

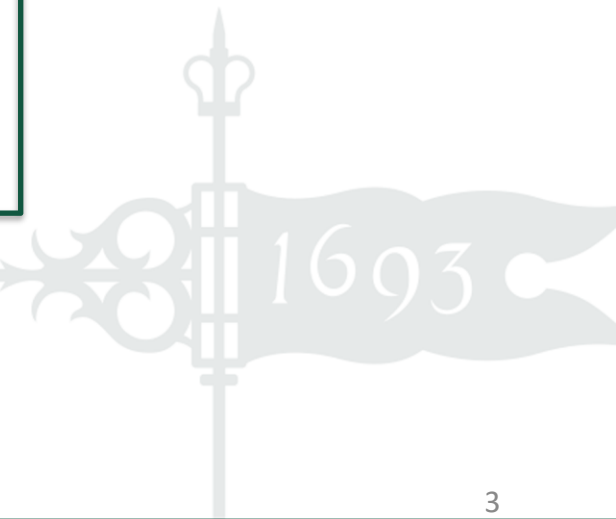
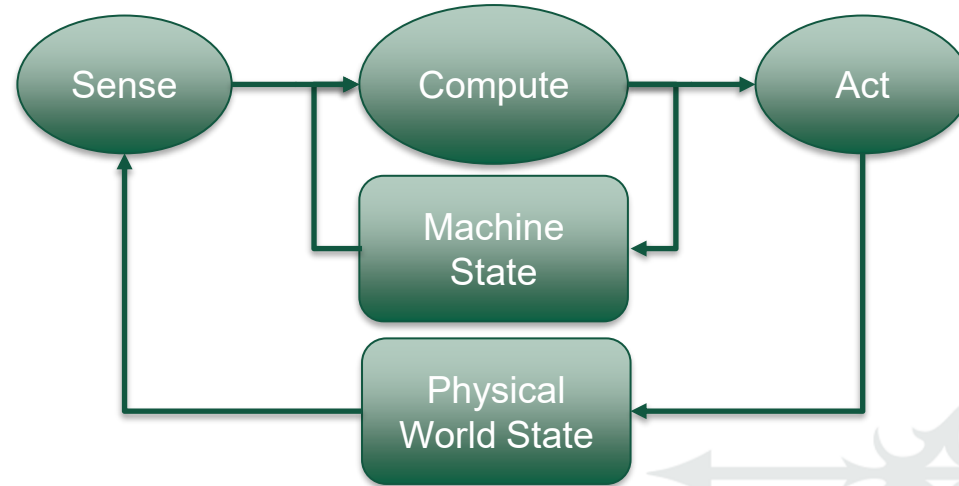
Perception

