Lecture 1:

Course Intro:
Welcome to Computer Graphics!

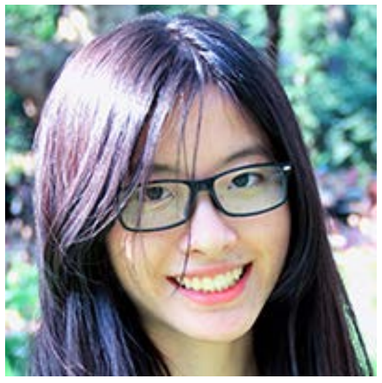
Computer Graphics
CMU 15-462/662

Hi!

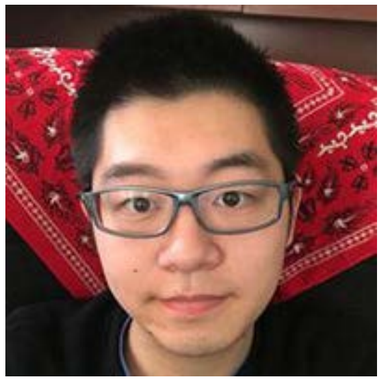
Keenan Crane

Katherine Ye

Adrian Biagioli



Jackie Li

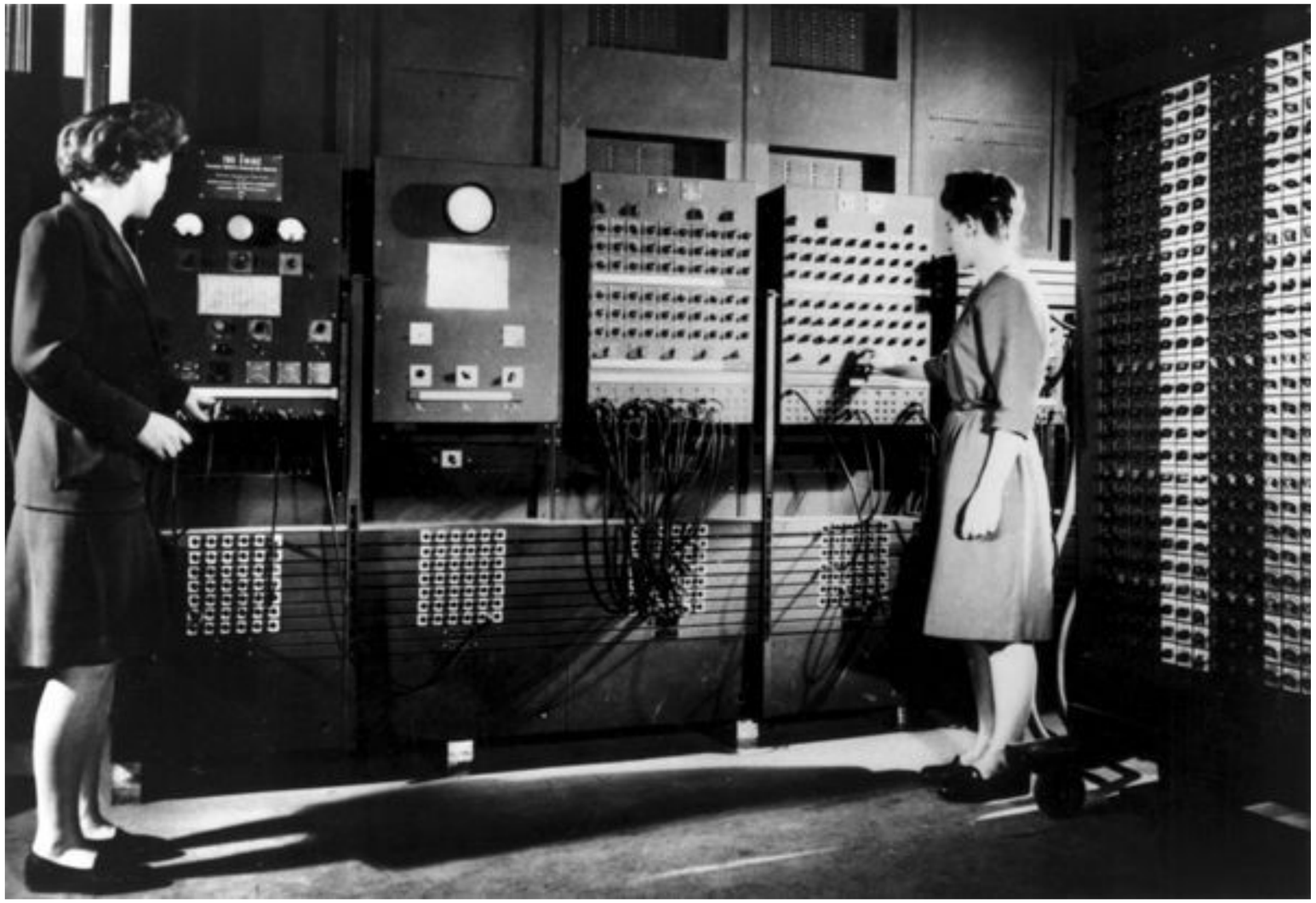


Connor Lin

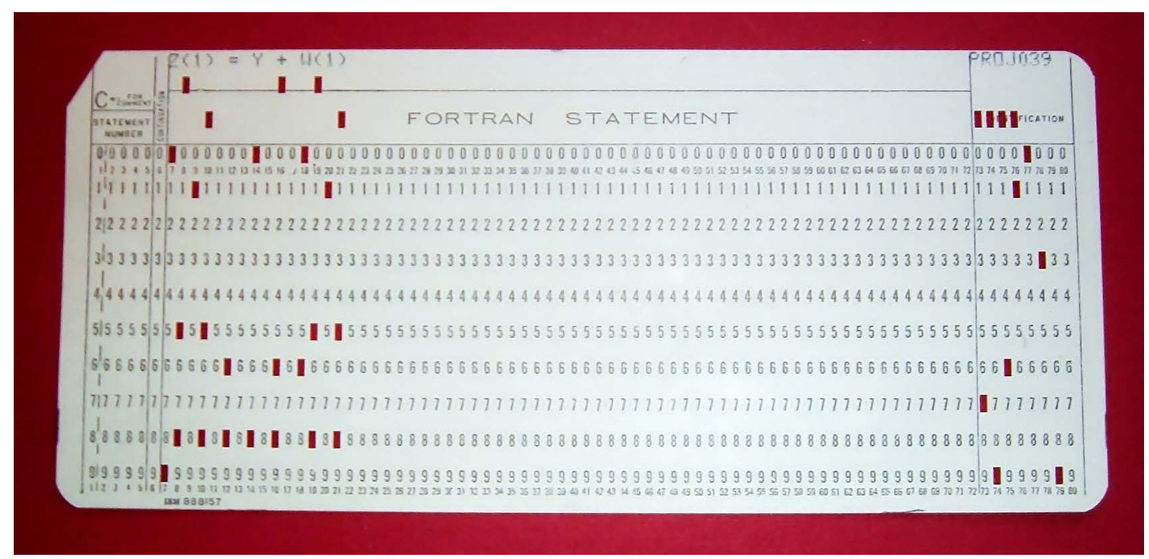
Q: What is computer graphics?

Probably an image like this comes to mind:

Q: …ok, but more fundamentally: what is computer graphics (and why do we need it)?

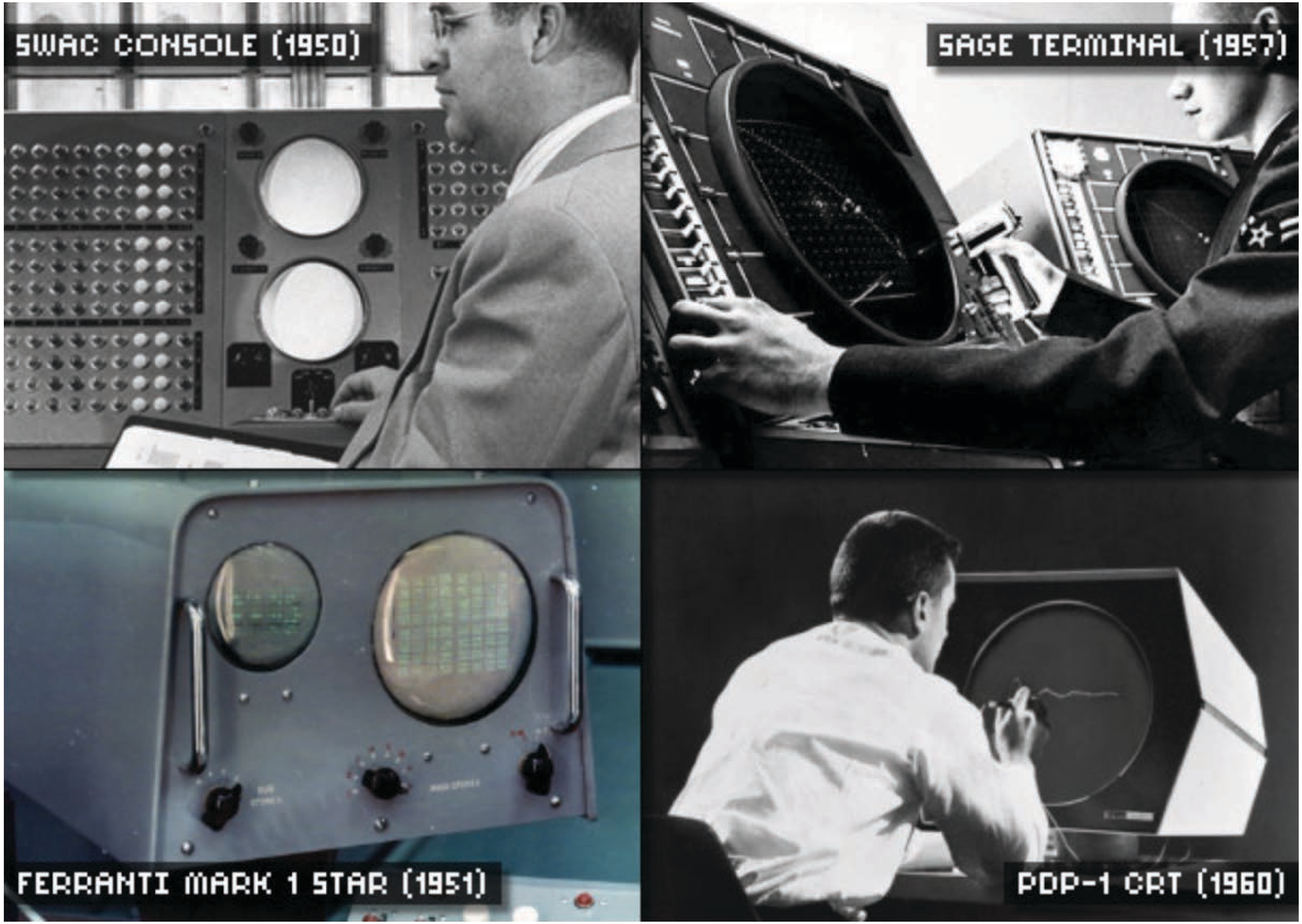


Early computer (ENIAC), 1945



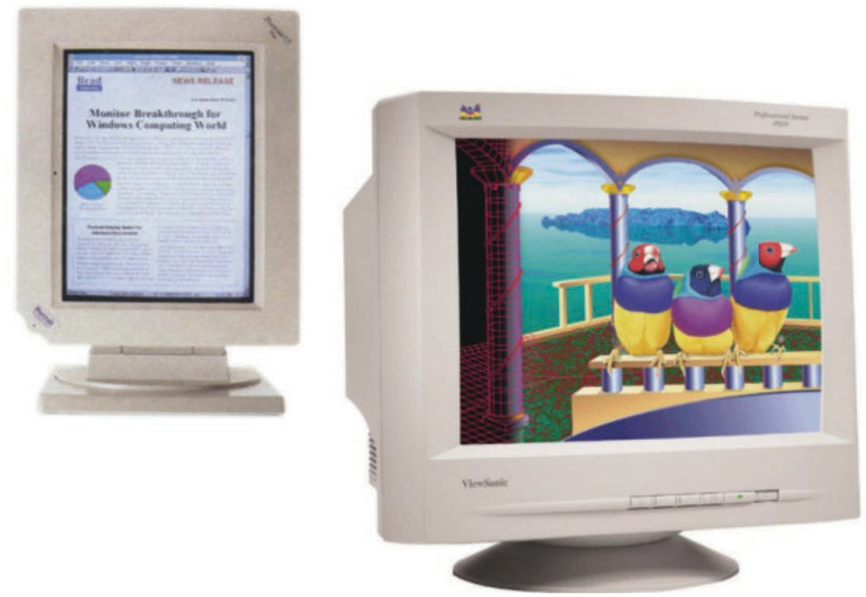
punch card (~120 bytes)

There must be a better way!



Credit: PC World, ***"A Brief History of Computer Displays"***

Sketchpad (Ivan Sutherland, 1963)



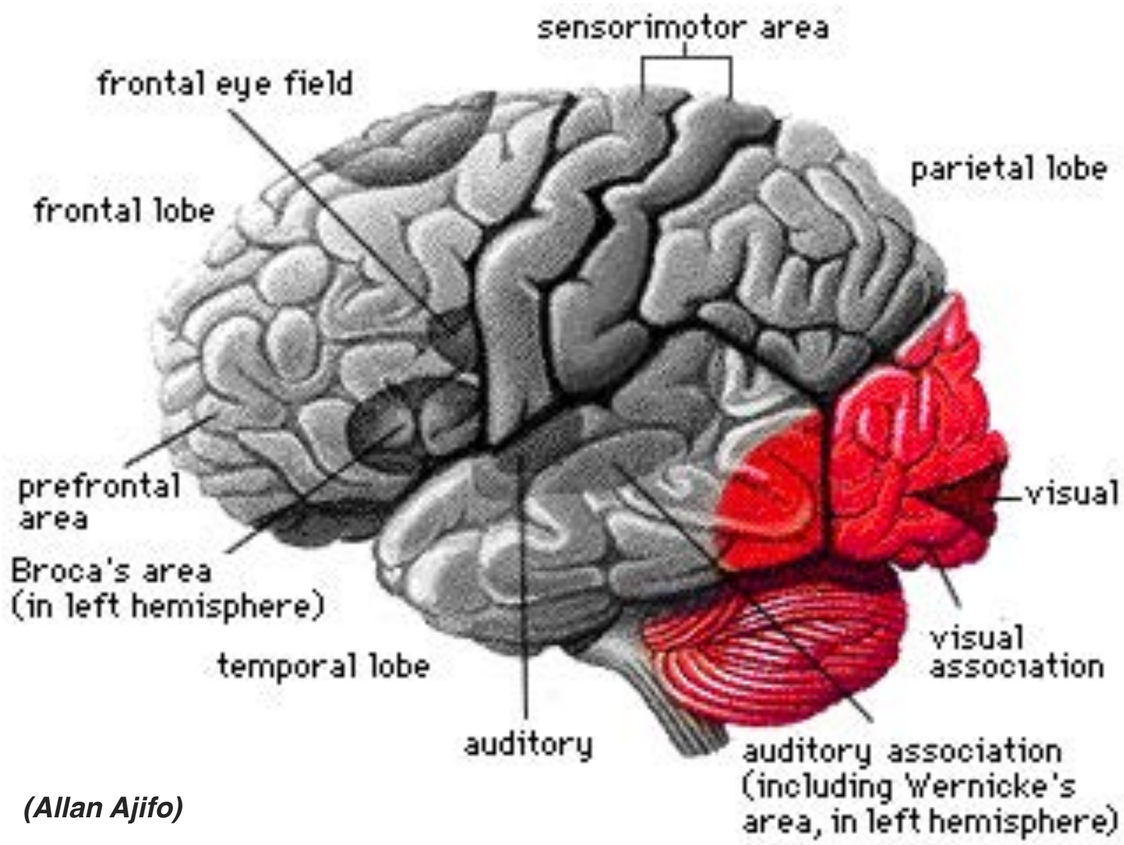
2018: Dell 8k monitor
7680x4320 (~95MB)

Coming down the pipe…

2018 Google/LG display: 2x 4800x3480 @ 1280Hz => 11.2GB/s

Why *visual* information?

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



(Allan Ajifo)

(Petar Milošević)

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

What is computer graphics?

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

digital information

computation

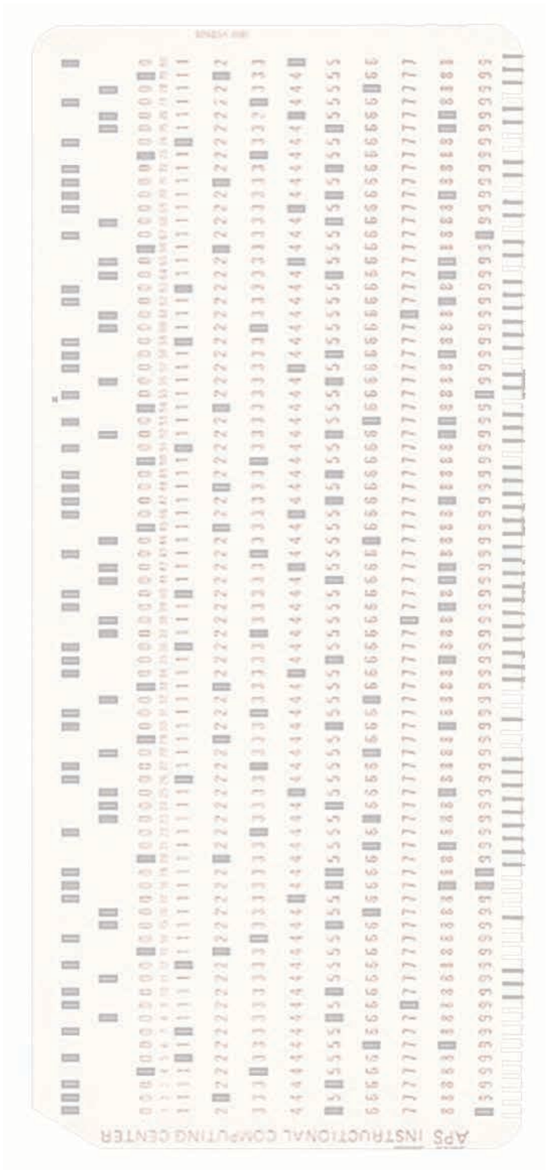
visual information

What is computer graphics?

com•put•er graph•ics /kəmˈpyoodərˈgrafiks/ *n.*

The use of computers to synthesize visual information.

Why only visual?



