Computer Vision

Jacob Vesco

OUTLINE

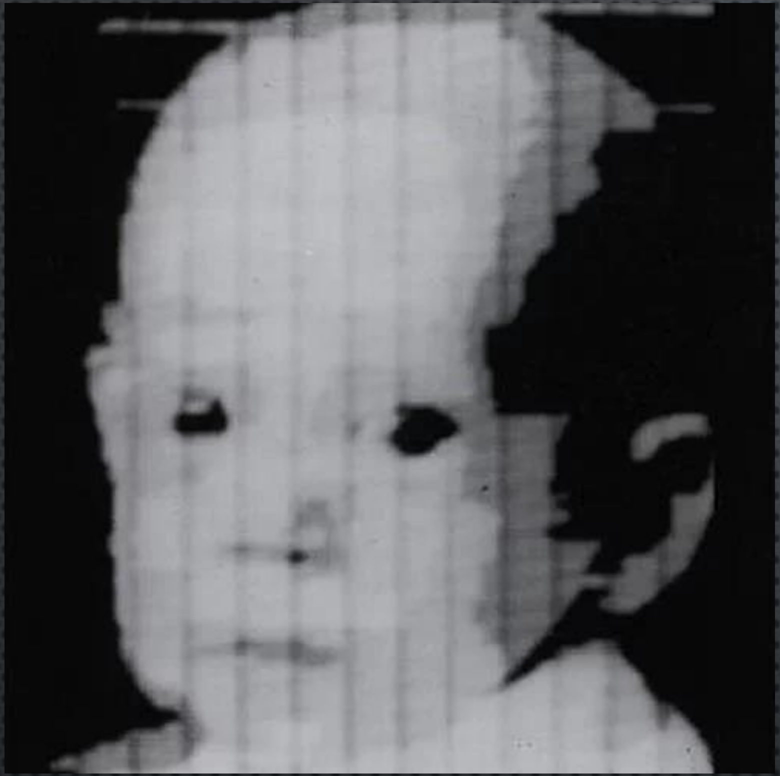
- What is computer vision?
- Feature detection
- Image classification
- Deep convolutional neural networks

WHAT IS IT

- The field of artificial intelligence that seeks to develop techniques for computers to “see” and understand images
- We’ve all seen and heard of it
  - Facial recognition
  - Self driving vehicles
- Easy for us, hard for computers

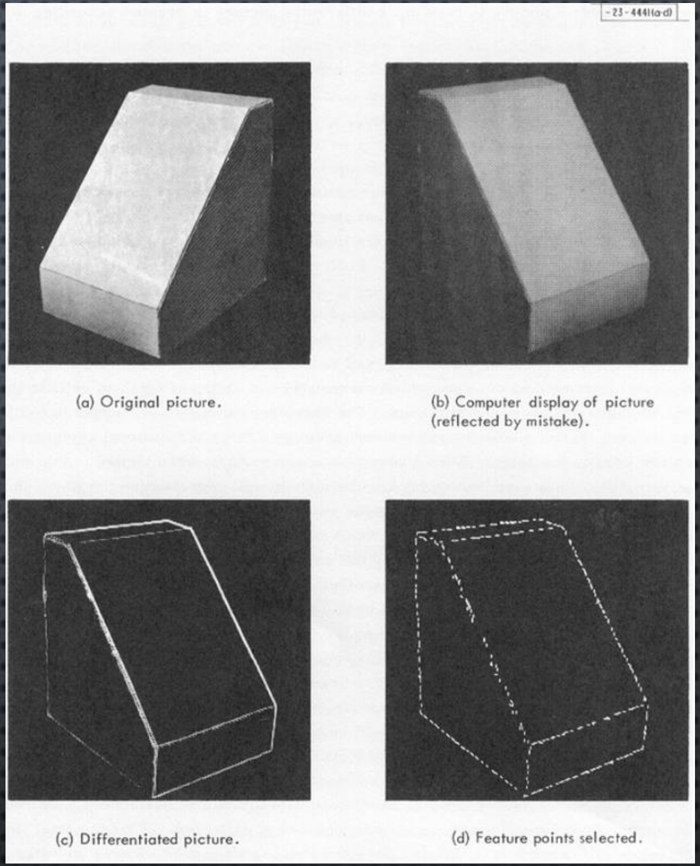
SIGHT AND IMAGES

- David Hubel and Torsten Wiesel’s discovery in 1959
    - The biological eye determines objects by looking at features of the object in order of least to most complexity
    - This is just like how deep learning works
- The first digital image in 1959
- “*Machine perception of three-dimensional solids*” in 1963
    - Considered to be a precursor to modern computer vision



