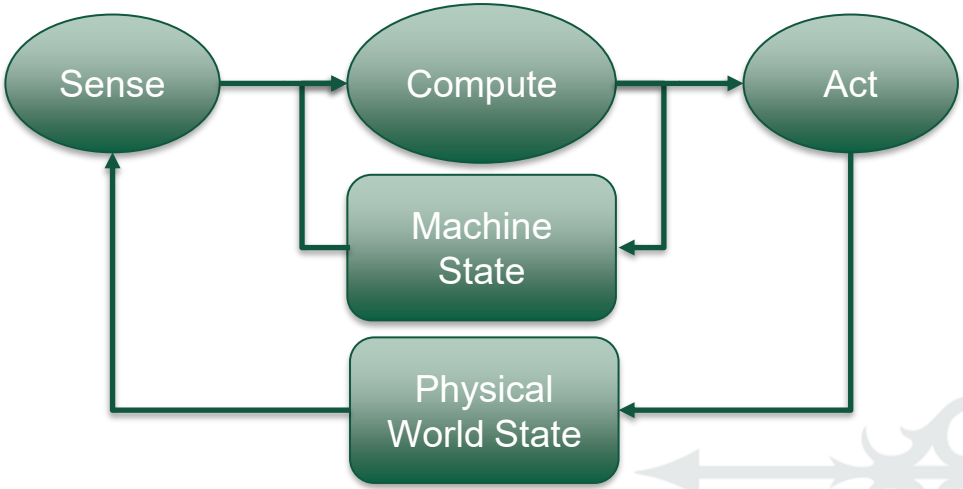
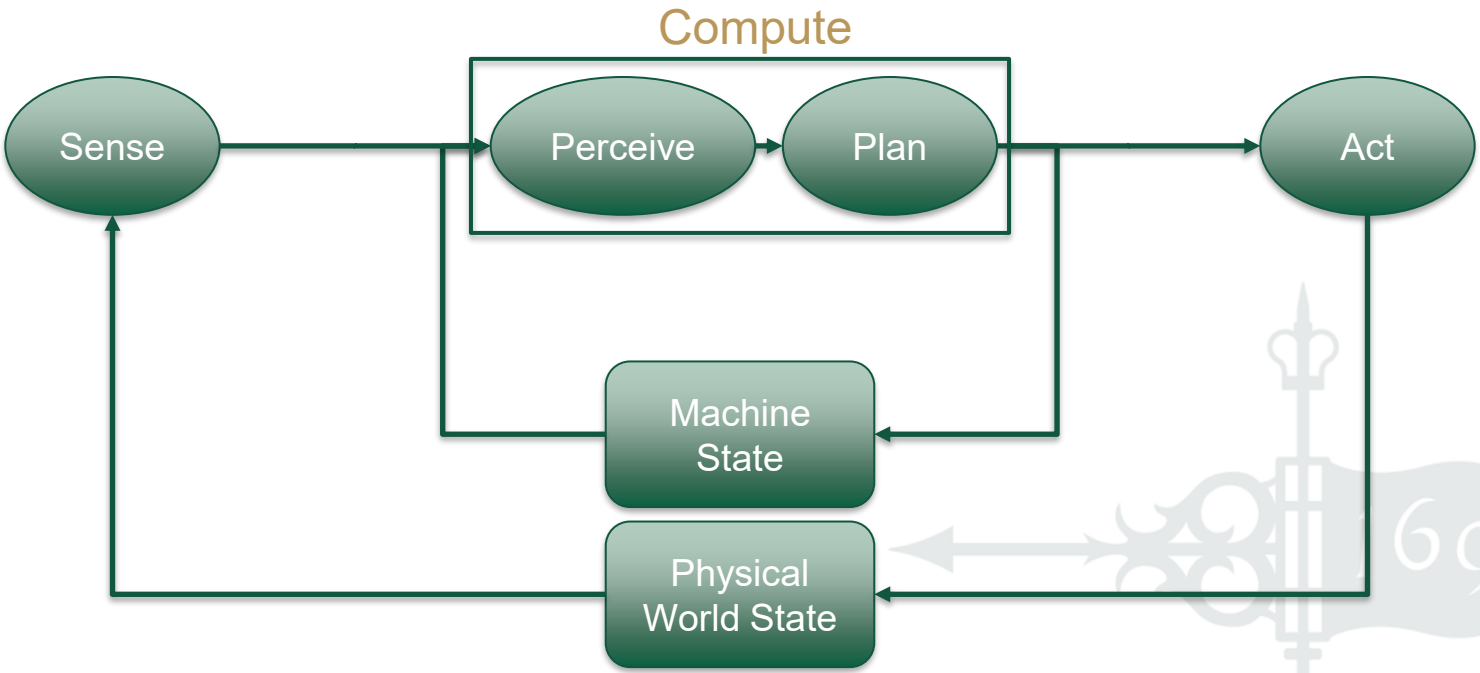
Perception

