# Mid-Level Operations for Segmentation

# Recall: Thresholding Example

original image

pixels above threshold

original image kidney.jpg

# Image Segmentation Methods from Dhawan (ch 10)

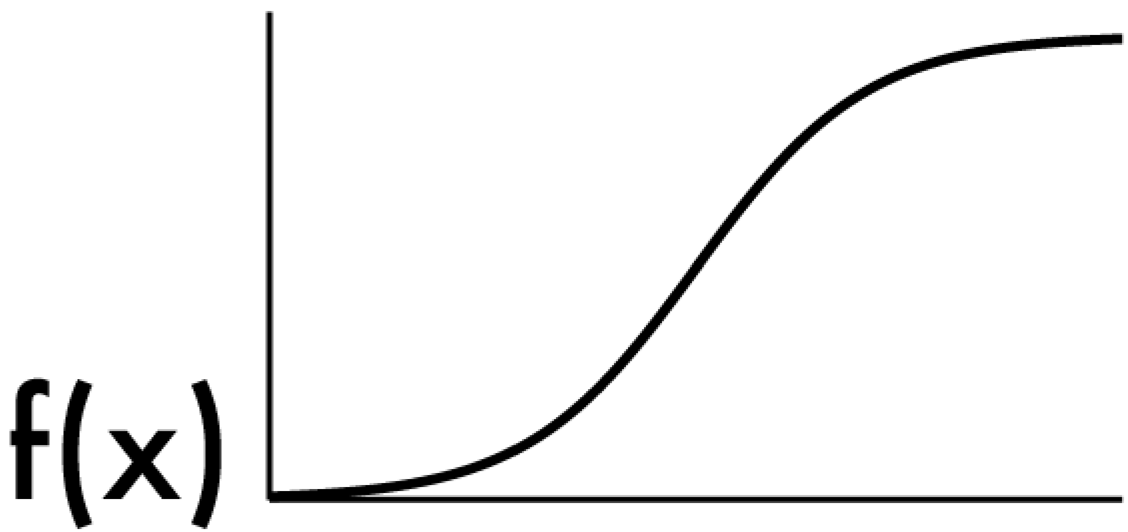
- Edge Detection
- Boundary Tracking
- Hough Transform
- Thresholding (we just covered)
- Clustering
- Region Growing (and Splitting)
- Estimation-Model Based
- Using Neural Networks (we do semantic segmentation this way)

# We’ll look at

- Thresholding (we just covered)
- Edge Detection
- Hough Transform
- Clustering
- Using Neural Networks (we do semantic segmentation this way)

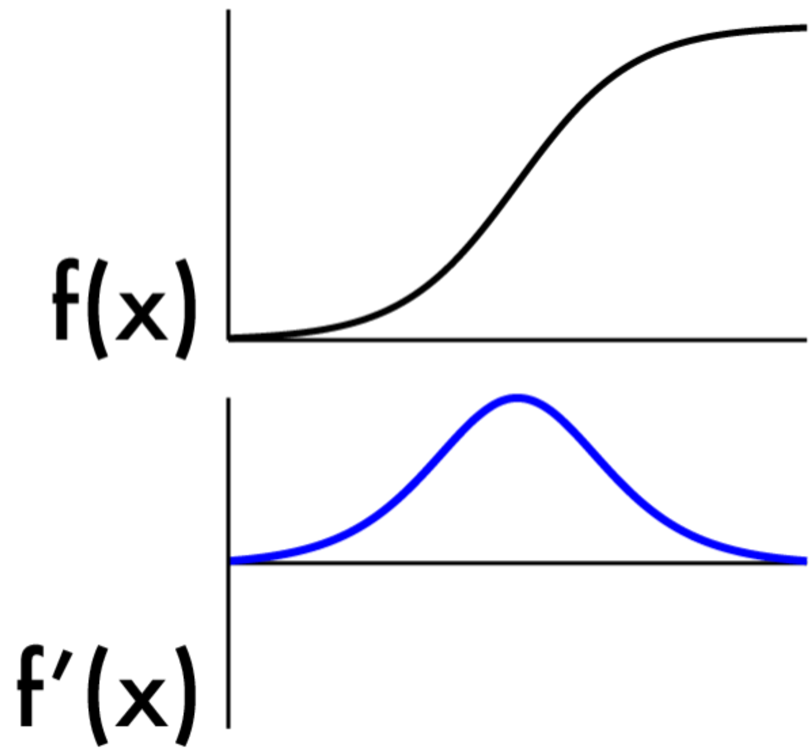
# What's an edge?