The first digital image, taken by Russel Kirsch in 1959

Image source
https://www.engadget.com/2010/06/30/russell-kirsch-helped-create-them-now-he-wants-to-kill-square-p/



(a) Original picture.

(b) Computer display of picture (reflected by mistake).

(c) Differentiated picture.

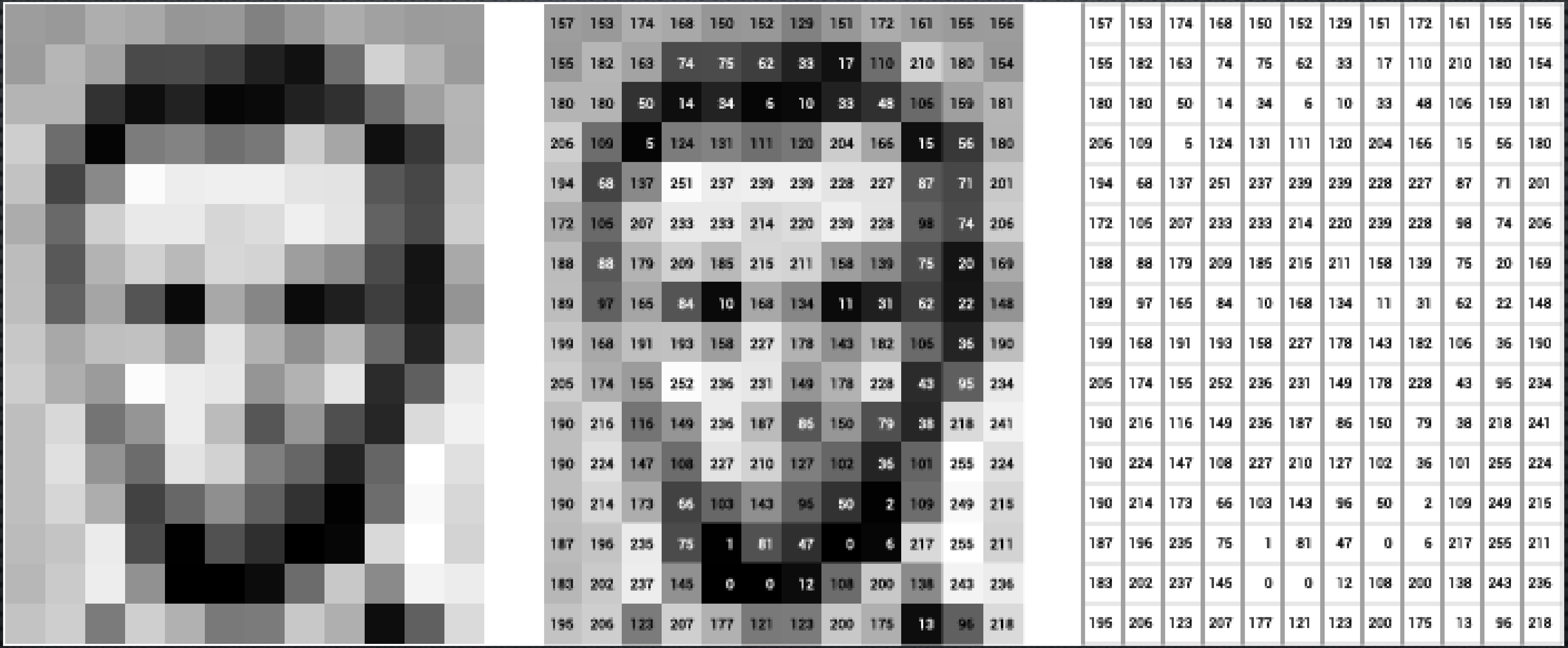
(d) Feature points selected.