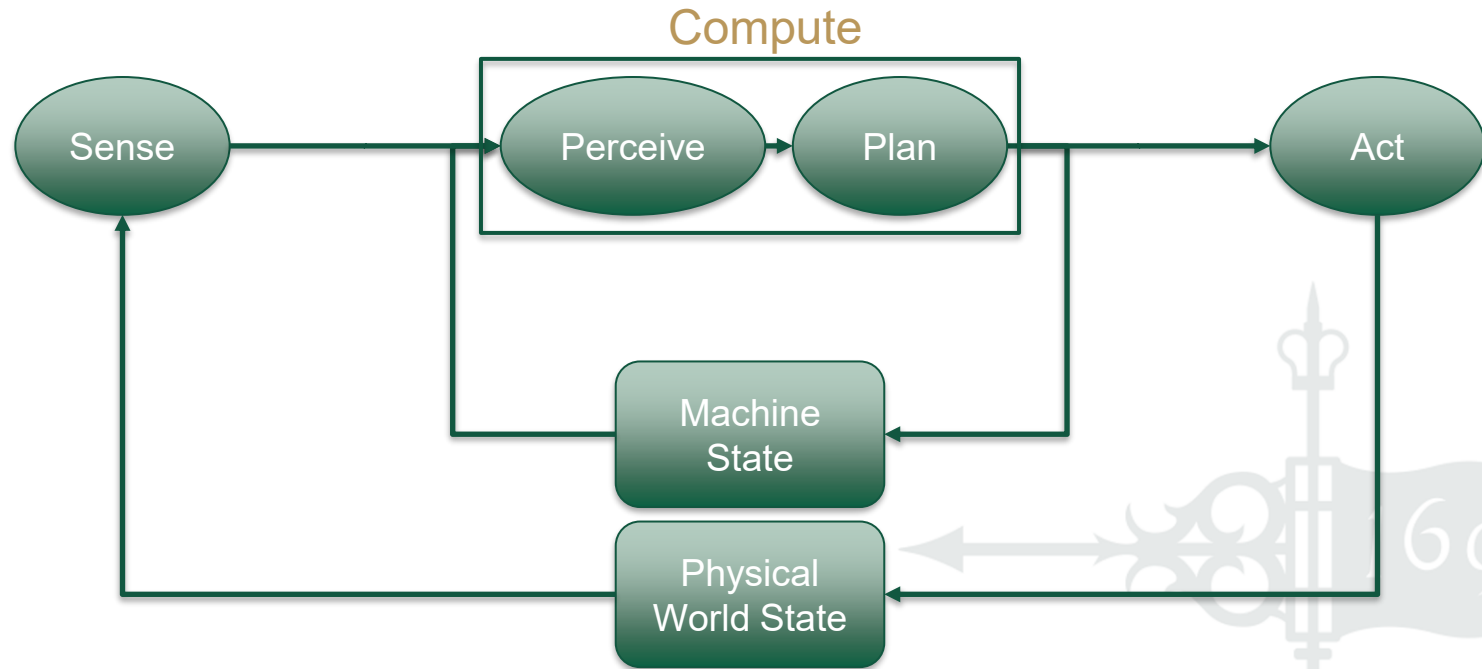
CSCI 420-04 Robotics



WILLIAM & MARY

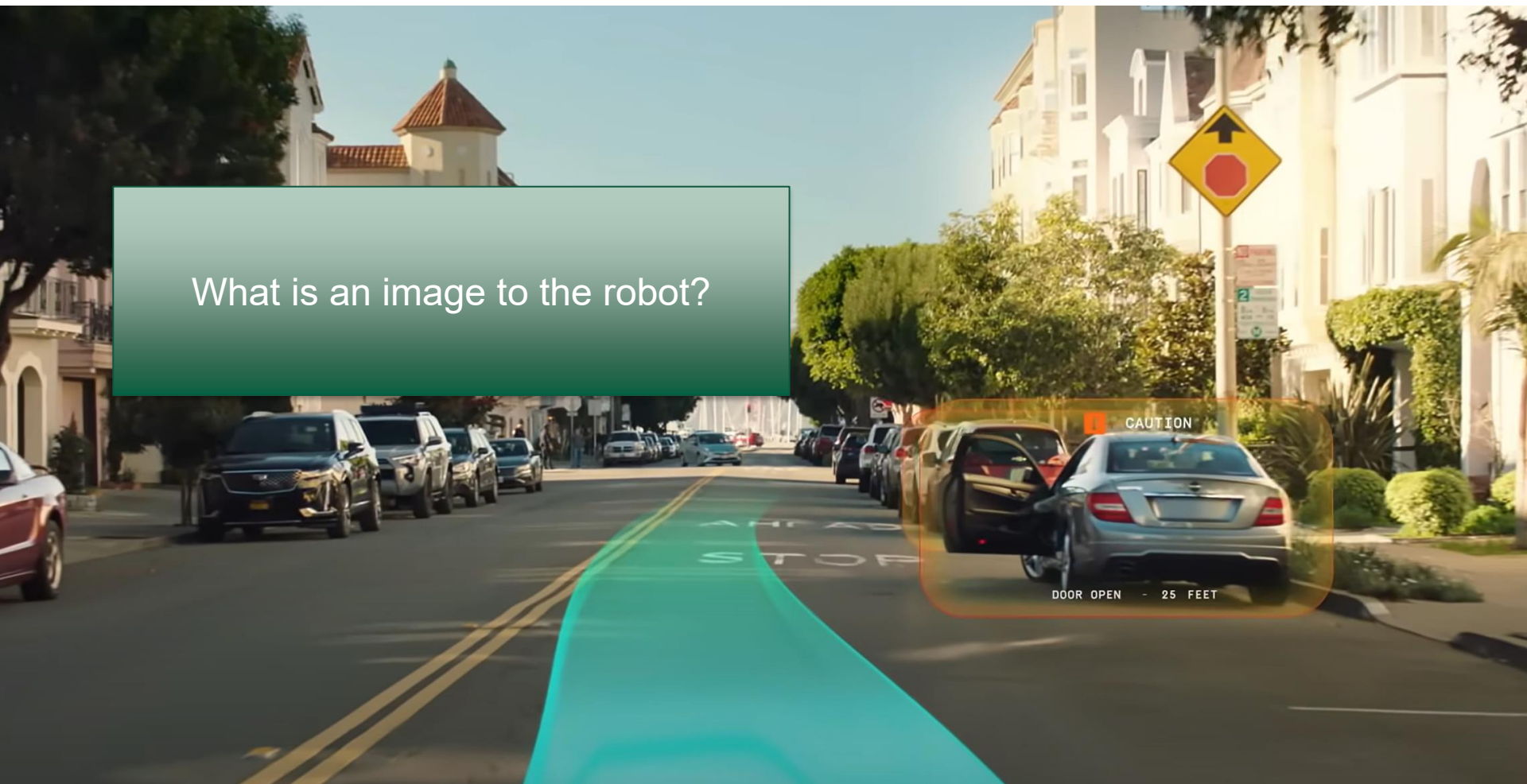
CHARTERED 1693







What is an image to the robot?



