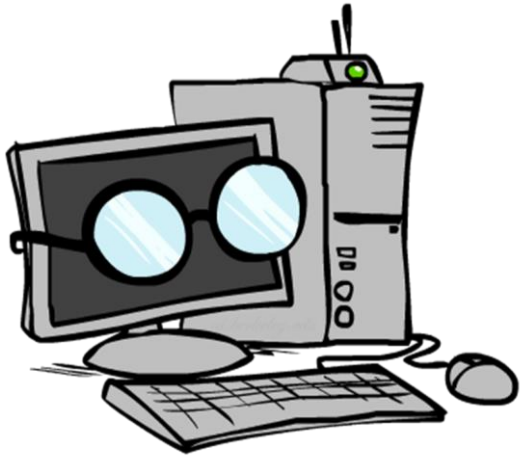
COMS W4733: Computational Aspects of Robotics

Lecture 24: Computer Vision 1



Instructor: Tony Dear

Vision and Perception

- Any intelligent system requires some form of feedback from environment
- Vision is a specific form of perception based on images—easy to gather using cameras and many ways to process and interpret

- Relevant tasks: Camera calibration, image processing, object recognition, pose estimation, motion analysis, scene reconstruction, visual servoing

- Methods: Camera physics, image transformation, feature extraction, segmentation, closed-loop control

Image Processing

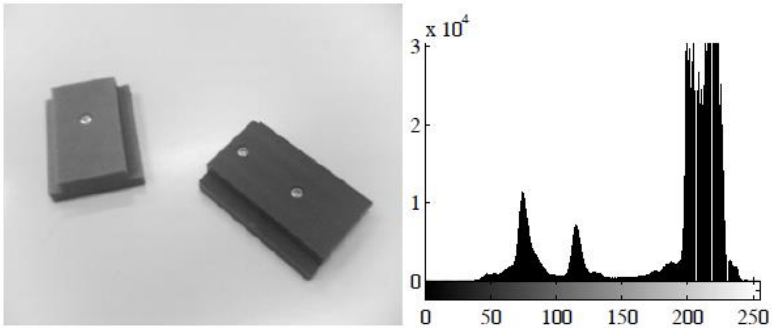
Semantic information

Geometric information

Segmentation

- Goal: Divide image up into meaningful components
  - Foreground vs background, different colors or shading, different classes of objects
- Related: Find significant features in image, e.g. edges or lines

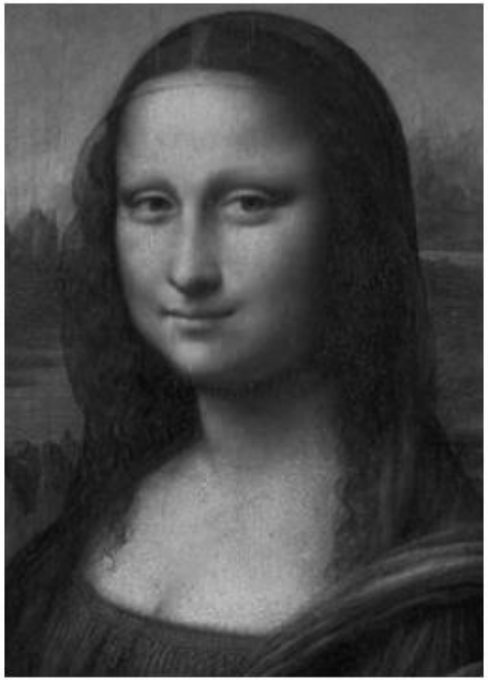
- First problem: How to numerically represent images? Histograms!