"*Machine perception of three-dimensional solids*" – Lawrence Roberts

Image Source
https://hackernoon.com/a-brief-history-of-computer-vision-and-convolutional-neural-networks-8fe8aacc79f3

TO A COMPUTER, IMAGES ARE NUMBERS

- The human brain uses the eye to grab sensory information, and automatically perceives what they represent
- Computers see everything in binary numbers
- An image to a computer is a series of numbers, representing a value of each and every pixel
- Grayscale values, RGB values



157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

Image Source
https://openframeworks.cc/ofBook/chapters/image_processing_computer_vision.html

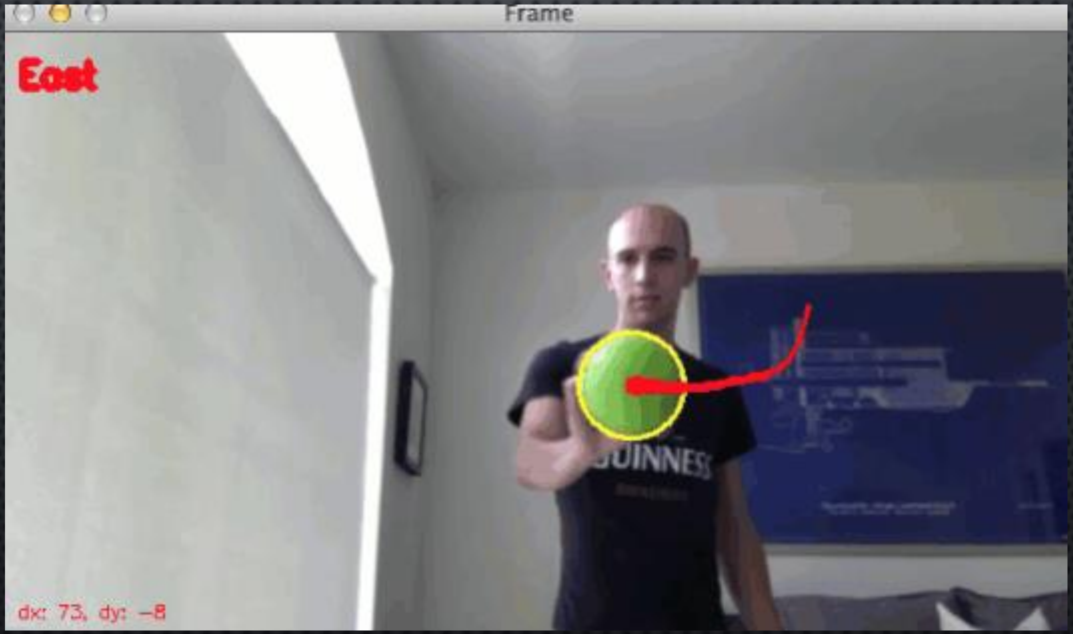
SCANNING THE PIXELS

- A basic form of computer vision comes from scanning the pixels
- Example: tracking a fly in a video
- An algorithm that scans each pixel in an image, and returns the pixel that most closely matches a desired quality
- Finds this pixel for every frame of the video, effectively tracking the pixel throughout the video

SCANNING THE PIXELS

- Multiple problems with current approach
  - Multiple pixels in image have similar or the same value
  - More than one fly – Can’t tell which fly to track
- Some improvements
  - Keep track of previous fly position
  - Track a group of pixels instead of just one