uint32 height

RGB, BGR,
MONO, HSV

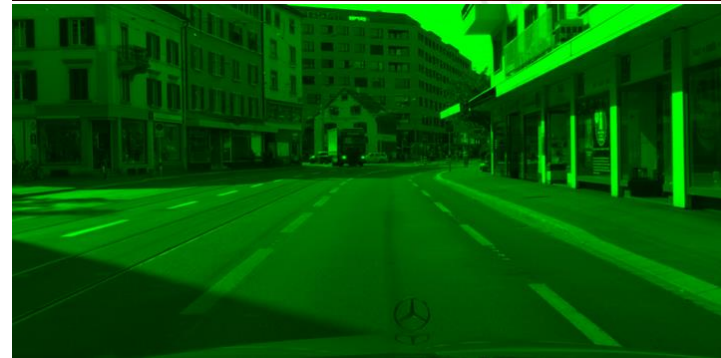
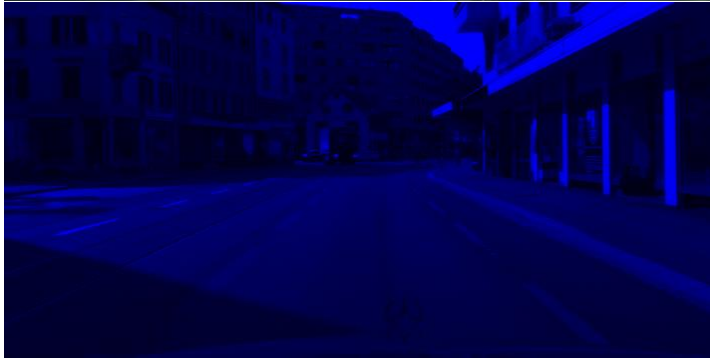
string encoding MONO, HSV