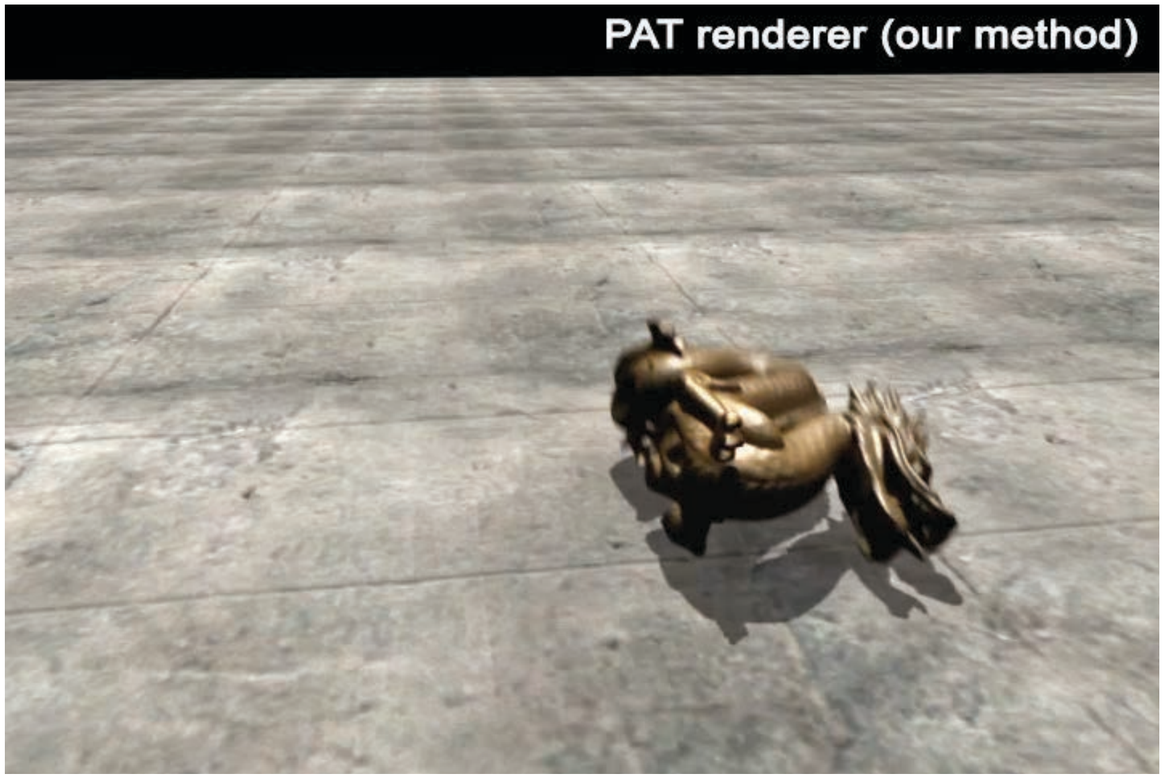
digital information

computation

visual information

Graphics has evolved a *lot* since its early days… no longer just about turning on pixels!

Turning digital information into sensory stimuli



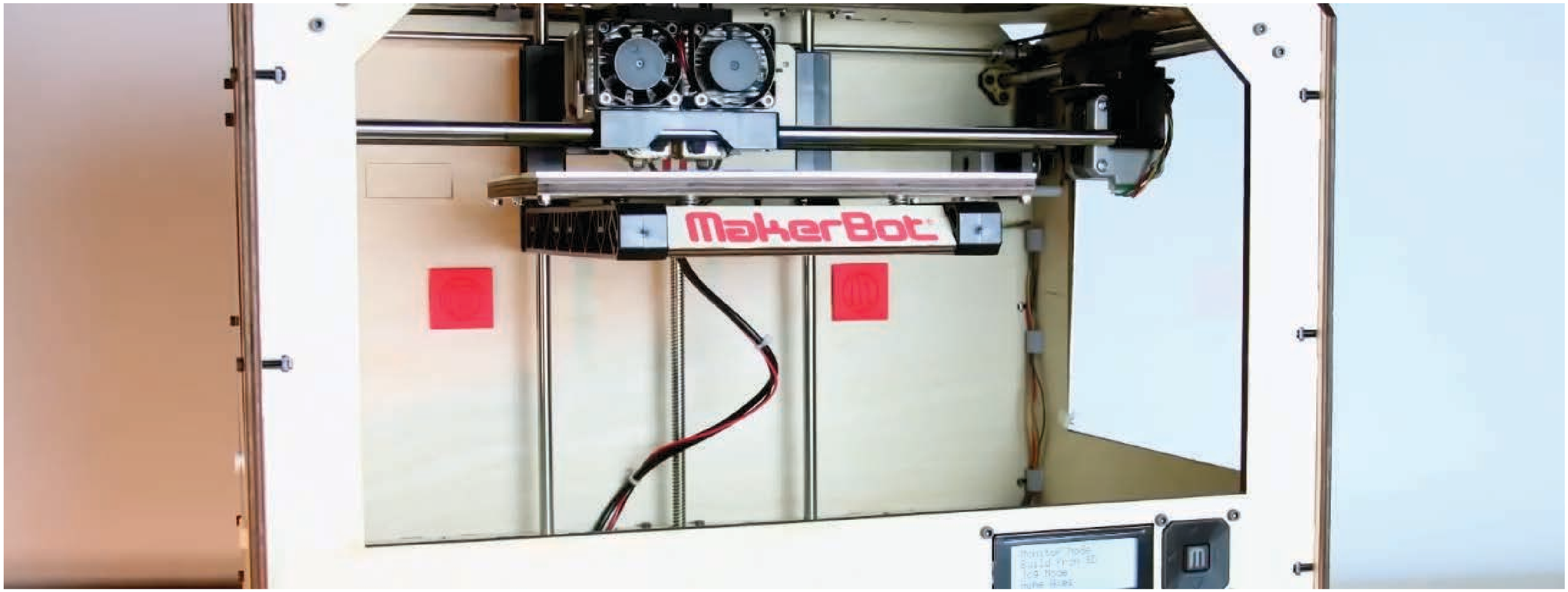
(sound)

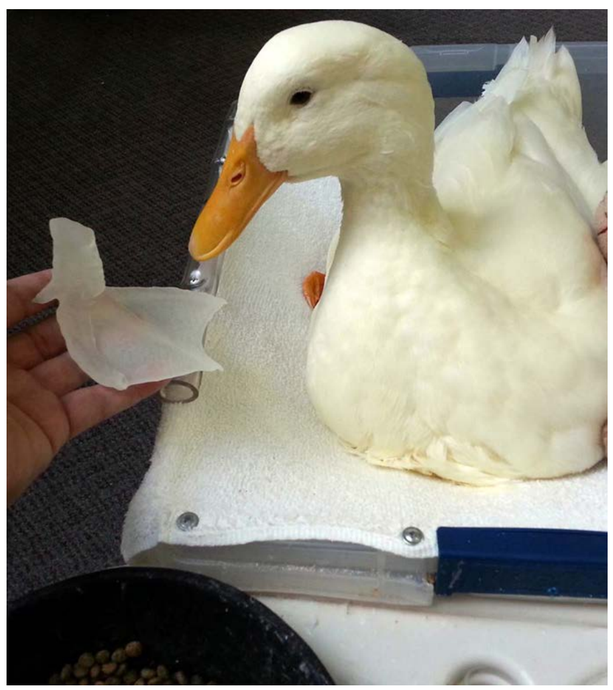
(touch)

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

(…What about taste? *Smell?!*)

Turning digital information into physical matter





Definition of Graphics, Revisited

com•put•er graph•ics /kəmˈpyo͞odər ˈgrafiks/ *n.*
The use of computation to turn digital information into **sensory stimuli**.

Even this definition is too narrow…

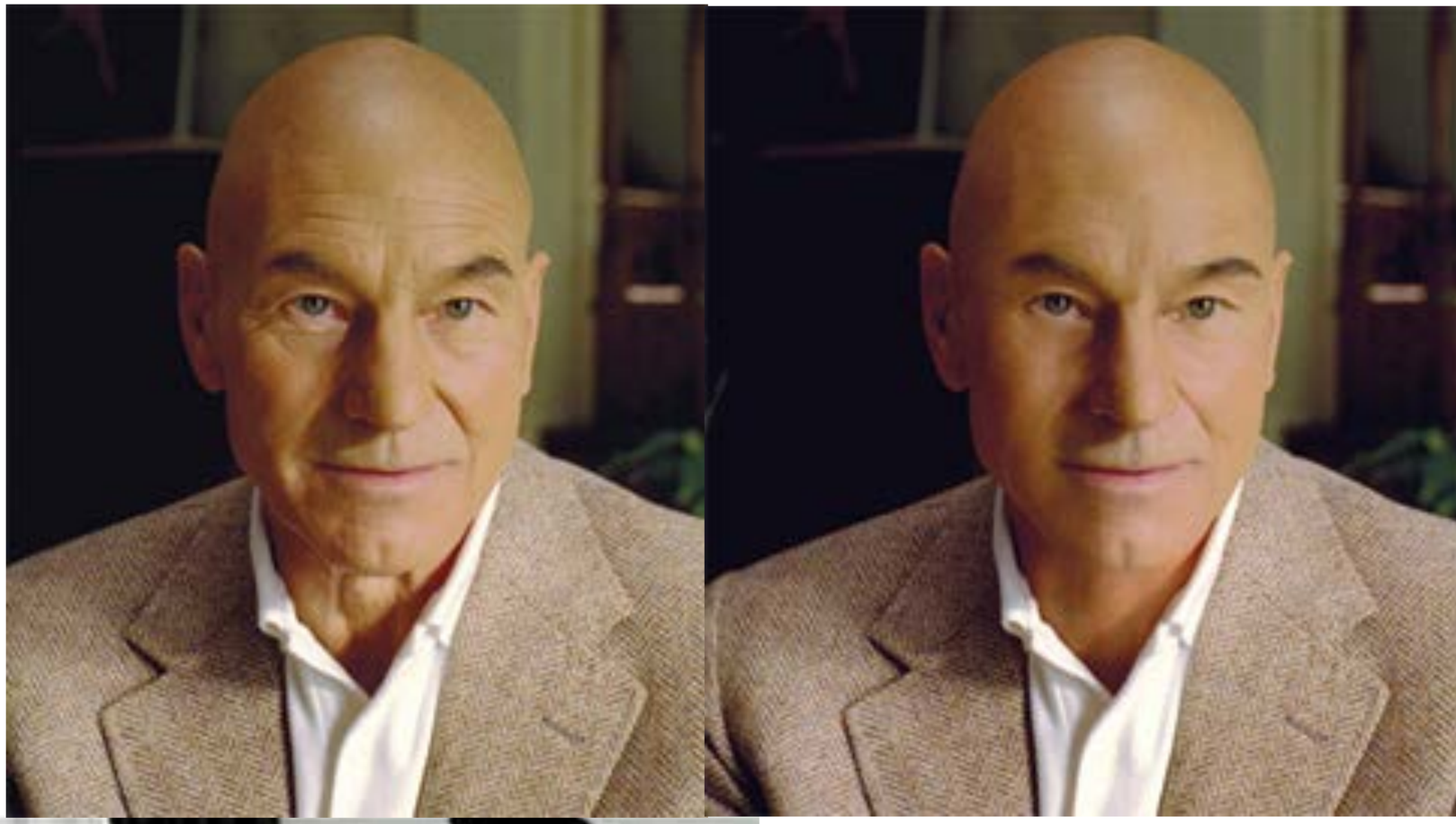
SIGGRAPH 2018 Technical Papers Trailer

Computer graphics is *everywhere!*

Entertainment (movies, games)

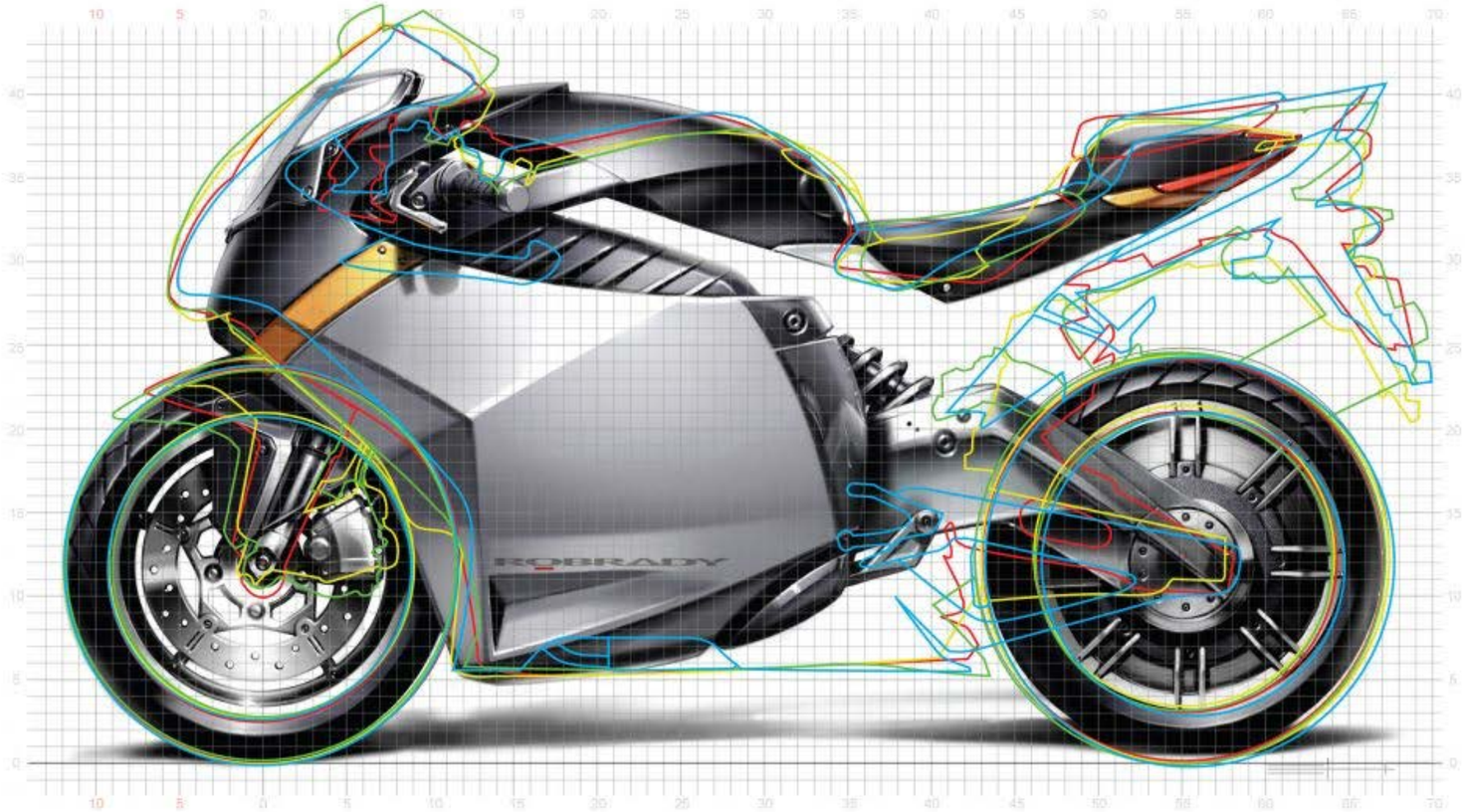
Entertainment

- Not just cartoons!

