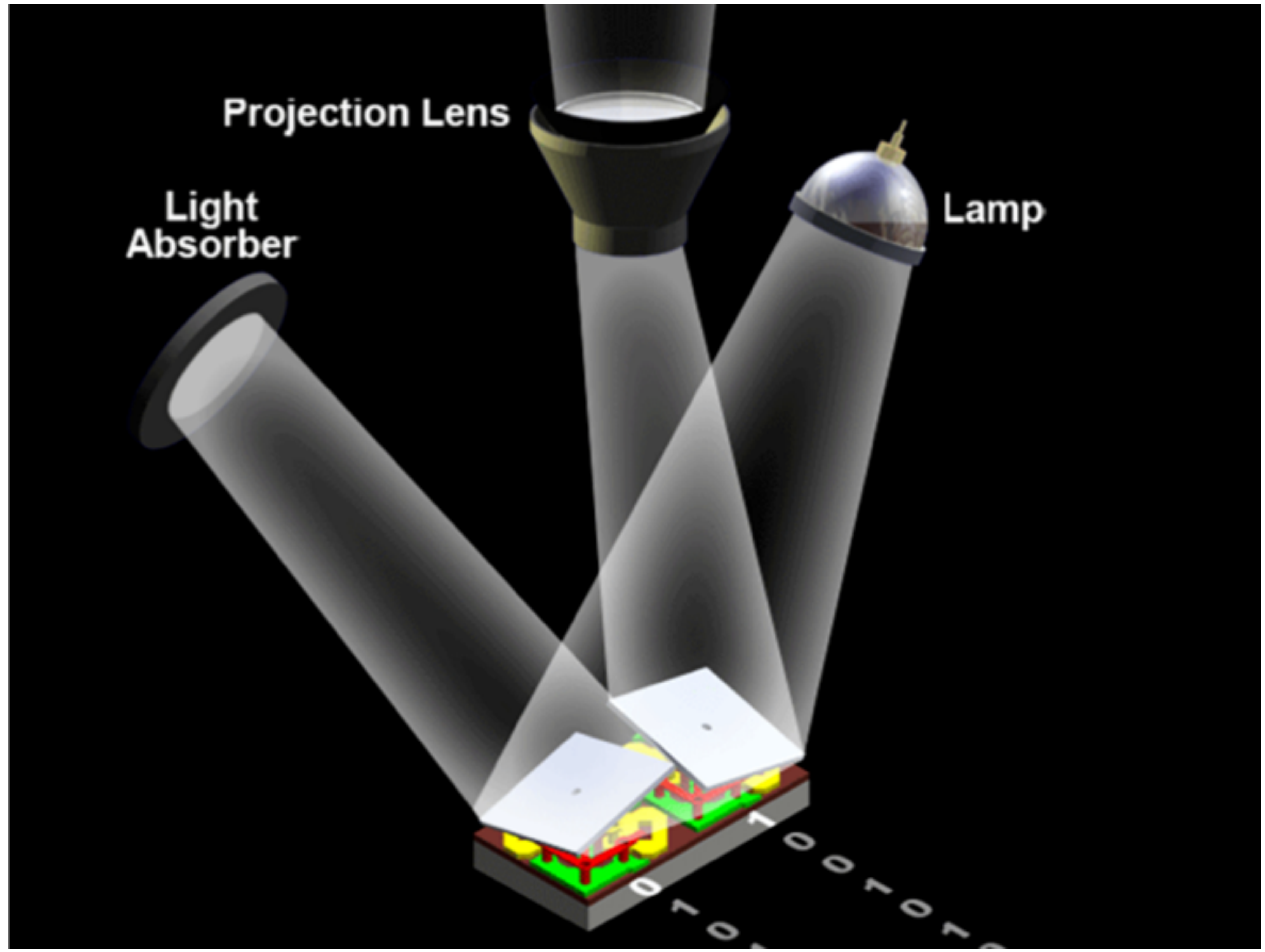
[Y.K. Rabinowitz; EKB Technologies

Array of micro-mirror pixels

DMD = Digital micro-mirror device

DMD projection display





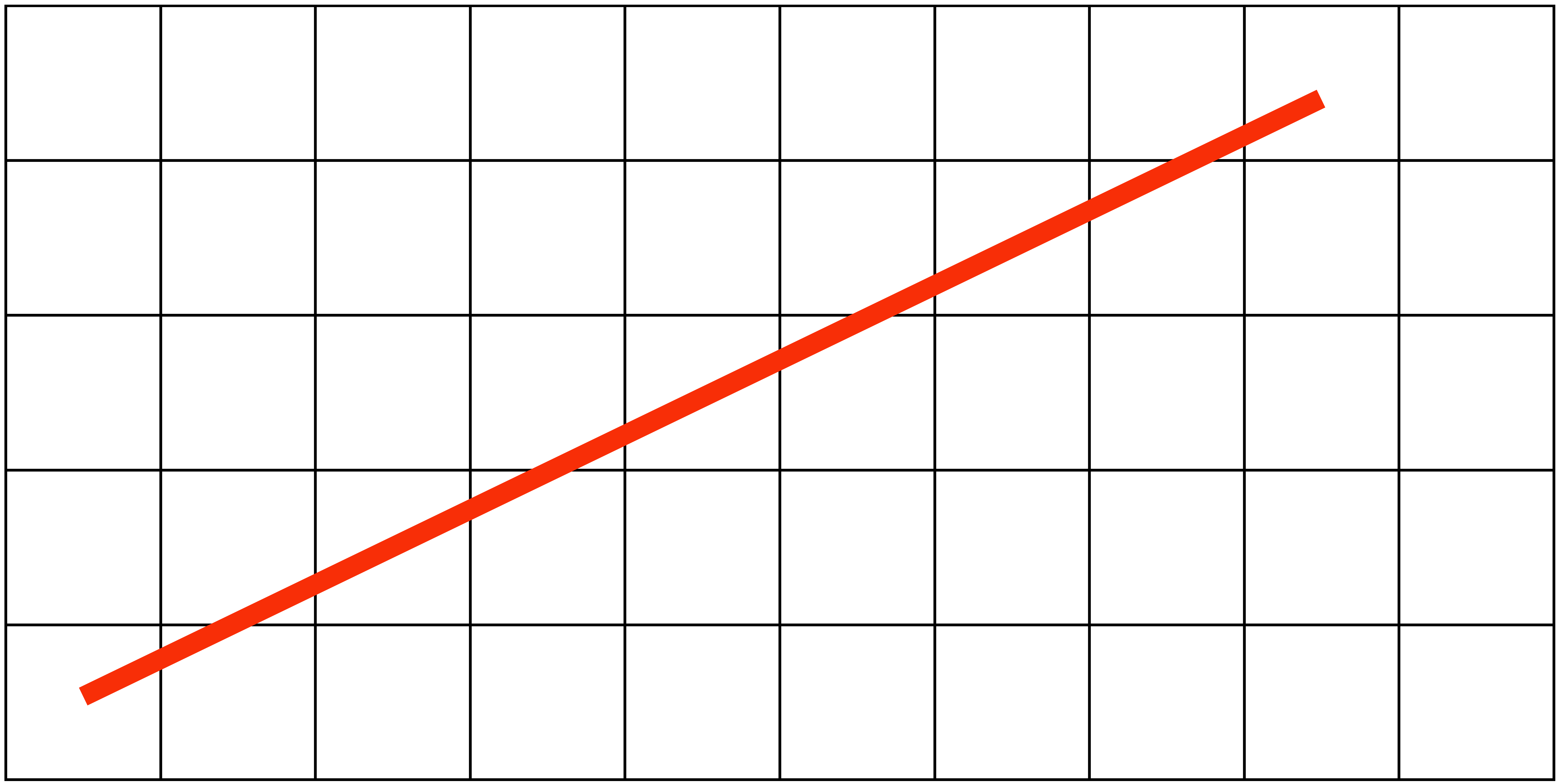
[Texas Instruments]