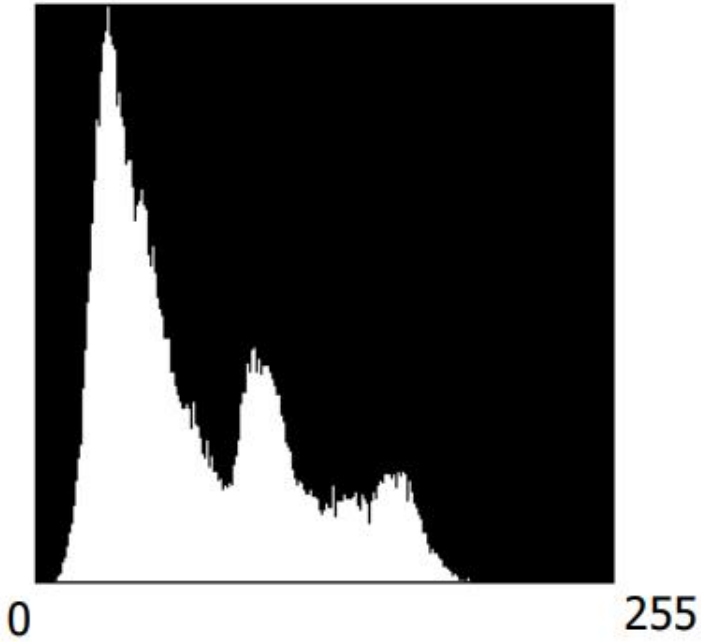


- X-axis: Pixel value (typically, pure black is 0 and pure white is 255)
- Y-axis: Number of pixels with pixel value

- We discard a lot of information (e.g. spatial) this way!

Example: Image Histogram





Binary Segmentation

- While simple, image histogram can be used to perform simple **thresholding**
- Pixels belong to one of two labels, depending on value relative to a threshold
- Reveal locations of interesting features while compressing image

- Simple approach: Decide on a value μ
- All pixels greater than μ: make them 255; otherwise, make them 0
- How to pick μ? Maybe start with 128

- More generally, use the histogram to decide on an appropriate threshold!

Choosing a Threshold

- The histogram is a “probability function” that a given pixel will have a given gray value
- One option: Use the image’s gray value mean as a threshold

- Good start, but can be improved since variance also provides information
  - Ex: Image with half white, half black vs image that is mostly gray

- Improved approach: Pick a threshold that would minimize some weighted combination of the two groups’ variances

Mean: $\mu_i = \sum_{z=0}^{255} \frac{H_i(z)}{\sum_{z=0}^{N-1} H_i(z)}$

Variance: $\sigma_i^2 = \sum_{z=0}^{255} (z - \mu_i)^2 \frac{H_i(z)}{\sum_{z=0}^{N-1} H_i(z)}$

Example: Thresholding

Grayscale	Binary threshold
	

Connected Components

- Images often contain more than just foreground and background
- Labels should be further differentiated between different **components**
- Component identification can be done by simply noting adjacent pixels

- Raster scanning (left to right, up to down) image for connected components

- First pass—for each potential component pixel, assign to either existing component if adjacent to a seen one or assign to new one if not
  - If more than one adjacent component, mark down equivalence
- Second pass—reassign pixels after combining equivalences

Example: Raster Scanning Components

0	0	0	0	0	0	0	0	0	0
0	X	X	X	0	0	0	0	0	0
0	X	X	X	0	0	0	0	0	0
0	X	X	X	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	X	0	0	X	X	0
0	0	0	0	X	0	0	X	X	0
0	0	0	0	X	X	X	X	X	0
0	X	X	X	X	X	X	X	X	0
0	X	X	X	X	X	X	X	X	0
0	0	0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	2	0	0	3	3	0
0	0	0	0	2	0	0	3	3	0
0	0	0	0	2	2	2	2	2	0
0	4	4	4	2	2	2	2	2	0
0	4	4	4	2	2	2	2	2	0
0	0	0	0	0	0	0	0	0	0

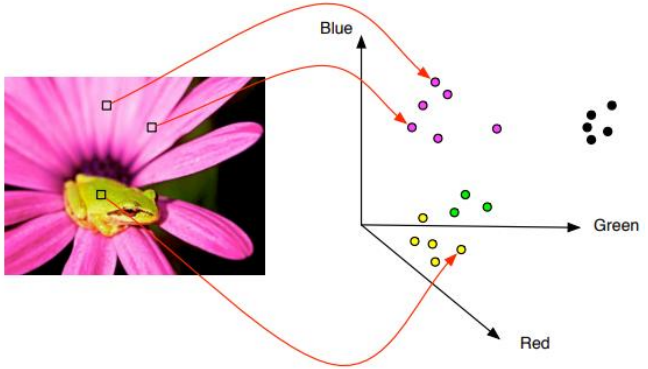
0	0	0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	2	0	0	2	2	0
0	0	0	0	2	0	0	2	2	0
0	0	0	0	2	2	2	2	2	0
0	2	2	2	2	2	2	2	2	0
0	2	2	2	2	2	2	2	2	0
0	0	0	0	0	0	0	0	0	0