Storage data

uint32 step Storage data

```
uint8[] data
```

RGB

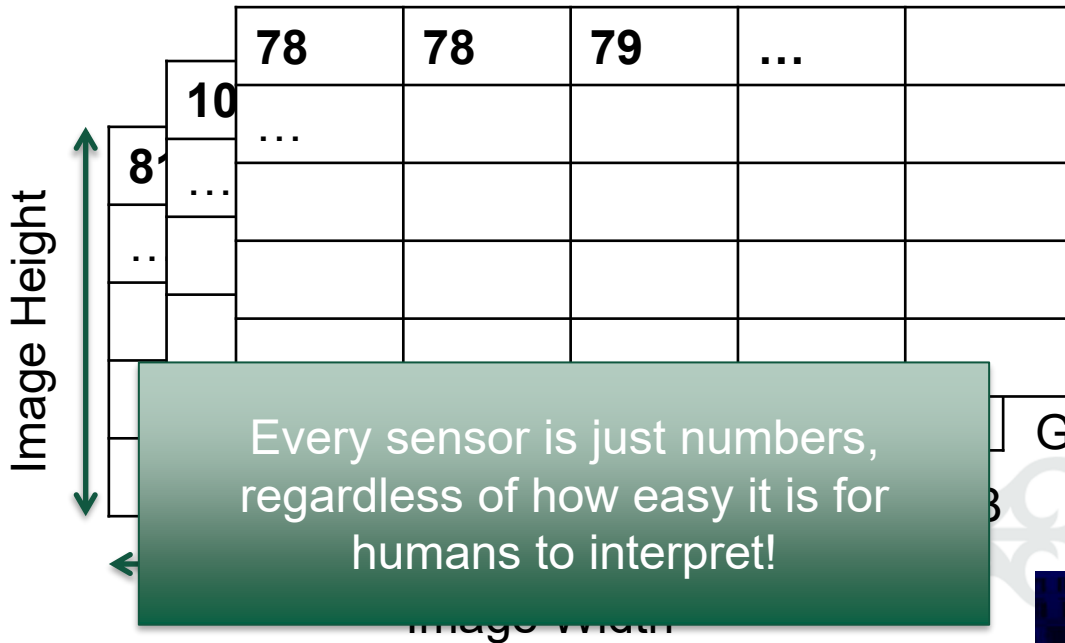


	64	76	79	87	98	97	88	80	81	81	79	80	81	79	43	20	21	21	21	22	22	23	23	23	24	23	32	64	66	56	57	58	63	66	65	57
	57	65	68	73	76	79	79	82	85	84	85	88	87	87	87	88	86	84	86	91	92	95	95	91	92	94	92	89	90	89	91	91	86	84	88	90
	88	88	87	83	78	79	82	82	80	80	79	79	79	79	80	80	79	78	77	77	79	80	80	80	78	73	74	75	72	71	79	112	111	99	114	127
126	128	129	127	124	121	124	124	125	122	116	115	107	99	100	97	97	100	100	94	85	90	88	68	58	57	55	54	55	54	55	55	55	55	54	53	53
[	54	55	54	56	54	55	55	54	53	52	53	52	51	51	51	51	52	53	53	53	52	51	49	53	50	49	49	49	48	48	51	71]				
	60	60	60	60	59	57	58	59	58	54	42	49	59	68	61	56	56	55	55	56	55	56	55	55	55	55	55	56	56	58	58	59	60	59	59	59
	60	60	57	49	50	47	53	58	57	56	56	56	57	56	51	53	49	42	42	26	23	24	24	24	25	23	22	29	35	35	35	34	30	28	27	27
	24	22	21	21	21	19	17	18	17	18	17	19	20	25	28	31	34	34	31	27	27	27	27	27	28	28	36	41	44	44	43	44	45	45	45	44
	37	34	39	36	33	36	39	42	45	45	45	45	45	44	43	41	40	38	36	36	33	31	32	32	32	31	31	31	30	29	30	29	29	31	30	30
	30	29	28	30	32	33	32	30	30	30	31	30	29	26	26	24	25	28	29	26	25	25	27	28	51	62	54	60	57	60	62	60	57	62	66	65
	65	77	83	83	97	98	90	81	80	80	79	80	81	80	48	21	22	22	21	23	23	23	24	23	22	28	60	69	58	56	57	63	67	65	58	
	55	64	67	70	74	77	79	82	83	84	85	88	89	87	86	88	87	84	85	89	92	95	93	91	93	94	92	89	90	89	91	91	87	83	87	90
	87	87	87	84	79	77	80	82	82	80	78	78	79	80	80	80	79	78	77	77	78	79	80	80	77	73	73	74	72	73	76	99	106	102	115	126
126	125	125	126	128	125	123	124	124	122	118	118	115	101	98	96	97	100	100	94	88	88	90	77	58	57	56	54	54	54	54	54	55	55	54	55	
[	55	54	53	55	55	55	54	53	54	53	53	52	50	51	51	52	52	51	53	53	52	52	48	50	50	48	49	49	49	48	50	68]				
	60	60	59	60	59	59	58	58	57	48	43	57	61	67	58	56	56	55	55	55	55	55	55	54	54	55	56	57	57	58	58	59	59	60	60	
	59	59	54	46	47	46	55	60	59	58	58	58	59	57	51	53	47	45	39	23	23	23	24	23	23	23	33	41	42	41	41	37	36	36	37	
	35	32	29	28	25	24	23	22	22	21	21	21	24	27	28	31	35	37	34	29	28	28	26	26	26	27	36	39	40	40	40	42	41	39	39	37
	31	28	34	31	30	37	41	42	45	44	44	45	43	41	38	38	37	35	34	33	32	32	33	33	32	31	32	31	32	32	30	30	31	30	28	31
	31	30	28	31	31	32	31	30	30	30	30	29	27	27	26	25	26	29	29	27	27	26	29	39	41	48	54	52	53	54	57	62	61	66	65	
	69	82	77	74	92	94	90	79	80	79	79	80	80	81	55	22	22	22	23	22	22	23	24	23	22	23	26	56	71	59	55	56	62	64	63	61
	55	63	66	68	73	76	77	81	83	85	85	88	89	87	86	88	87	85	85	87	91	94	94	90	92	95	92	90	91	91	91	91	89	85	89	90
	89	89	87	83	80	76	77	82	83	80	78	78	78	80	80	80	78	76	76	76	78	77	77	72	72	74	73	73	75	88	98	102	121	122		
120	123	127	130	127	126	122	123	120	119	120	120	106	95	92	95	100	100	95	93	92	90	83	60	56	57	55	54	53	54	56	55	56	54	53		
[	55	54	54	53	54	55	56	53	54	53	53	54	52	52	51	50	52	51	50	50	50	49	48	50	50	47	48	49	48	49	50	66]				
	61	61	60	59	59	59	58	58	55	42	48	60	63	66	58	55	55	55	55	55	56	55	55	56	55	55	56	56	56	58	58	58	59	60	60	60
	59	58	58	56	56	55	59	60	59	59	59	59	60	57	54	56	46	48	34	22	23	24	24	24	23	24	25	39	45	46	45	45	41	40	41	43
	42	42	39	34	32	32	31	29	28	28	28	28	31	33	34	36	40	41	41	39	37	35	33	33	32	35	42	43	43	44	43	43	41	38	38	38
	32	27	28	30	32	37	40	41	42	43	41	41	40	38	35	35	35	32	31	31	30	30	30	31	32	31	31	32	32	33	31	30	30	31	29	31
	31	31	30	30	30	30	29	28	27	27	26	25	24	24	23	22	23	26	27	26	25	25	27	27	43	45	46	54	51	52	54	58	64	62	66	64
	68	75	67	71	87	87	91	80	79	79	78	80	80	81	61	24	21	22	22	22	22	23	23	22	22	24	24	51	71	57	53	55	60	62	62	63
	55	59	66	68	74	76	75	79	83	83	87	88	89	86	86	86	86	85	85	87	81	84	83	83	83	83	80	80	81	80	80	80	87	88	89	90
	88	89	87	86	82	74	76	82	82	81	78	78	78	78	80	79	80	78																		
124	125	129	128	127	127	123	120	117	117	120	119	118	111	97	91	88	98	1																		
[	53	54	54	54	55	55	53	54	53	54	54	52	53	52	50	50	51																			
	60	60	60	60	59	58	58	51	40	54	60	63	65	57	55	55	55																			
	58	58	58	59	60	59	59	60	60	60	60	61	57	58	55	47	47																			
	43	43	41	36	35	36	34	31	30	30	30	31	35	34	37	39	41	44																		
	42	36	30	34	36	35	38	38	39	39	38	35	33	32	30	29	28	27																		
	30	30	28	28	28	27	26	24	23	23	23	22	21	21	22	25																				
	63	65	59	71	96	96	93	82	79	80	79	80	81	66	25	21	22																			
	56	57	66	66	72	75	74	77	83	81	85	87	88	85	85	85	87	86																		
	88	87	88	87	83	76	75	81	83	81	79	77	77	78	78	79	82	80																		
127	126	130	134	131	128	126	123	121	120	120	117	111	112	103	97	88	91	98	92	91	98	91	89	86	67	56	58	56	55	55	56	56	54	53	52	
	53	51	53	55	56	53	54	54	54	53	54	53	52	51	51	52	51	51	51	52	50	50	49	50	49	49	50	49	48	48	50	60]]]				