Array of micro-mirror pixels

DMD = Digital micro-mirror device

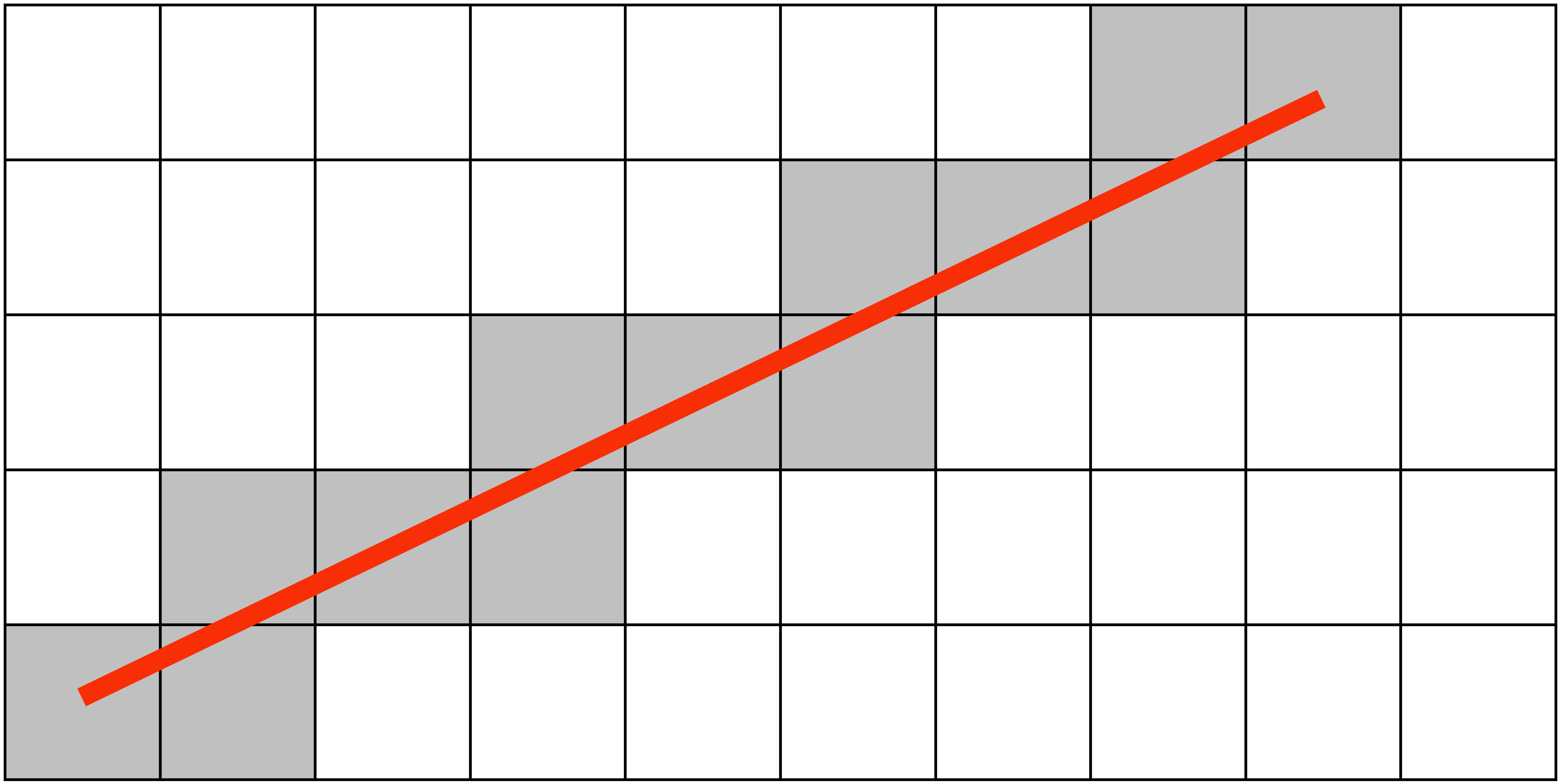
What pixels should we color in to depict a line?

"Rasterization": process of converting a continuous object (a line, a polygon, etc.) to a discrete representation on a "raster" grid (pixel grid)



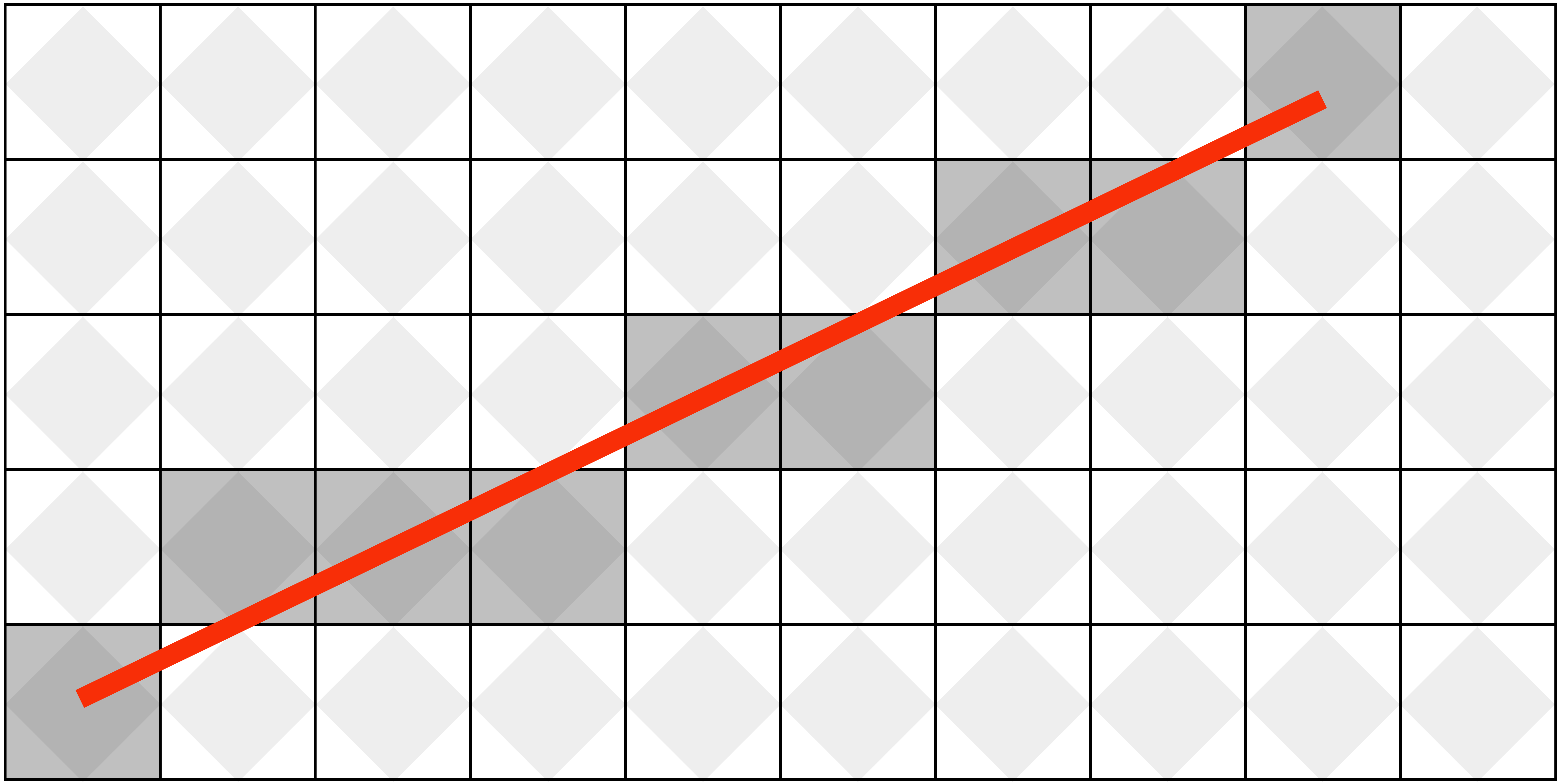
What pixels should we color in to depict a line?

Light up all pixels intersected by the line?