Example: Connected Components

Segmentation by Clustering

- Image segmentation can also be done via clustering methods
- Groups of “similar” pixels tend to clump together
- k-means allows one to specify number of components beforehand

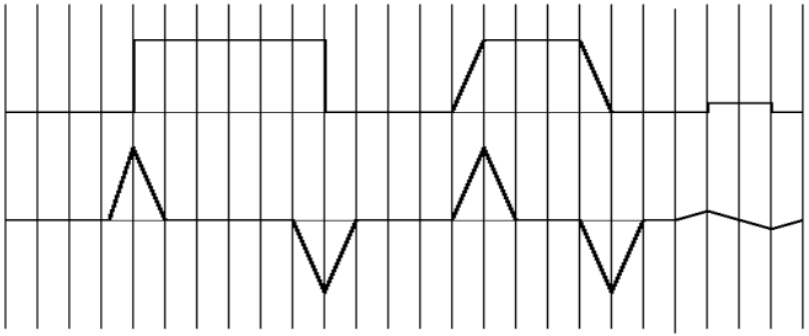
- Randomly select k points as cluster means
- Repeat until convergence:
  - Assign each point to the closest mean
  - Move each mean to average of its assigned points



Boundary Segmentation

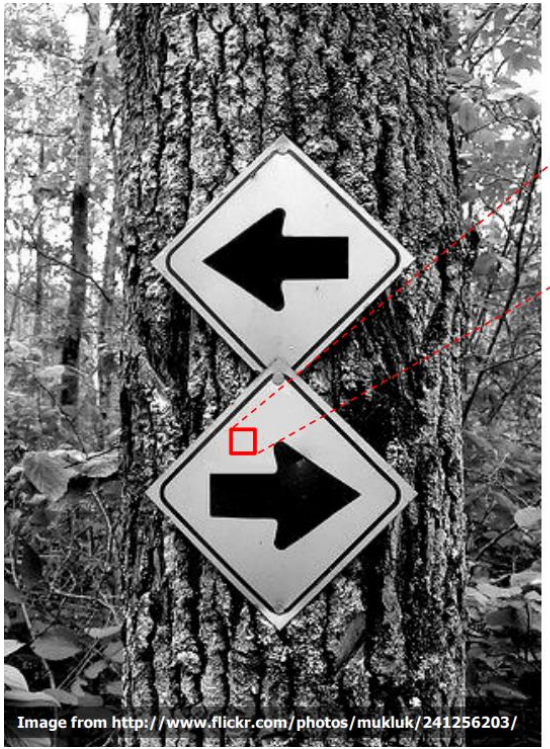
- Idea: Instead of grouping pixels, find local discontinuities in the grayness level
- We can define functions on pixels such that gradient on boundaries is large
- Edge detection: Take all pixels whose gradient is higher than some threshold

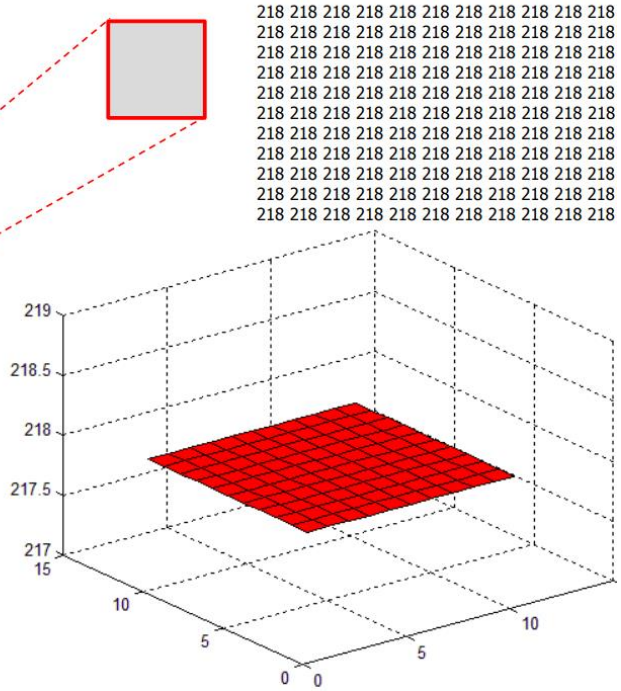
- Since images are discretized, “gradients” really just mean local differences



- Scanline of a row of pixels and gray values
- Gradient, or difference between values of successive pixels