CSCI 420-04 Robotics

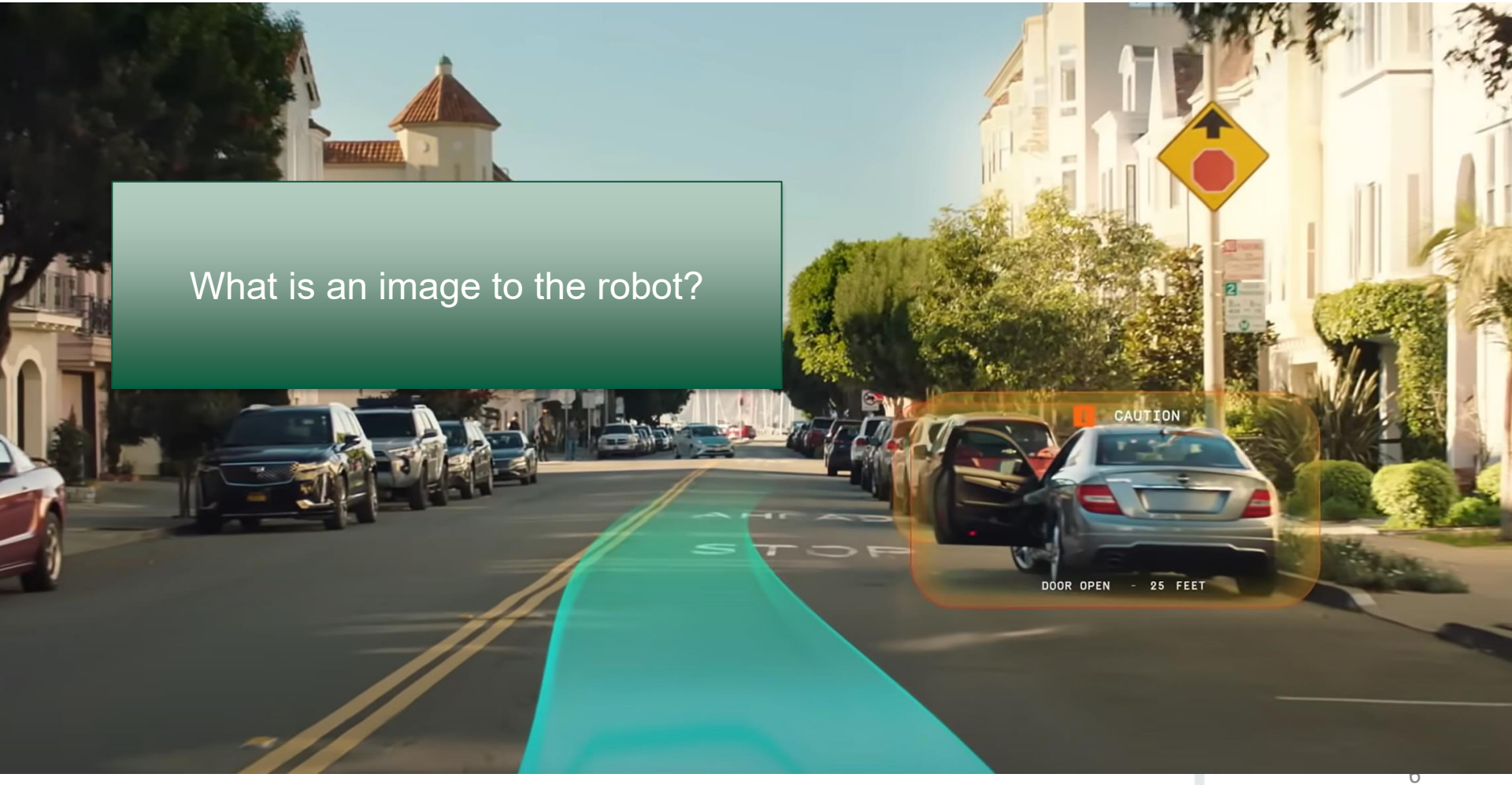
WILLIAM & MARY

CHARTERED 1693


Sense
Compute
Act
Machine State
Physical World State


Compute
Sense
Perceive
Plan
Act
Machine State
Physical World State


RADAR
CAMERA


What is an image to the robot?
CAUTION
DOOR OPEN - 25 FEET

sensor_msgs/Image Message

std_msgs/Header header

uint32 height

uint32 width

string encoding

RGB, BGR, MONO, HSV

