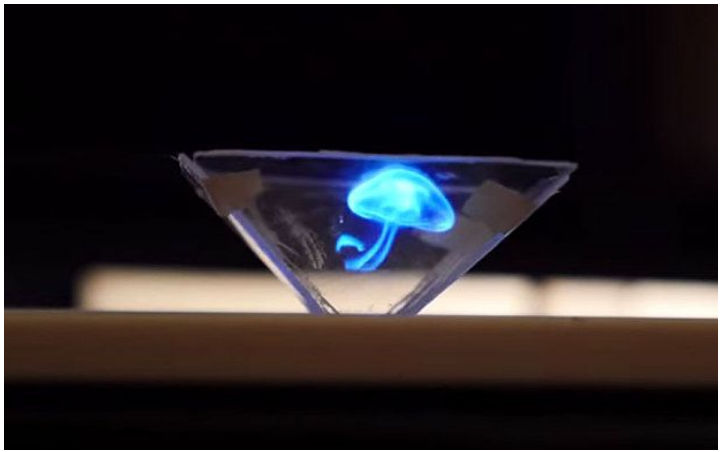
Futuristic Pepper's Ghost Approximation (FPGA)

Jeremy McCulloch, Adam Potter, Sreya Vangara

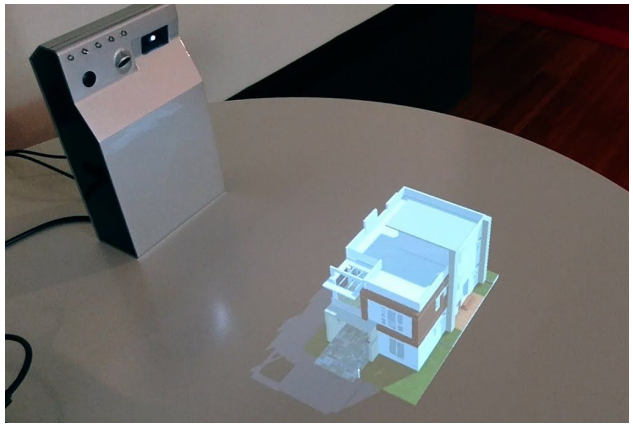
Inspiration



- Pepper’s Ghost tabletop simulation