- Image is a function
- Edges are rapid changes in this function



# Finding edges

- Could take derivative
- Edges = high response



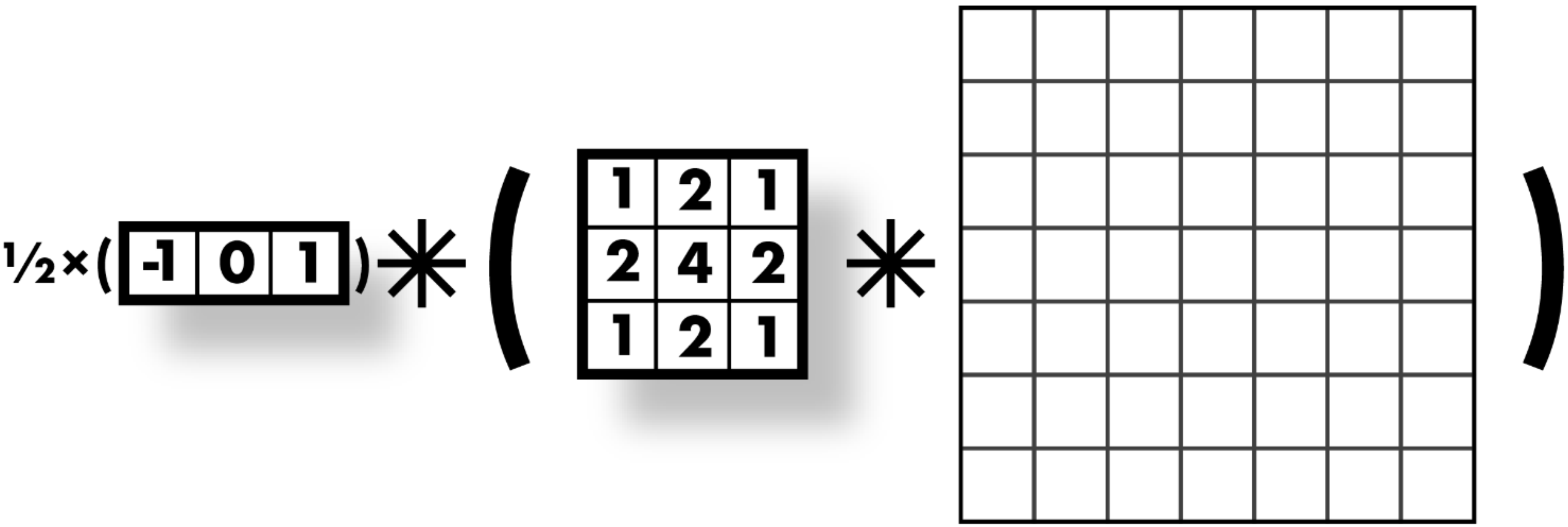
# To find edges, we use filters

- Use masks for filters
- Define a small mask
- Apply it to every pixel position in the image to produce a new output image
- In general, this is called filtering.
- We call linear filters CONVOLUTIONS (even though they are really correlations).

# Averaging Filters

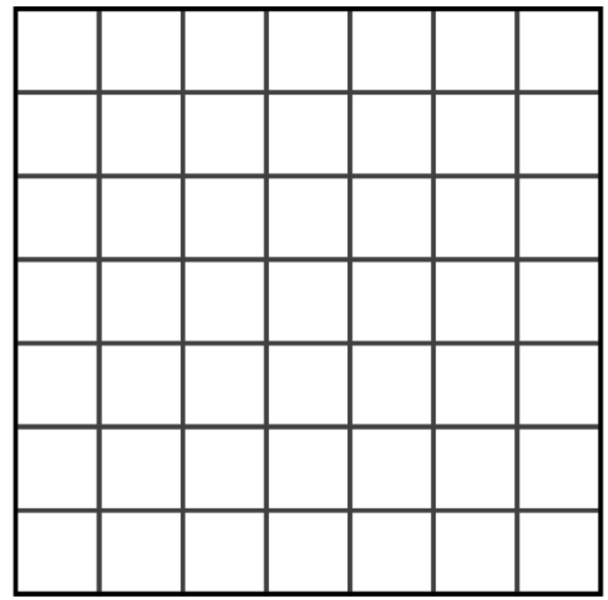
$$\frac{1}{9}\times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad \frac{1}{16}\times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