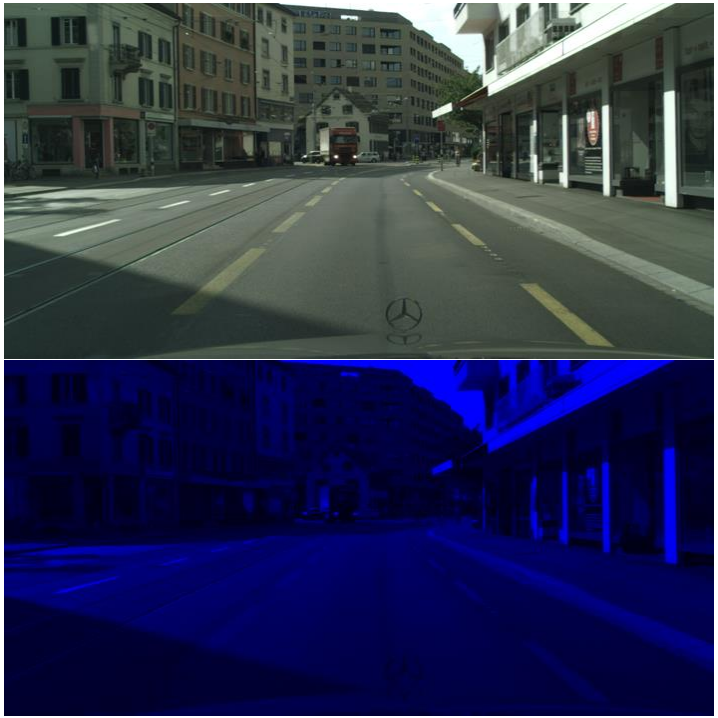
uint8 is_bigendian

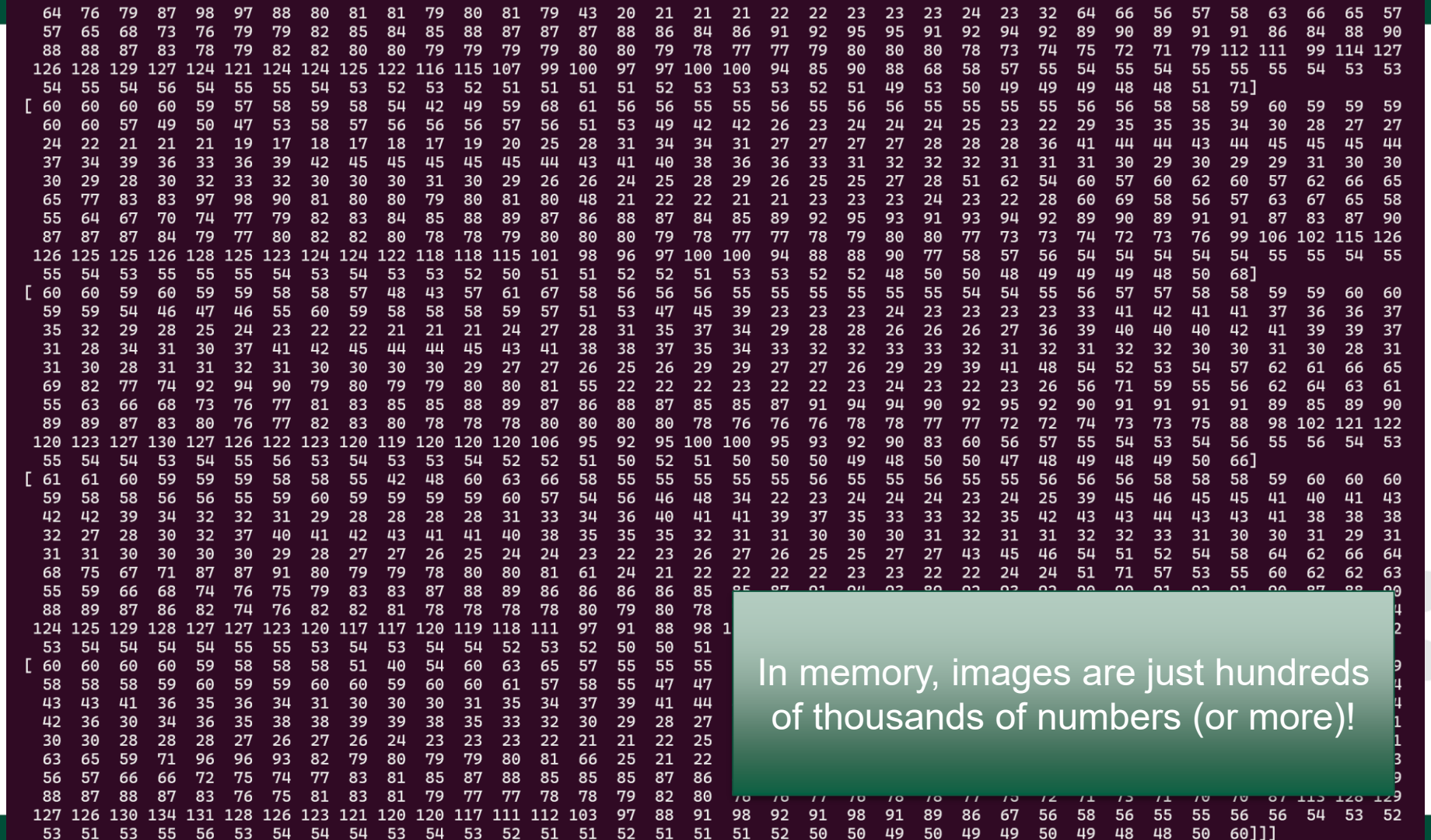
uint32 step

Storage data

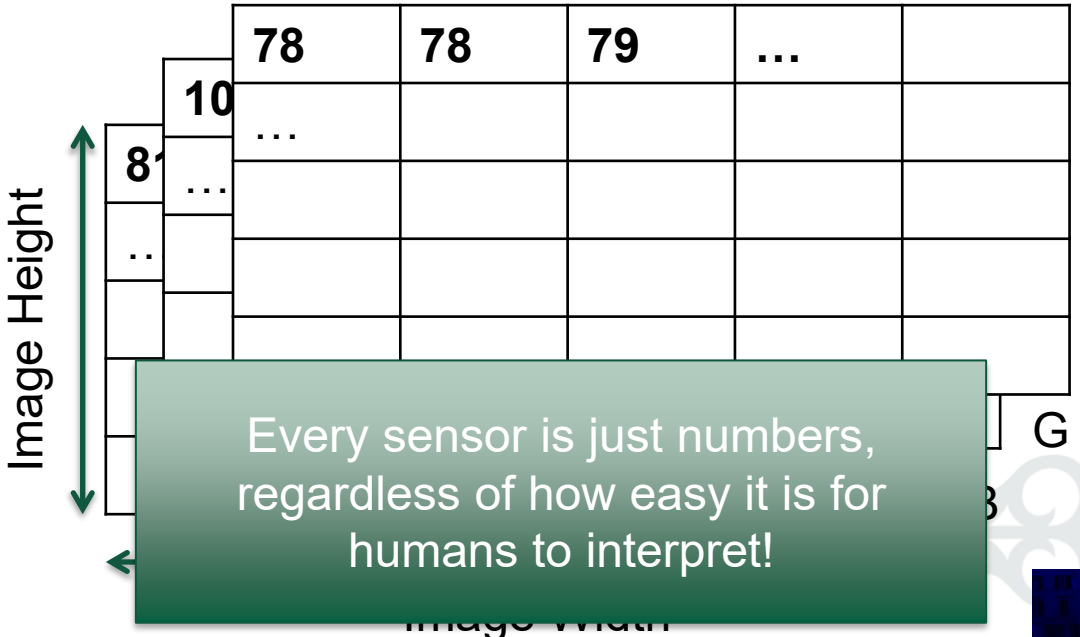
uint8[] data

RGB



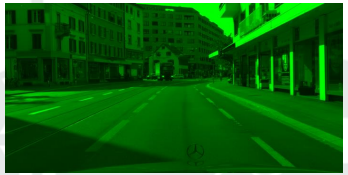

In memory, images are just hundreds of thousands of numbers (or more)!

RGB in memory



Every sensor is just numbers, regardless of how easy it is for humans to interpret!