- Render from the user’s perspective

Our Take

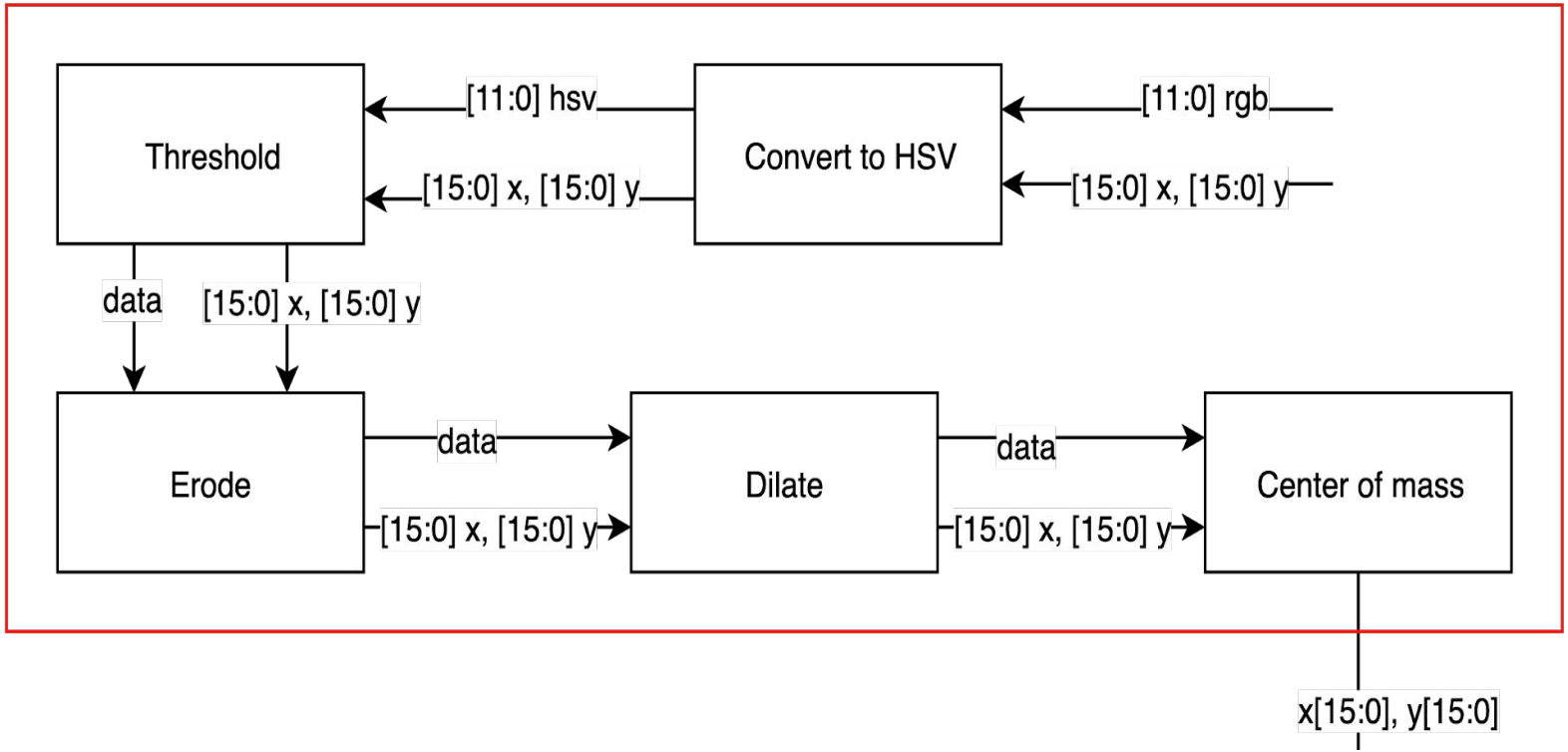
HoloLamp



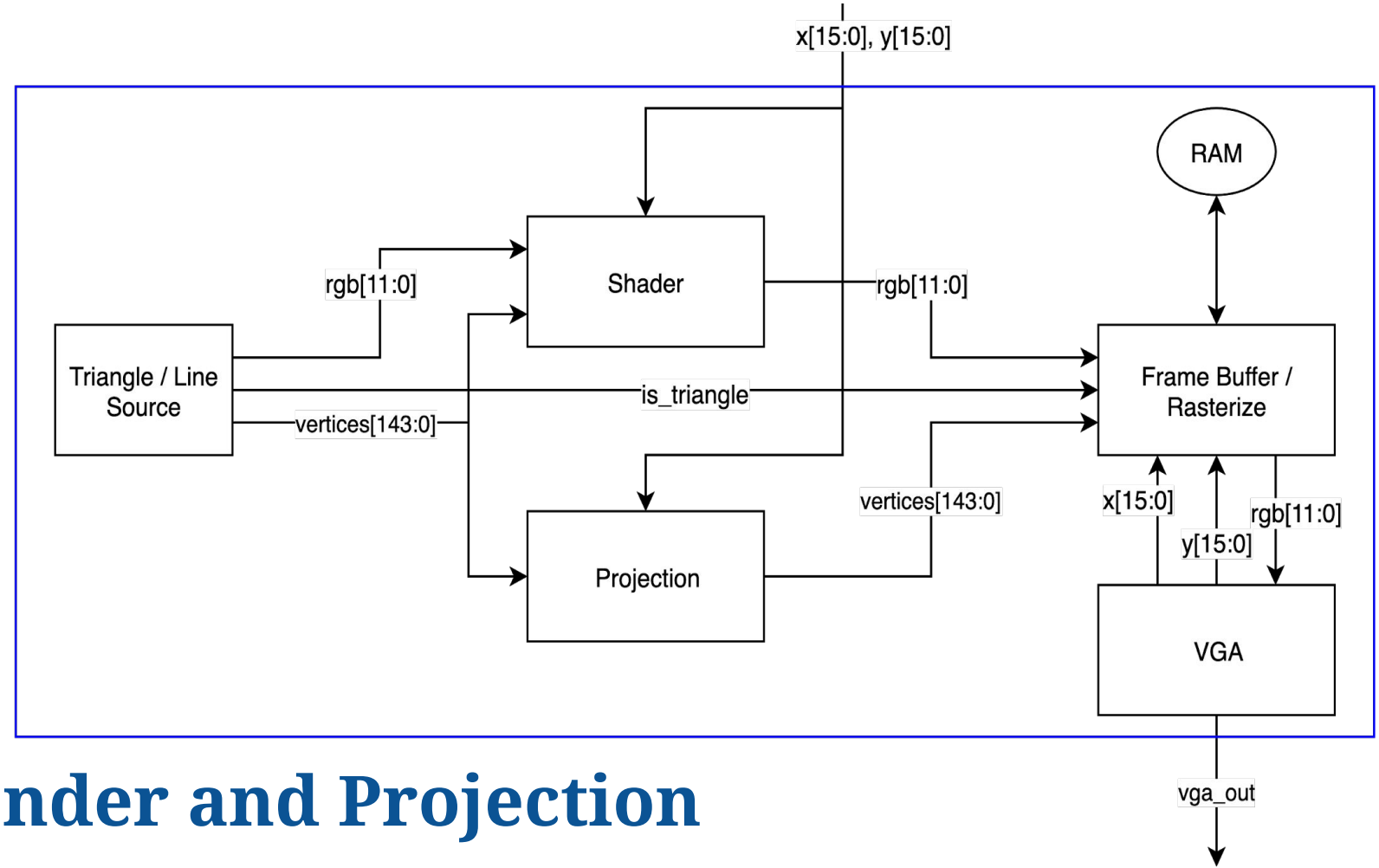
- Transform and project onto table
- Track user, re-render to simulate object