# Smooth first, then derivative

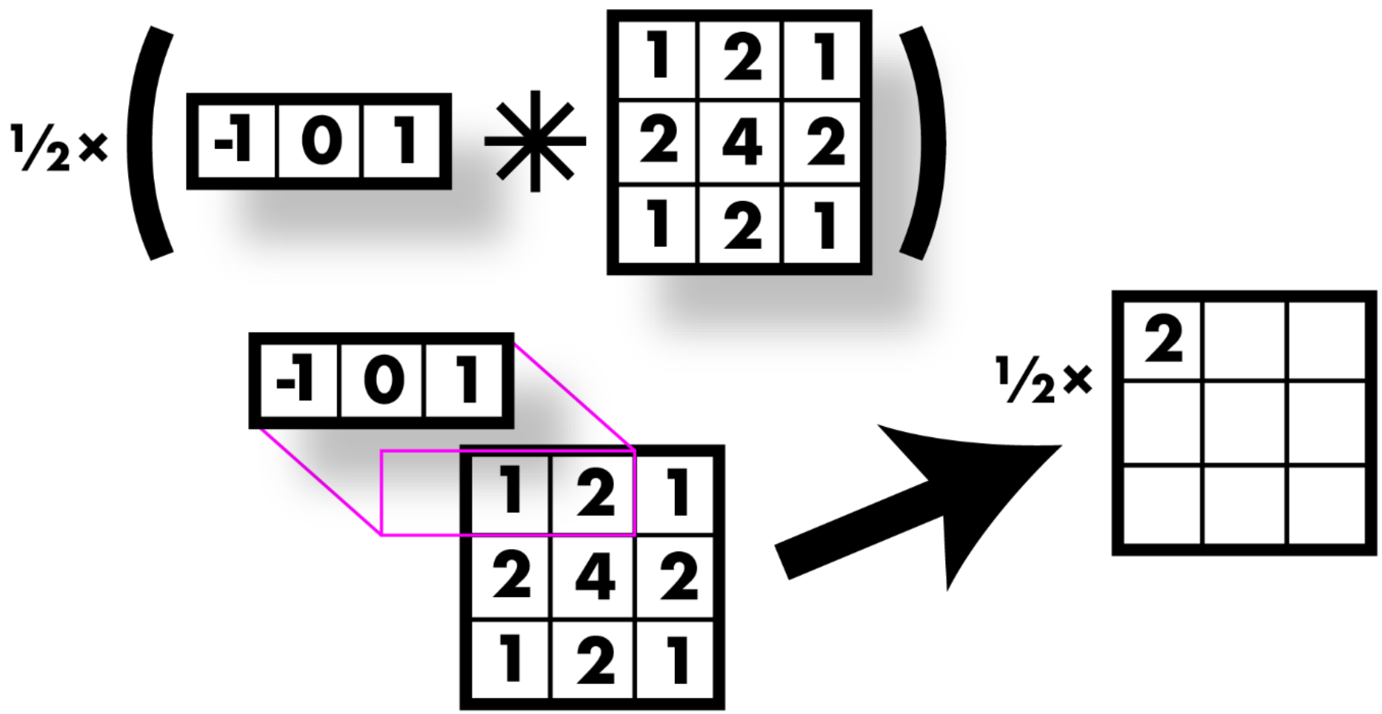


# Smooth first, then derivative

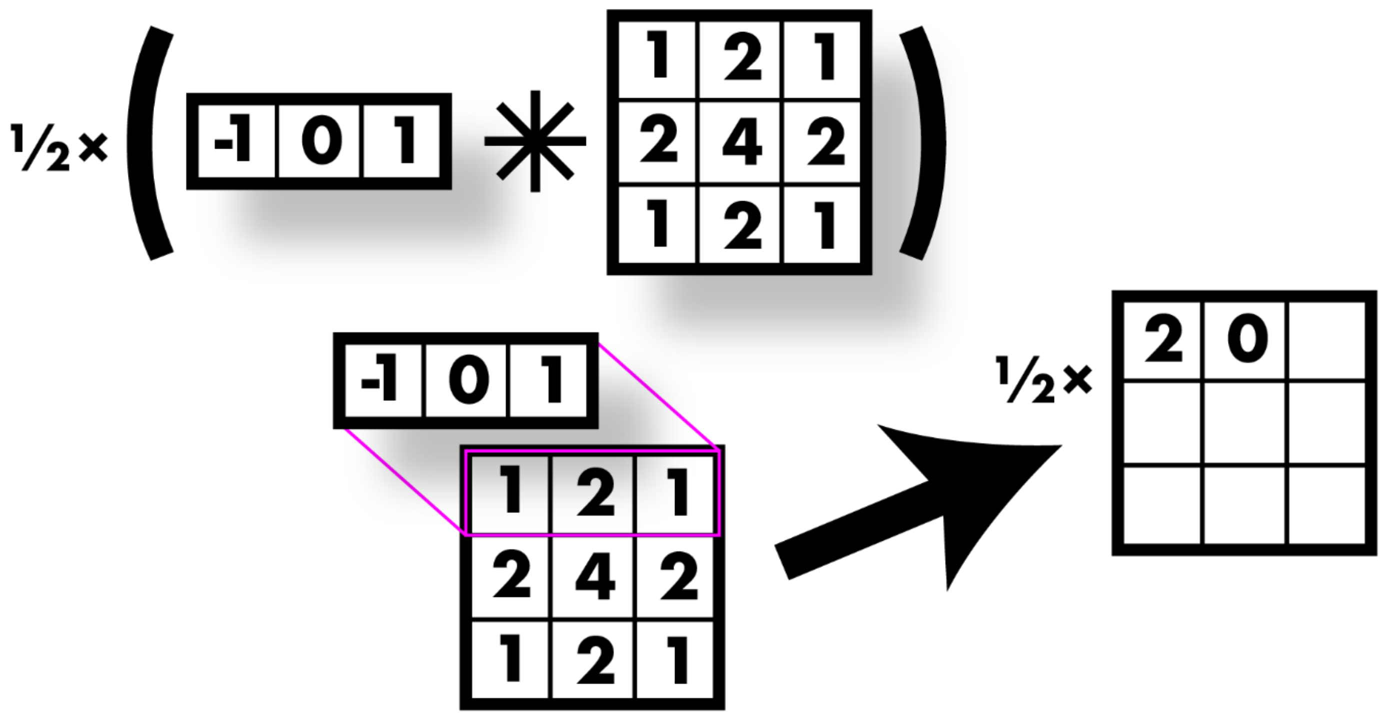
$$\frac{1}{2}\times\left(\begin{bmatrix}-1 & 0 & 1\end{bmatrix} * \begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}\right) *$$