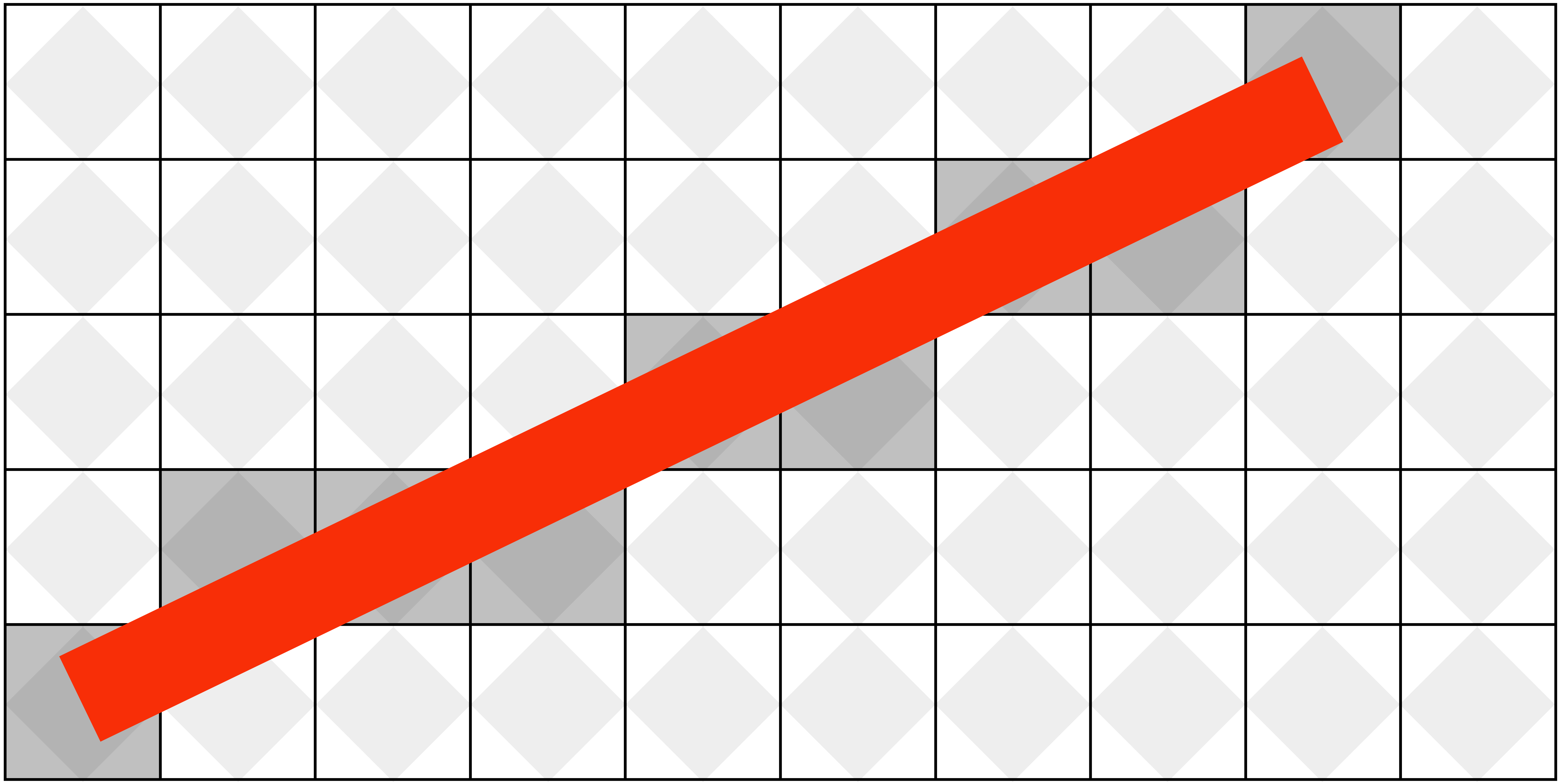
What pixels should we color in to depict a line?

Diamond rule (used by modern GPUs):
light up pixel if line passes through associated diamond



What pixels should we color in to depict a line?

Is there a right answer?
(consider a drawing a "line" with thickness)



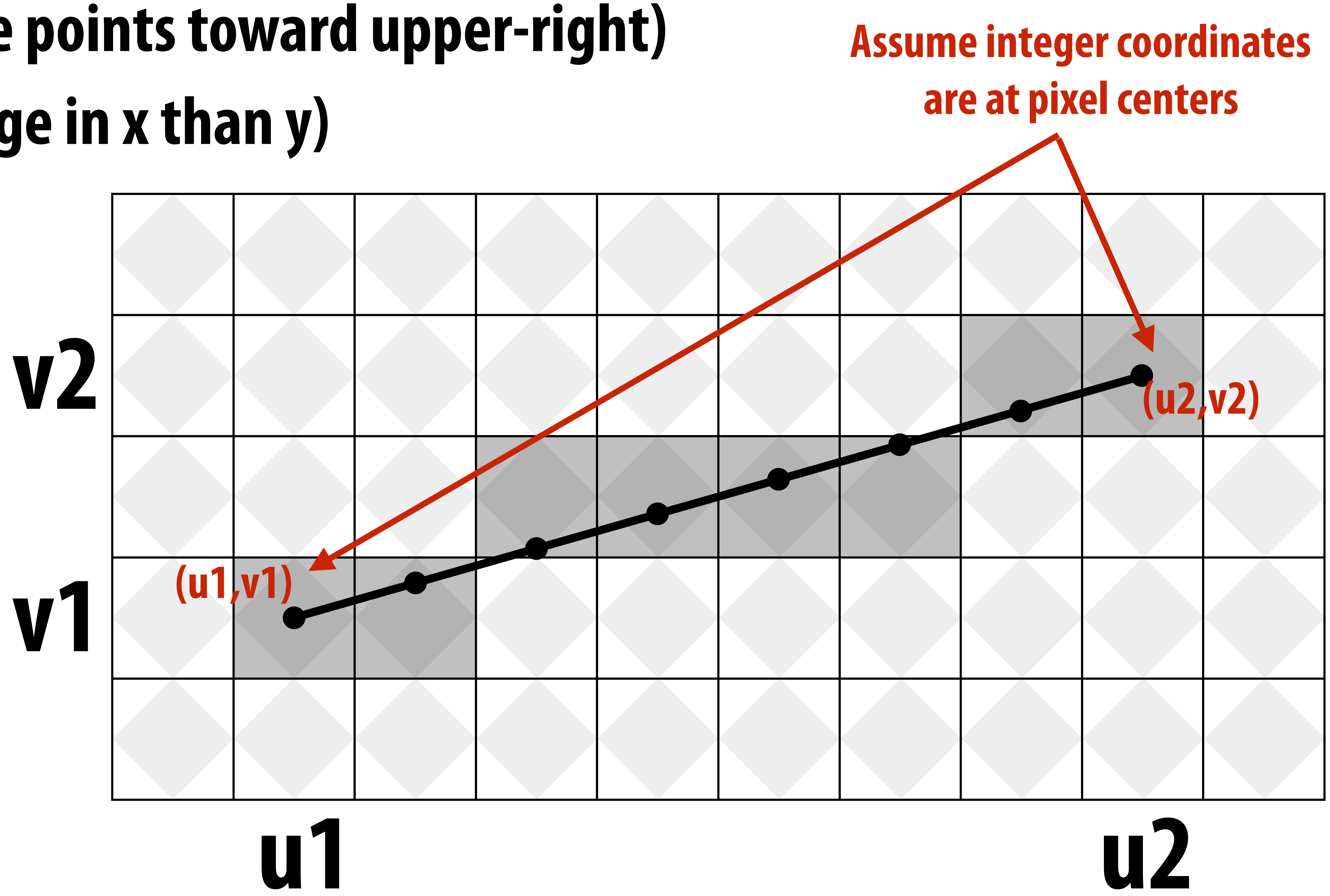
How do we find the pixels satisfying a chosen rasterization rule?

- Could check every single pixel in the image to see if it meets the condition...
  - $O(n^2)$ pixels in image vs. at most $O(n)$ “lit up” pixels
  - *must* be able to do better! (e.g., seek algorithm that does work proportional to number of pixels in the drawing of the line)

Incremental line rasterization

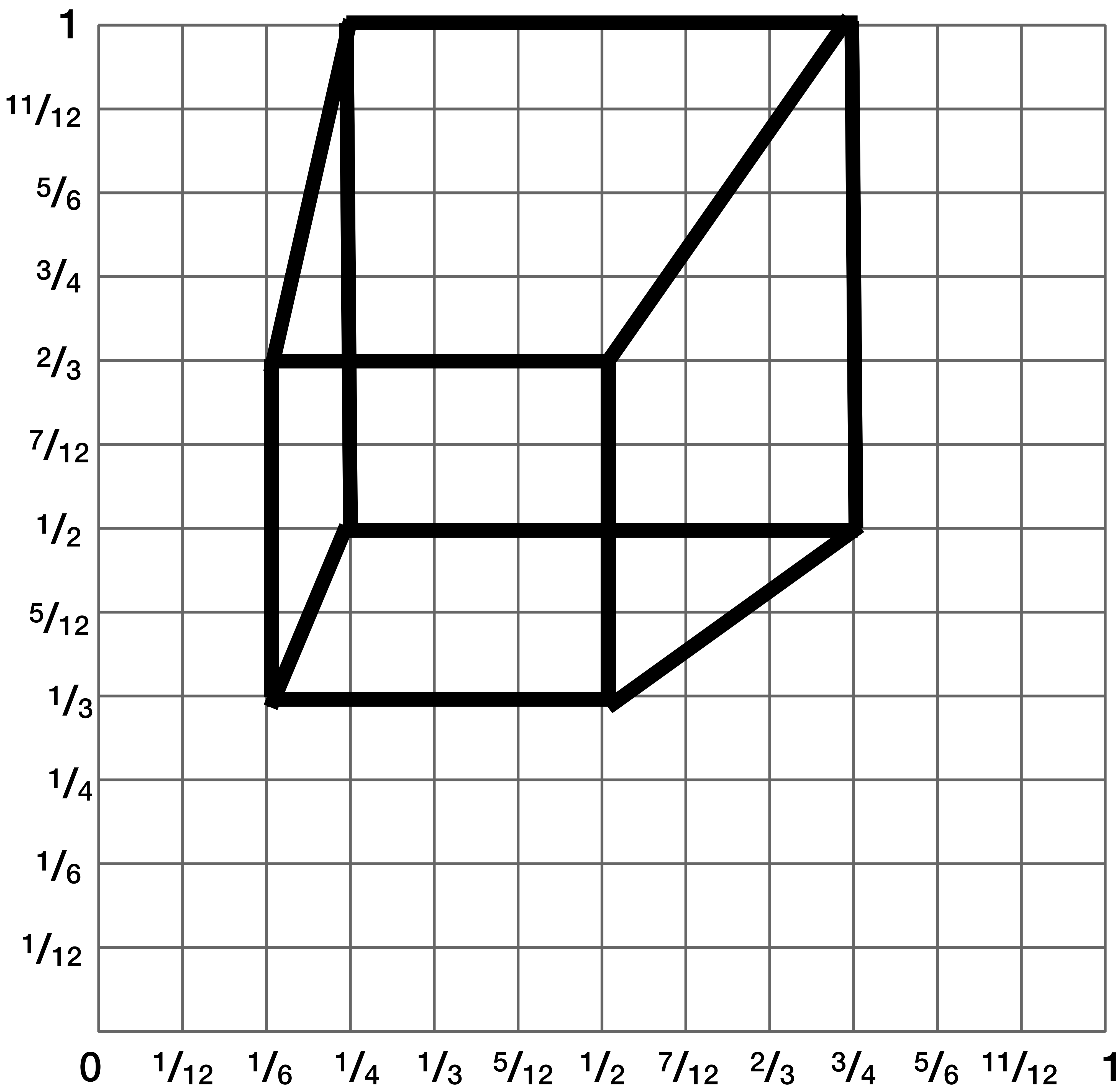
- Let's say a line is represented with integer endpoints: (u1,v1), (u2,v2)
- Slope of line: s = (v2-v1) / (u2-u1)
- Consider an easy special case:
  - u1 < u2, v1 < v2 (line points toward upper-right)
  - 0 < s < 1 (more change in x than y)

```
v = v1;
for( u=u1; u<=u2; u++ )
{
   v += s;
   draw( u, round(v) )
}
```