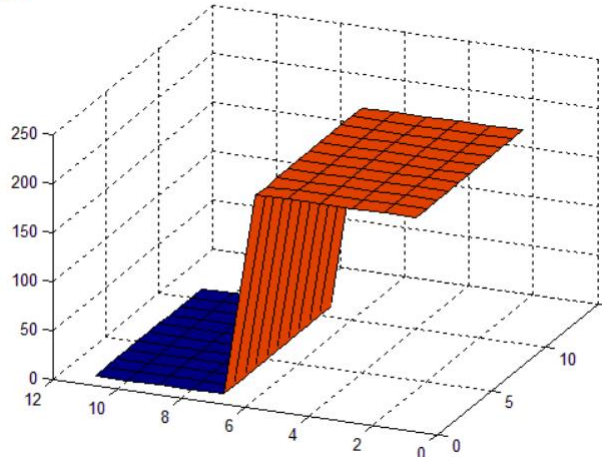
Example: Edge Detection



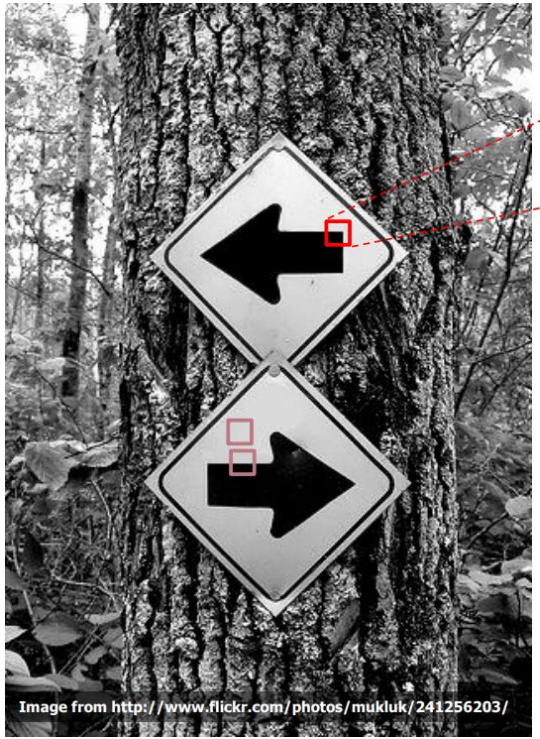


Example: Edge Detection

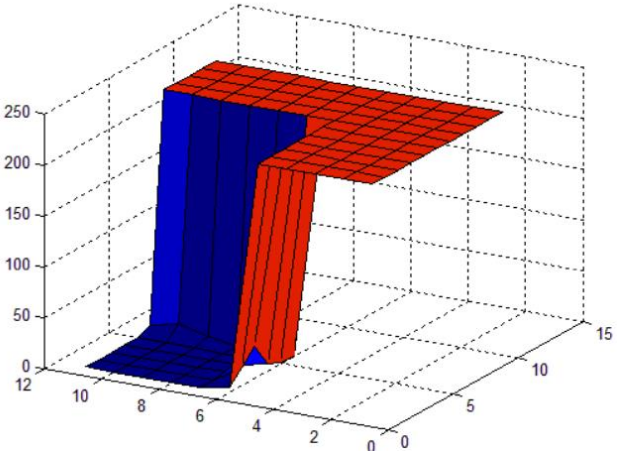
208	208	208	208	208	208	208	208	208	208	208
208	208	208	208	208	208	208	208	208	208	208
208	208	208	208	208	208	208	208	208	208	208
208	208	208	208	208	208	208	208	208	208	208
208	208	208	208	208	208	208	208	208	208	208
208	207	208	208	208	208	208	208	208	208	208
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0



Example: Edge Detection



229	229	229	229	229	229	229	229	229	229	229
229	229	229	229	229	229	229	229	229	229	229
229	229	229	229	229	229	229	229	229	229	229
229	229	229	229	229	229	229	229	229	229	229
229	229	229	229	229	230	229	229	229	229	229
5	17	31	7	1	0	229	229	229	229	229
0	0	1	0	0	0	229	229	229	229	229
0	0	0	0	0	0	229	229	229	229	229
0	0	0	0	1	4	229	229	229	229	229
0	0	0	0	0	11	229	229	229	229	229
0	0	0	0	0	5	229	229	229	229	229



Edge Detection

- Process of taking local differences at each pixel is really a process of applying *masks* on the image pixels via convolution