R



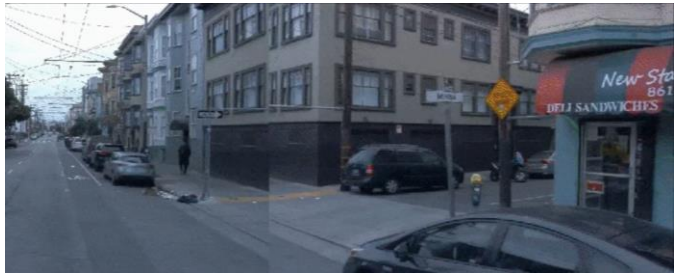
Perception

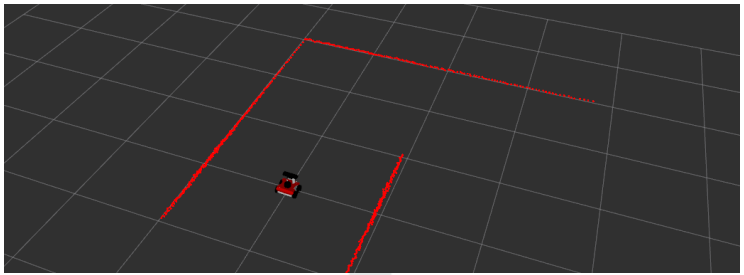
Perception refers to the ability of an autonomous system to collect information and extract relevant knowledge from the environment.

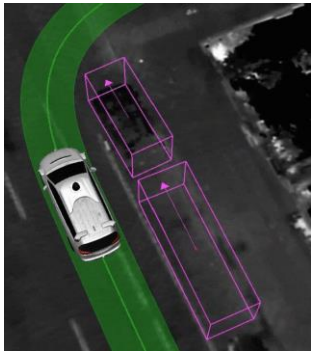
Pendleton, Scott Drew, et al. "Perception, planning, control, and coordination for autonomous vehicles." Machines 5.1 (2017)