Common optimization: rewrite algorithm to use only integer arithmetic (Bresenham algorithm)

Line drawing of cube



2D coordinates:

A: (1/4, 1/2)
B: (3/4, 1/2)

C: (1/4, 1)
D: (3/4, 1)

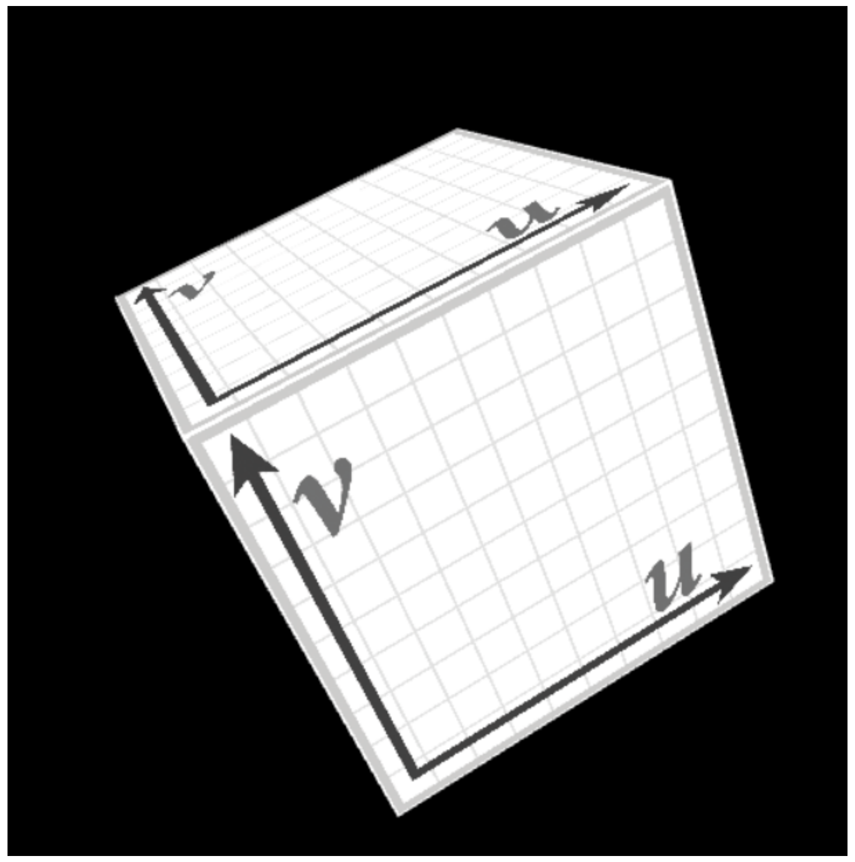
E: (1/6, 1/3)
F: (1/2, 1/3)

G: (1/6, 2/3)
H: (1/2, 2/3)

*** keep in mind, this image is mirrored since we simulated the result of pinhole projection**

We just rendered a simple line drawing of a cube.

But to render more realistic pictures (or animations) we need a much richer model of the world.