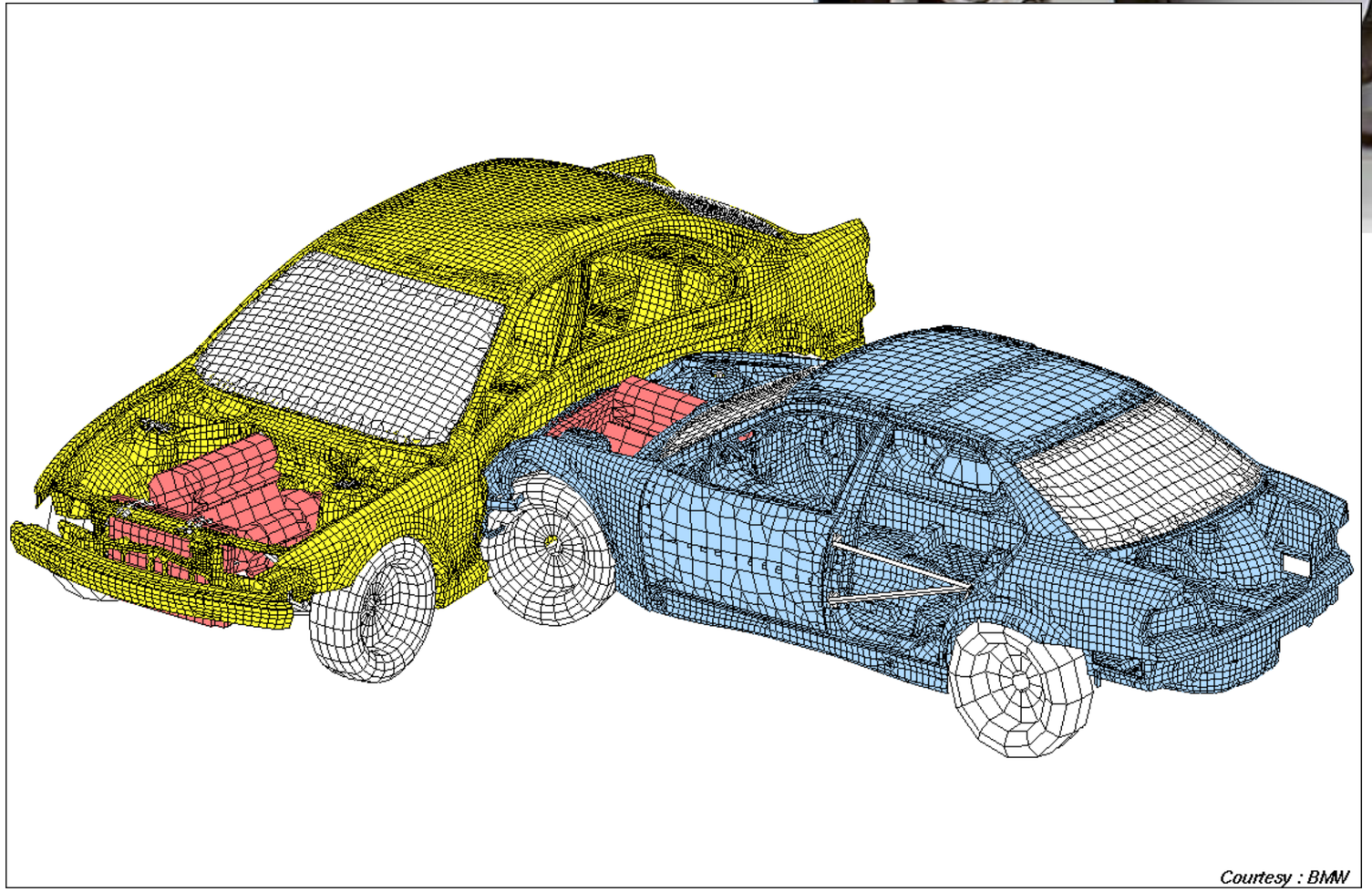


Art and design

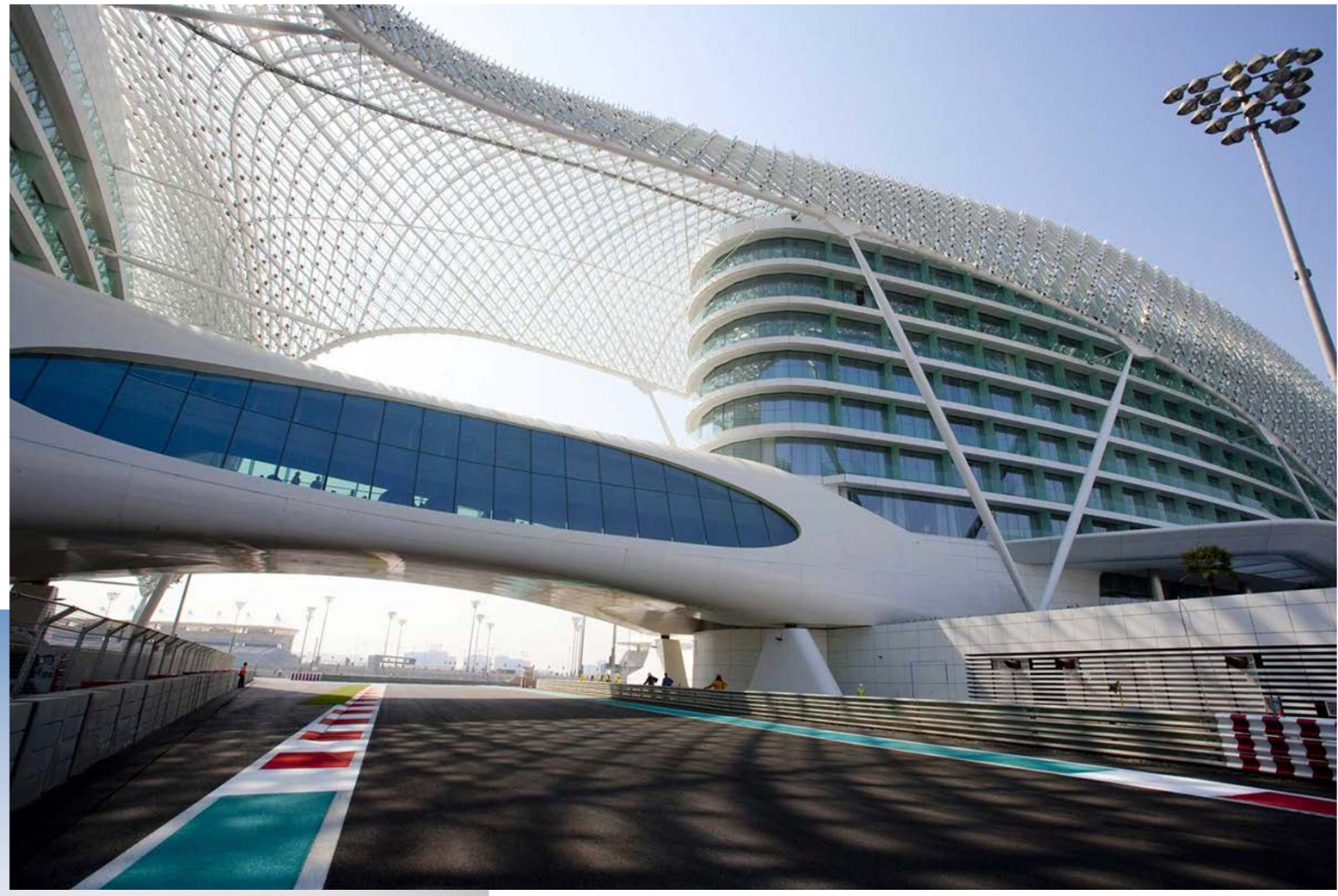
Industrial design

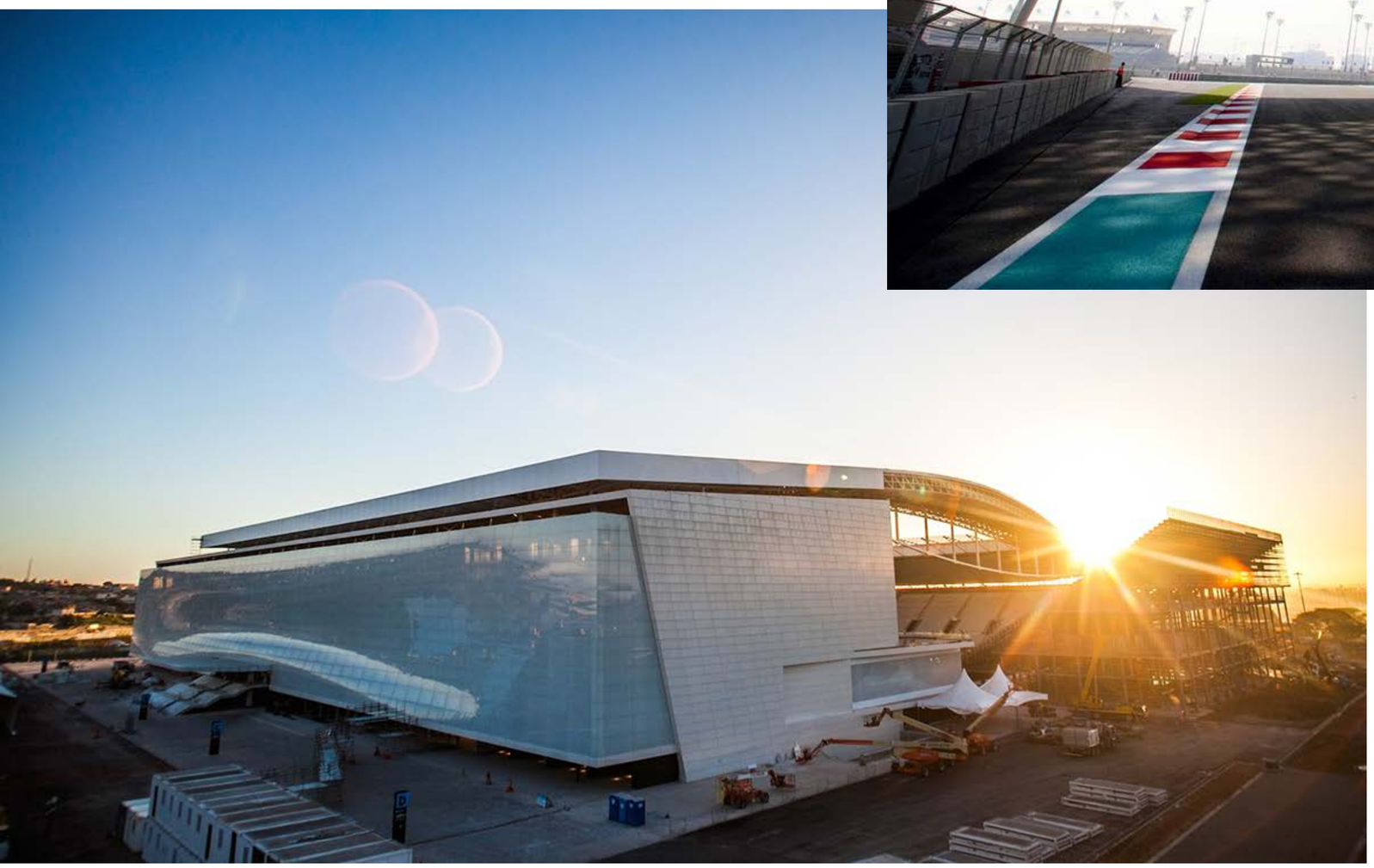


Computer aided engineering (CAE)