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

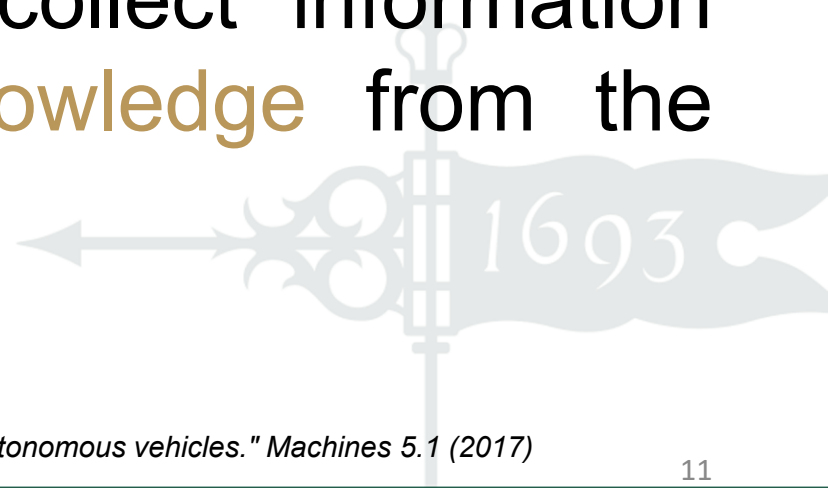
In memory, images are just hundreds of thousands of numbers (or more)!

RGB in memory



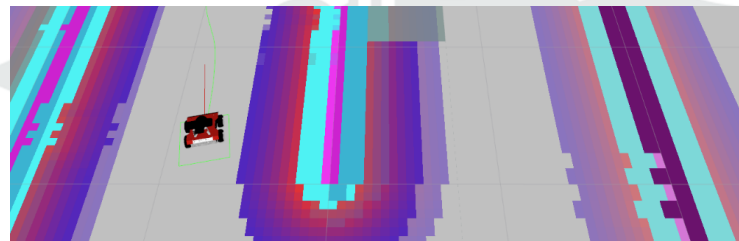
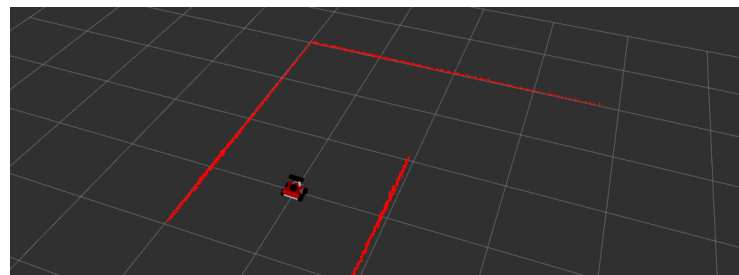
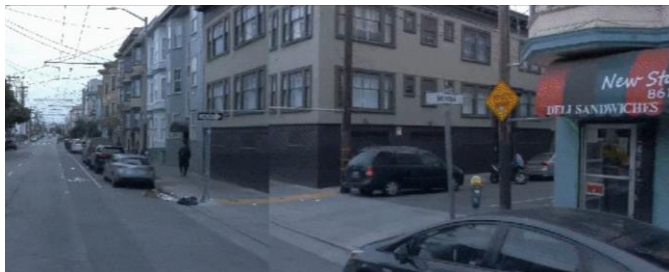
Perception

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



Perception

Extract high-level abstraction from sensor data



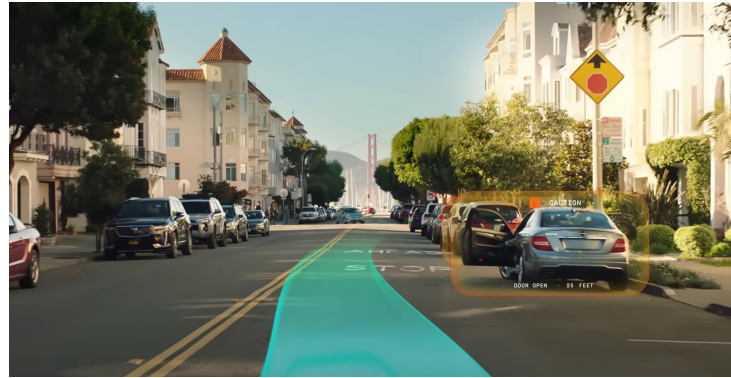
Color: Red

Main types of Perception

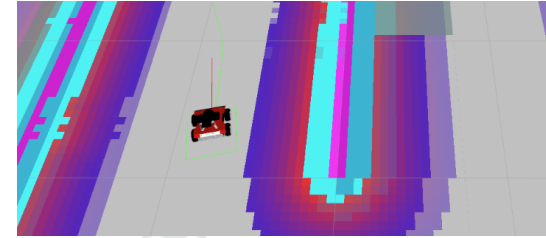
Classification



Object Detection

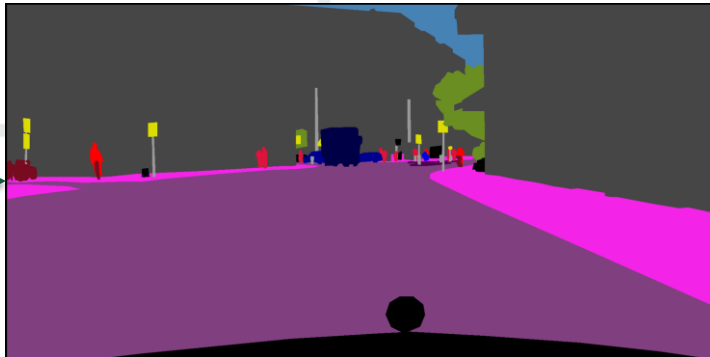


Interpretation



What is Perception?