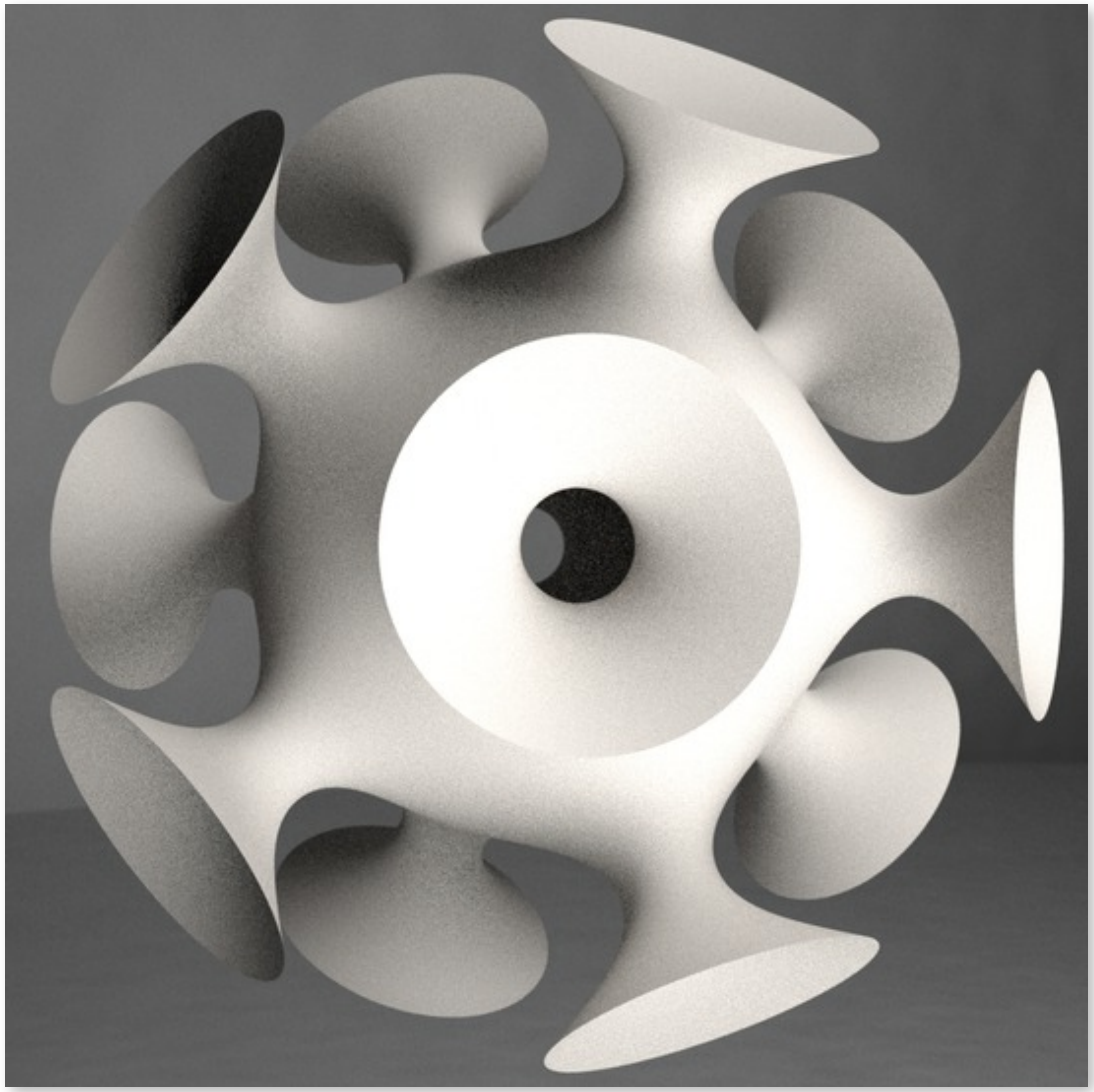


surfaces
motion
materials
lights
cameras

2D shapes

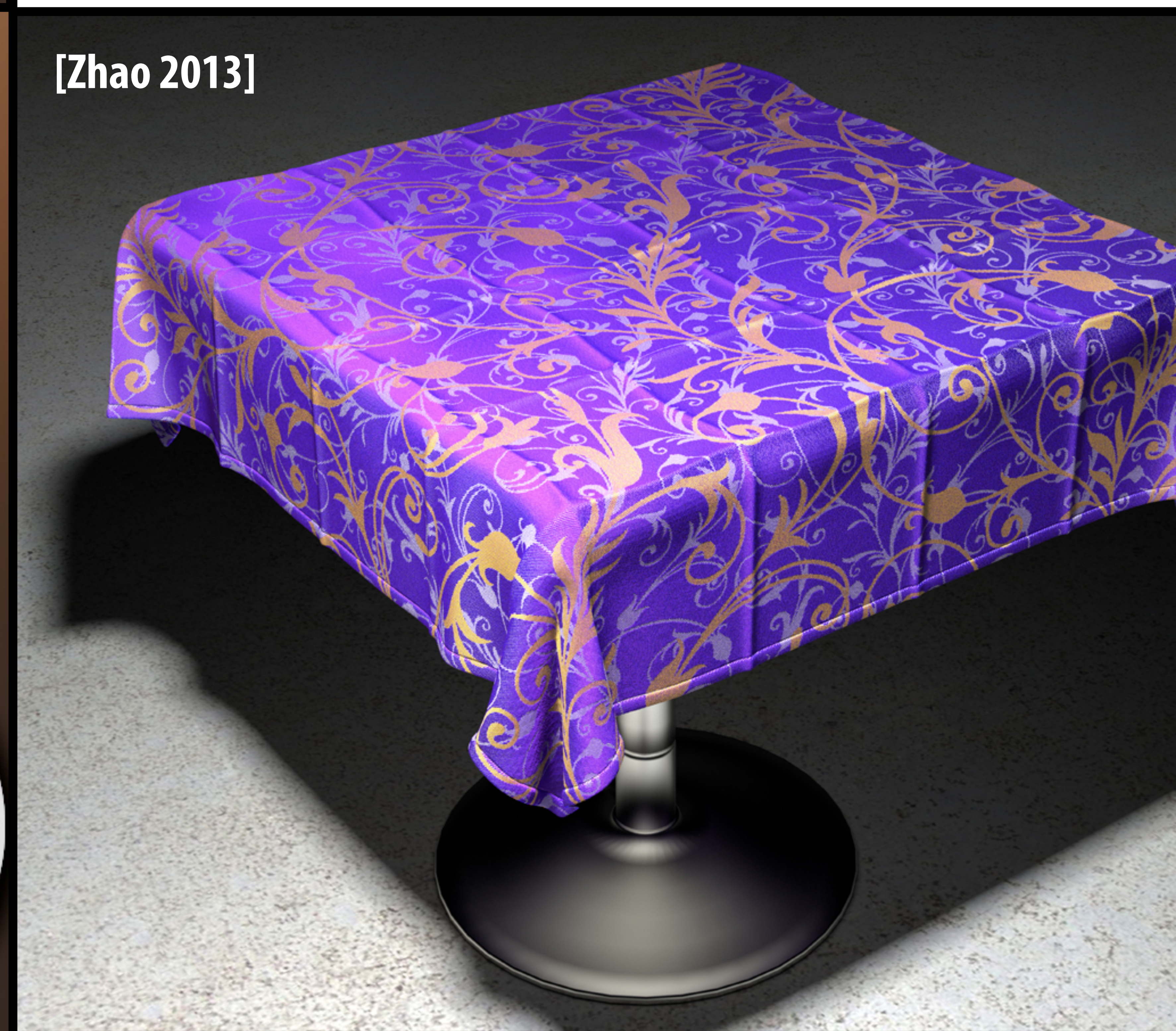
[Source: Batra 2015]

Complex 3D surfaces



Platonic noid

Modeling material properties

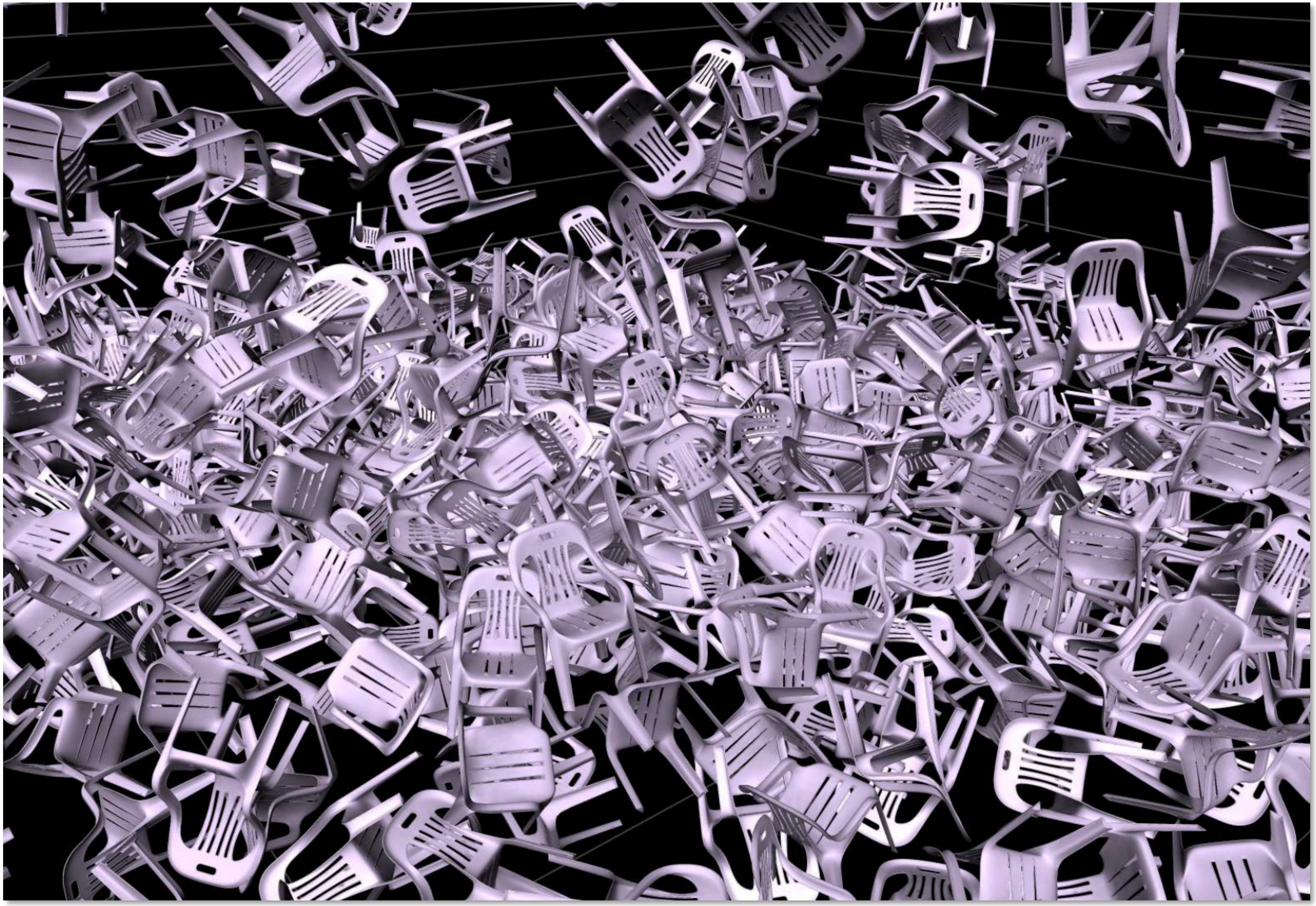


Realistic lighting environments

Animation: modeling motion

https://www.youtube.com/watch?v=6G3O60o5U7w

Physically-based simulation of motion



https://www.youtube.com/watch?v=tT81VPk_ukU

[James 2004]

Course Logistics

About this course

- **A broad overview of major topics and techniques in interactive computer graphics: geometry, rendering, animation, imaging**

- **Learn by implementing:**
  - **Focus on implementing fundamental data structures and algorithms that are reused across all areas of graphics**