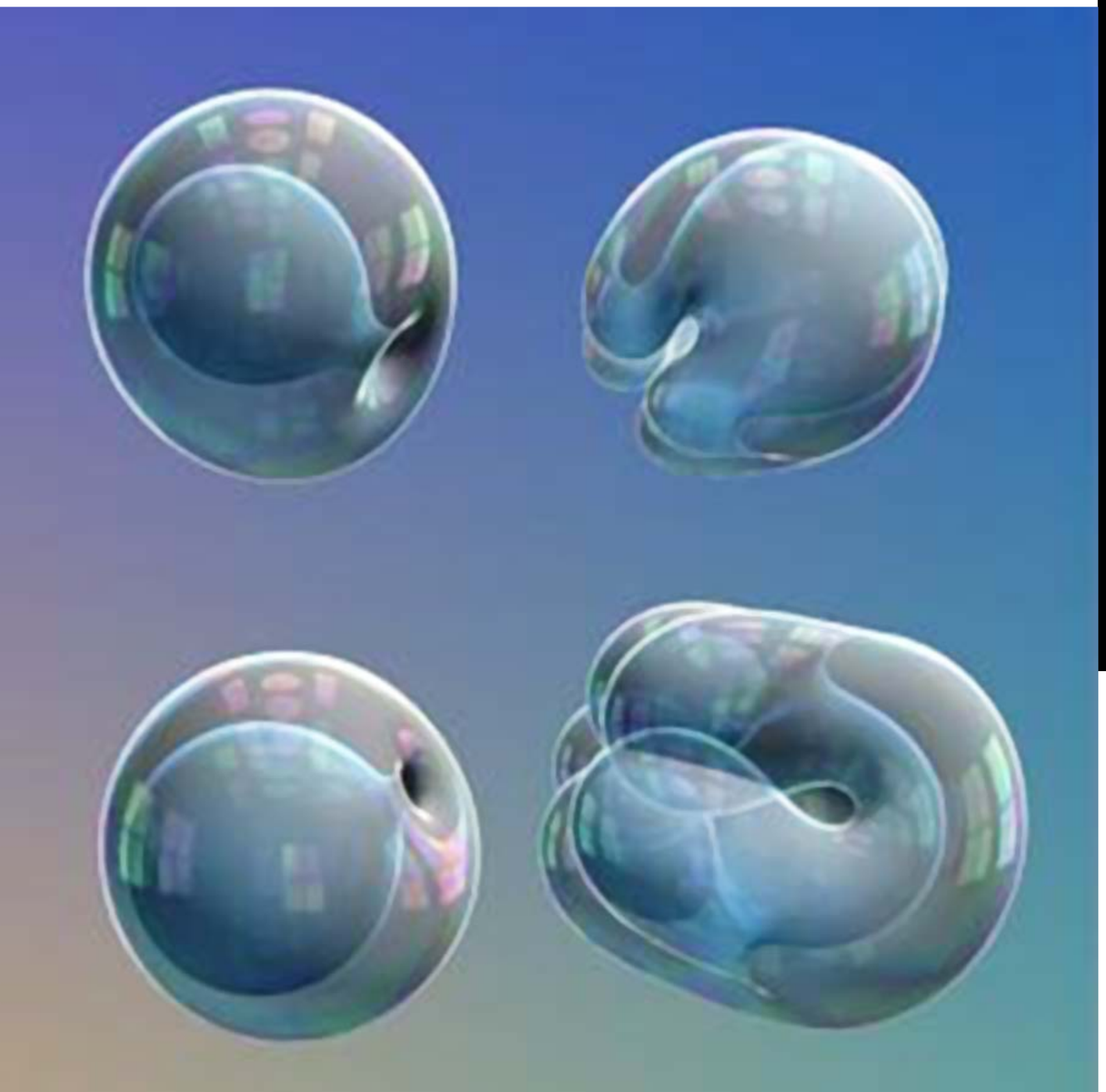


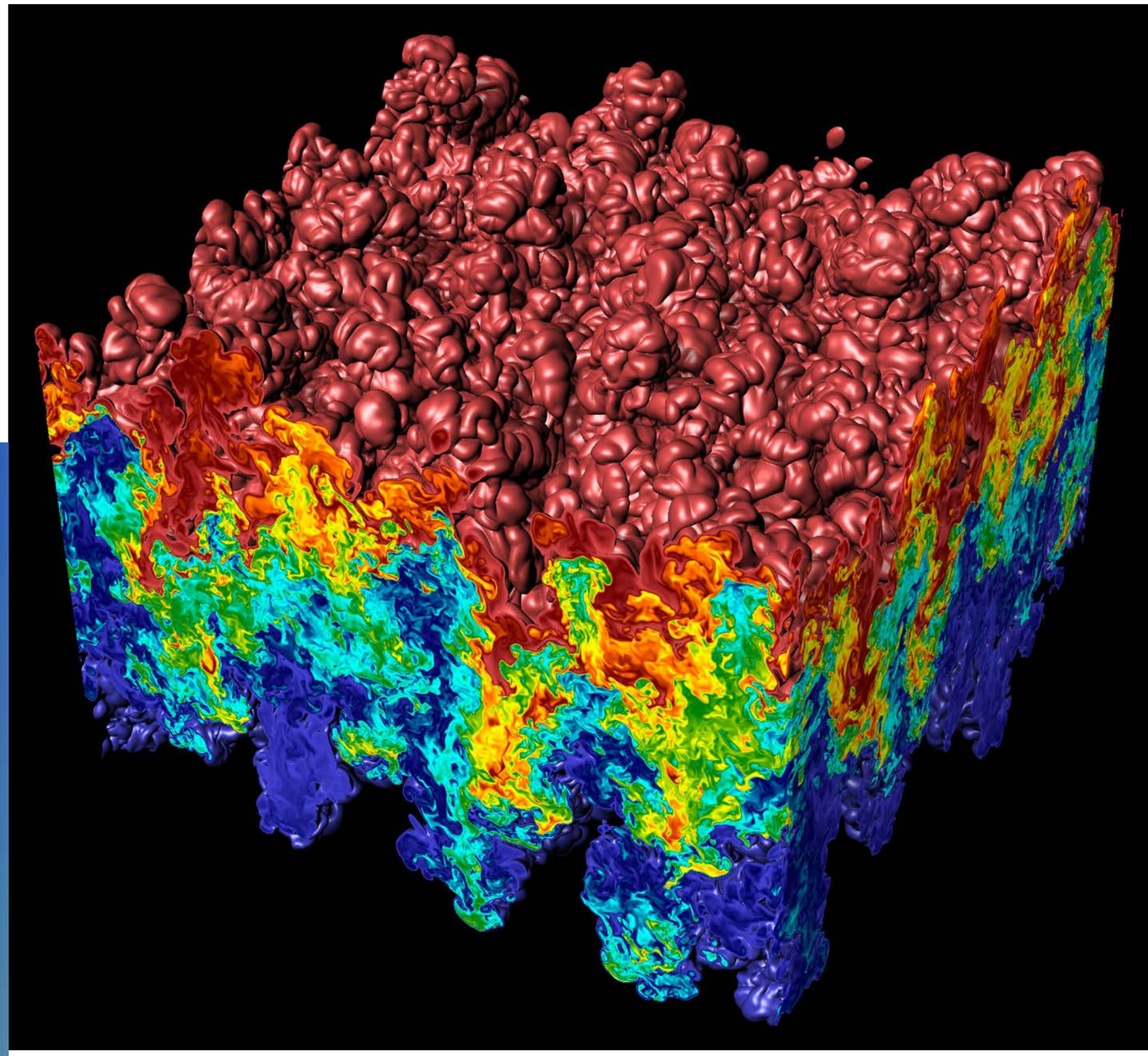
Architecture



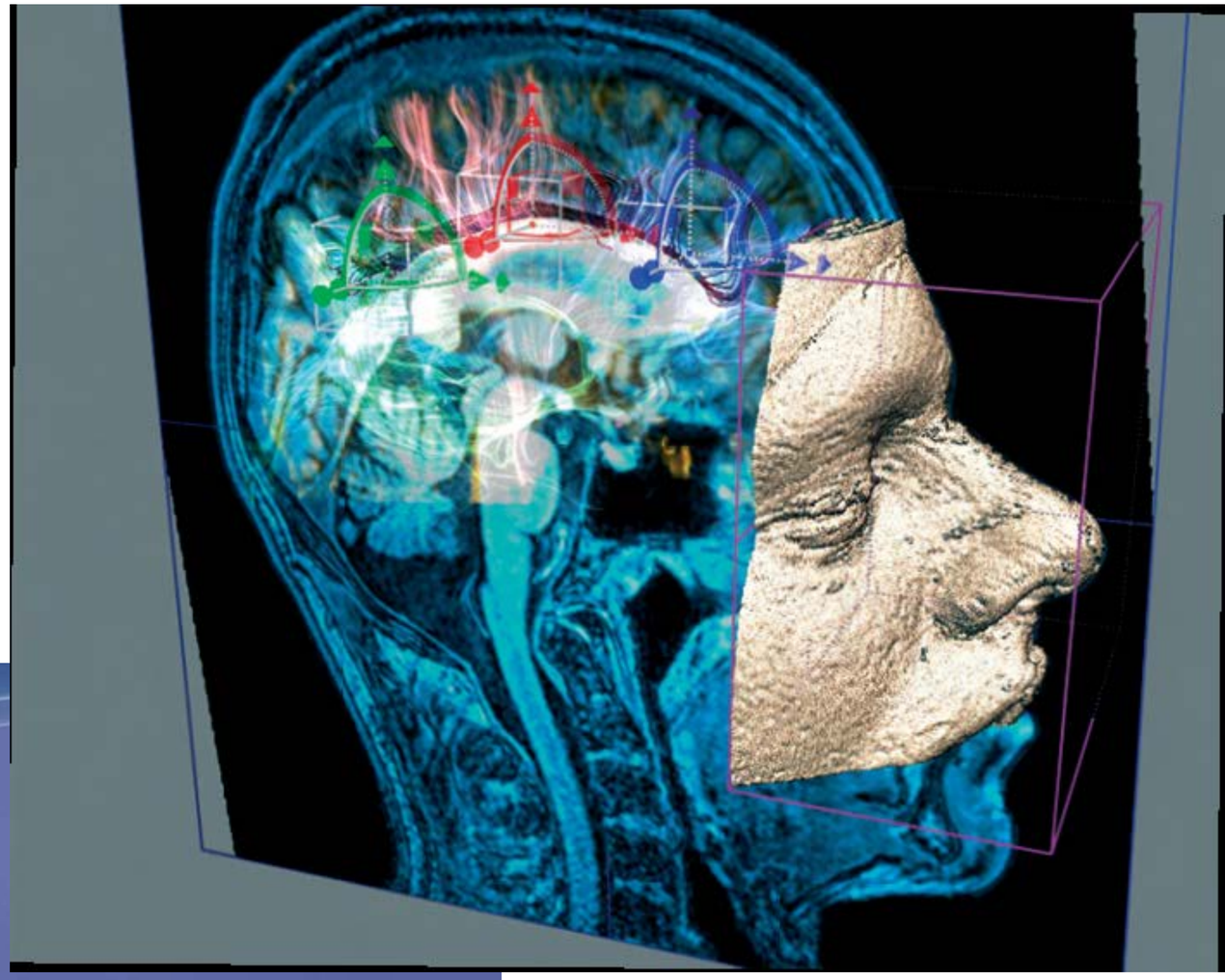


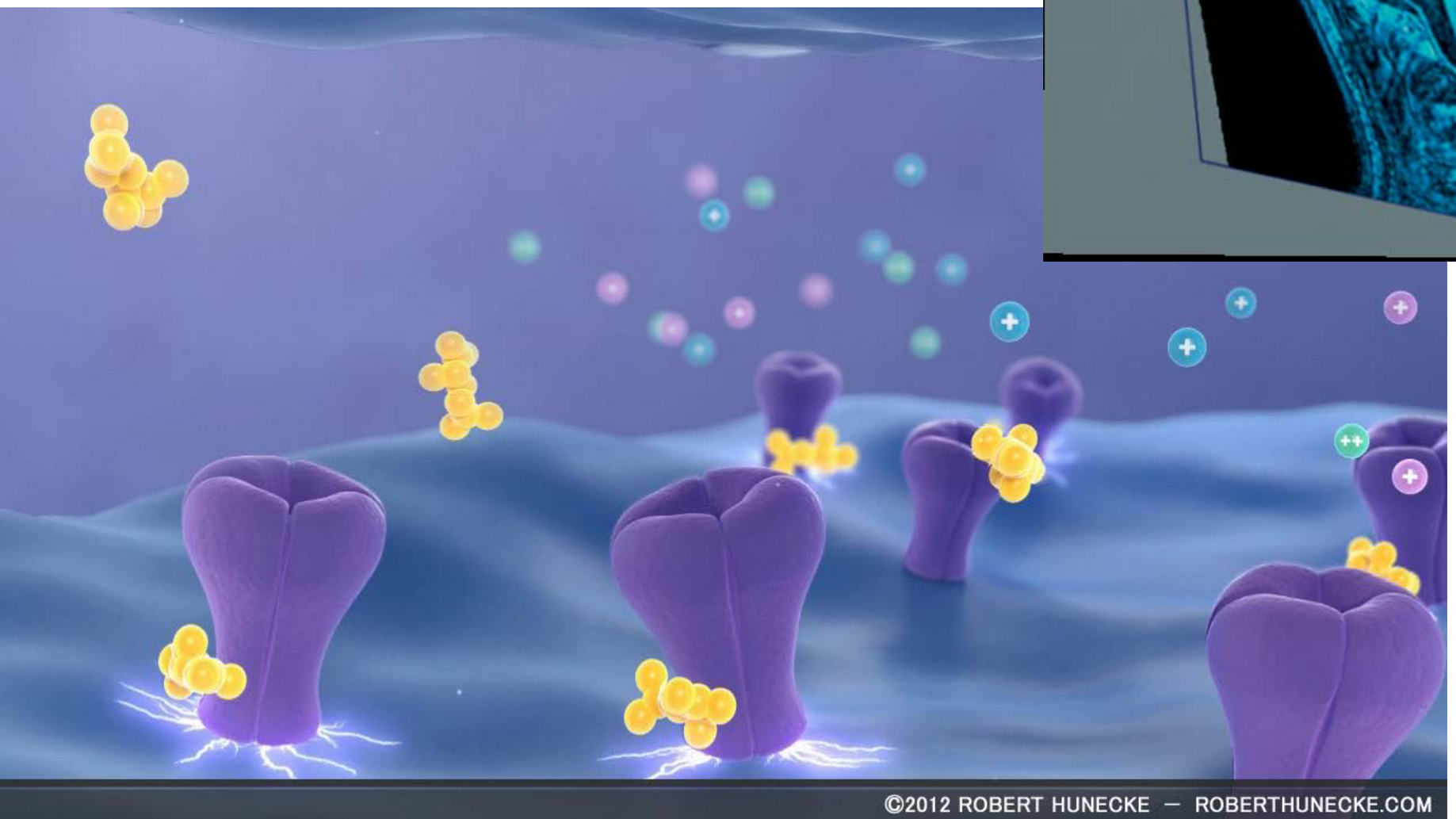
Scientific/mathematical visualization



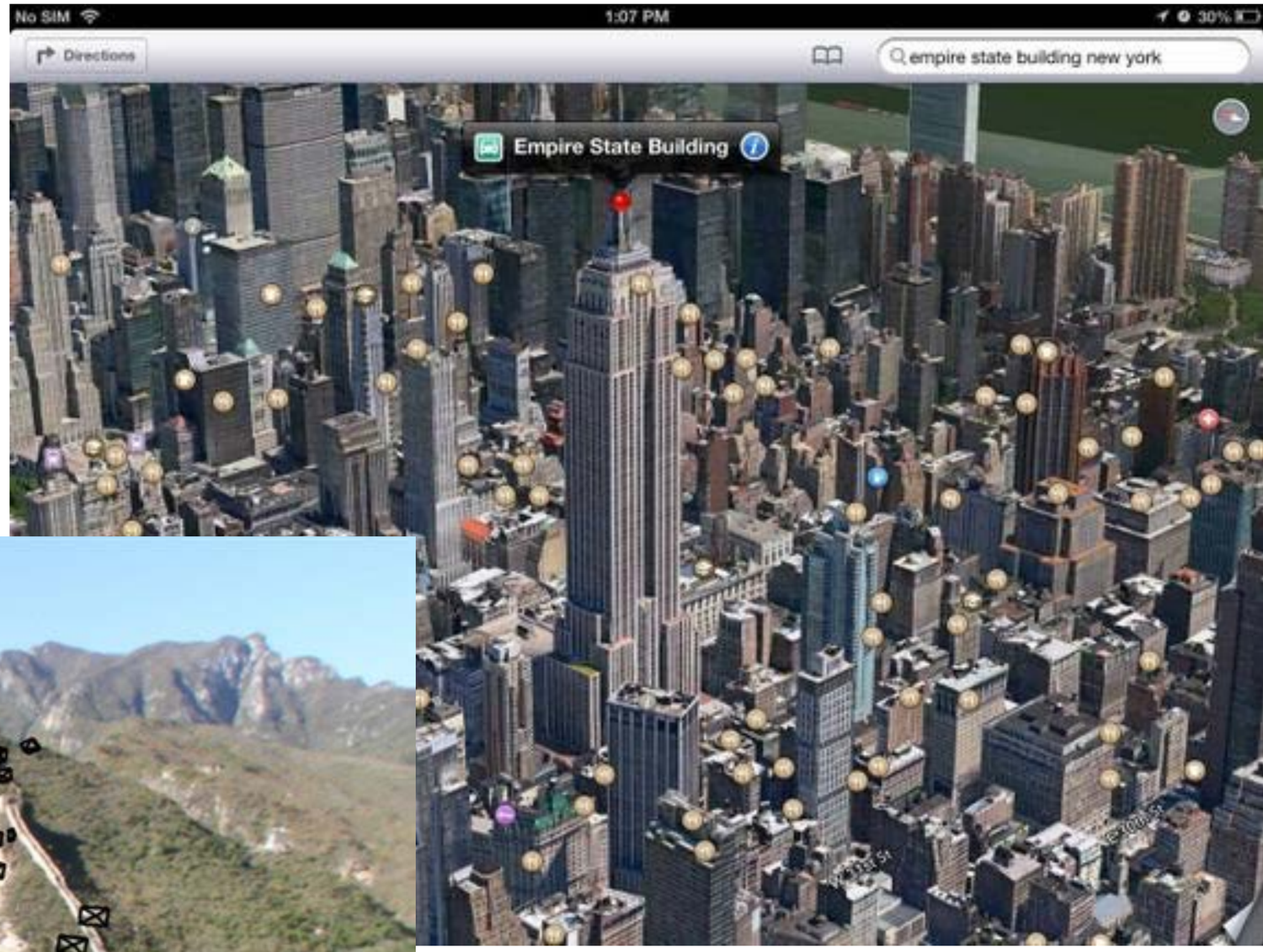


Medical/anatomical visualization

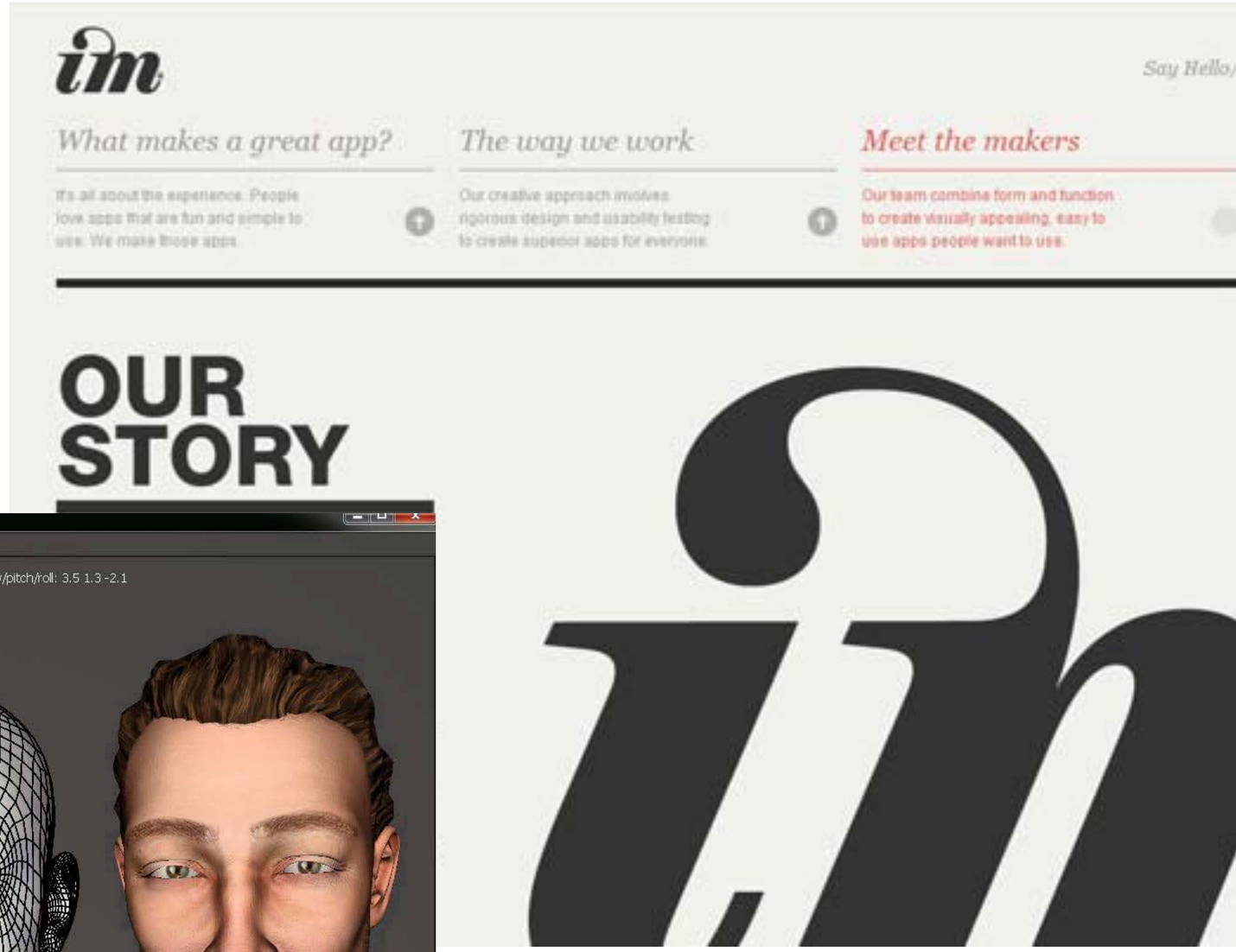


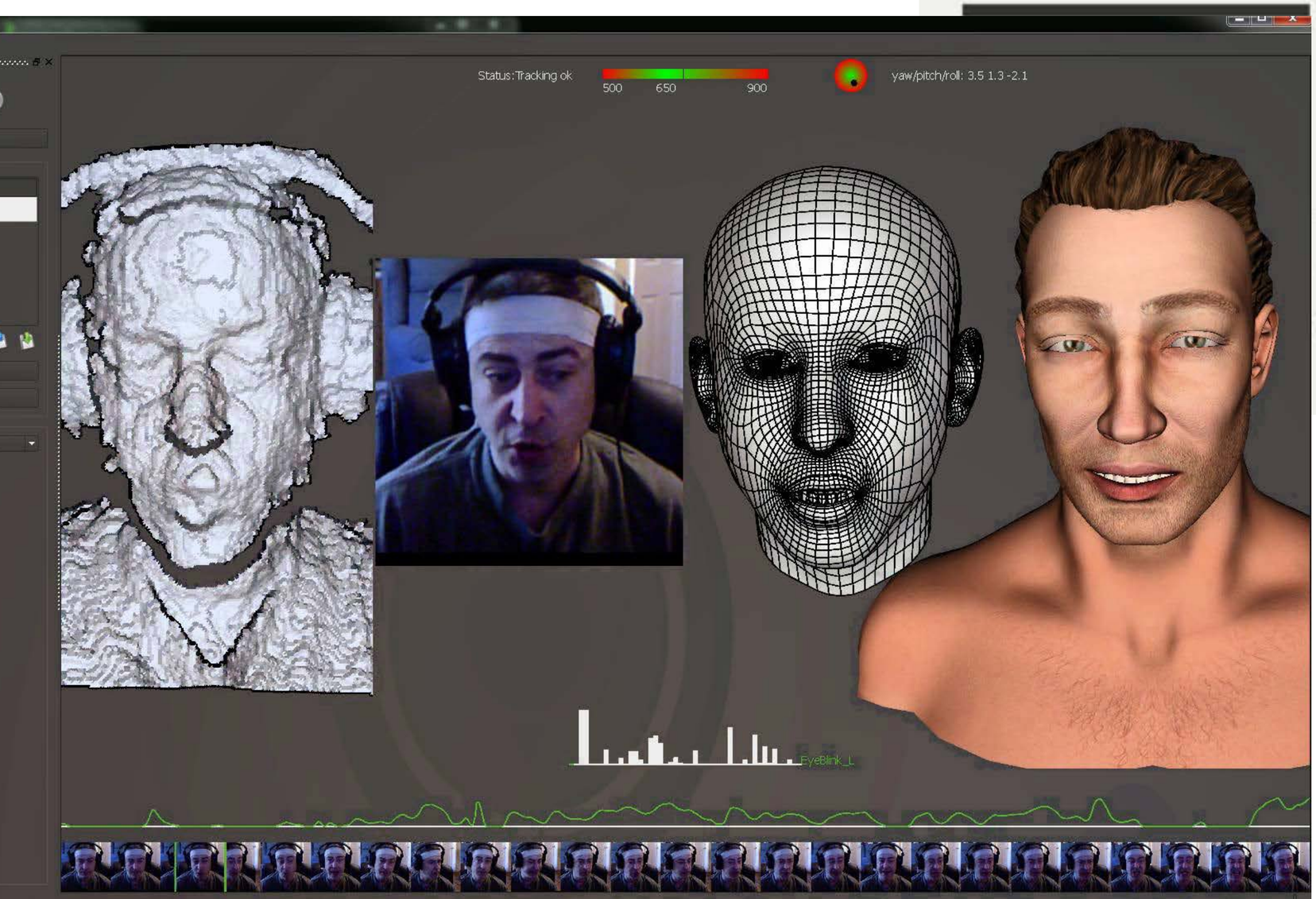


Navigation


No SIM
1:07 PM
30%
Directions
empire state building new york
Empire State Building

Communication





Foundations of computer graphics

- **All these applications demand *sophisticated* theory & systems**
- **Theory**
  - **basic representations** *(how do you digitally encode shape, motion?)*
  - **sampling & aliasing** *(how do you acquire & reproduce a signal?)*
  - **numerical methods** *(how do you manipulate signals numerically?)*
  - **radiometry & light transport** *(how does light behave?)*
  - **perception** *(how does this all relate to humans?)*
  - **…**
- **Systems**
  - **parallel, heterogeneous processing**
  - **graphics-specific programming languages**
  - **…**

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 2x2x2
  - edges are aligned with x/y/z axes
- 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