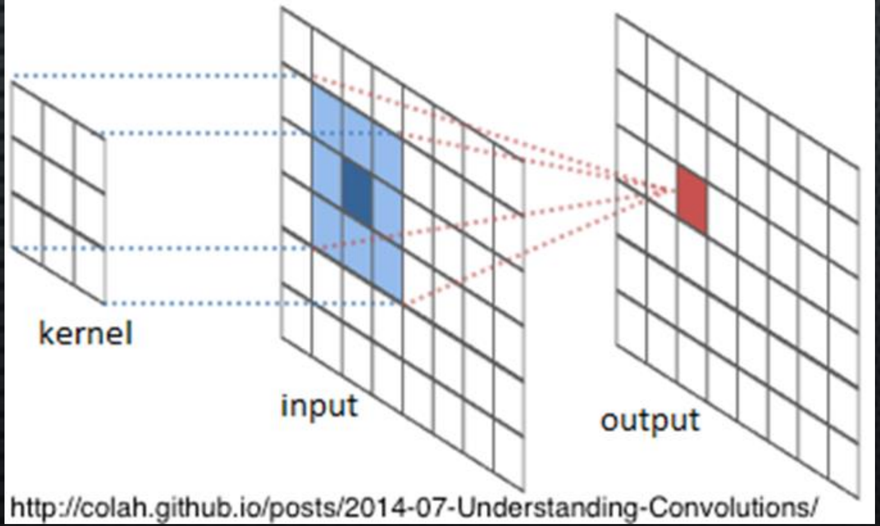
SCANNING THE PIXELS



```
East
dx: 73, dy: -8
```

KERNELS

- Square matrixes, with the cells corresponding to pixels, used to revalue the center pixel
- A weight is applied to each pixel value in the matrix
- The new value of the center pixel is the sum of all the weighted values in the matrix



http://colah.github.io/posts/2014-07-Understanding-Convolutions/

Image Source https://stackoverflow.com/questions/51008505/kernels-and-weights-in-convolutional-neural-networks

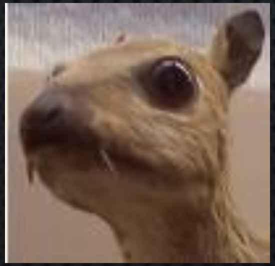
KERNELS

- Used for:
  - Edge detection
  - Sharpening
  - Blurring
- Different uses require different weighting and different sized matrixes
- Another term for them is “Prewitt Operators”, named after their inventor Judith M. S. Prewitt

KERNELS

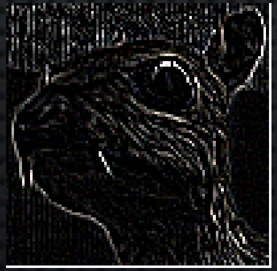
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Identity



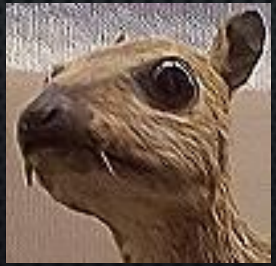
$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Edge Detection



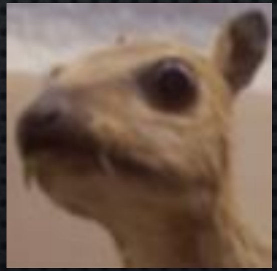
$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Sharpening



$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

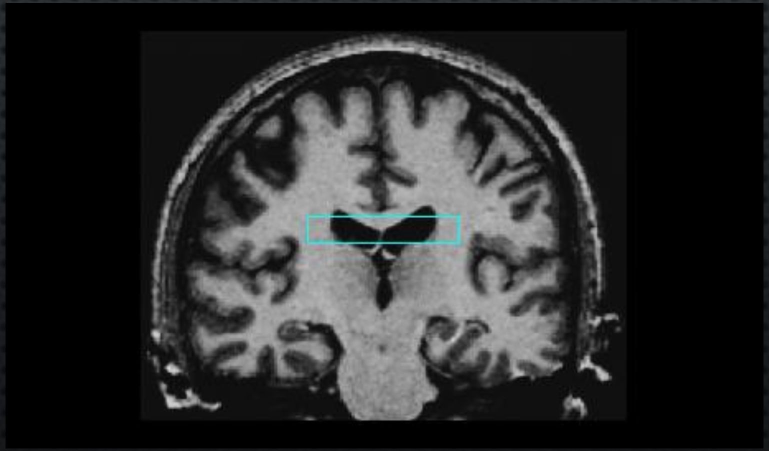
Normalized
Box Blur



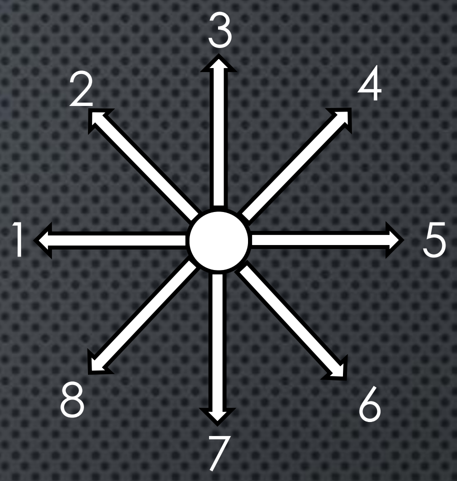
Images snipped from
Wikipedia page over kernels