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



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

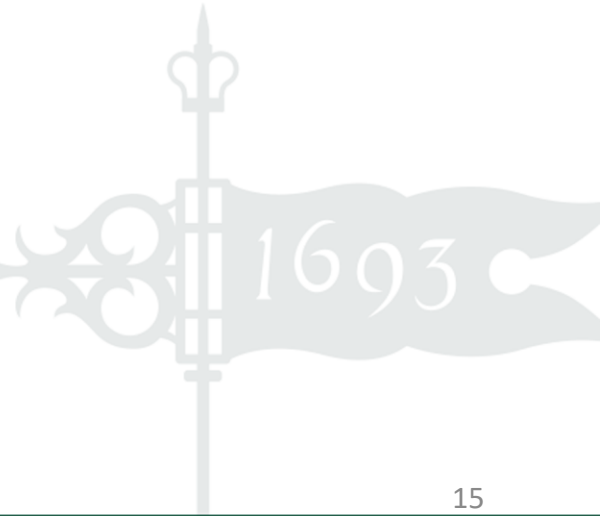


Image Processing

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



NumPy^N



SciPy

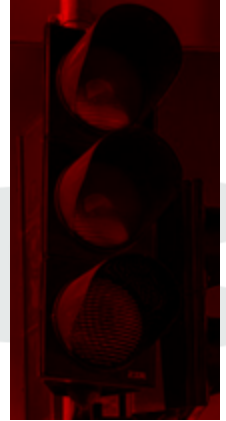


Color Filtering

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

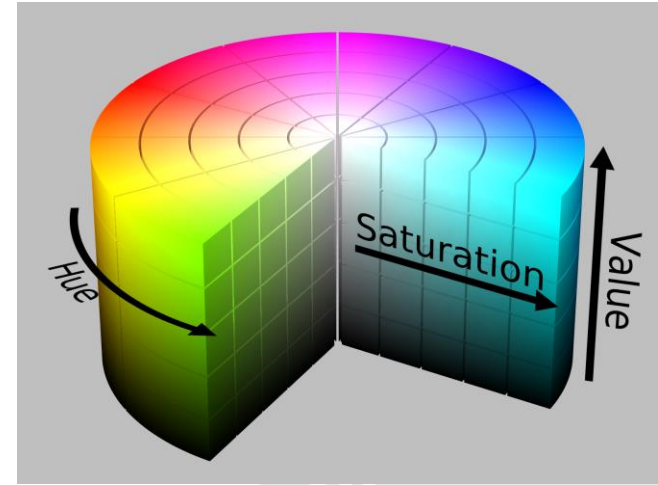


The green filter can identify the green light!



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 we filter out more colors?
- HSV stores color information 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

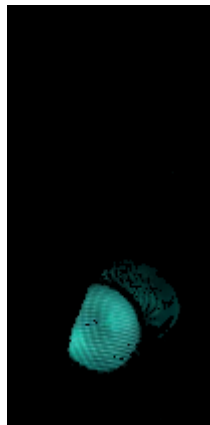
Original



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])
```

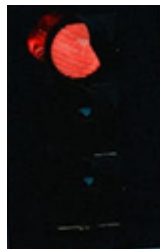
```
upper_bound = np.array([90, 255, 255])
```

```
# 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)
```



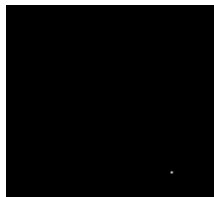
Different programs use different scales!
OpenCV uses 0-180 for Hue,
and 0-255 for S and V

Color Filtering

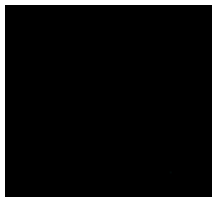
Original



Mask



Result



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

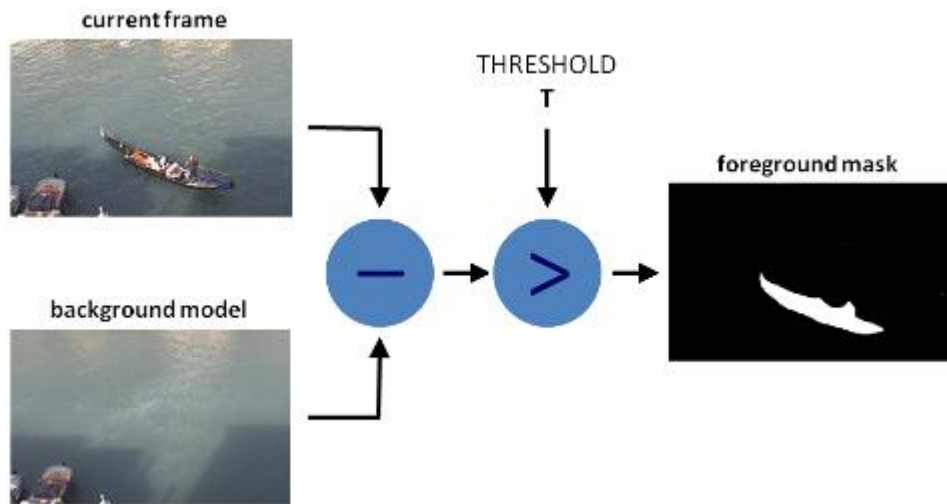
# Set up filter values
lower_bound = np.array([85, 128, 0])
upper_bound = np.array([90, 255, 255])
```