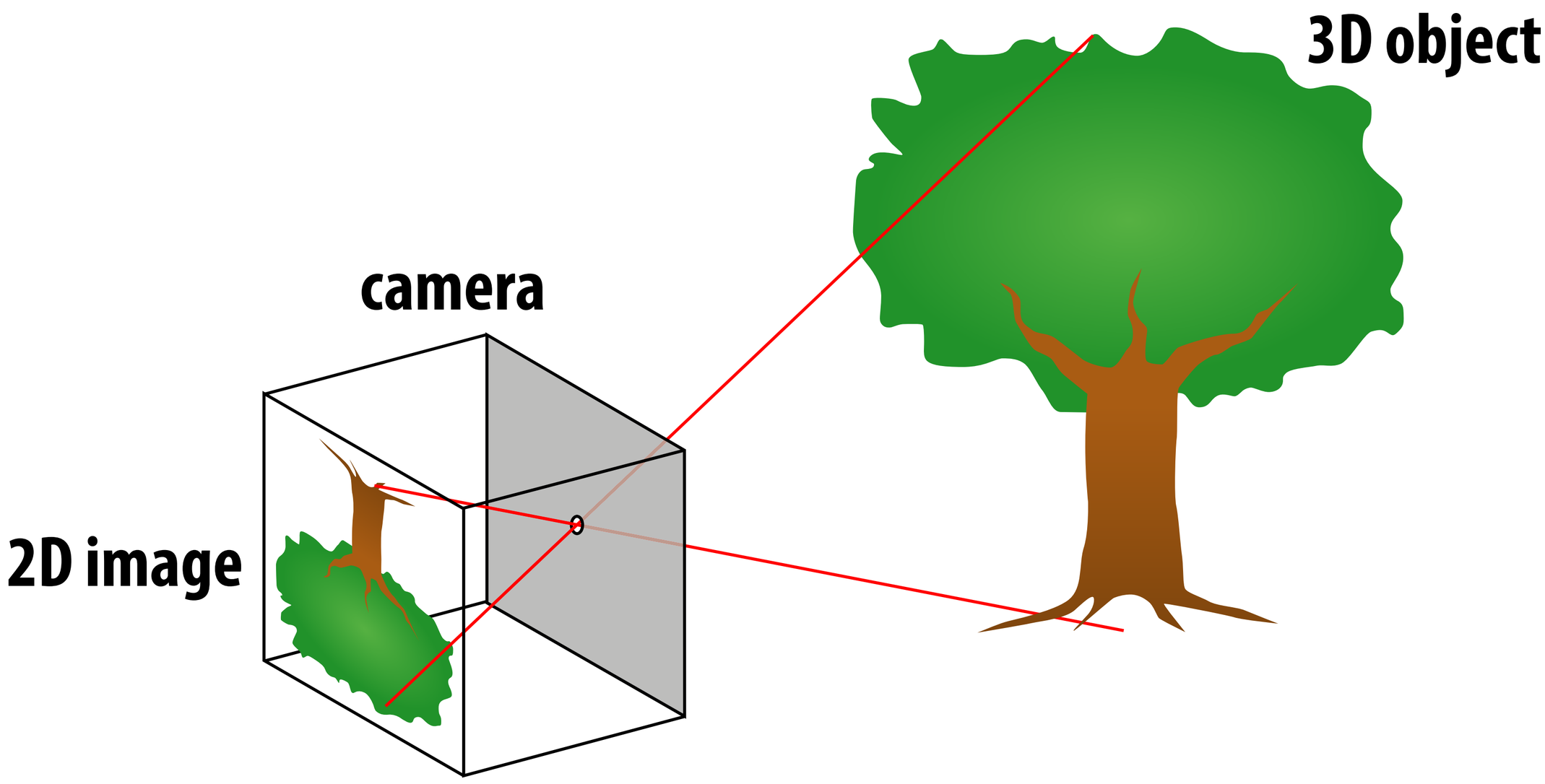
- Now have a digital description of the cube:

```
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. map 3D vertices to 2D points in the image
  2. connect 2D points with straight lines
- ...Ok, but how?

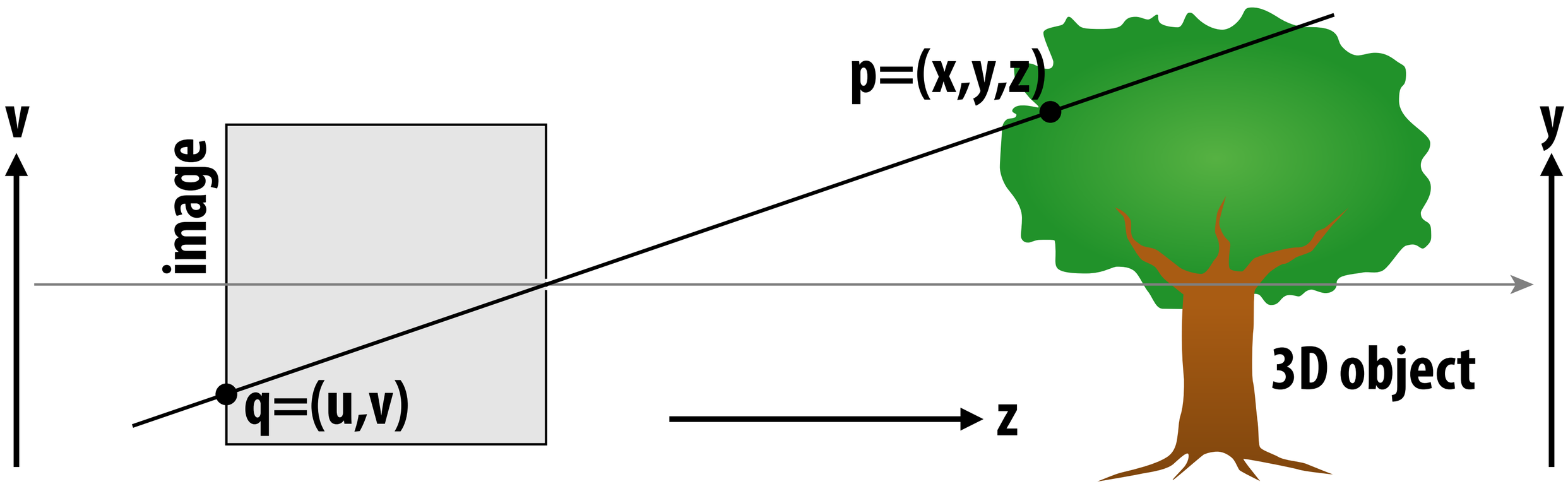
Perspective projection

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



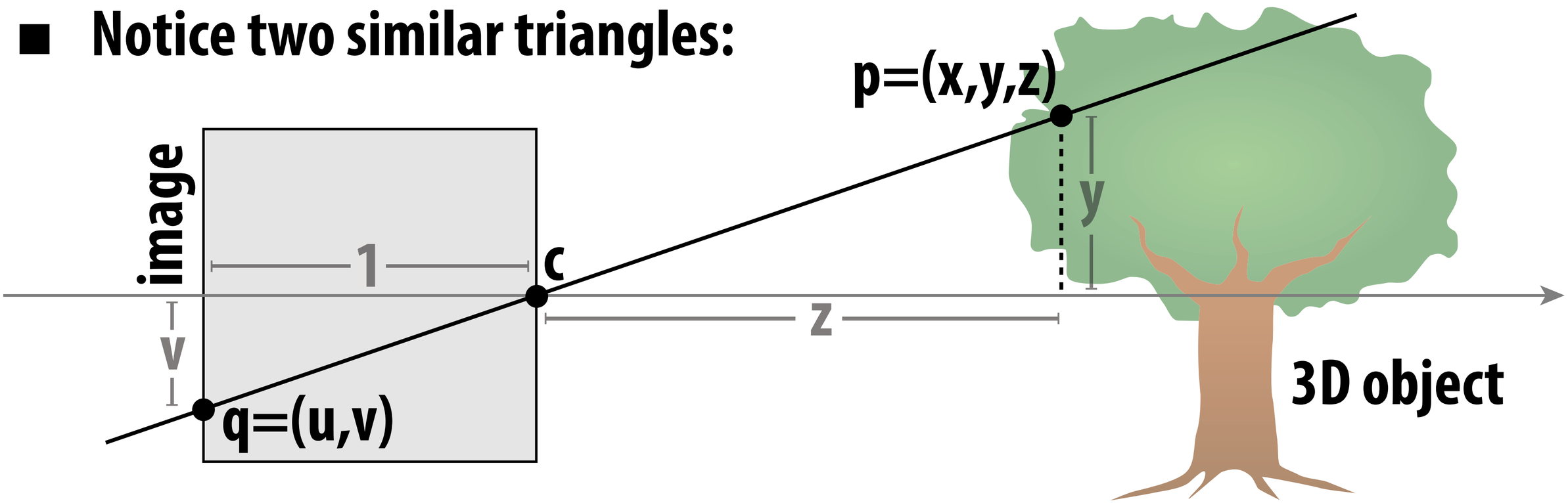
Perspective projection: side view

- 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:



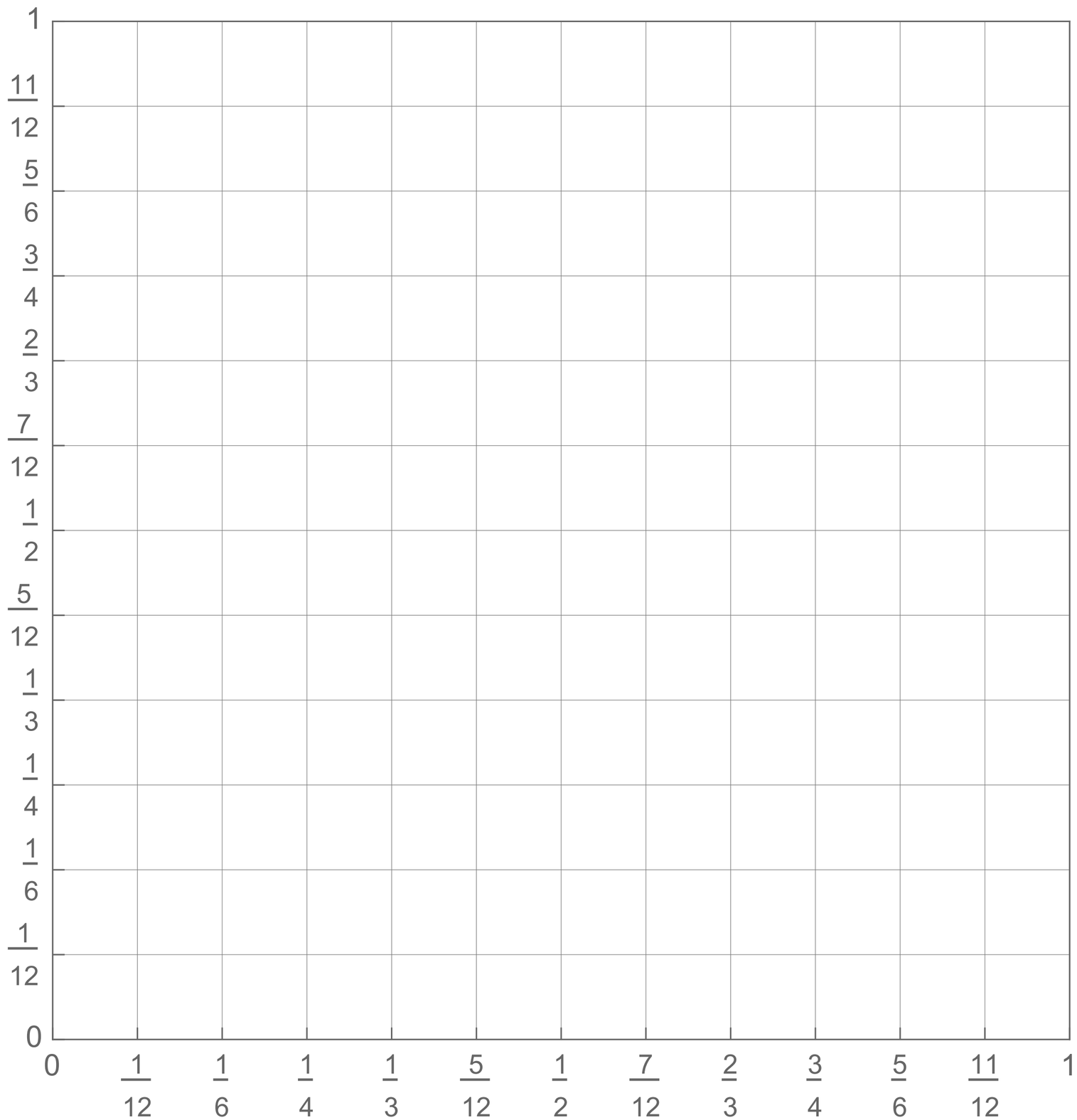
- Assume camera has unit size, origin is at pinhole c
- Then v/1 = y/z, i.e., vertical coordinate is just the slope y/z
- Likewise, horizontal coordinate is u=x/z

ACTIVITY: now draw it!

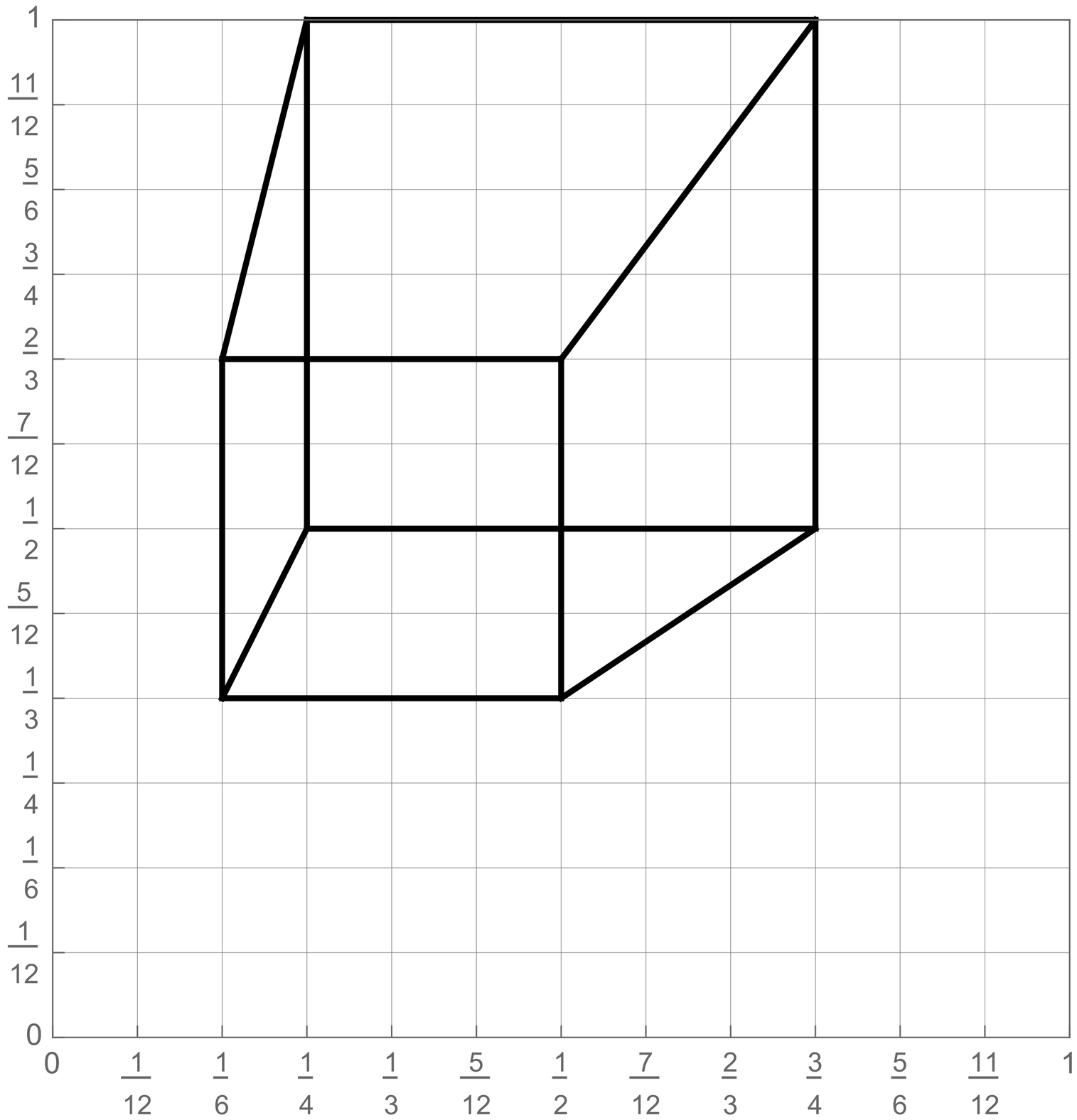
- Need 12 volunteers
  - each person will draw one cube edge
  - assume camera is at c=(2,3,5)
  - convert (X,Y,Z) of both endpoints to (u,v):
    1. subtract camera c from vertex (X,Y,Z) to get (x,y,z)
    2. divide (x,y) by z to get (u,v)—*write as a fraction*
  - draw line between (u1,v1) and (u2,v2)

```
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: output on graph paper



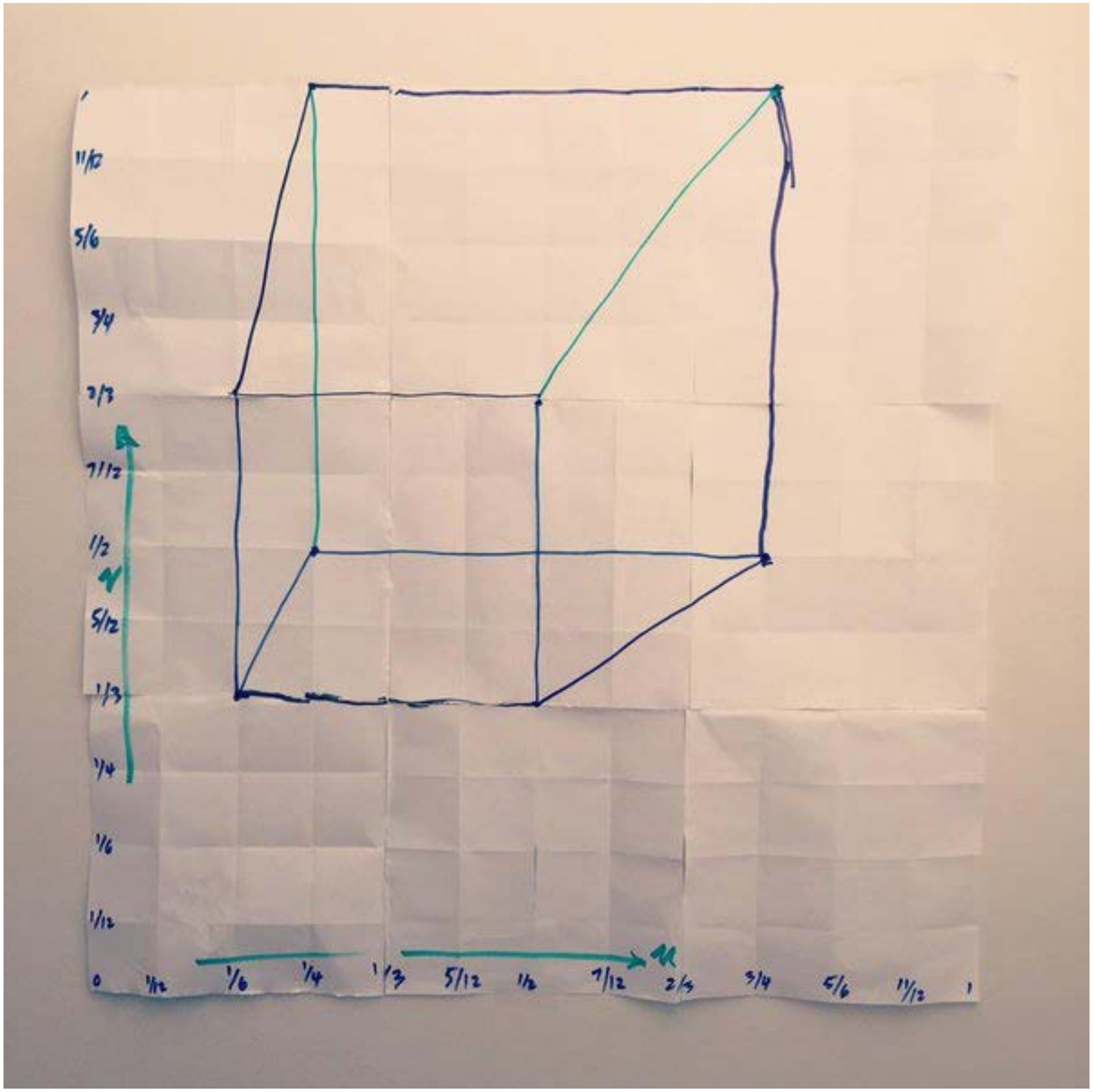
ACTIVITY: How did we do?



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

ACTIVITY: Previous year's result



Success! We turned purely <u>digital</u> information into purely <u>visual</u> information, using a completely <u>algorithmic</u> procedure.