CONTOURS

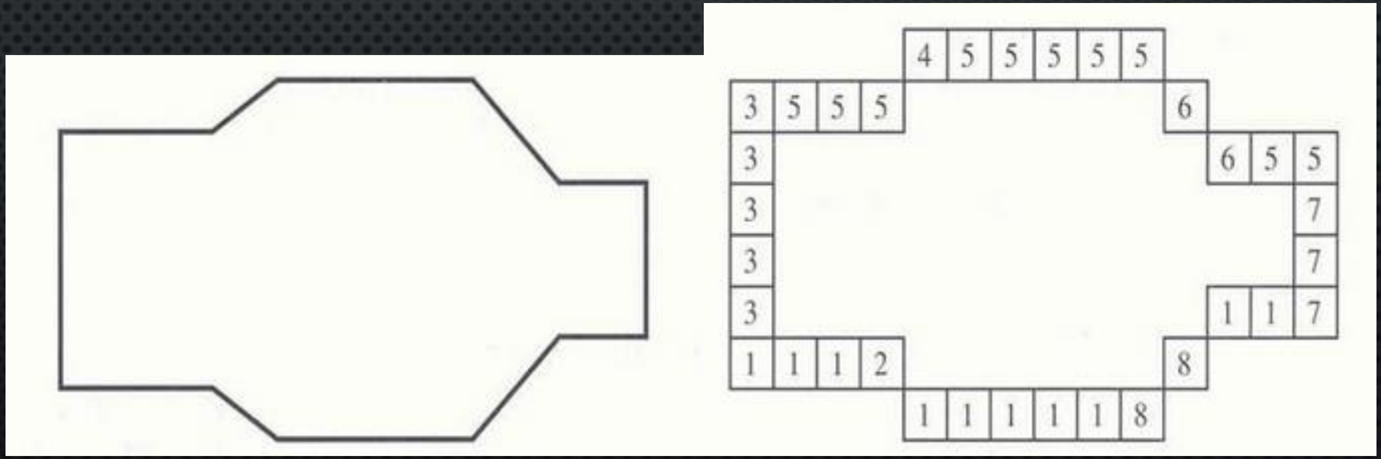
- Finding the boundaries between objects
- Utilizes the edges found previously
- Two types
  - Closed contours – define a regional boundary
  - Open contours – contains gaps, which makes them less reliable