When does this excel? When does this struggle?

```
mask = cv2.inRange(hsv, lower_bound, upper_bound)
result = cv2.bitwise_and(image, image, mask=mask)
```

Background Subtraction

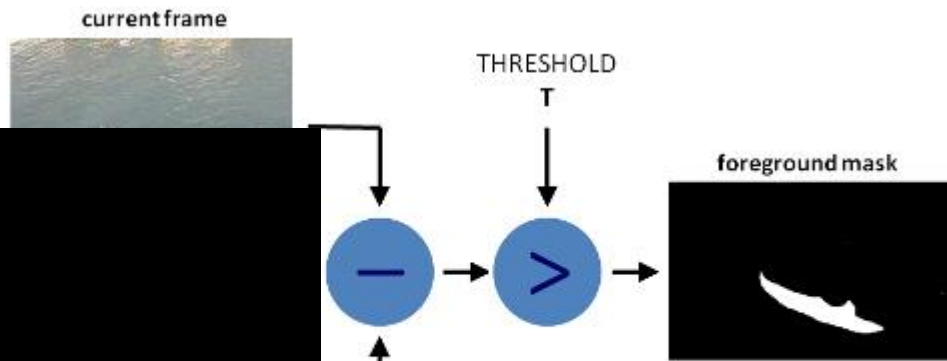
- Idea: the interesting parts move while the background is static