<u>digital</u> information

computation

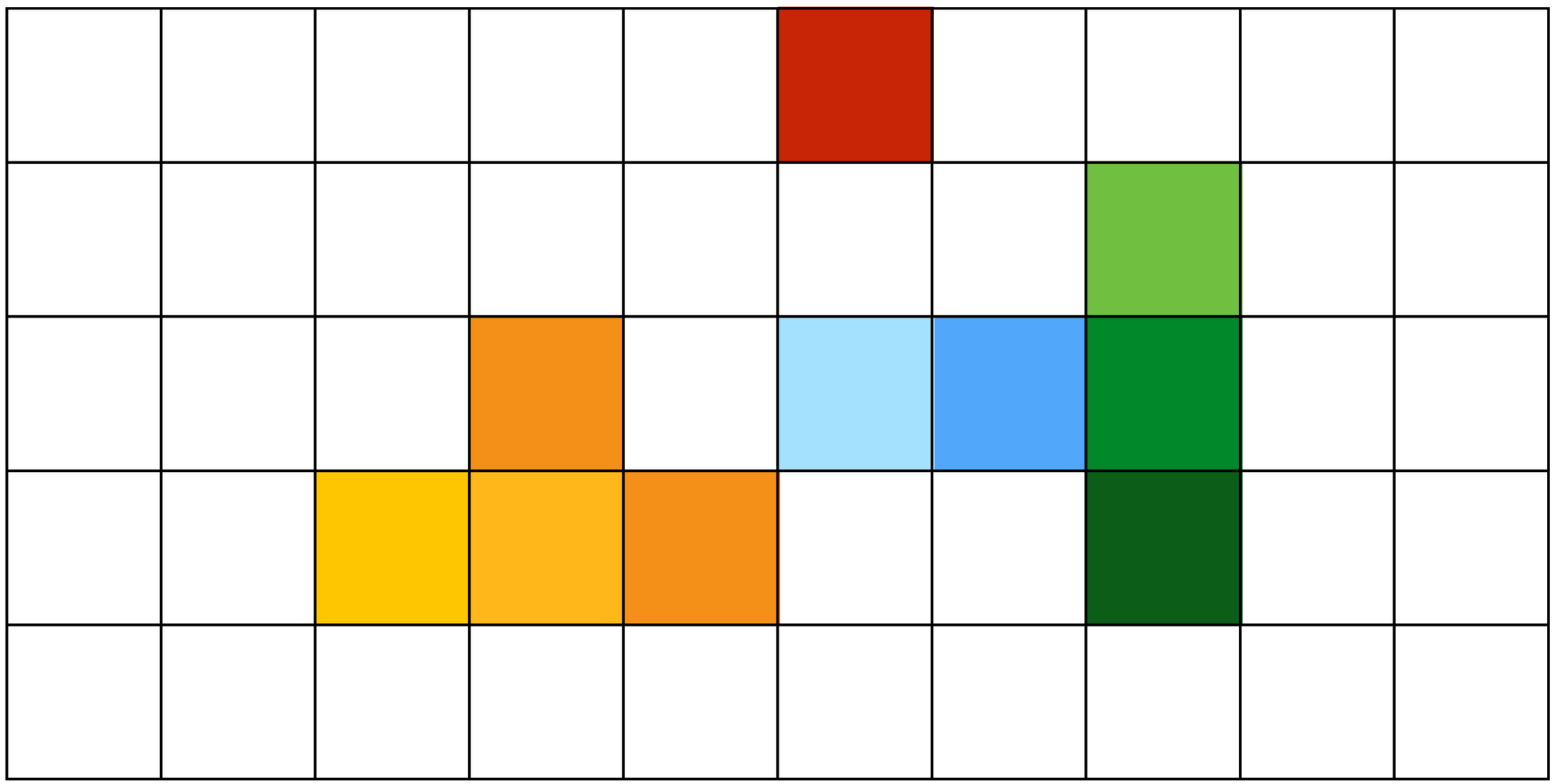
<u>visual</u> information

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

Close up photo of pixels on a modern display

Output for a raster display

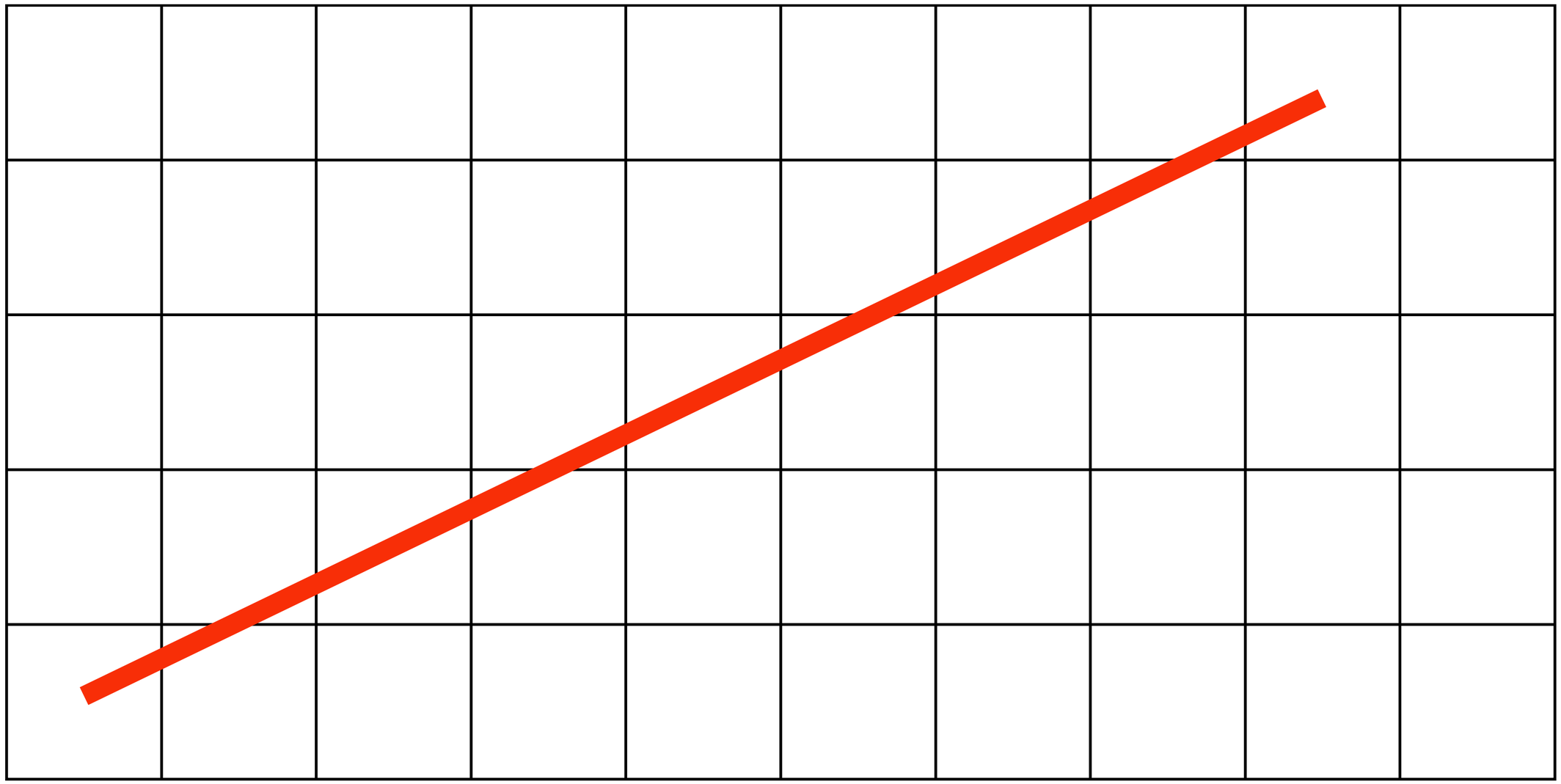
- Common abstraction of a raster display:
  - Image represented as a 2D grid of "pixels" (picture elements) **
  - Each pixel can can take on a unique color value



** We will strongly challenge this notion of a pixel "as a little square" soon enough.
But let's go with it for now. ;-)

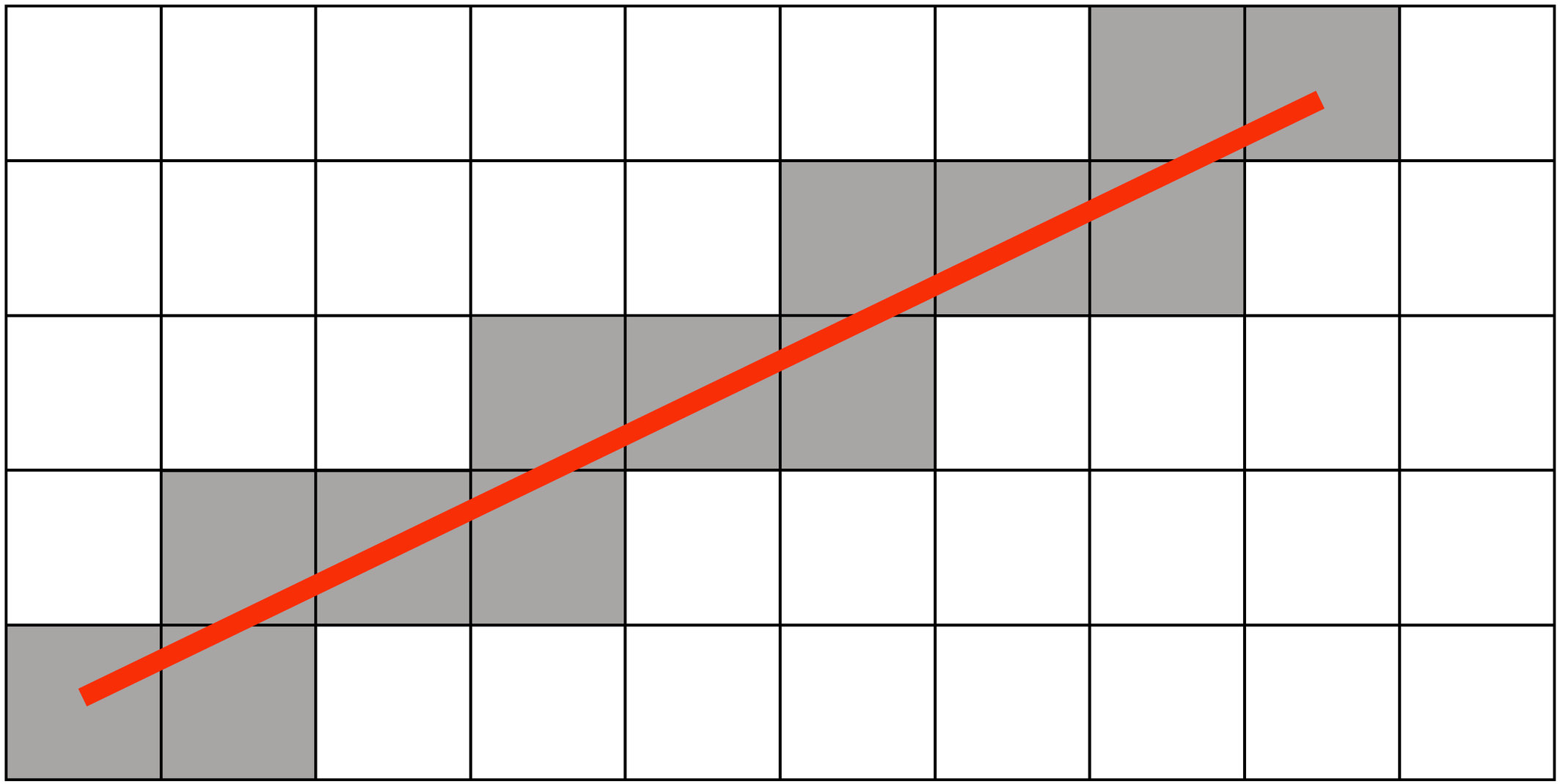
What pixels should we color in to depict a line?

"Rasterization": process of converting a continuous object to a discrete representation on a raster grid (pixel grid)



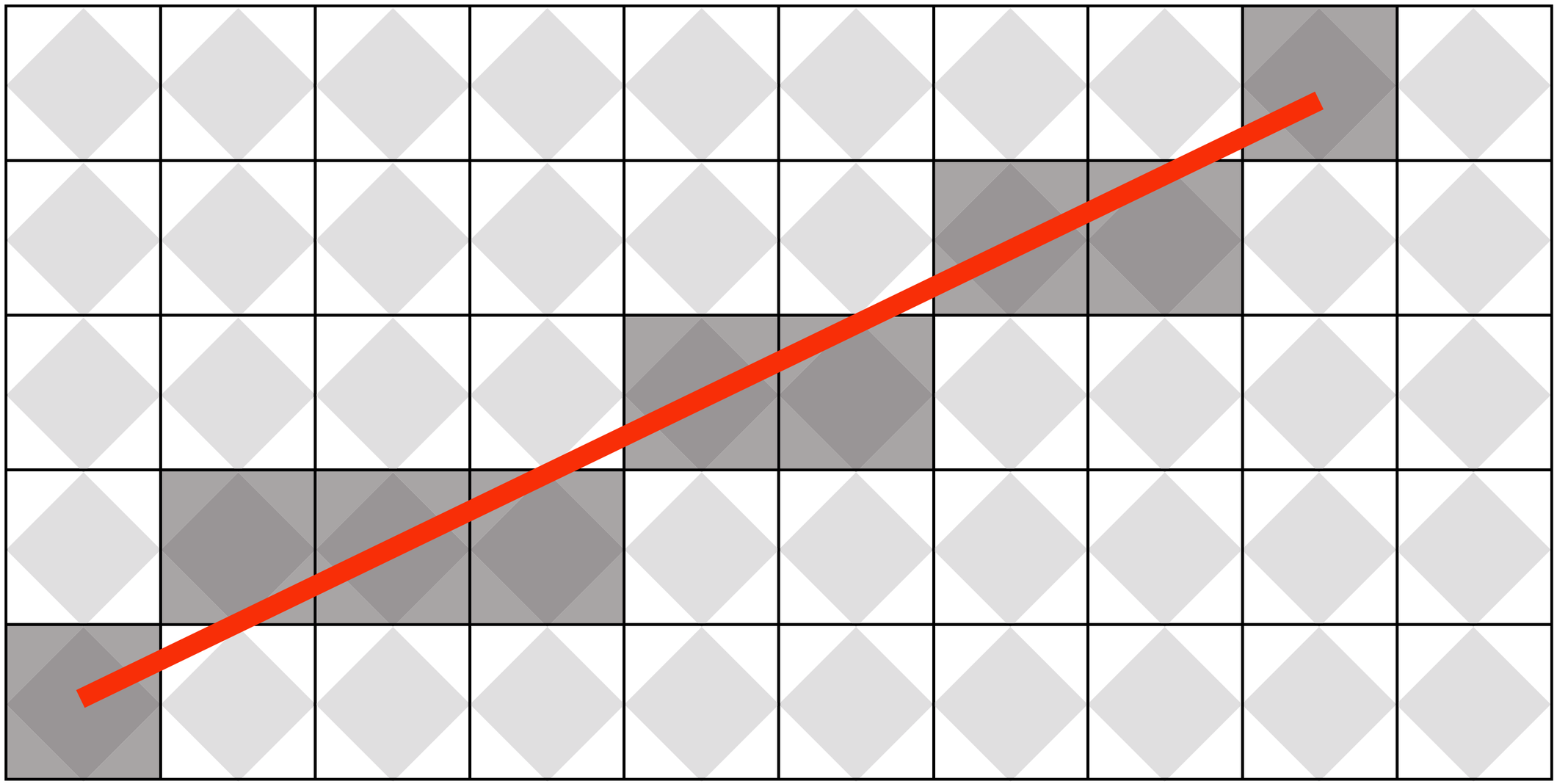
What pixels should we color in to depict a line?

Light up all pixels intersected by the line?



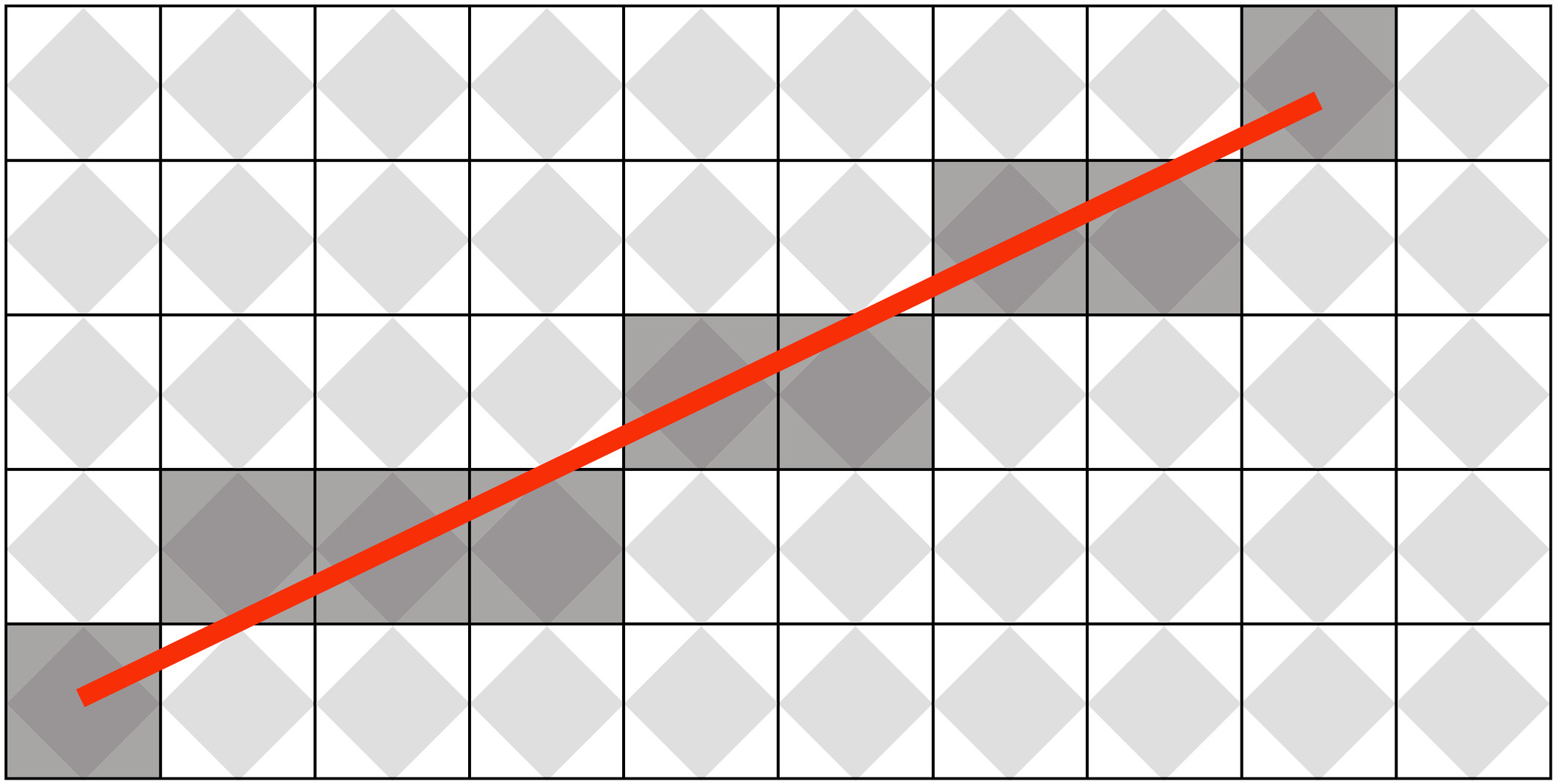
What pixels should we color in to depict a line?

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)



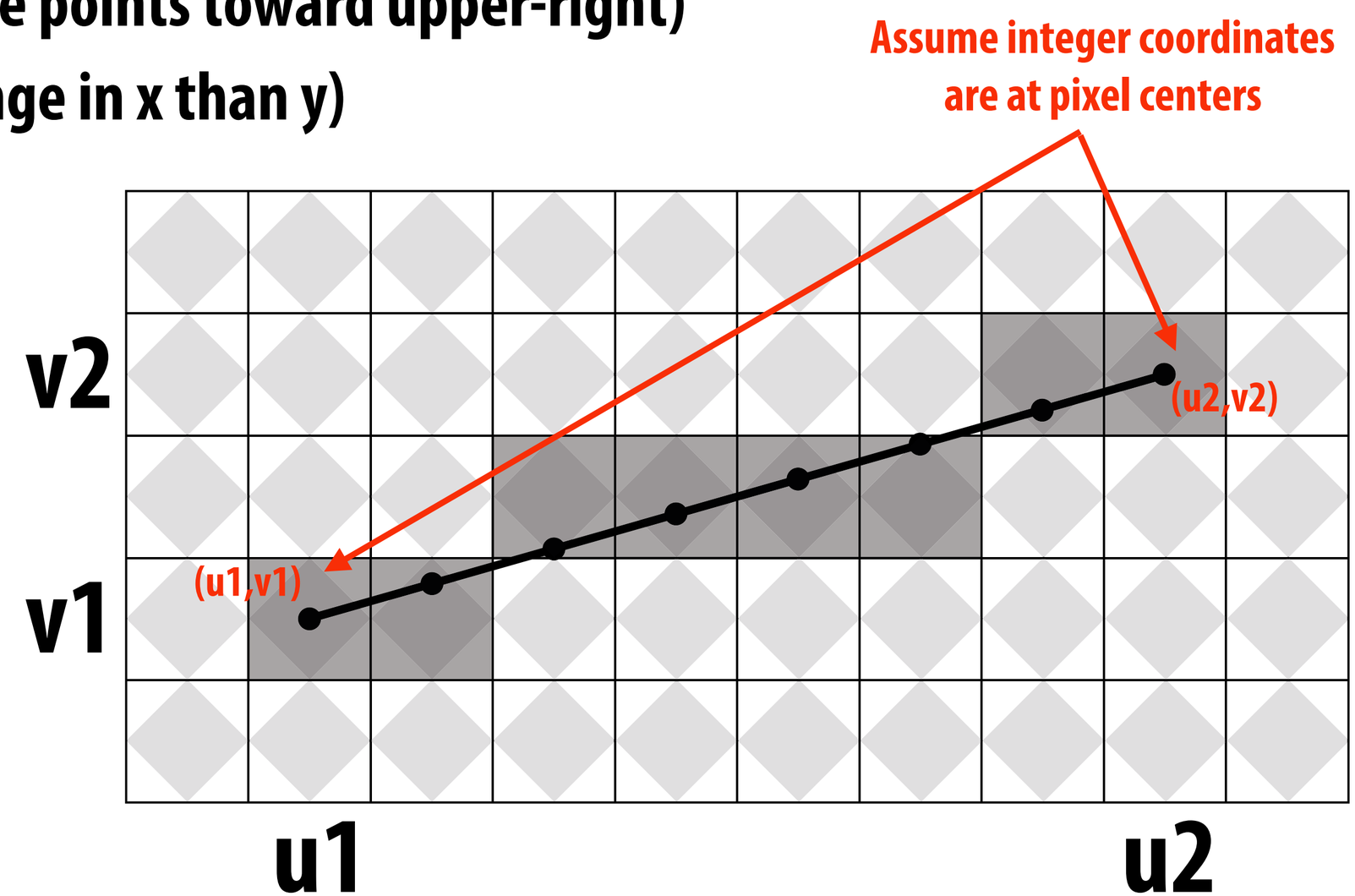
How do we find the pixels satisfying a chosen rasterization rule?