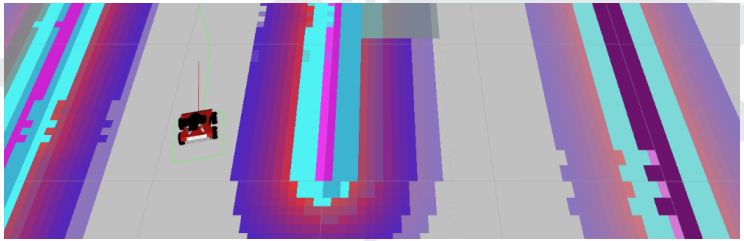
Perception

Extract high-level abstraction from sensor data









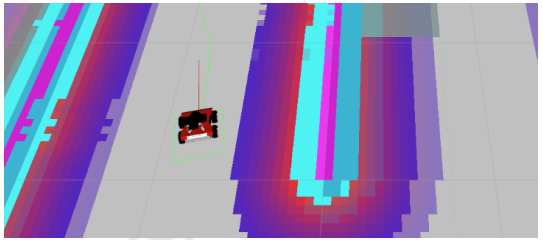
Color: Red

Main types of Perception

Classification

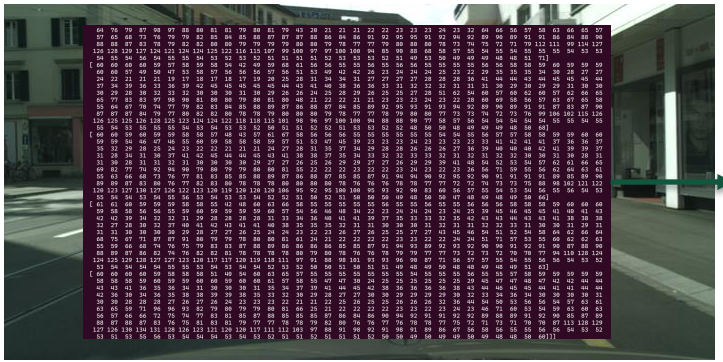
Object Detection

Interpretation

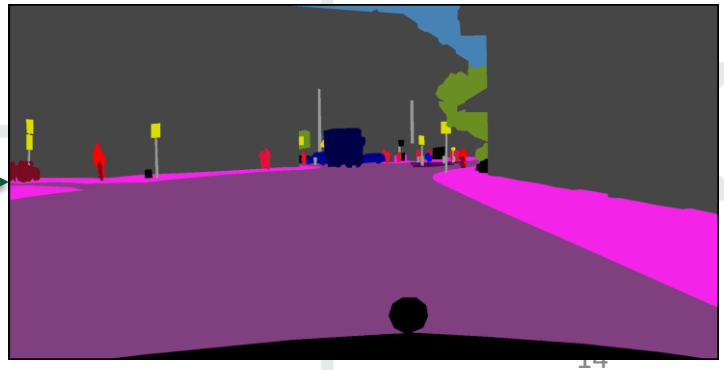


What is Perception?

- Goal: Extract Relevant Knowledge
- From: Raw sensor data
- To: Classify/Detect/Interpret



Perception



How?

- Perception abstracts the environment
  - Data (Image) Processing
    - A defined process (algorithmic)
    - Human-defined semantics
  - Machine Learning
    - Gather data to estimate behavior
    - Computer learns the function

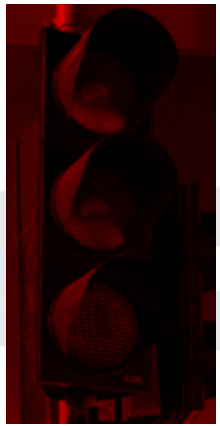
Image Processing

- Color filtering
- Blurring
- Smoothing
- Background subtraction
- Edge Detection
- …

Color Filtering

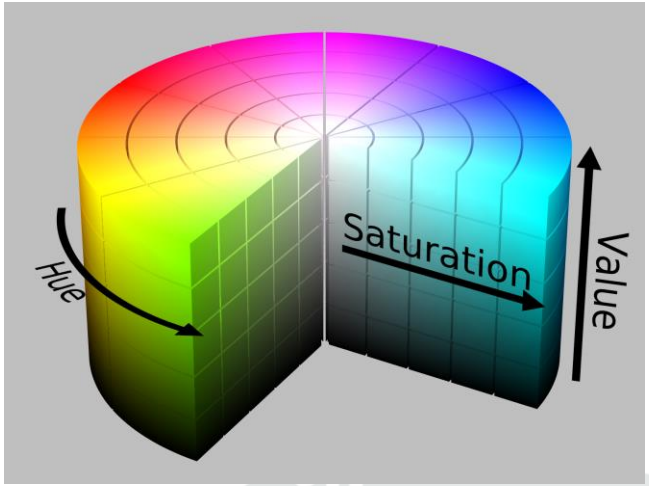
- Idea: Keep only the parts of the image that are a certain color, the count the amount

The green filter can identify the green light!