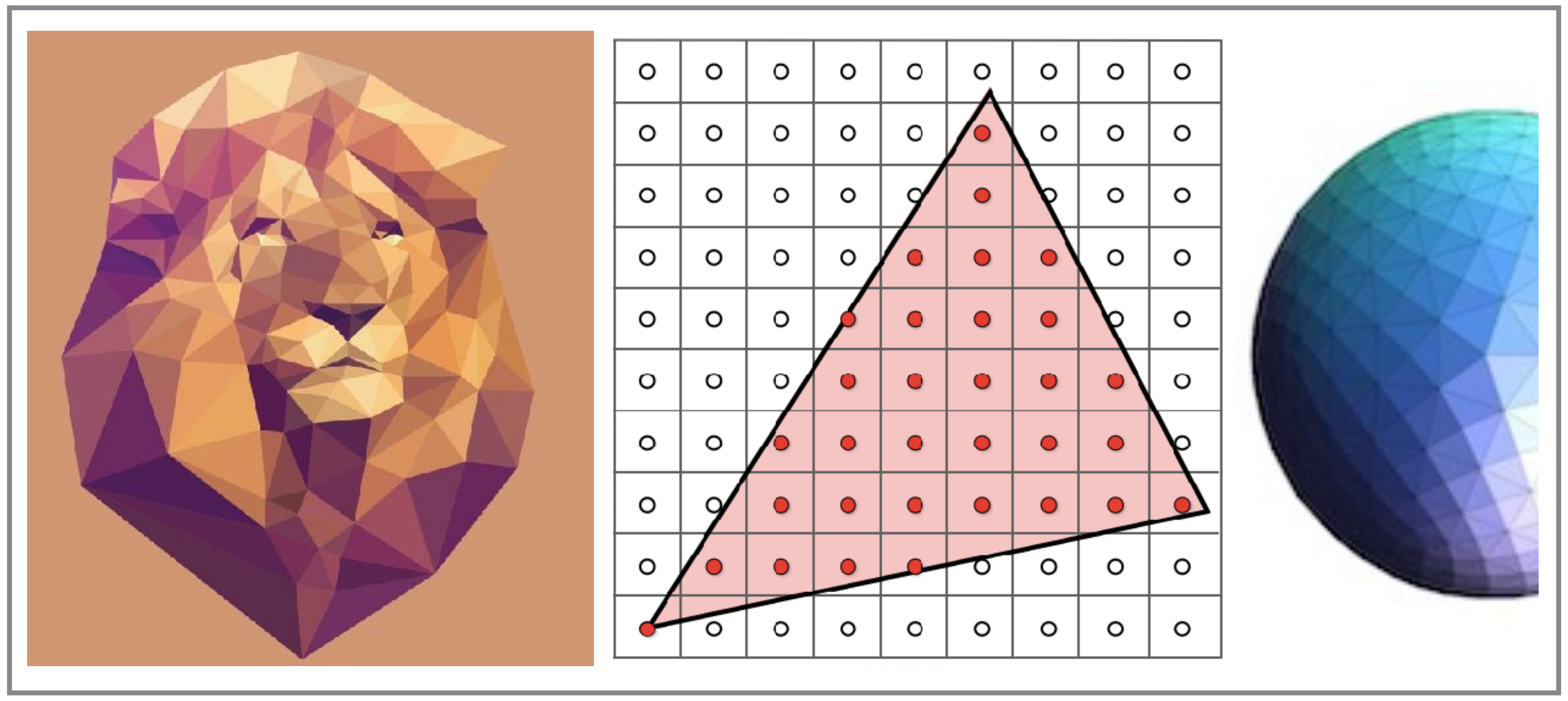
Getting started

- Sign up for an account on the course web site
  - http://cs248.stanford.edu

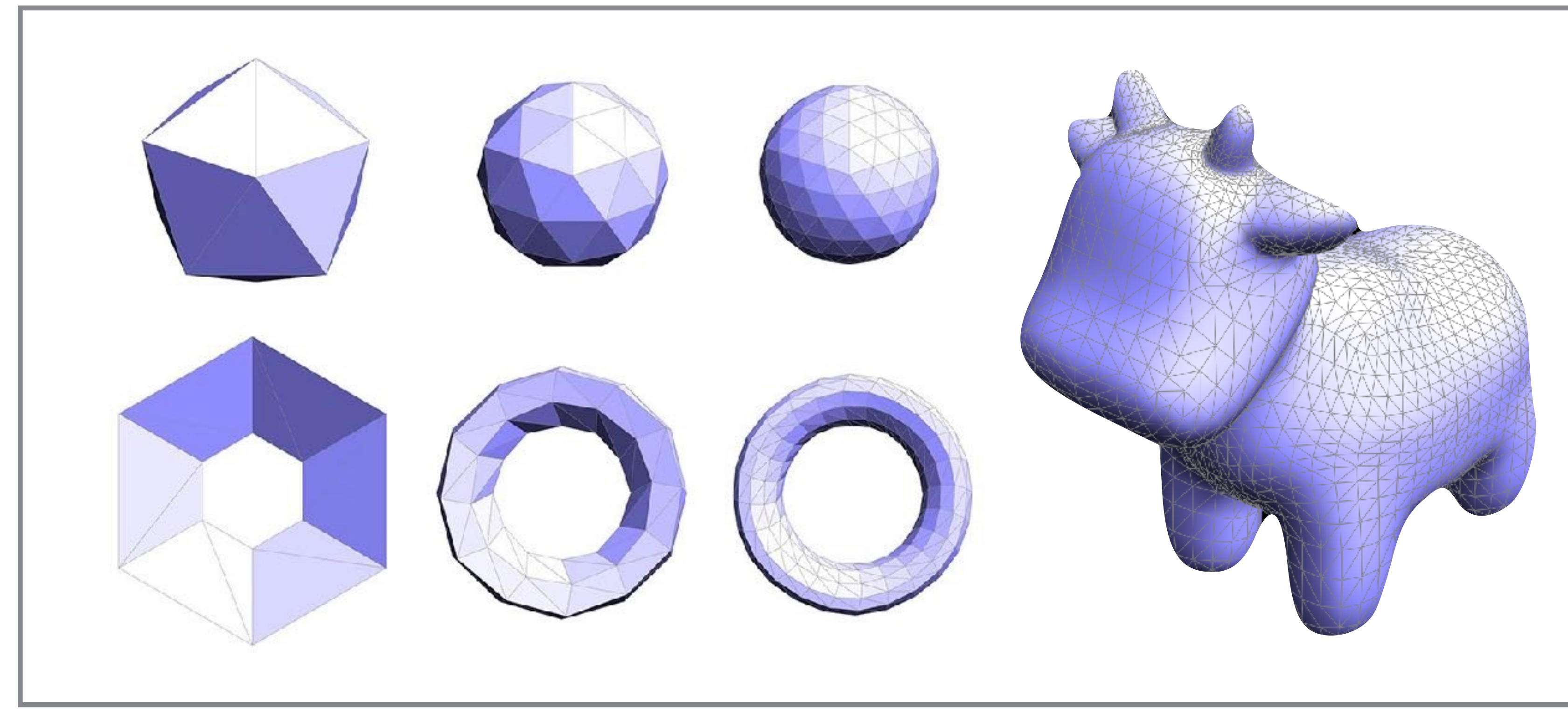
- Sign up for the course on Piazza
  - http://piazza.com/stanford/winter2020/cs248

- There is no textbook for this course, but please see the course website for references (there are some excellent graphics textbooks)