- 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., work proportional to number of pixels in the drawing of the line)

Incremental line rasterization

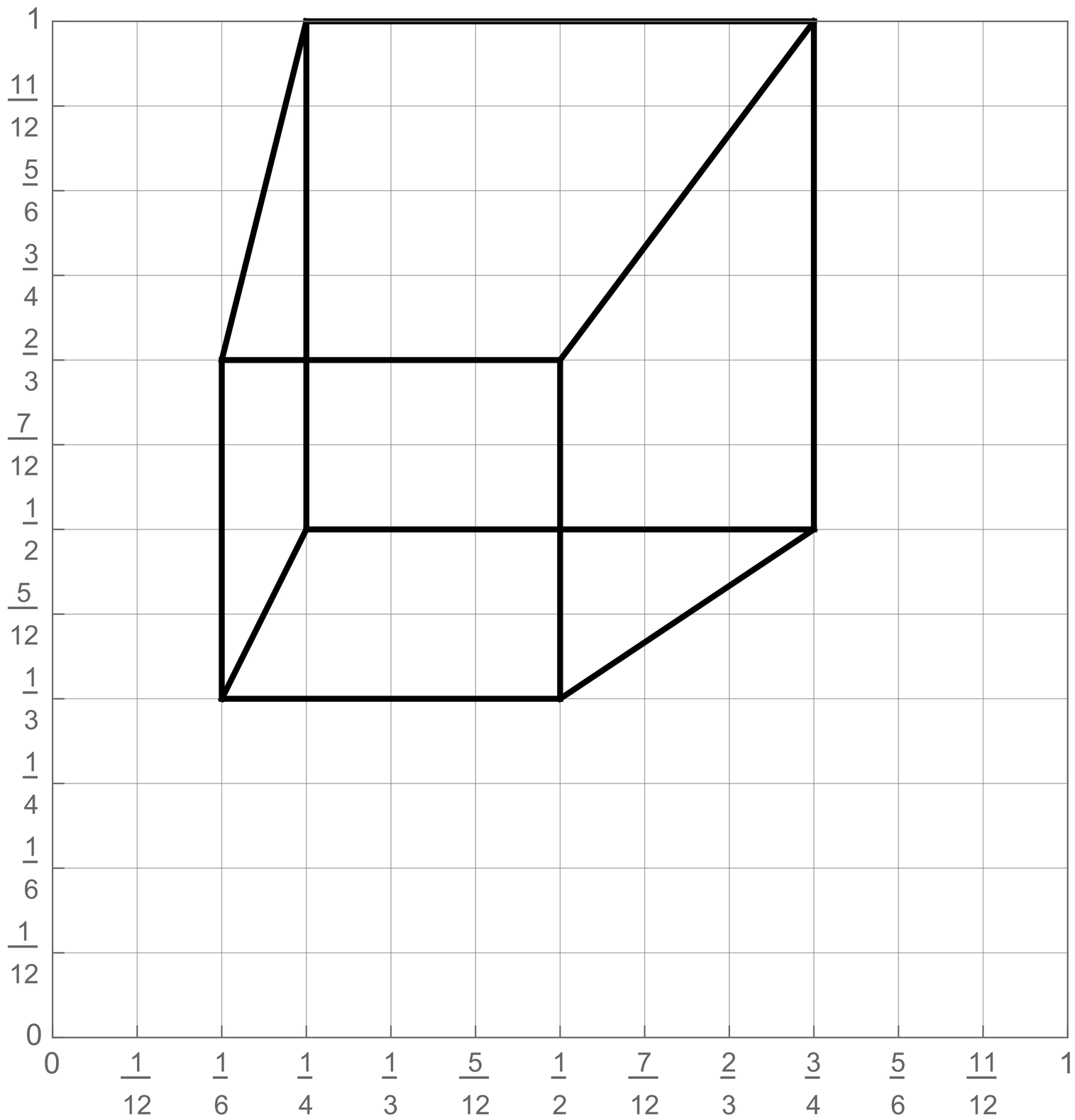
- Let's say a line is represented with integer endpoints: (u1,v1), (u2,v2)
- Slope of line: s = (v2-v1) / (u2-u1)
- Consider a very easy special case:
  - u1 < u2, v1 < v2 (line points toward upper-right)
  - 0 < s < 1 (more change in x than y)

```
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)

Our line drawing!



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
```

We just rendered a simple line drawing of a cube.

But for more realistic pictures, we need a *<u>much</u>* richer model of the world:

GEOMETRY
MATERIALS
LIGHTS
CAMERAS
MOTION
...

Will see all of this (and more!) as our course progresses.

Goal: create images like this!

[Source: Batra 2017]

Goal: create images like this!

Goal: create images like this!


Big Hero 6 (Disney 2014)

Goal: create images like this!

Unreal Engine Kite Demo (Epic Games 2015)

Goal: create images like this!

[Mirror's Edge 2008]

Course Logistics

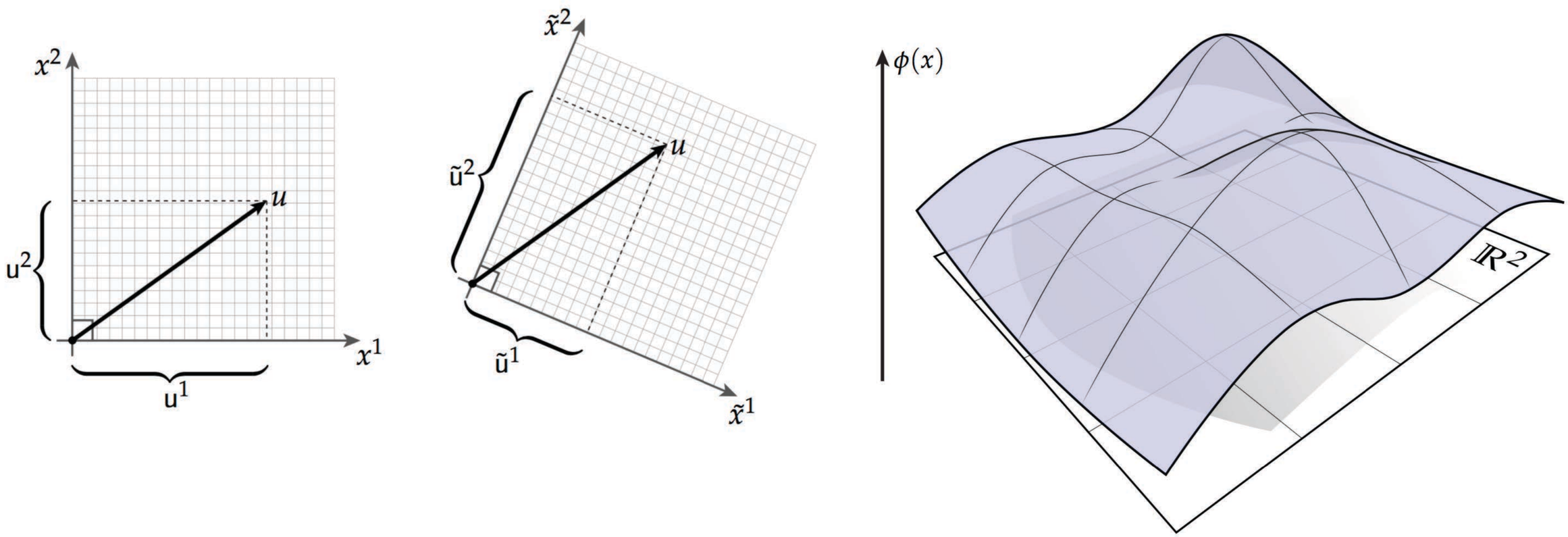
About this course

- Outline:
  - Focus on fundamental data structures and algorithms that are reused across all areas of graphics
  - Major areas of focus:
    - IMAGING — how do computers store/generate images?
    - GEOMETRY — how do we represent shape?
    - RENDERING — how do we simulate light?
    - ANIMATION — how do we synthesize motion?
  - Goal: develop skills needed to derive, develop, and implement modern graphics algorithms.

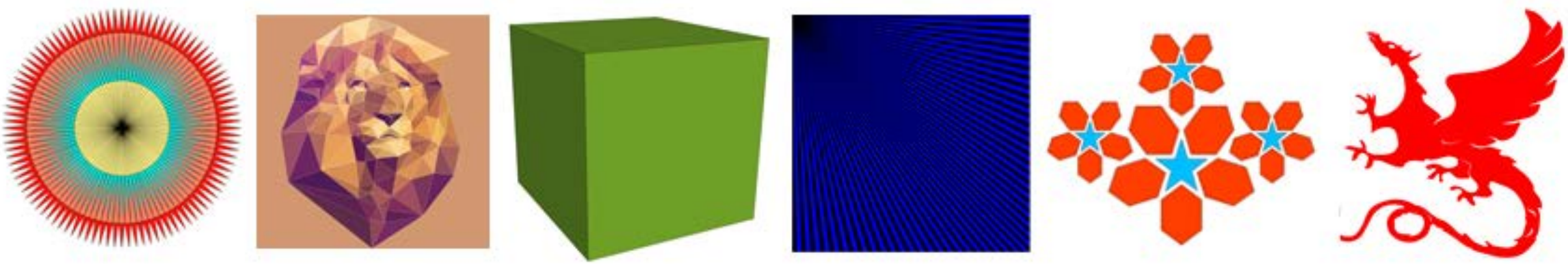
Assignments / Grading

- (5%) Warm-up Math (P)Review
  - Written exercises on basic linear algebra and vector calc. (individually)
- (60%) Four programming assignments
  - Four programming assignments
  - Each worth 15% of overall course grade
- (10%) Take-home quizzes
  - One per lecture
  - Must be turned in BY YOU at the beginning of the next lecture
- (20%) Midterm / final
  - Both cover cumulative material seen so far
- (5%) Class participation
  - 3% per-class comments on website, 2% other contributions to class

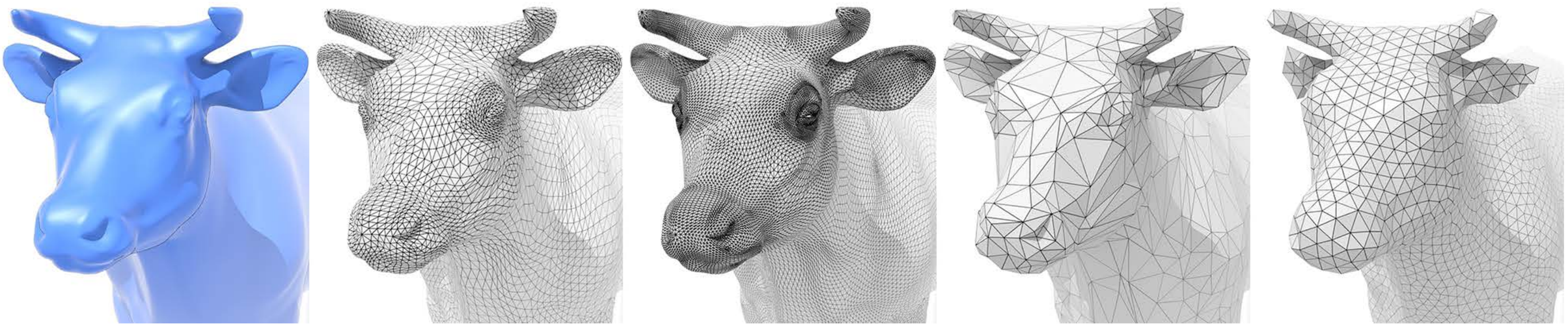
Assignment 0: Math (P)Review



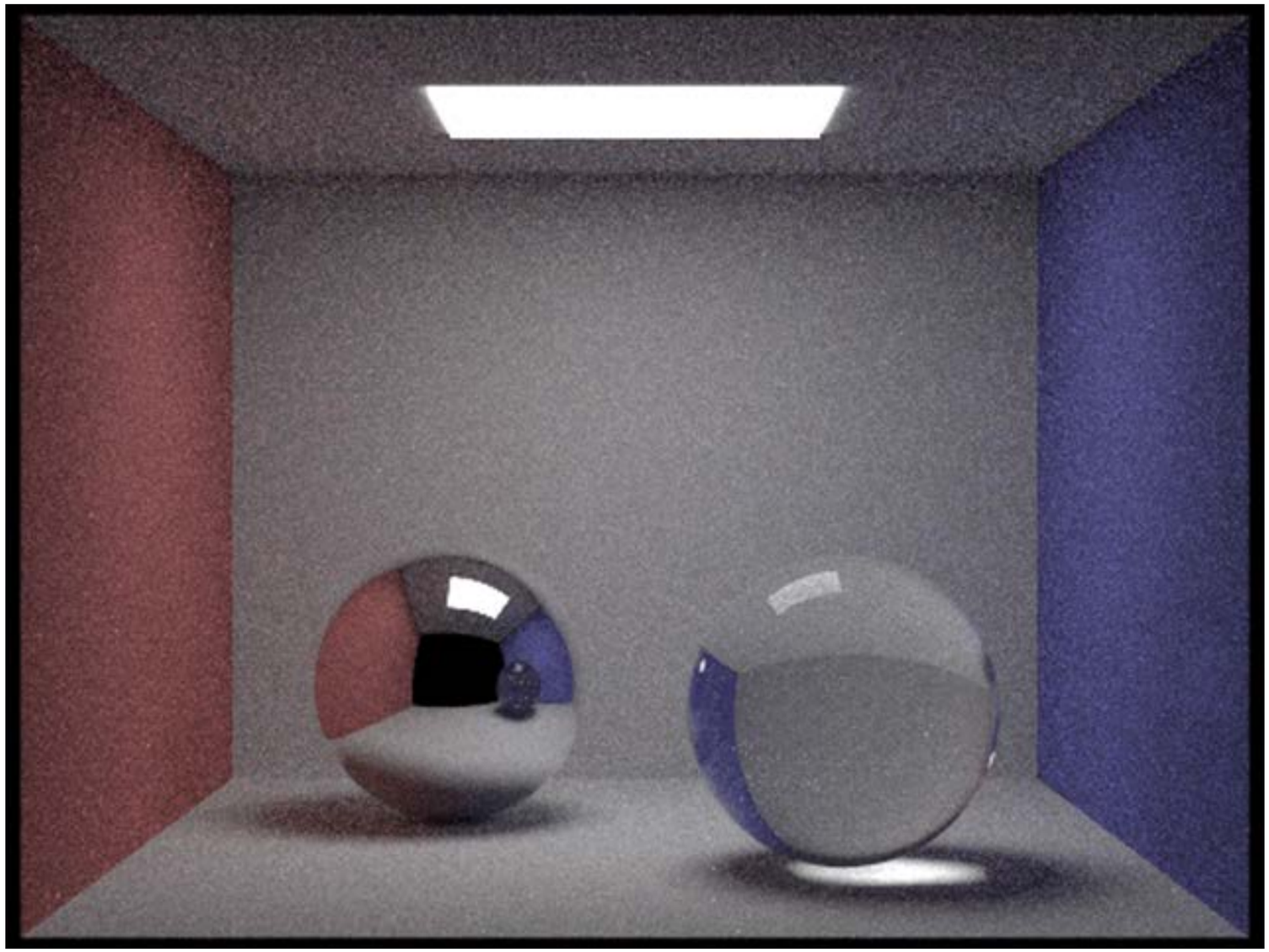
Assignment 1: Rasterization



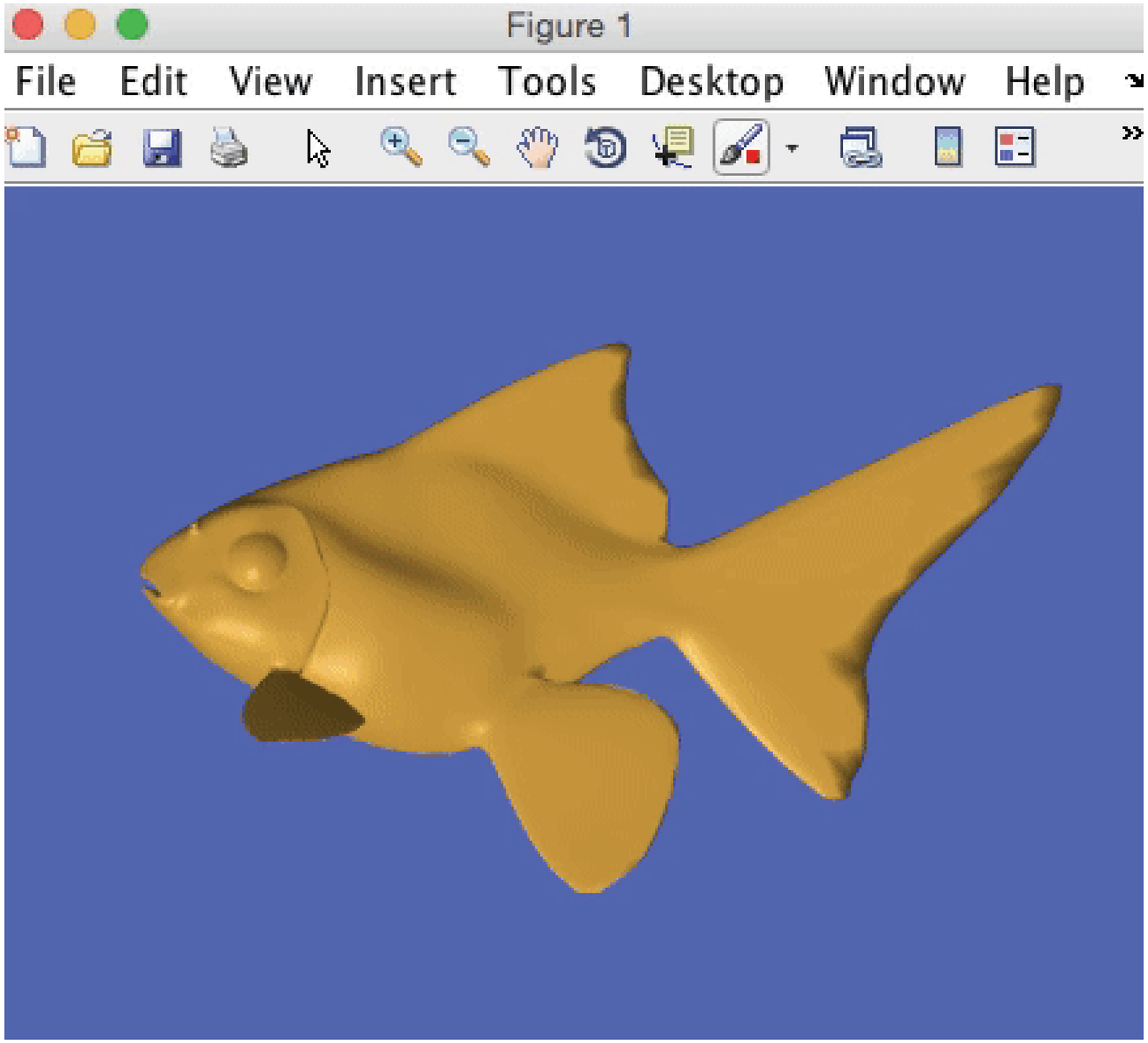
Assignment 2: Geometric Modeling



Assignment 3: Photorealistic Rendering



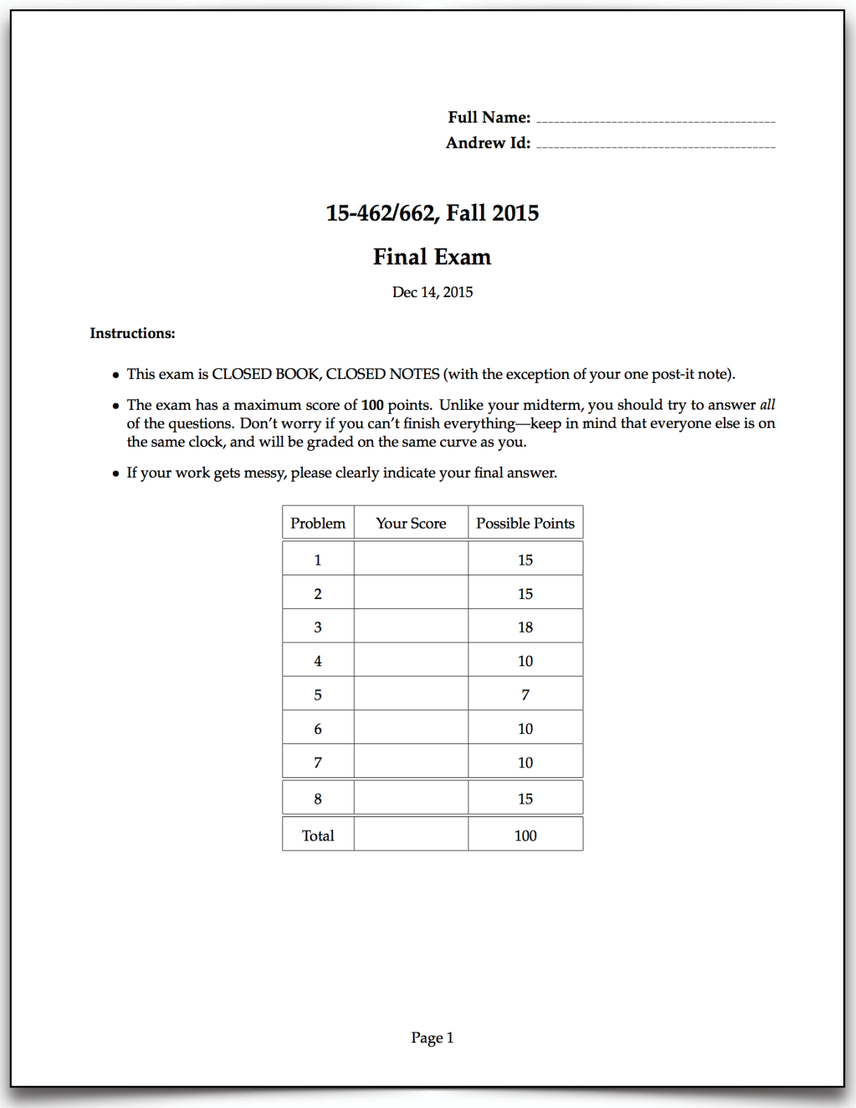
Assignment 4: Animation



(cribbed from Alec Jacobson)