Color Filtering

- How do we expand to more colors?
- HSV stores color based on *hue*



https://en.wikipedia.org/wiki/HSL_and_HSV#/media/File:HSV_color_solid_cylinder_saturation_gray.png

Color Filtering

- How do
more co
- HSV sto
on *hue*

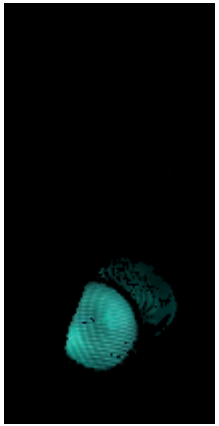
```
# Convert from BGR to HSV
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

# Set up filter values
lower_bound = np.array([85, 128, 0])
upper_bound = np.array([90, 255, 255])

# Identify pixels that meet the filter
mask = cv2.inRange(hsv, lower_bound, upper_bound)

# Apply the filter
result = cv2.bitwise_and(image, image, mask=mask)
```

Color Filtering

Original

Mask

Result



```
# Convert from BGR to HSV
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

# Set up filter values
lower_bound = np.array([85, 128, 0])
upper_bound = np.array([90, 255, 255])

# Identify pixels that meet the filter
mask = cv2.inRange(hsv, lower_bound, upper_bound)

# Apply the filter
result = cv2.bitwise_and(image, image, mask=mask)
```

Different programs use different scales!
OpenCV uses 0-180 for Hue,
and 0-255 for S and V

Color Filtering

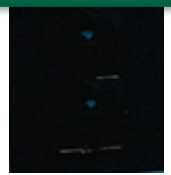
Original

Mask

Result

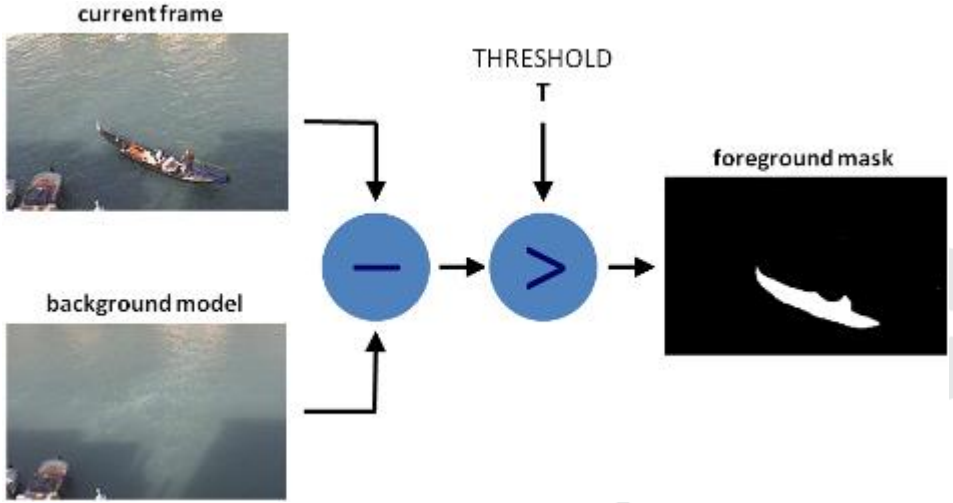
```
# Convert from BGR to HSV
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

# Set up filter values
lower_bound = np.array([85, 128, 0])
upper_bound = np.array([90, 255, 255])

ls that meet the filter
Range(hsv, lower_bound, upper_bound)

ter
twise_and(image, image, mask=mask)
```

When does this excel? When does this struggle?



Background Subtraction

- Idea: the interesting parts move while the background is static

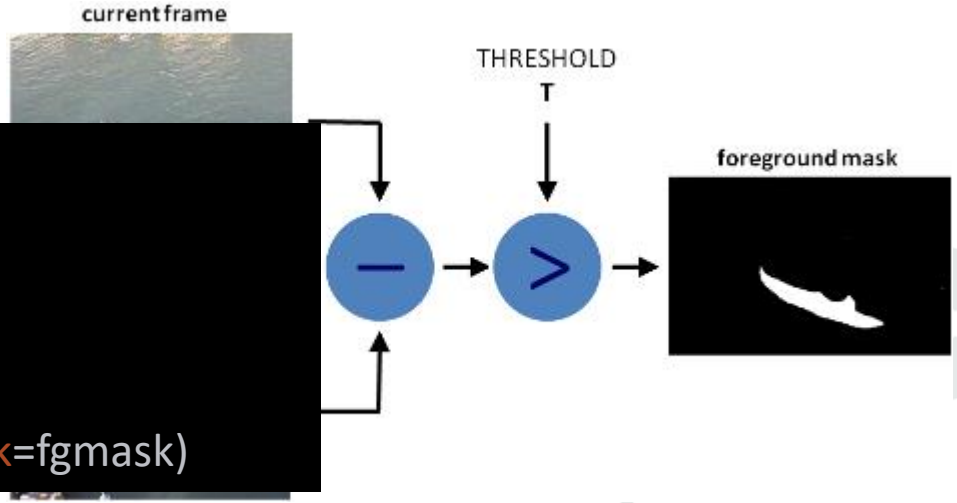


https://docs.opencv.org/3.4/d1/dc5/tutorial_background_subtraction.html

Background Subtraction

- Idea: the interesting parts move while the background is static

```
# Read the image
frame = cv2.imread(f'./{file}')
# Mask for the foreground
fgmask = fpbg.apply(frame)
# Apply the filter
result = cv2.bitwise_and(frame, frame, mask=fgmask)
```