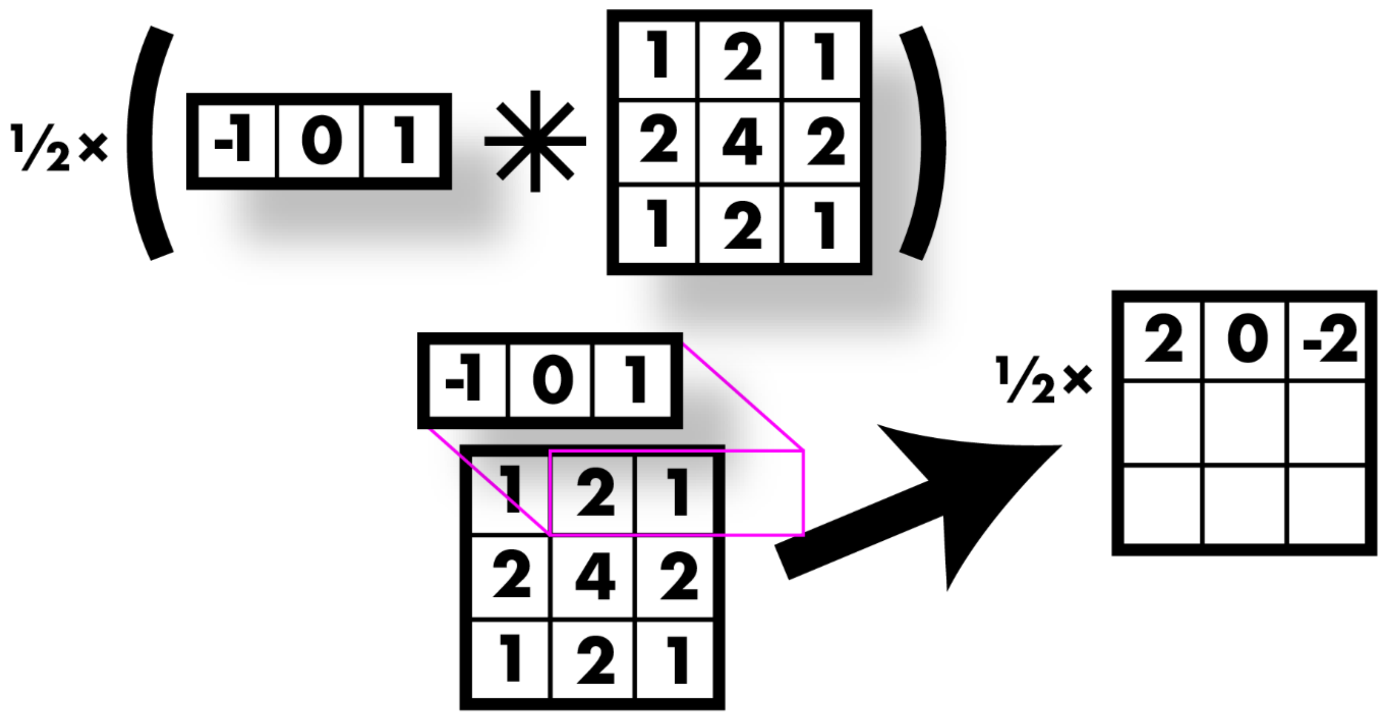
# Smooth first, then derivative



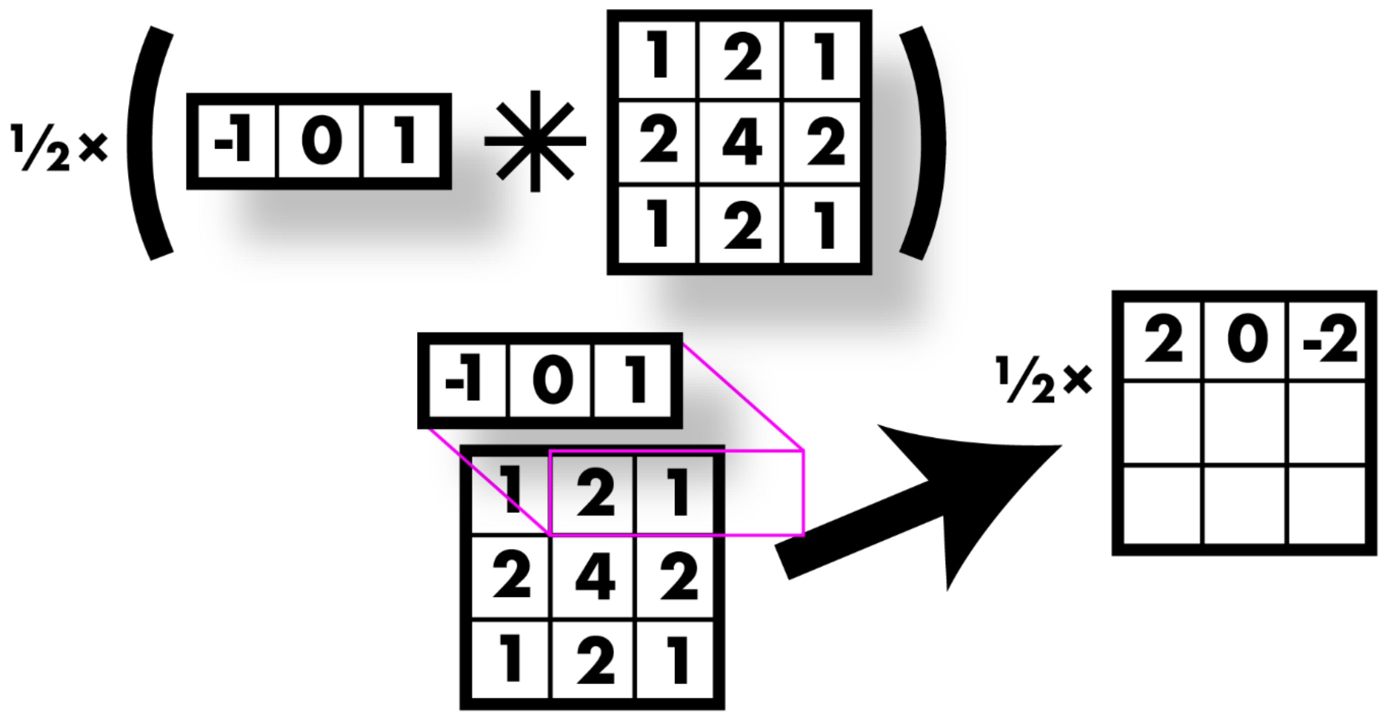