Functionality Goals

BLOCK DIAGRAMS



Computer Vision



Render and Projection

Hardware

- Camera
- VGA Projector
- Projector and Camera Mount
- Green Hat

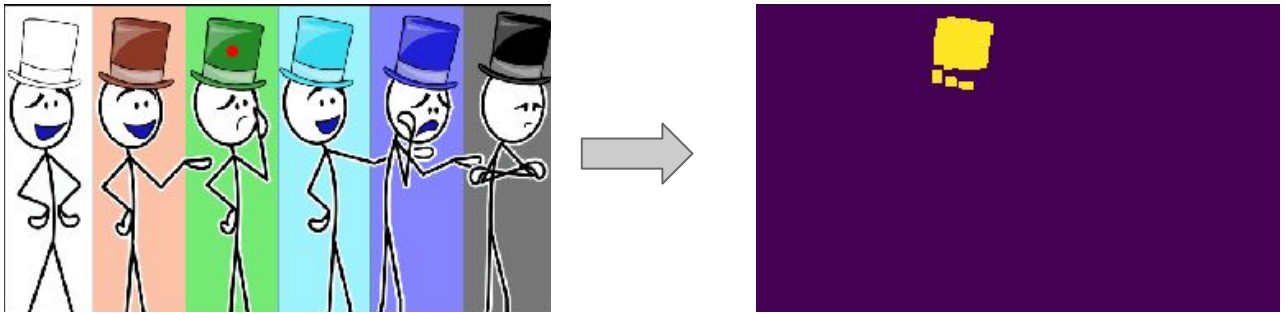
Limitations

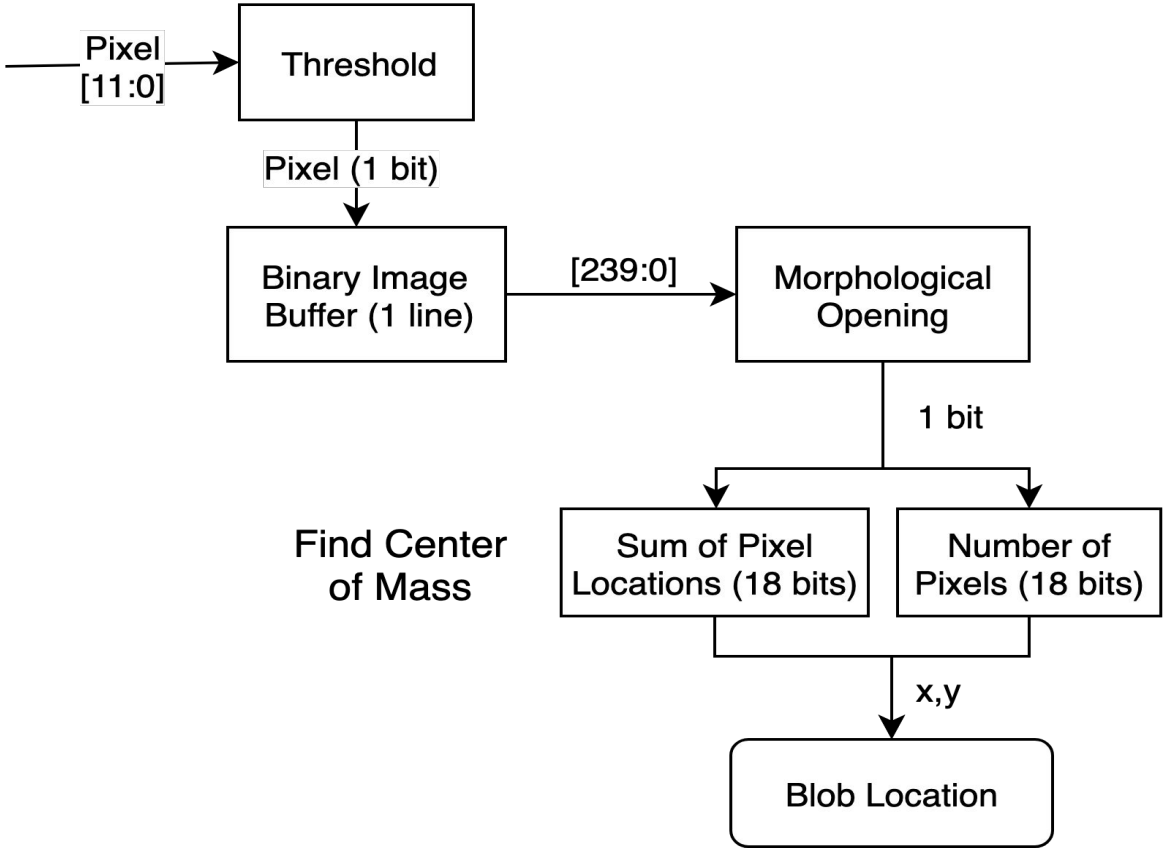
- Board memory, frame rate, and camera specs (FOV)