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)
```



Background Subtraction



511.VDOT.Virginia.gov



511.VDOT.Virginia.gov



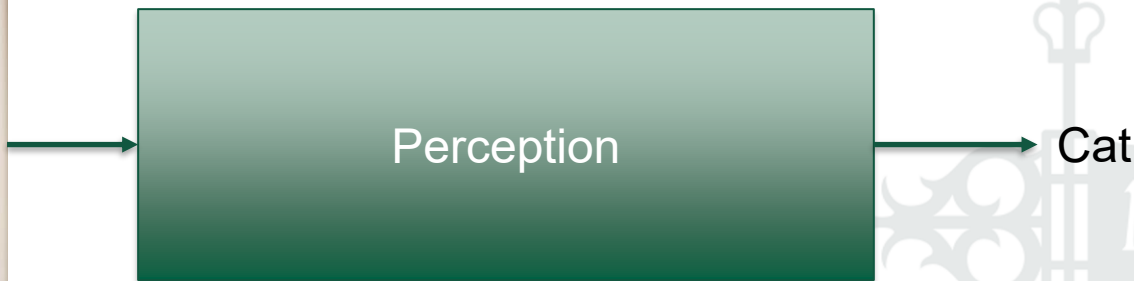
Data (Image) Processing

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



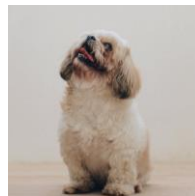
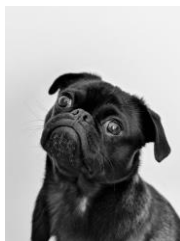
Machine Learning

- 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



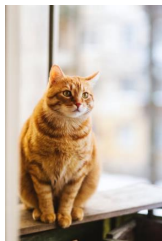
Label: Cat

Label: Dog

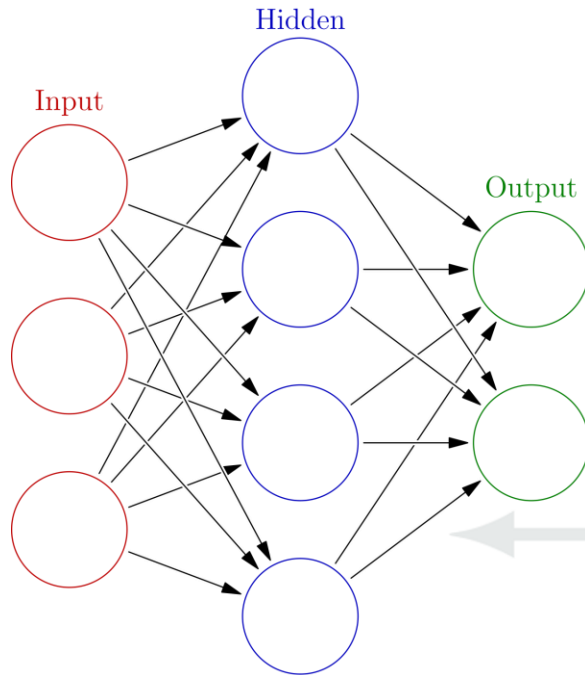


Machine Learning

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



Neural Networks

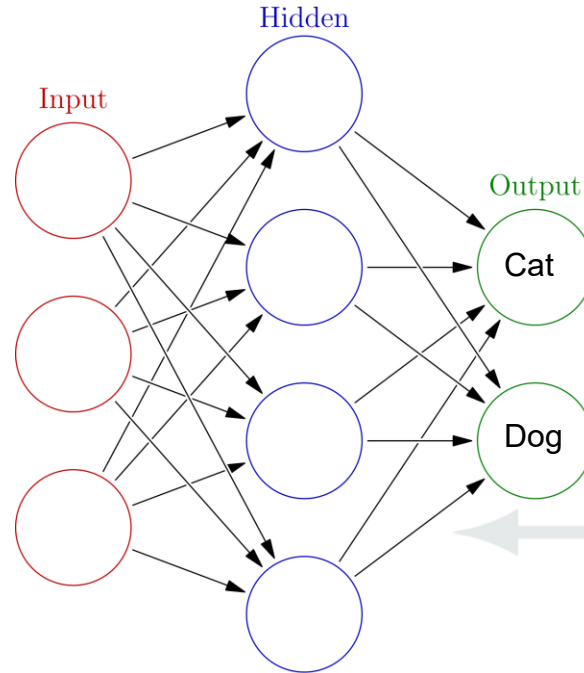


[https://en.wikipedia.org/wiki/Neural_network_\(machine_learning\)#/media/File:Colored_neural_network.svg](https://en.wikipedia.org/wiki/Neural_network_(machine_learning)#/media/File:Colored_neural_network.svg)

Cat

1693

Neural Networks



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

x_1 x_2 x_3 x_4 x_5 ... x_n

Cat

Dog

1693

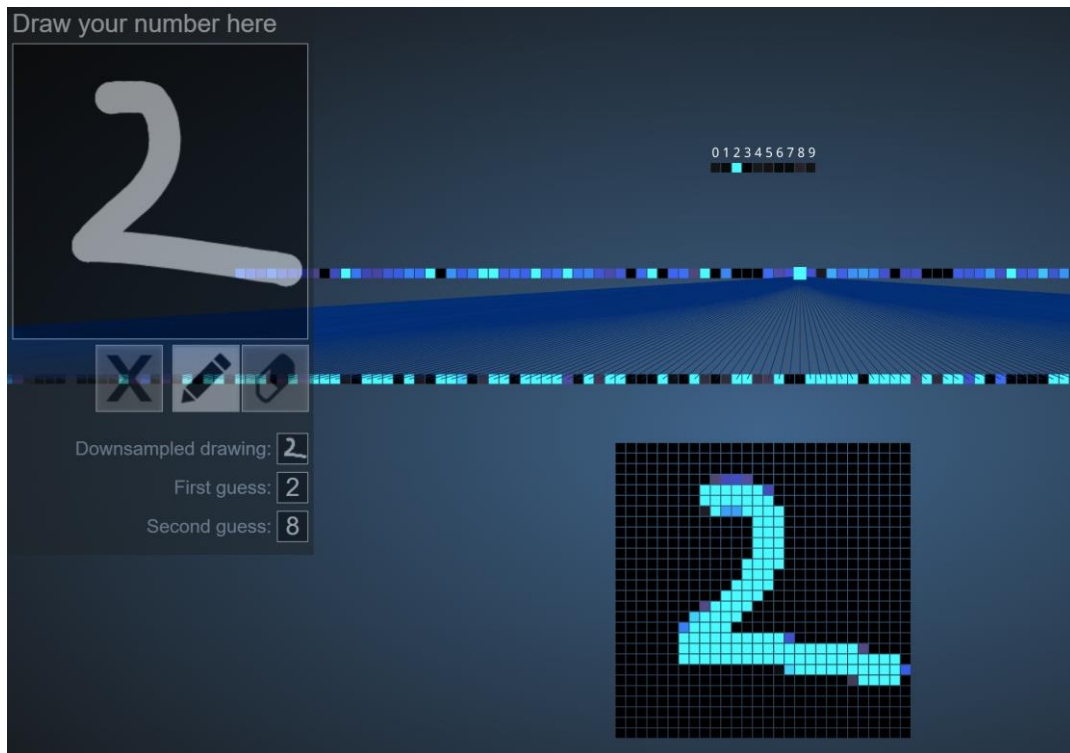
What's in a Neuron?

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

ReLU Activation:

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

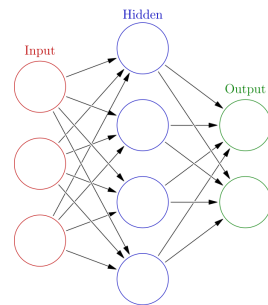
Visualizing NN



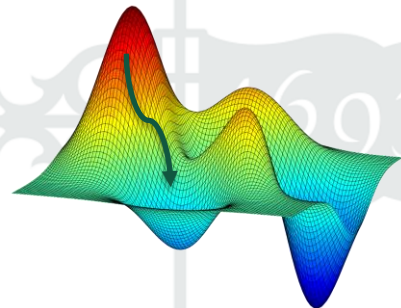
Try it! https://adamharley.com/nn_vis/mlp/2d.html

How do NNs train?

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



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