# Smooth first, then derivative



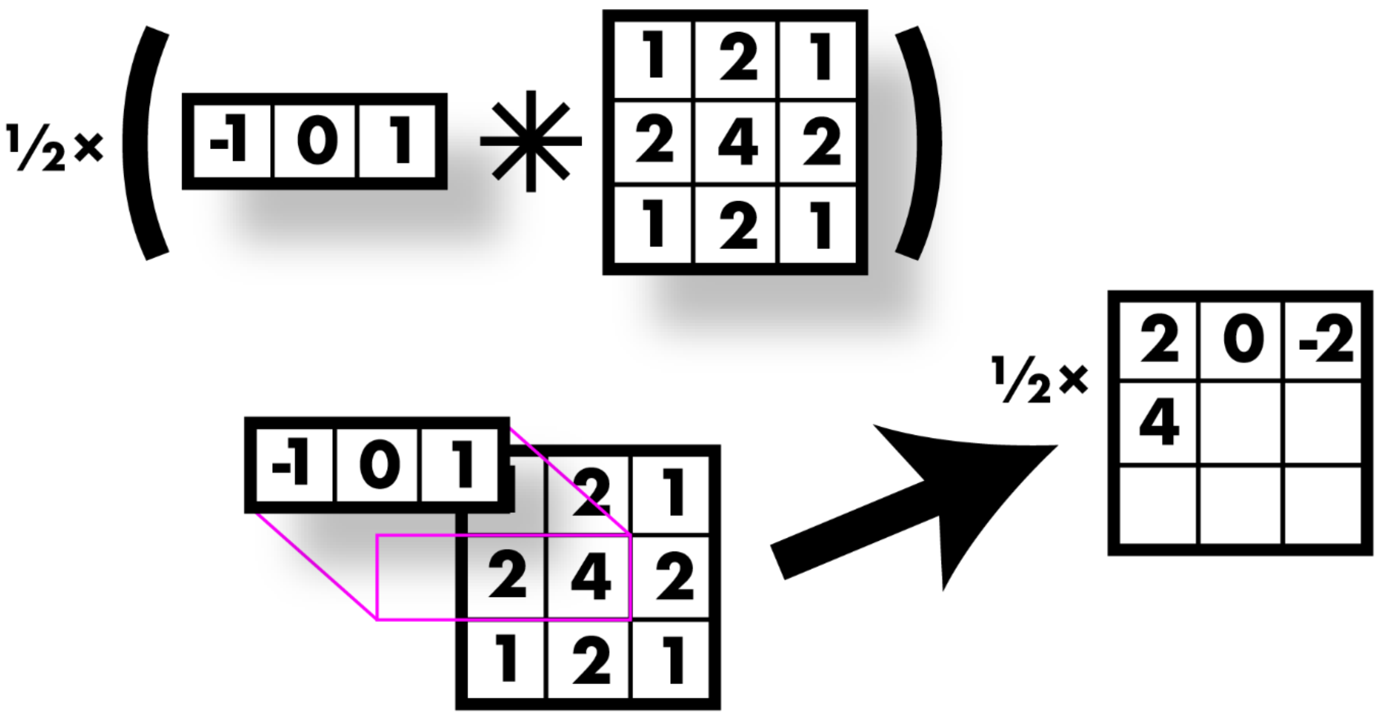
# Smooth first, then derivative