https://docs.opencv.org/3.4/d1/dc5/tutorial_background_subtraction.html

Background Subtraction

Data (Image) Processing

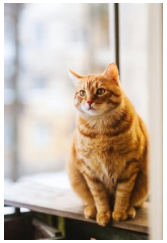
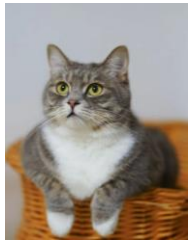
- Pros:
  - Does not require extra inputs
  - Interpretable by humans
  - Simple, quick computations
  - Ready-made libraries
- Cons:
  - Only works for simple functions

Machine Learning

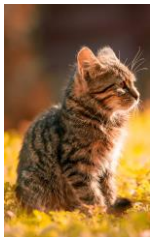
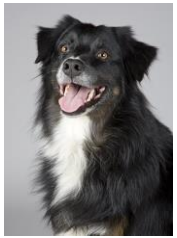
- What if we don’t know how to express what we want, but we know what it looks like?

Machine Learning

- Given enough examples, *learn* the function



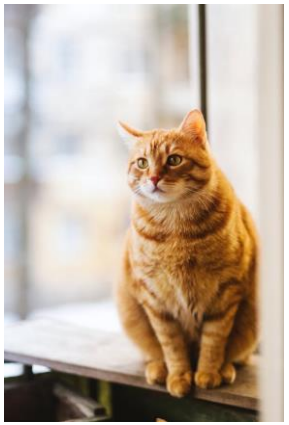

Label: Cat

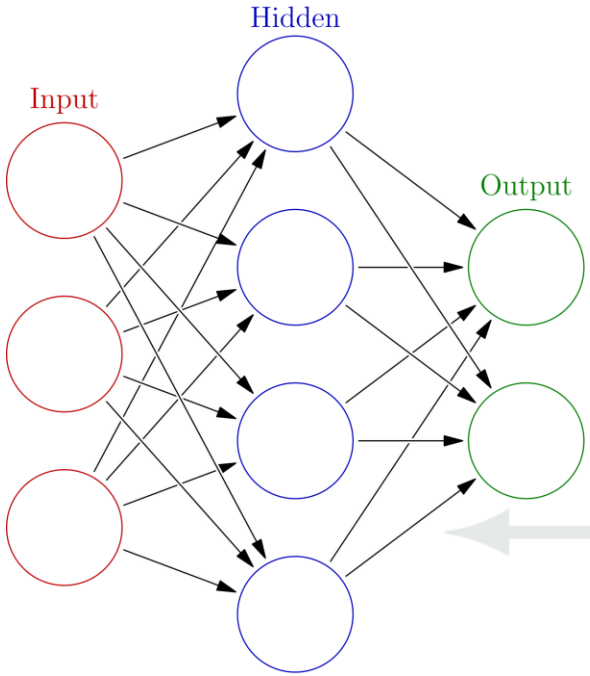
Label: Dog

Machine Learning

- Machine learning needs lots of data to learn what is important
- Data Augmentation!
  - Apply transformations
  - Edit image but keep the label

Neural Networks

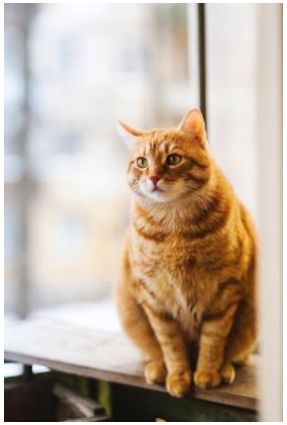




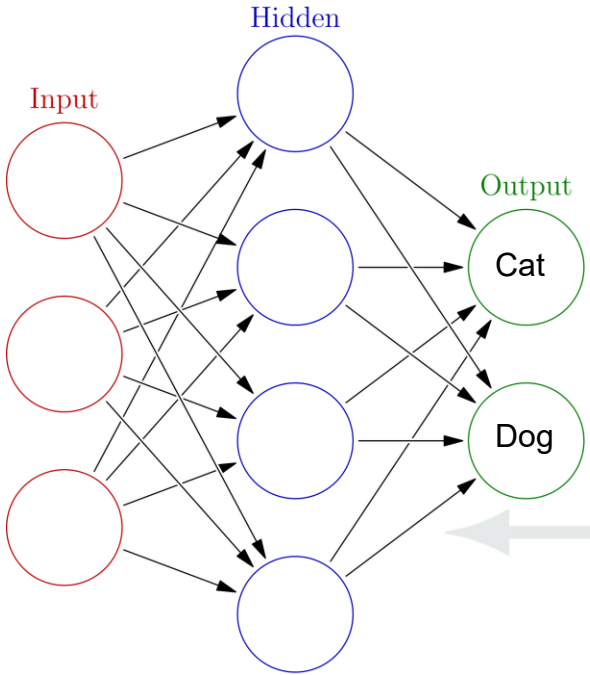
https://en.wikipedia.org/wiki/Neural_network_(machine_learning)#/media/File:Colored_neural_network.svg