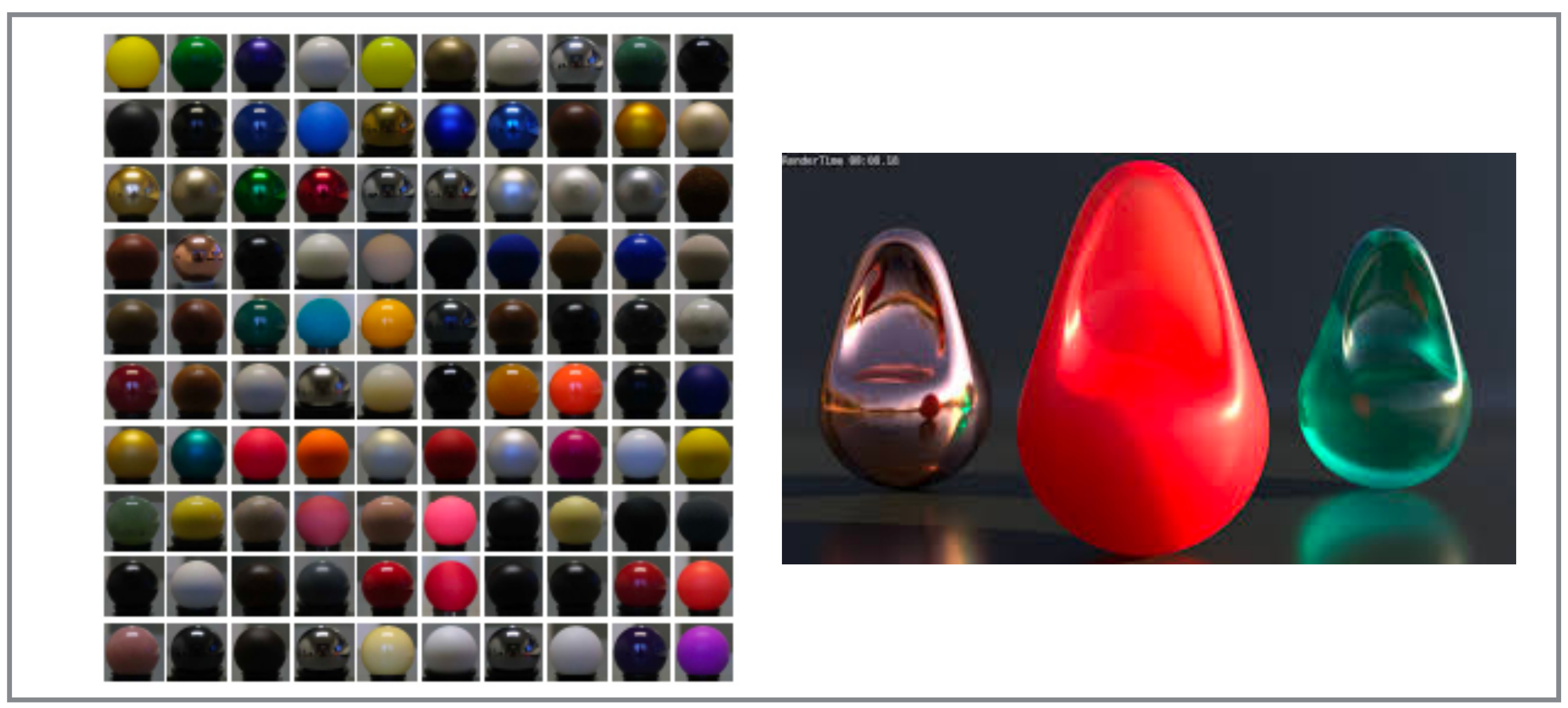
Course programming assignments



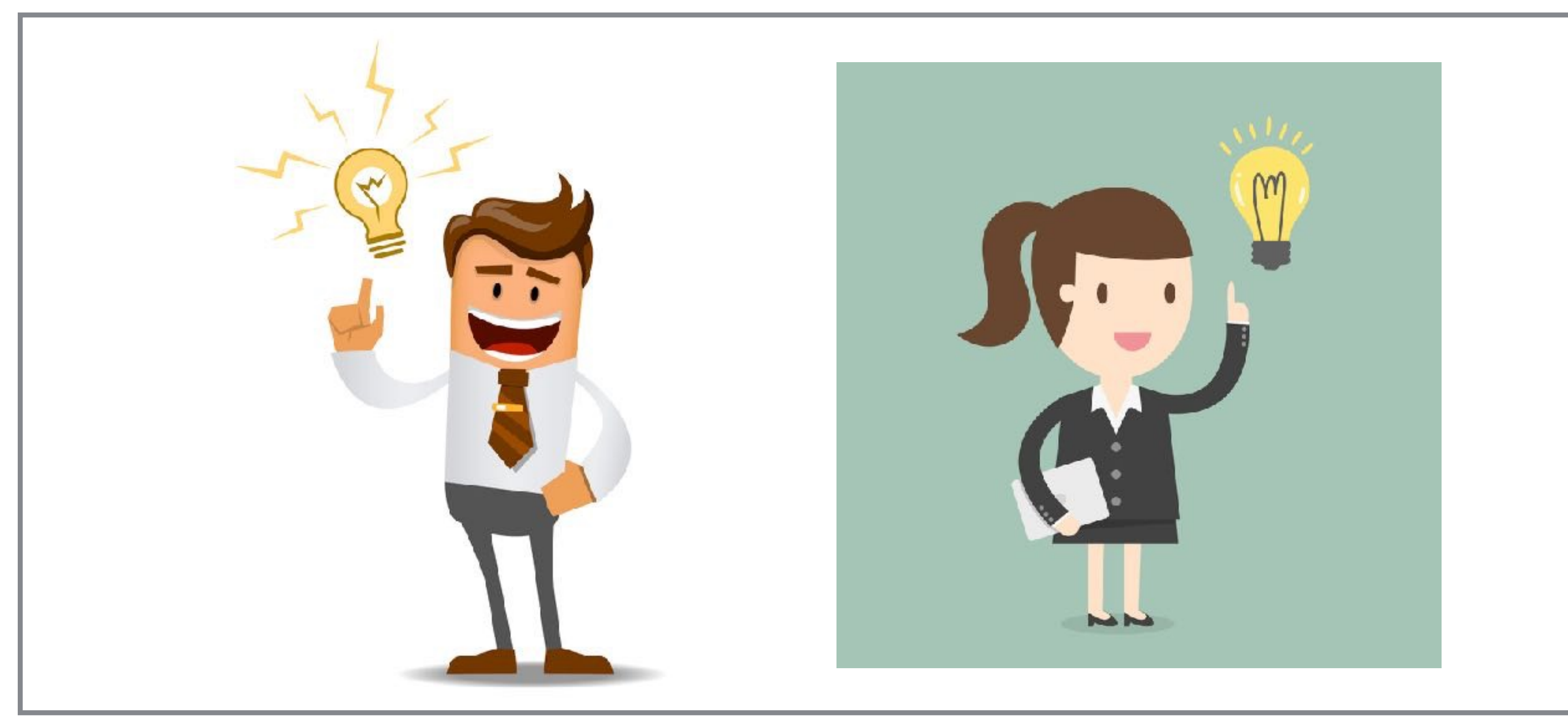
1. 2D drawing (2 weeks)



2. Geometry editing (2 weeks)



3. Materials and lighting in a 3D renderer (2 weeks)



4. Self-selected project
extend existing project, take on optional animation project, choose your own
(~3 weeks)

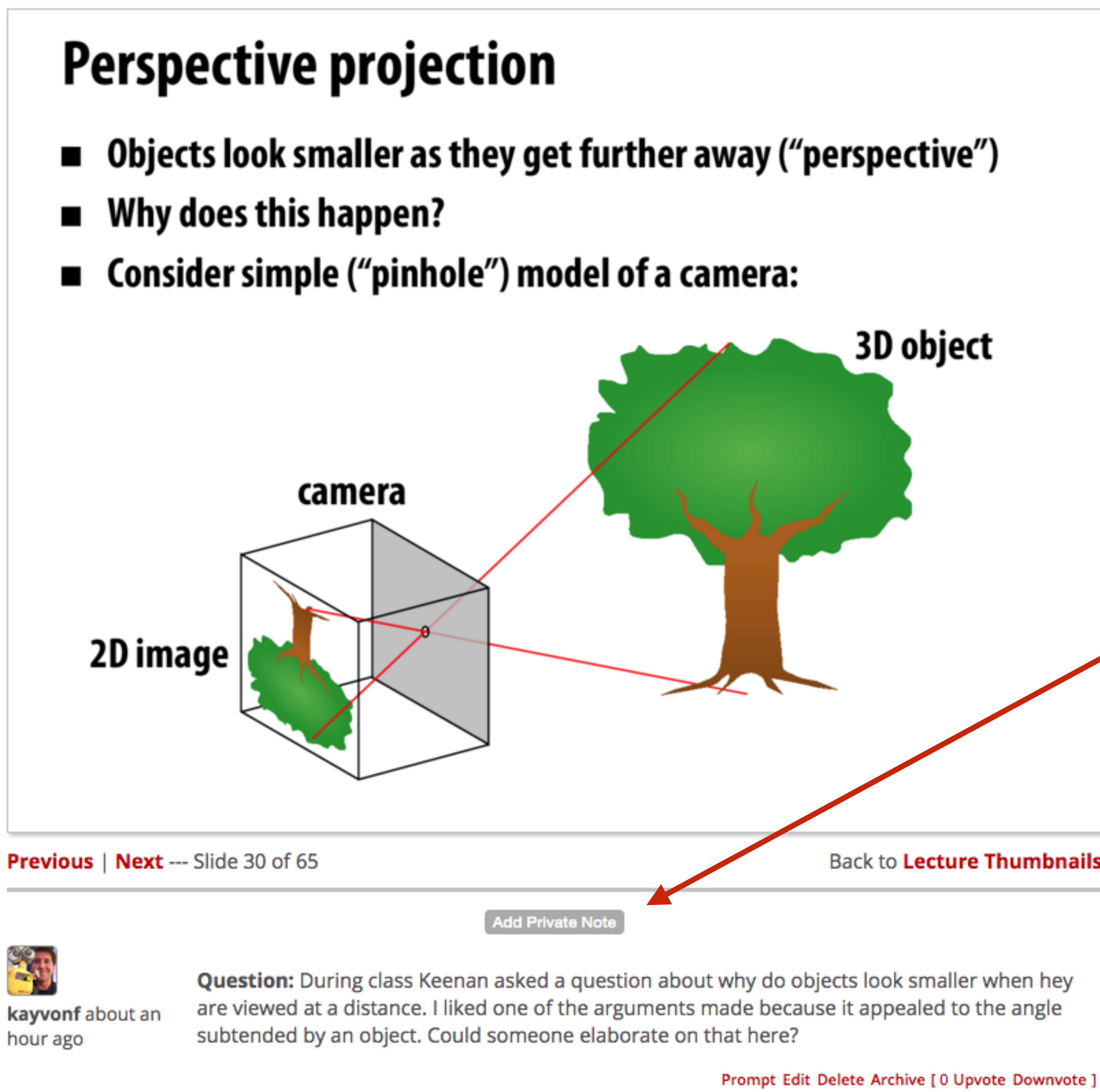
Man Designed by Alekksall / Freepik. Woman designed by Dooder / Freepik

Assignments / Grading

- **(40%) Three programming assignments**
  - In teams of up to two students (yes, you can work alone if you wish)
- **(10%) Written exercises (weekly)**
  - Released on Tues night, due Friday 10am (starting next week)
  - Graded on participation only
- **(25%) Final exam**
  - Given in week 9 of the course
- **(25%) Self-selected final project**
  - Extend an earlier assignment, or do your own thing!