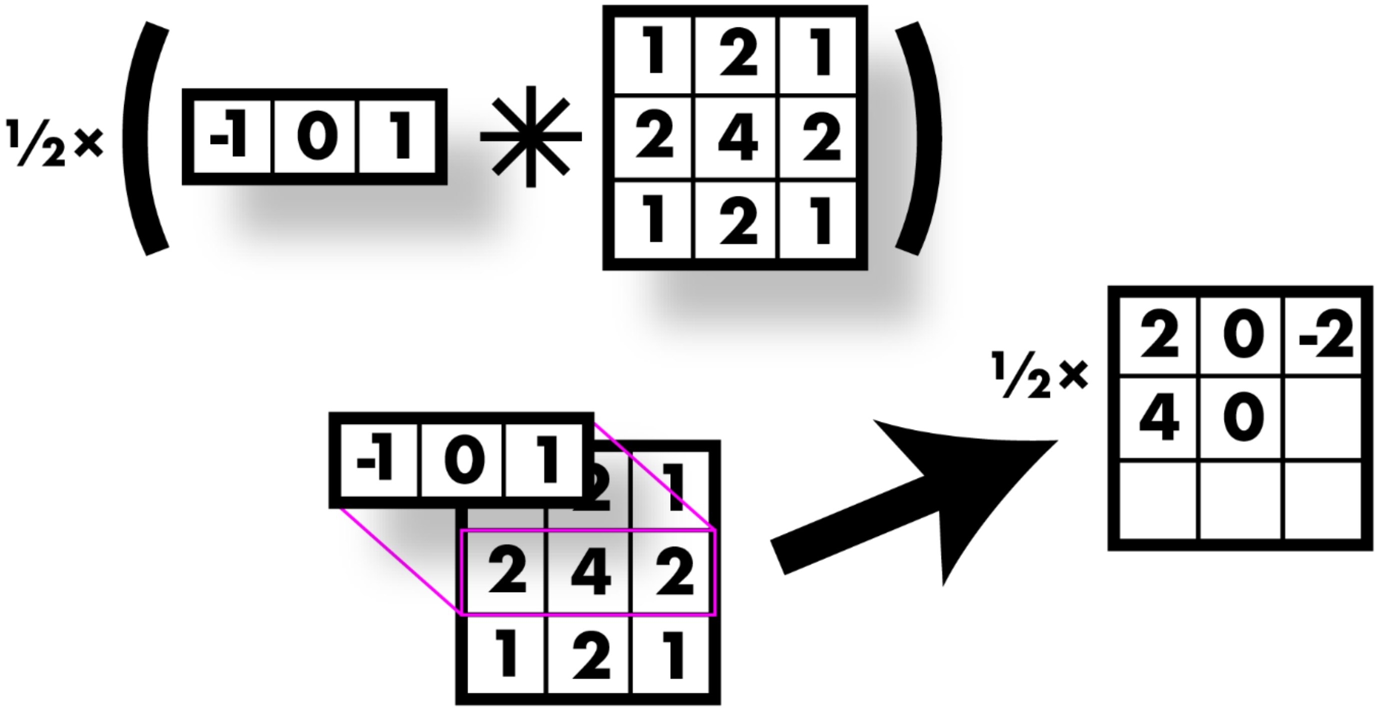
# Smooth first, then derivative



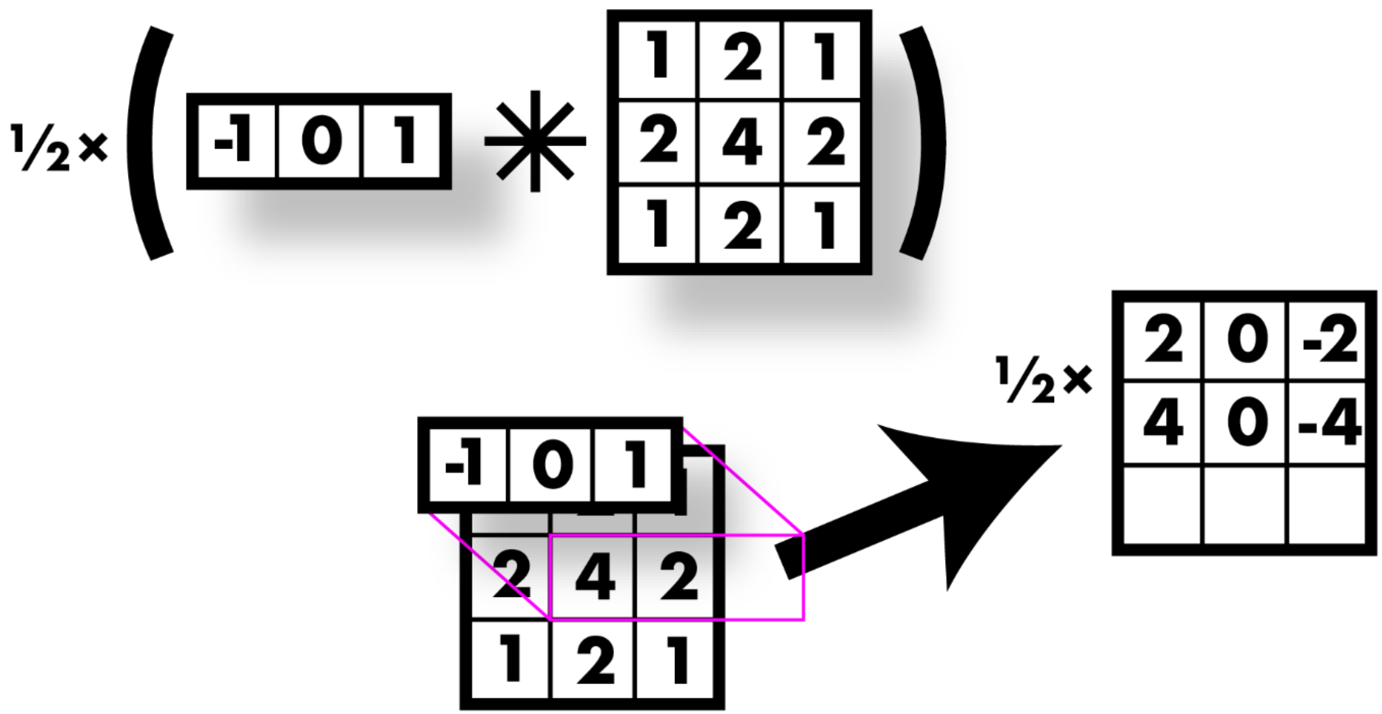