- Horizontal edge: $[-1 \quad 1]^T$
- Vertical edge: $[-1 \quad 1]$

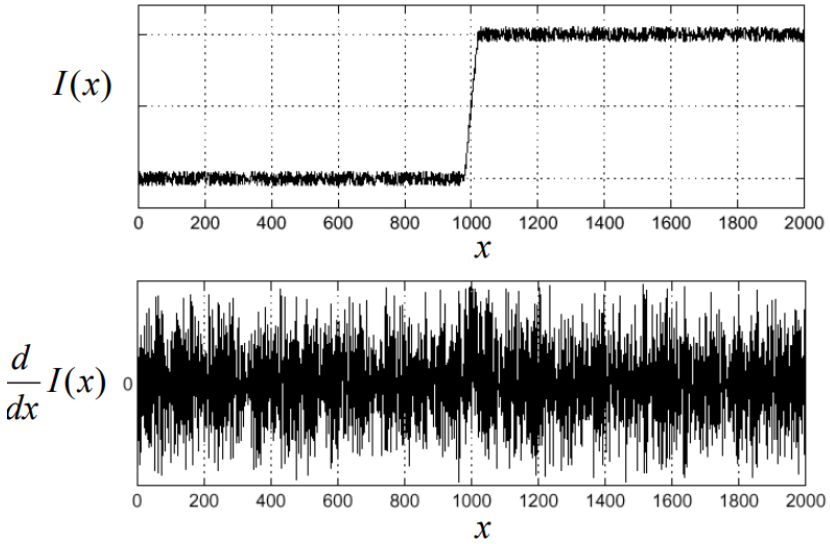
Gradient magnitude: $\|G\| = \sqrt{G_x^2 + G_y^2}$

- More sophisticated masks also possible, e.g. Sobel operators

Horizontal: $\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$ Vertical: $\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$

Masks and Smoothing

- Notion of applying masks to an image is important for another reason
- Real images are noisy! Taking differences on raw images may be a bad idea
- Smoothing should be applied first

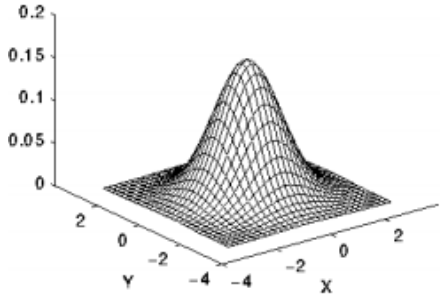


Masks and Smoothing

- Convolution of a mask with an image: Replace each pixel value with dot product of mask and nearby pixel values
- Usually used for averaging (smoothing), but masks can be anything
- Ex: 3×3 mean averaging filter:

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

- Ex: Gaussian smoothing filter:



$$\frac{1}{273}$$

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

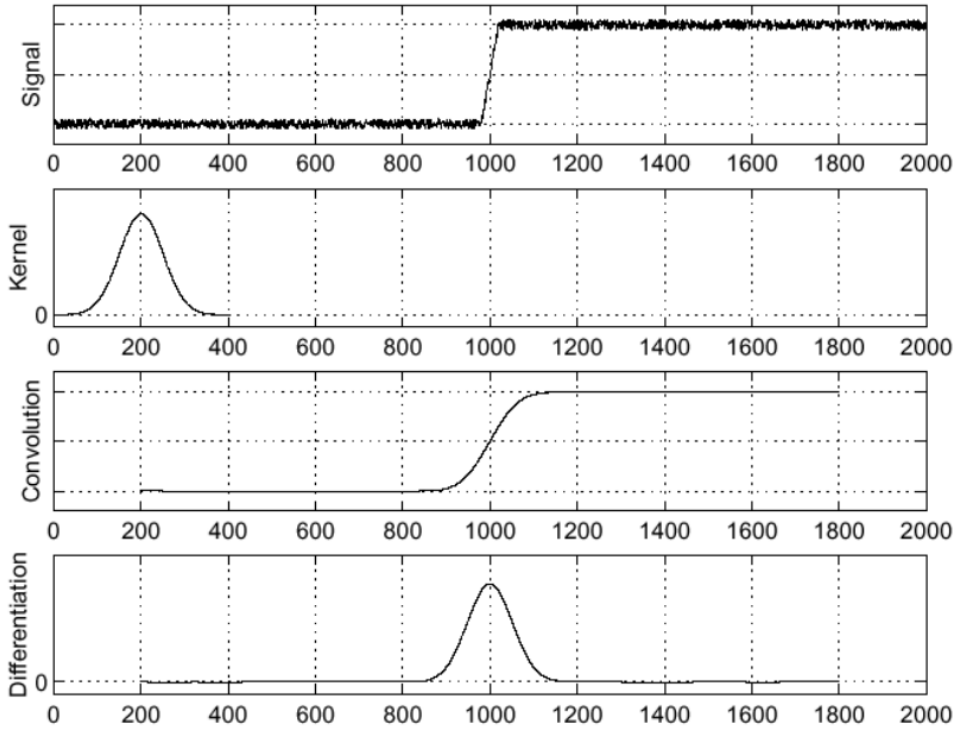
Example: After Smoothing

$I(x)$

$G_\sigma(x)$

$s(x) = I(x) * G_\sigma(x)$

$s'(x) = \frac{d}{dx}(s(x))$



Canny Edge Detector

- Combine smoothing with edge detection, plus some more steps for robustness
- Criteria: Locate all edges, localize all edges, and minimize false edges

1. Gaussian smoothing to reduce noise
2. Find intensity gradients (e.g. Sobel)
3. Non-maximum suppression to *thin out* edges
4. Double thresholding to identify strong edges and discard weak ones
5. Edge tracking to see whether weak edges are isolated and should be removed

Feature Detection

- In addition to being used for segmentation, edges can also serve as *features*
- Features: Any interesting part of an image useful for later tasks
- Should be repeatable (easily identifiable elsewhere on the image), may be examined at the pixel level

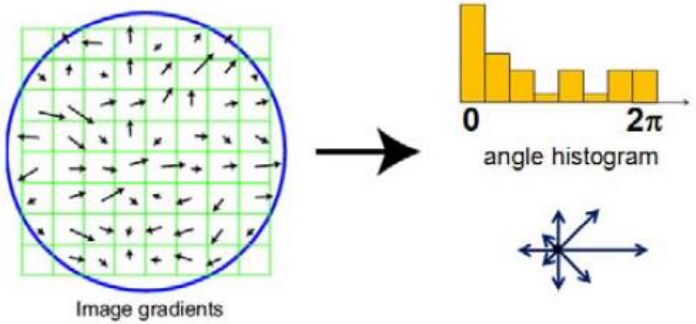
- Other feature types: Corners, blobs, ridges
- All have specialized detection algorithms

- Can we have some more generalized feature descriptors?

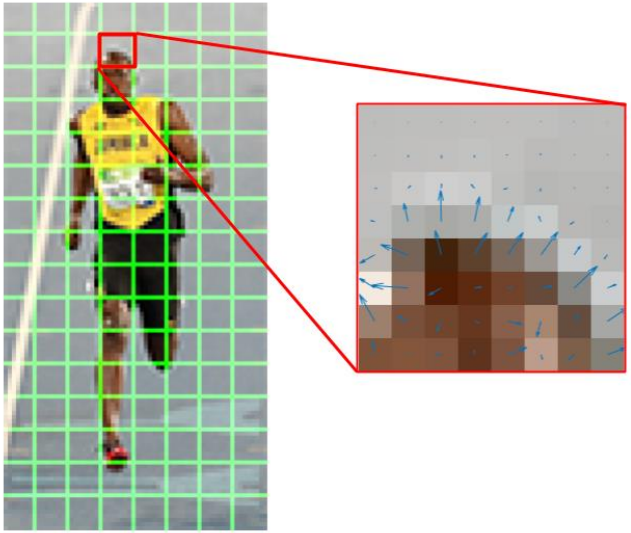
Histogram of Oriented Gradients

- Idea: Distribution of gradients in a local area may provide some useful information about object appearance and shape

- After computing gradient information at each pixel, look at local patches of discrete cells to find weighted histogram over all possible gradient angles
- All histograms in image are combined together as one giant feature vector