Cat

Neural Networks



x_1 x_2 x_3 x_4 x_5 … x_n



https://en.wikipedia.org/wiki/Neural_network_(machine_learning)#/media/File:Colored_neural_network.svg

Cat

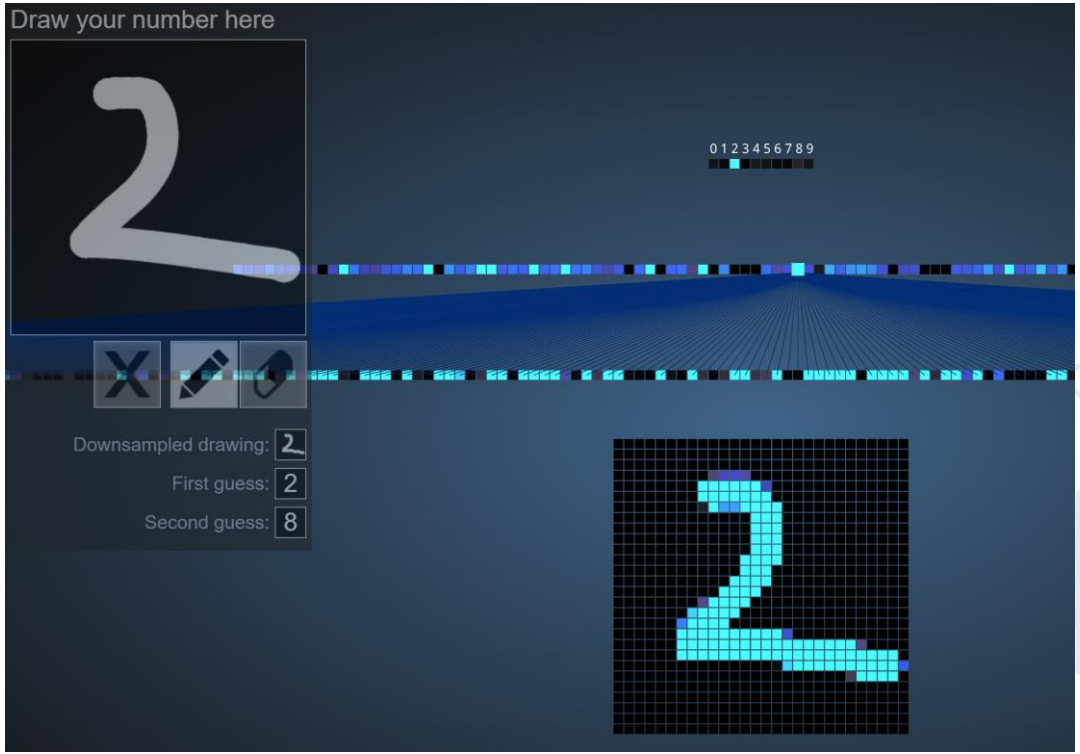
What's in a Neuron?

- Neuron Function: $y = \sigma(w^T x + b)$
  - $y =$ output
  - $x =$ input
  - $\sigma =$ activation function
  - $w =$ weights
  - $b =$ bias

ReLU Activation:

$$\sigma(z) = \begin{cases} 0, z < 0 \\ z, z \geq 0 \end{cases}$$

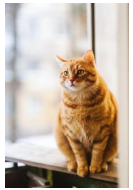
Visualizing NN

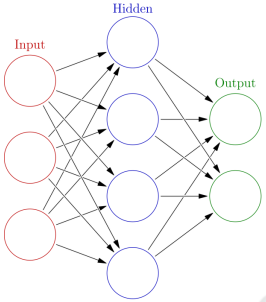


Try it! https://adamharley.com/nn_vis/mlp/2d.html

How do NNs train?

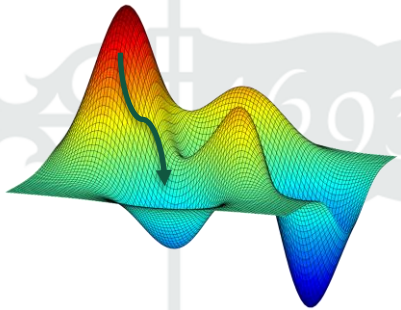
- For each image
  - Evaluate the image
  - Compute Error:
    - Here, binary cross entropy loss: 1.46182
  - Adjust weights based on Loss





https://en.wikipedia.org/wiki/Neural_network_(machine_learning)#/media/File:Colored_neural_network.svg

Cat: 30%
Dog: 70%



Machine Learning

- Pros:
  - Learn arbitrary functions
  - Improves with more data
- Cons:
  - Requires huge amounts of data and compute
  - Difficult to interpret
  - Brittle to changes in data distribution