CONTOURS – CHAIN CODES

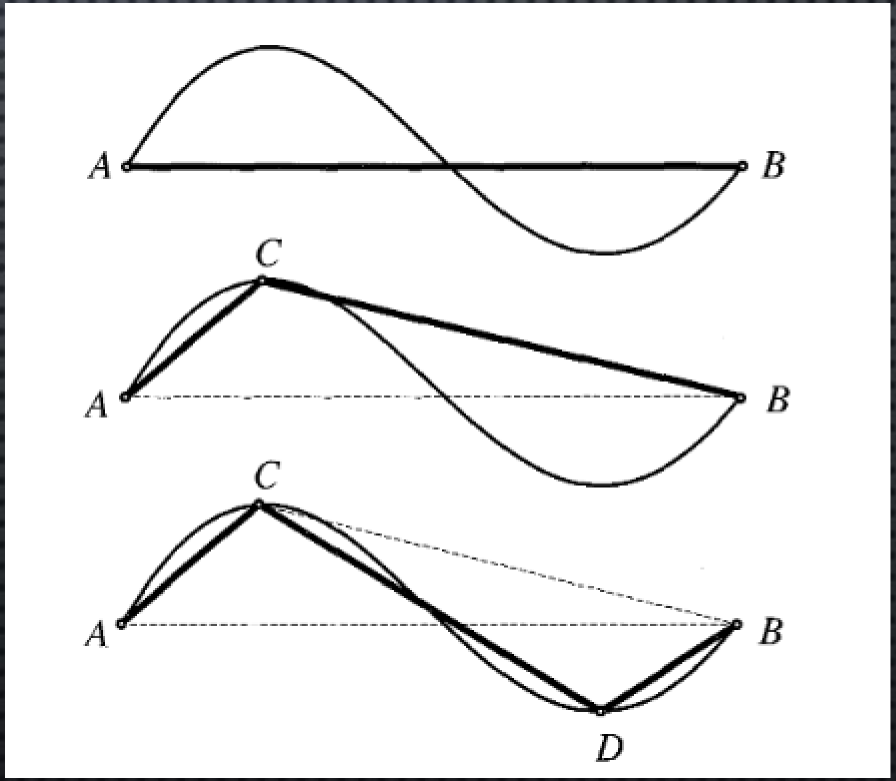


- A way of connecting pixels in an edge
- Each cardinal direction has a value
- Moving along an edge, the value of the chain at a pixel is the direction moved to get to that pixel from the previous pixel
- Slope representation
- Slope densities



CONTOURS – CURVE FITTING

- Four types of curves:
  - Line segment
  - Circular arc
  - Conical sections
  - Cubic splines
- Polyline representation

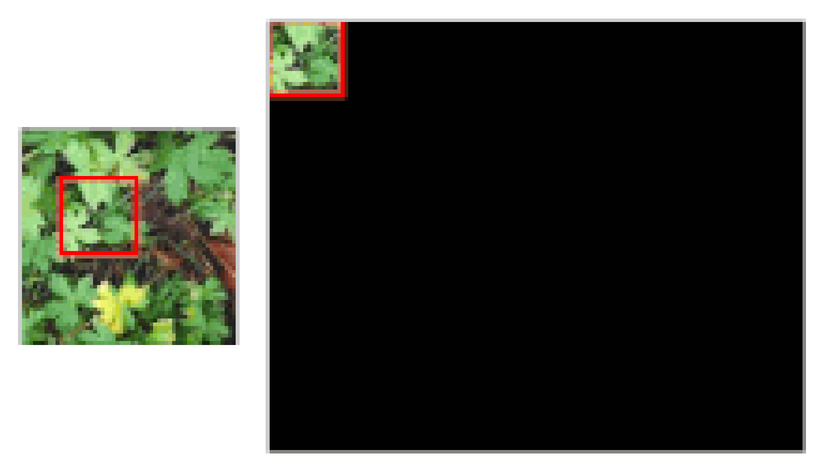


TEXTURES

- Problems that arise in detecting contours
- 2 overlapping objects that look too similar
- Using the light intensity of pixels to find textures
- Can detect seemingly camouflaged objects

TEXTURES – ISSUES

- Main problems of texture detection
  - Segmentation
  - Synthesis
  - Shape
  - Classification

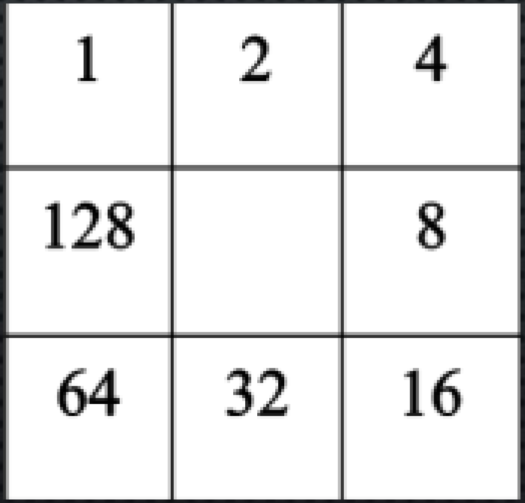


TEXTURES – APPROACHES

- Four approaches to analyzing textures
  - Statistical
  - Structural
  - Model Based
  - Transform

TEXTURES – LBP

- Local Binary Patterns
- A commonly used statistical approach
- Filter through each pixel, calculating a measure of texture via weighting it and its neighbors
- 3x3 weight matrix
- Compare intensities of neighbors to center
- Add up activated weights to get value of center pixel