The course web site

We have no textbook for this class and so the lecture slides and instructor/TA/ student discussions on the web are the primary course reference



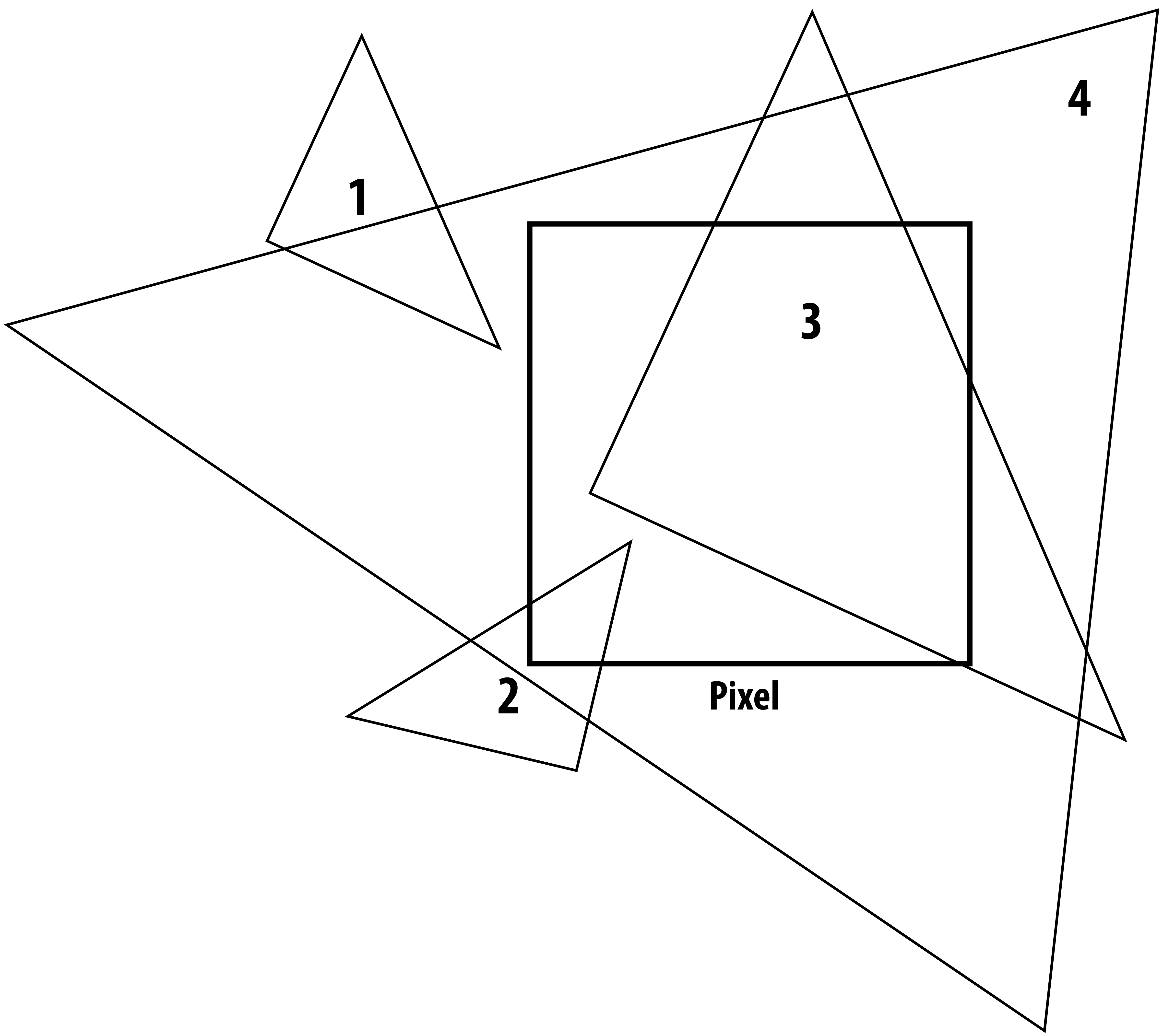
"Add private note" button:

You can add notes to yourself about this slide here.

Slide comments and discussion

Thought question for next time:
What does it mean for a pixel to be covered by a triangle?

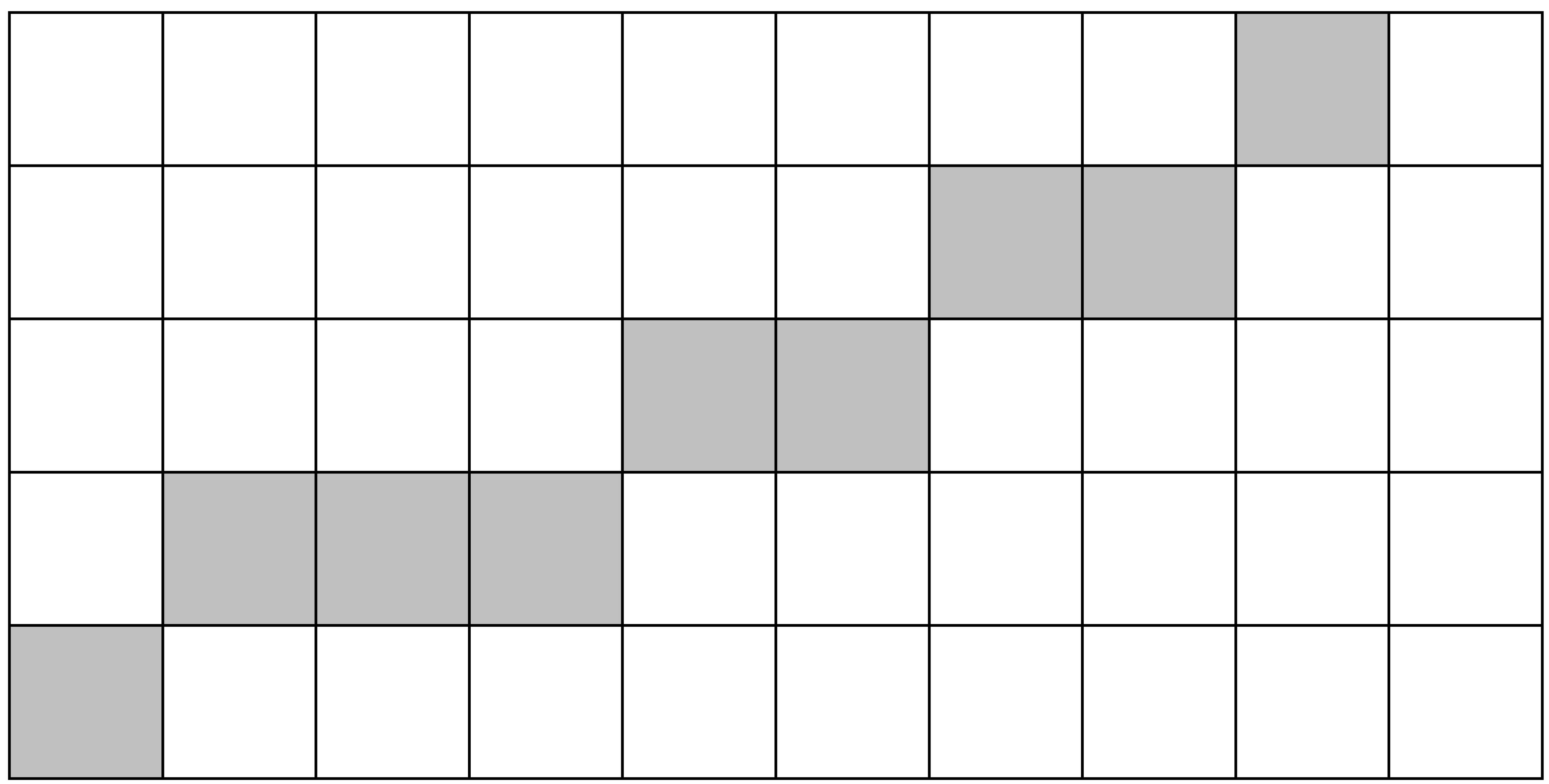
Question: which of these four triangles "cover" this pixel?



See you next time!

Next time, we'll talk about drawing a triangle

- **And it's a lot more interesting than it might seem…**
- **Also, what's up with these "jagged" lines?**



Slide acknowledgements:
Thanks to Keenan Crane and Ren Ng