Example: HOGs



2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

Gradient Magnitude

80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

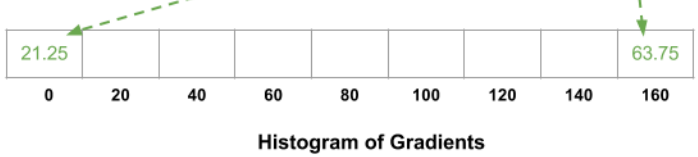
Gradient Direction

80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

Gradient Direction

2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

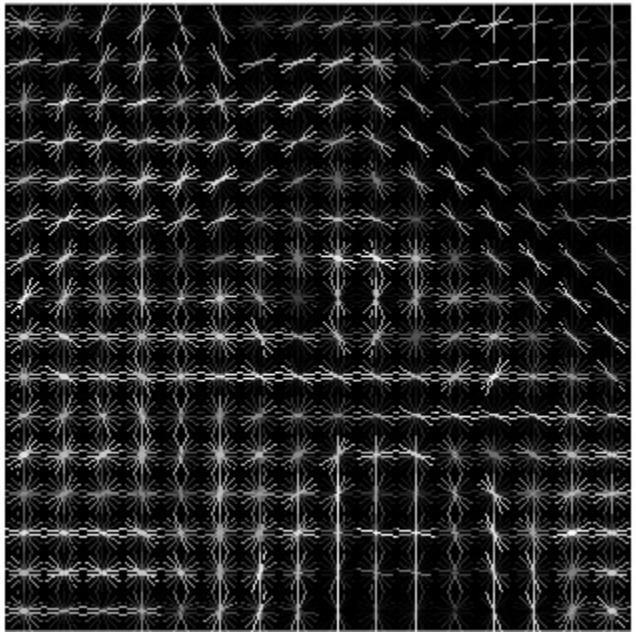
Gradient Magnitude



Histogram of Gradients

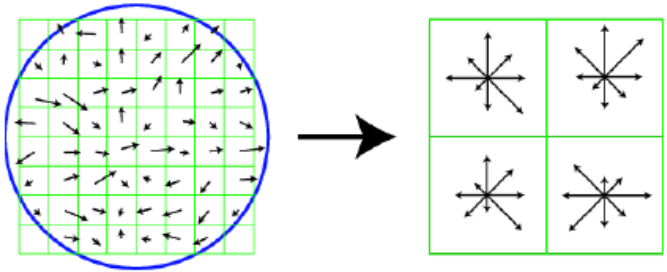
https://www.learnopencv.com/histogram-of-oriented-gradients/

Example: HOGs



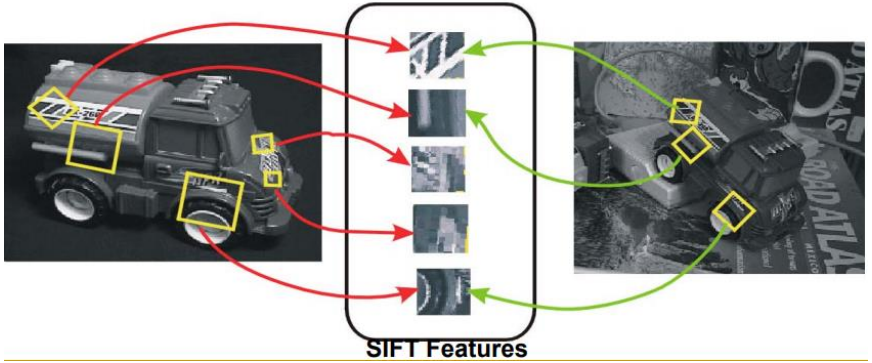
Scale-Invariant Feature Transform

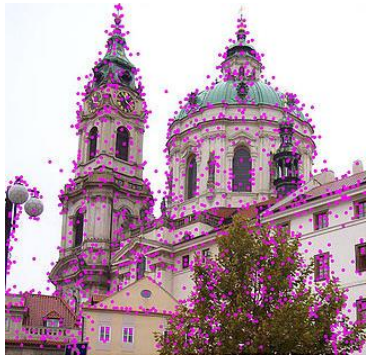
- We need to be able to match similar features across different images
- Same object might be viewed at different scales, angles, lighting, viewpoints
- High-level overview:



- Find key points as extrema on Gaussian-blurred image
- Refine localization of key points using thresholding
- Assign orientations to key points for robustness to rotations
- Create feature descriptor using orientation histograms

Example: SIFT





Summary

- Lots of different approaches in processing and understanding images!

- Segmentation: Dividing image up into meaningful components
- Feature detection: Find points of interest without extraneous information

- Techniques: Filters, thresholding, edge detection, HOGs, SIFT, …

- Next time: Vision used in robotics, including pose estimation and visual control