1	2	4
128		8
64	32	16

IMAGE CLASSIFICATION

- Finally, we can start identifying hotdogs
  - You can detect other things, but… why though
- Use the feature detectors discussed earlier
- Needs to train and test
- Requires a mountain of training information
- Multiple methods of classifying test images

IMAGE CLASSIFICATION

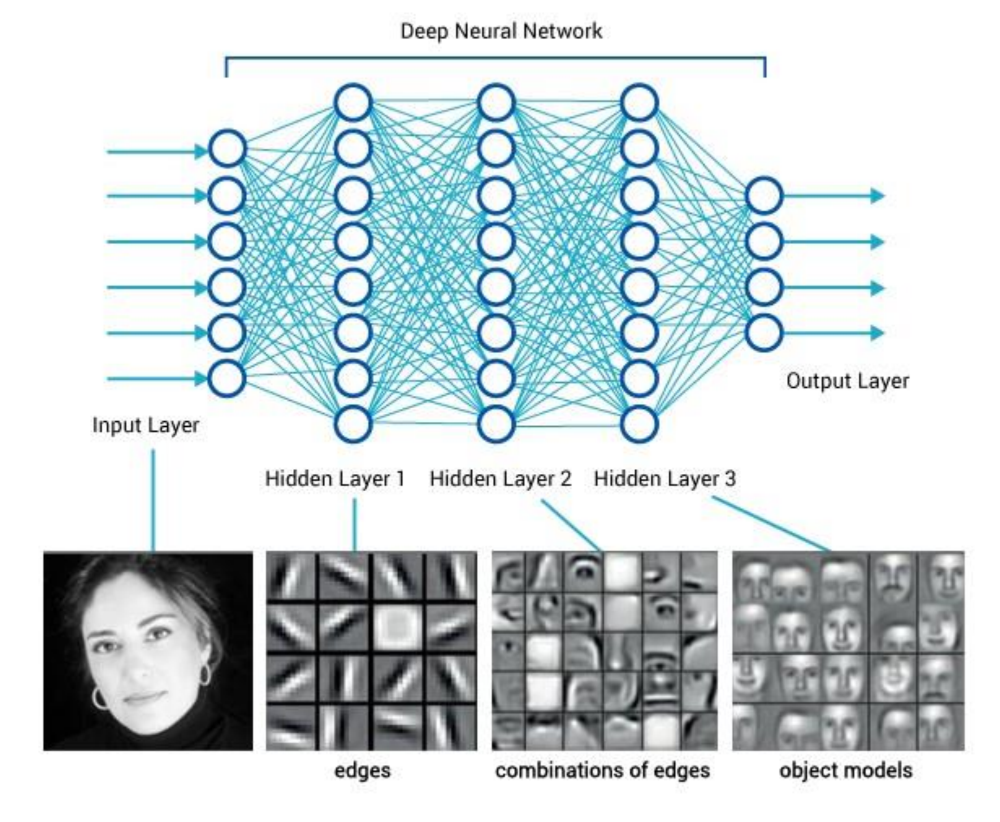
- A small brain way of classifying test images
- Compare test image to training images – find differences between each pixel for each training image
- Find the training image that gives the smallest total difference
- Classify the test image as the training image’s label
- This is what we did before we knew about better algorithms, like deep convolutional neural networks

DEEP CONVOLUTIONAL NEURAL NETWORKS

- Alexnet - Revolutionized computer vision in 2012
- Classify images with incredible accuracy and speed
- Use all the features we talked about
  - Edge detection
  - Region boundaries
  - Textures
  - Far more than necessary to mention