MODULES

Computer Vision

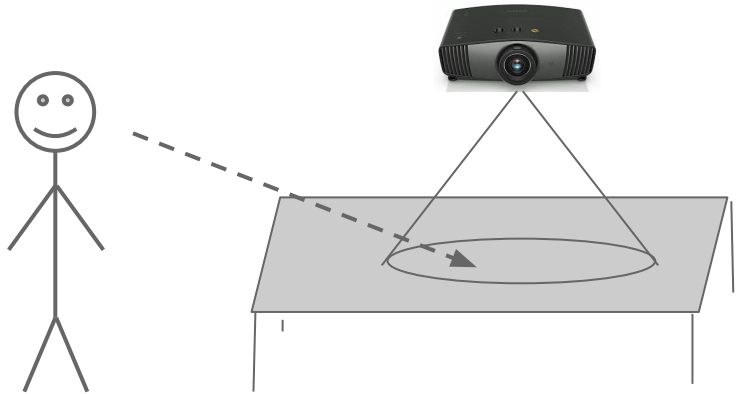
- Input: Camera Data
- Output: (x,y) of user's head
- Use chroma keying to pick out green hue
- Use erode and dilate to find largest green blob




Pixel
[11:0]
Threshold
Pixel (1 bit)
Binary Image
Buffer (1 line)
[239:0]
Morphological
Opening
1 bit
Find Center
of Mass
Sum of Pixel
Locations (18 bits)
Number of
Pixels (18 bits)
x,y
Blob Location

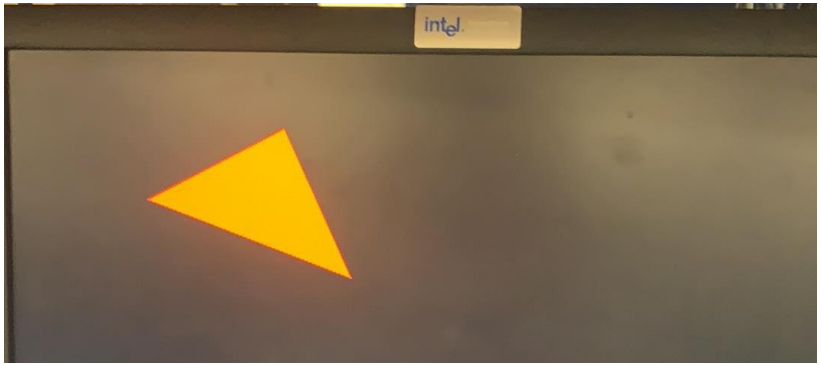
Projection

- Calculate where to draw a point on table based on user’s position and model position
- Adjust brightness of triangle based on angle of plane to user



Rasterize

- Use projection to map all vertices of triangle
- Iterate points in triangle, interpolate z coordinate
- Framebuffer to store RGB and z coordinate for each pixel



Timeline

Rasterize
Yesterday!

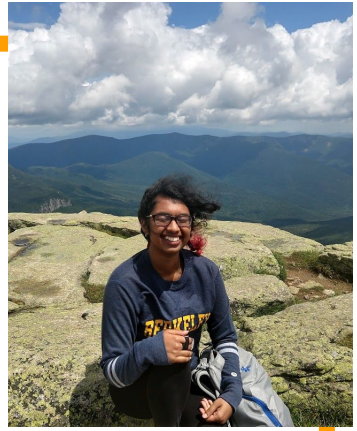
Projection
11/17

Shading
11/17

CV
11/17

Integrate
11/24

	Week 1 (11/04 - 11/10)	Week 2 (11/11 - 11/17)	Week 3 (11/18 - 11/24)	Week 4 (12/2 - 12/8)
Jeremy	Rasterize + Framebuffer	Render Simple Model	Integration	Attempt to Add Animation
Adam	CV Prototyping	Shader, Better Projection	Integration, Mounting	Demo
Sreya	Camera working	Computer Vision	Integration	Add Wireframe



Q&A