Midterm / Final

- Both cover cumulative material seen so far
- In-class, proctored exam
- Can bring one sticky note (both sides) w/ any information on it



Full Name: _____

Andrew Id: _____

15-462/662, Fall 2015

Final Exam

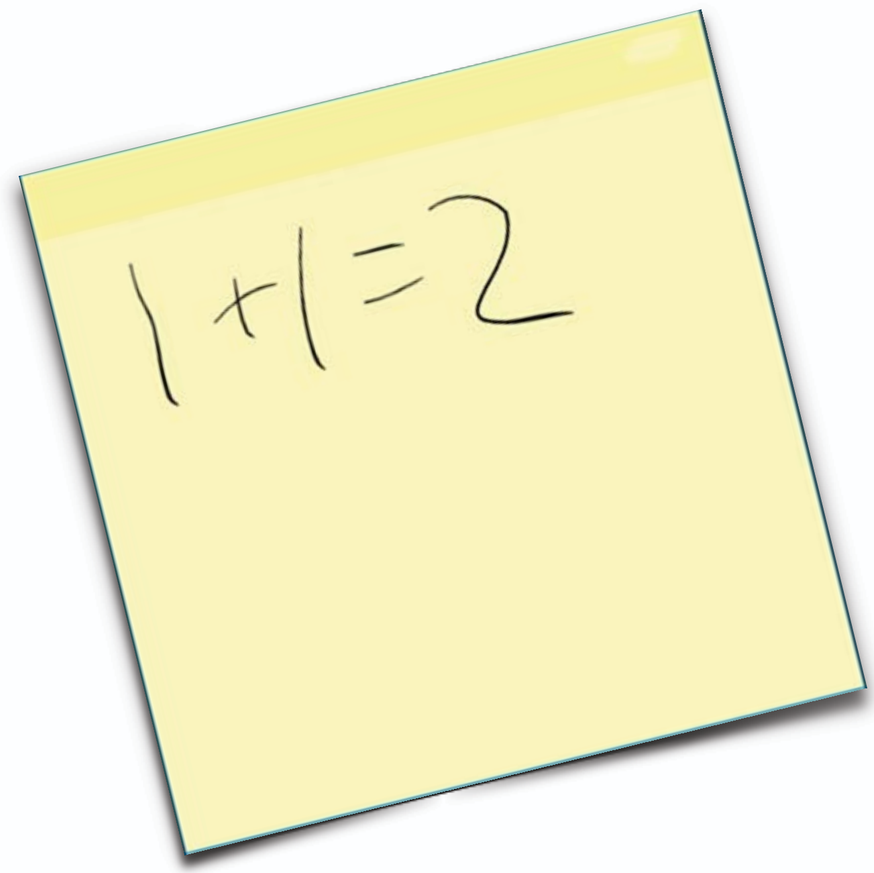
Dec 14, 2015

Instructions:

- This exam is CLOSED BOOK, CLOSED NOTES (with the exception of your one post-it note).
- The exam has a maximum score of **100** points. Unlike your midterm, you should try to answer *all* of the questions. Don't worry if you can't finish everything—keep in mind that everyone else is on the same clock, and will be graded on the same curve as you.
- If your work gets messy, please clearly indicate your final answer.

Problem	Your Score	Possible Points
1		15
2		15
3		18
4		10
5		7
6		10
7		10
8		15
Total		100

Page 1



Getting started

- Sign up for the course on Piazza
  - http://piazza.com/cmu/fall2016/15462
- Create an account on the course web site:
  - http://15462.courses.cs.cmu.edu
  - `signup code` is on Piazza!
- Review "Course Information" in detail
  - lots of important information about grading, late policy, …
- Note: *no official textbook* (but see course website for recommendations)

[Home] [Course Info] [Feed] [Lectures] [Quizzes] [Admin Console] Welcome, Keenan [Logout]

COMPUTER GRAPHICS (CMU 15-462/662)

Basic Info

Mon/Wed 1:30-2:50pm
GHC 4307
Instructor: Keenan Crane

See the course info page for more info on policies and logistics.

Getting Started

To get started with the class you need to do just three things:

1. Sign up for the course Piazza.
2. Sign up for an account on this webpage. (We'll give you the signup code on Piazza, so sign up for that first!)
3. Carefully read through the Course Info.

Note also that *all assignments for the semester are available* ***now***, at the bottom of this page. So if you're eager to get started, go for it! Due dates are listed inline in the course schedule below.

Fall 2018 Schedule

Aug 27 (Mon)	Course Introduction Assignment 0.0 out
Aug 29 (Wed)	Math Review Part I: Linear Algebra
Sep 3 (Mon)	**No class (Labor Day)** Assignment 0.0 due/Assignment 0.5 out
Sep 5 (Wed)	Math Review Part II: (Vector Calculus)
Sep 10	Drawing a Triangle

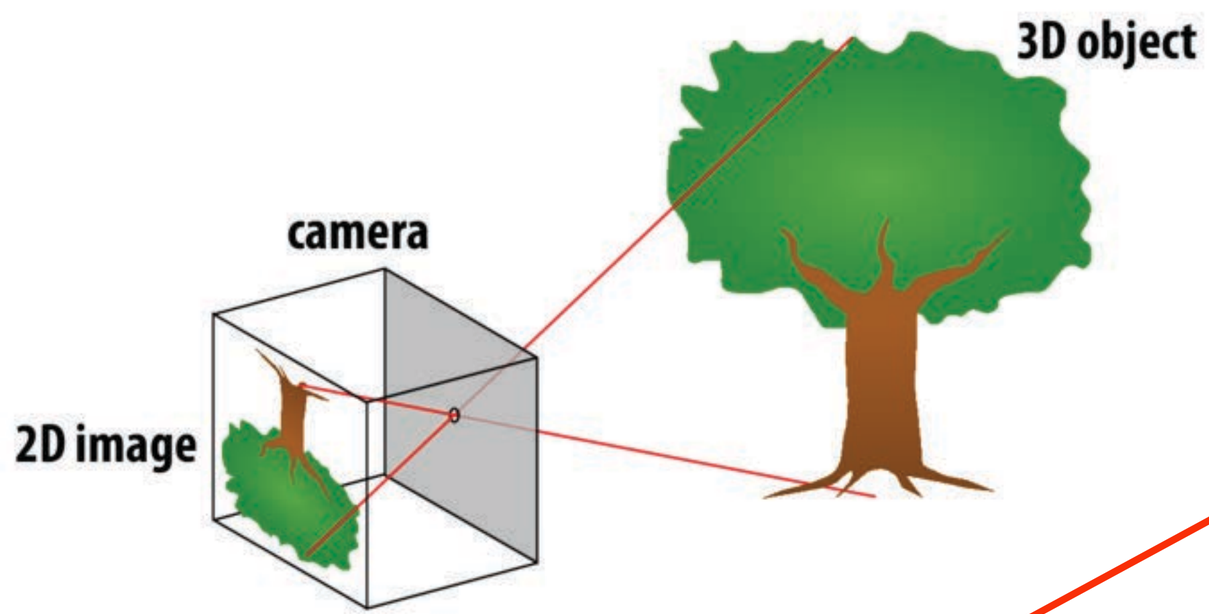
The course web site

Can discuss lecture slides directly on course web page

<u>Each lecture you will make one question/comment about the lecture</u>

Perspective projection

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



CMU 15-462/662, Fall 2015

Previous | Next --- Slide 30 of 65 Back to Lecture Thumbnails

Add Private Note

"Add private note" button:
You can add notes to yourself about this slide here.

kayvonf about an hour ago

Question: During class Keenan asked a question about why do objects look smaller when hey are viewed at a distance. I liked one of the arguments made because it appealed to the angle subtended by an object. Could someone elaborate on that here?

Slide comments and discussion

Prompt Edit Delete Archive [0 Upvote Downvote]

Take-home Quizzes

- Most lectures will have a "take-home quiz" to solidify concepts
- Turn in at the beginning of next lecture
  - *you have to turn it in yourself!* (not a friend)
- Quantized grading scale:
  - 100% — correct idea, details are correct
  - 85% — correct idea, some details are wrong
  - 60% — good faith attempt, but clearly wrong
  - 35% — no answer, but you explain what you didn't understand
  - 25% — no answer, just write "I don't know"
  - 0% — nothing handed in / too sloppy to read
- Remember to write your name / AndrewID!

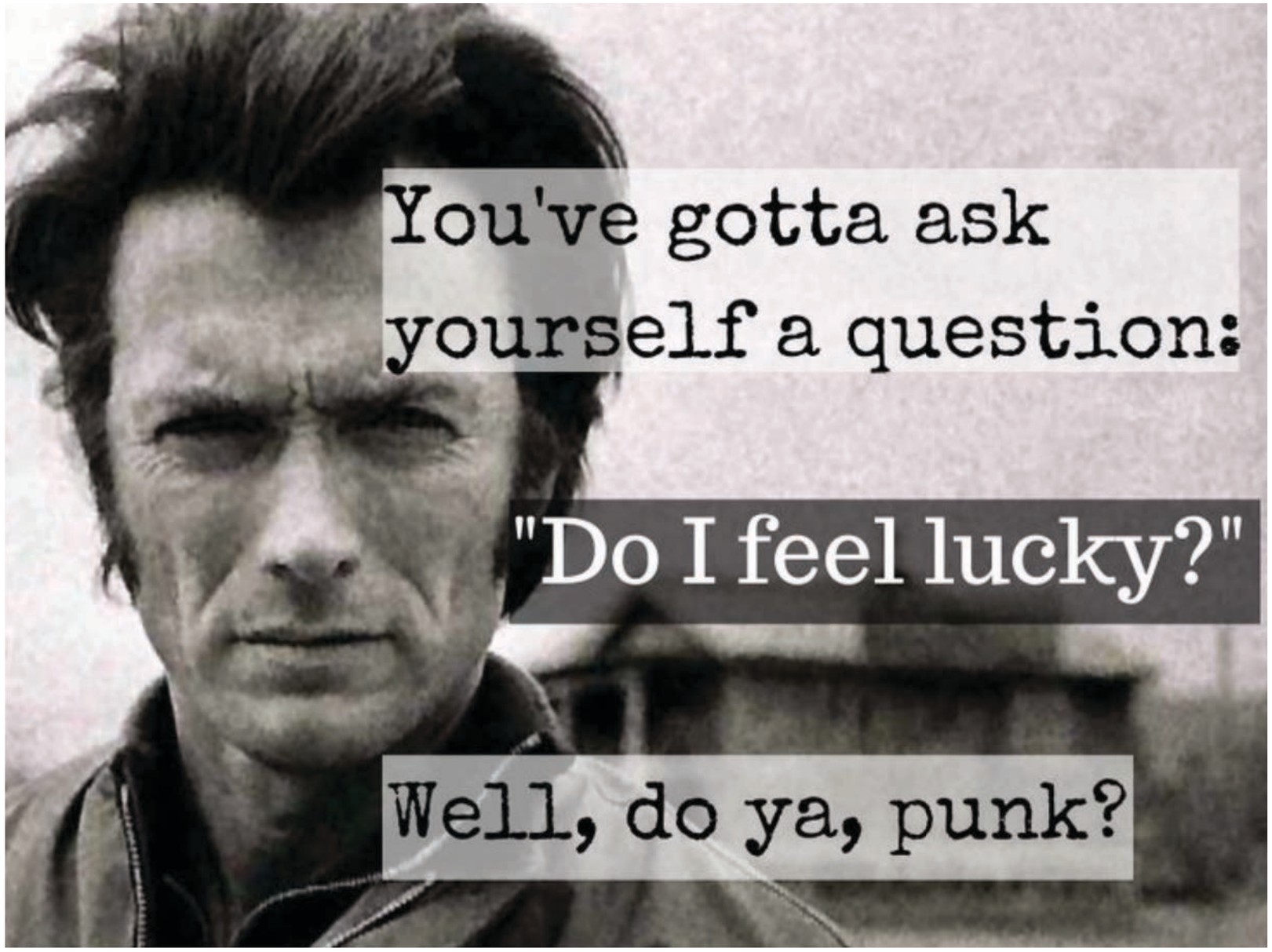
Assignments

- Short math review (linear algebra/vector calculus)
  - you should know *most* of this already!
- Four major coding assignments
  - A1 — DrawSVG
  - A2 — MeshEdit
  - A3 — PathTracer
  - A4 — Animation
- Coding in C++, skeleton provided via GitHub repository
- Documentation in Wiki
  - we will not specify every little detail! Learn good SE
  - part of each assignment is to *improve documentation*

Late policy

- Daily Quizzes
  - You can skip up to 4 with no penalty
- Programming assignments
  - You have five "late day points" for the entire semester
  - Can use on first four programming assignments only
  - No more late points? 10% penalty per day
  - No assignments will be accepted more than 3 days past the deadline

Cheating Policy



Let's keep it simple: if you are caught cheating, you will get a zero for the <u>entire course</u> (not just the assignment).

Our philosophy

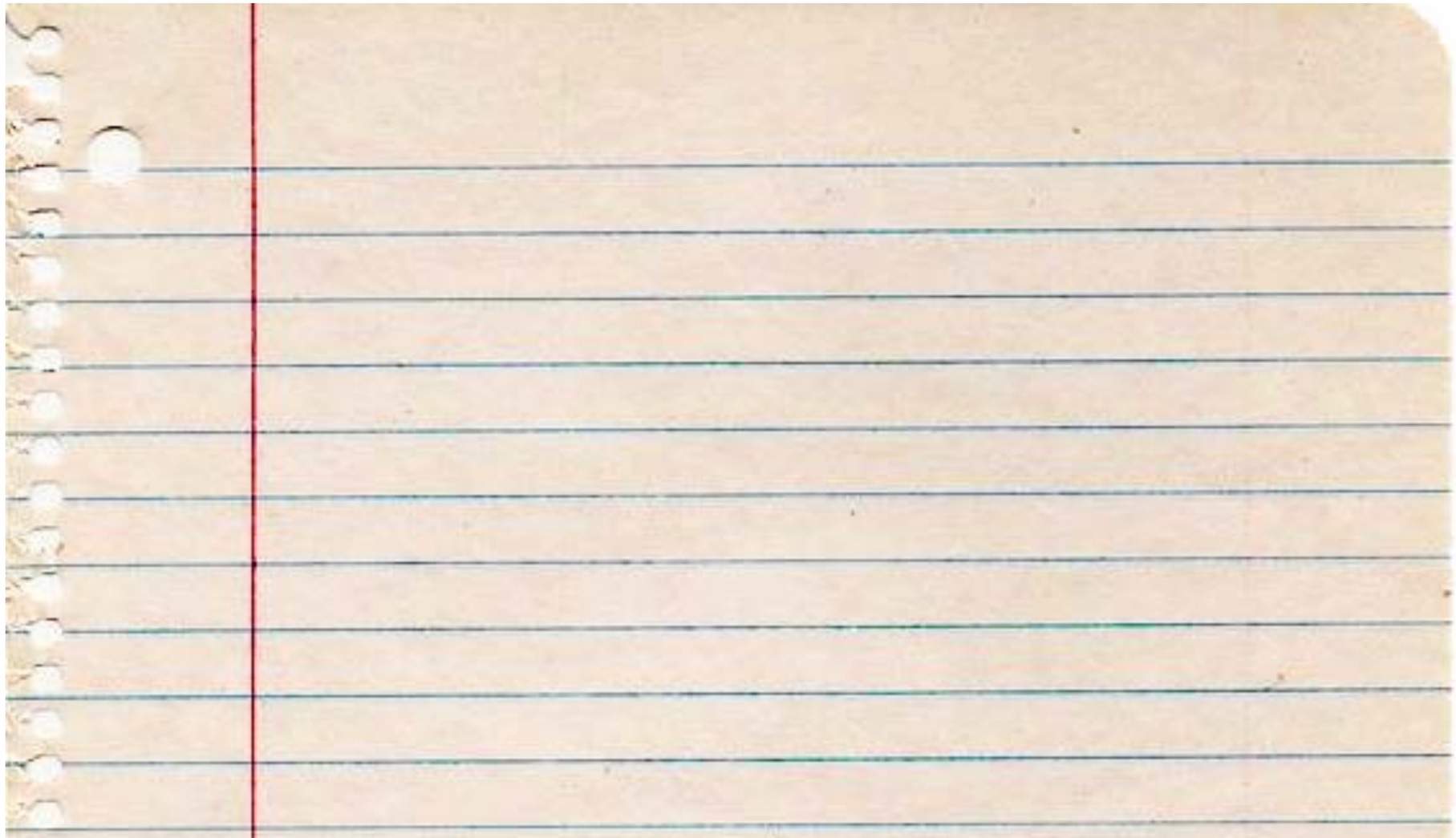
- We want a very active class: come to class, participate in the class, contribute to the web site
- Challenging assignments (with tons of "going further" opportunities: see what you can do!)
- Challenging exams (see what you can do!)
- *Very reasonable grading* (ask your friends! :-))

TLDR: Here's the algorithm for success :-)

- TODAY: sign up for Piazza / course web page
- EVERY WEEK:
  - Attend class
    - turn in <u>your</u> quiz *at the beginning of class*
  - After each class:
    - make one comment on course web page
    - complete the take-home quiz
- Do the coding assignments
  - due every ~2.5 weeks
  - come to office hours!!
- Study for the midterm & final
- Don't cheat :-)

QUIZ 0

- This one is easy: write one thing you want to learn from this course and/or one reason you decided to take the course.
- Write answer on <u>physical paper</u>.
- Must be turned in BY YOU in-class at the START of the next lecture.



See you next time!

- Next time, we'll do a math review & preview
  - Linear algebra, vector calculus
  - Help make the rest of the course easier!