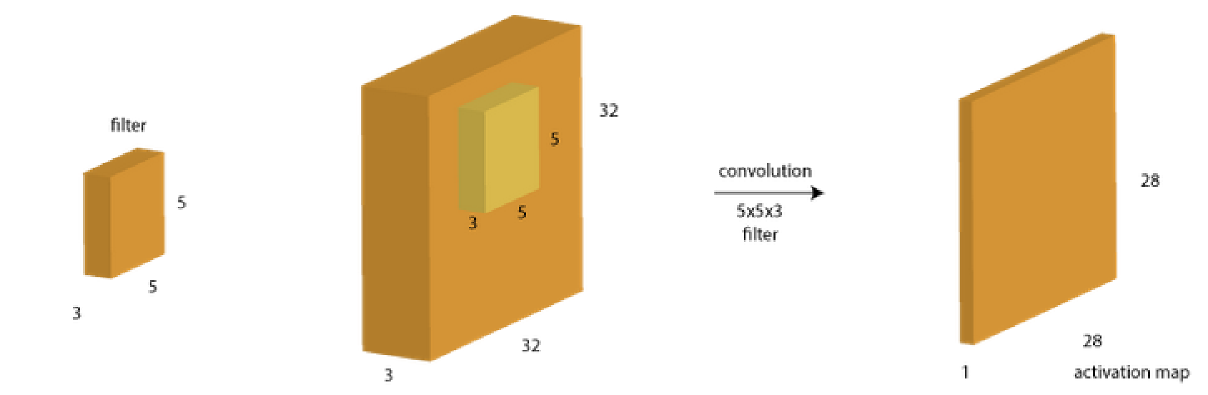
DEEP CONVOLUTIONAL NEURAL NETWORKS

- Three layer types in deep cnn's
  - Convolution layer
  - Pooling layer
  - Fully-connected



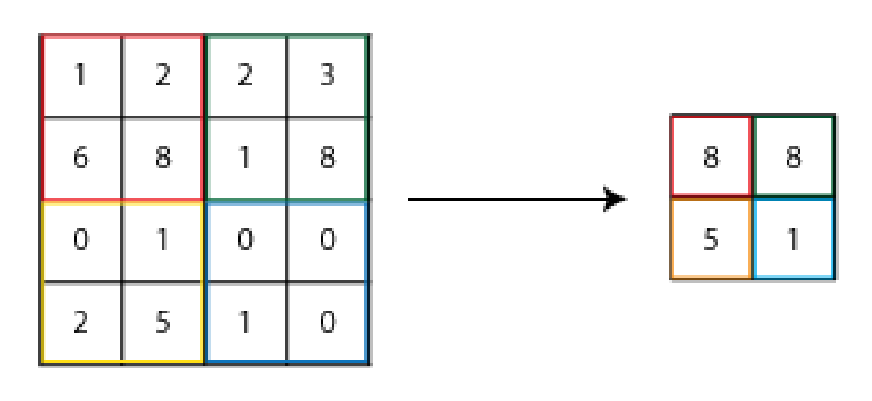
DEEP CONVOLUTIONAL NEURAL NETWORKS

- Convolution layers
- Take in an input layer – output activation maps
- Pass a filter through the layer
- The filter acts just like a kernel, weighting pixels and scoring one pixel based on its neighbors
- Each time the filter is passed through, another activation map is formed



DEEP CONVOLUTIONAL NEURAL NETWORKS

- Pooling layer
- Focuses on down sampling feature detection
- Makes the algorithm more robust to problems like image rotation

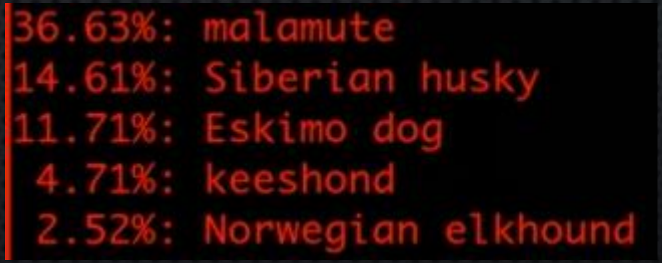


1	2	2	3
6	8	1	8
0	1	0	0
2	5	1	0

8	8
5	1

DEEP CONVOLUTIONAL NEURAL NETWORKS

- Fully connected layer
- Flattens previous layer output into a vector for input into the next stage
- The vector can be thought of as a list of tuples, representing each feature and the most likely classification
- The last fully connected layer will give the probabilities associated with each possible classification of the image



```
36.63%: malamute
14.61%: Siberian husky
11.71%: Eskimo dog
 4.71%: keeshond
 2.52%: Norwegian elkhound
```

CONCLUSION