# Smooth first, then derivative



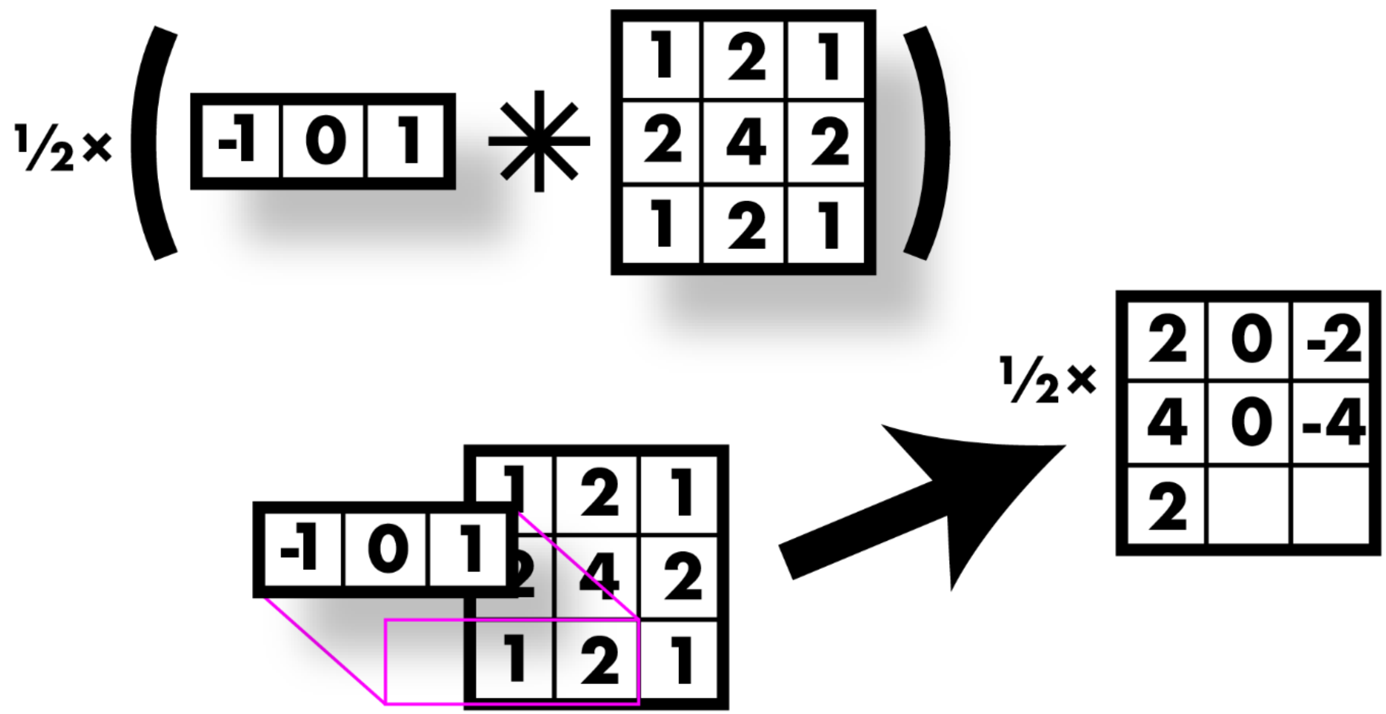
# Smooth first, then derivative



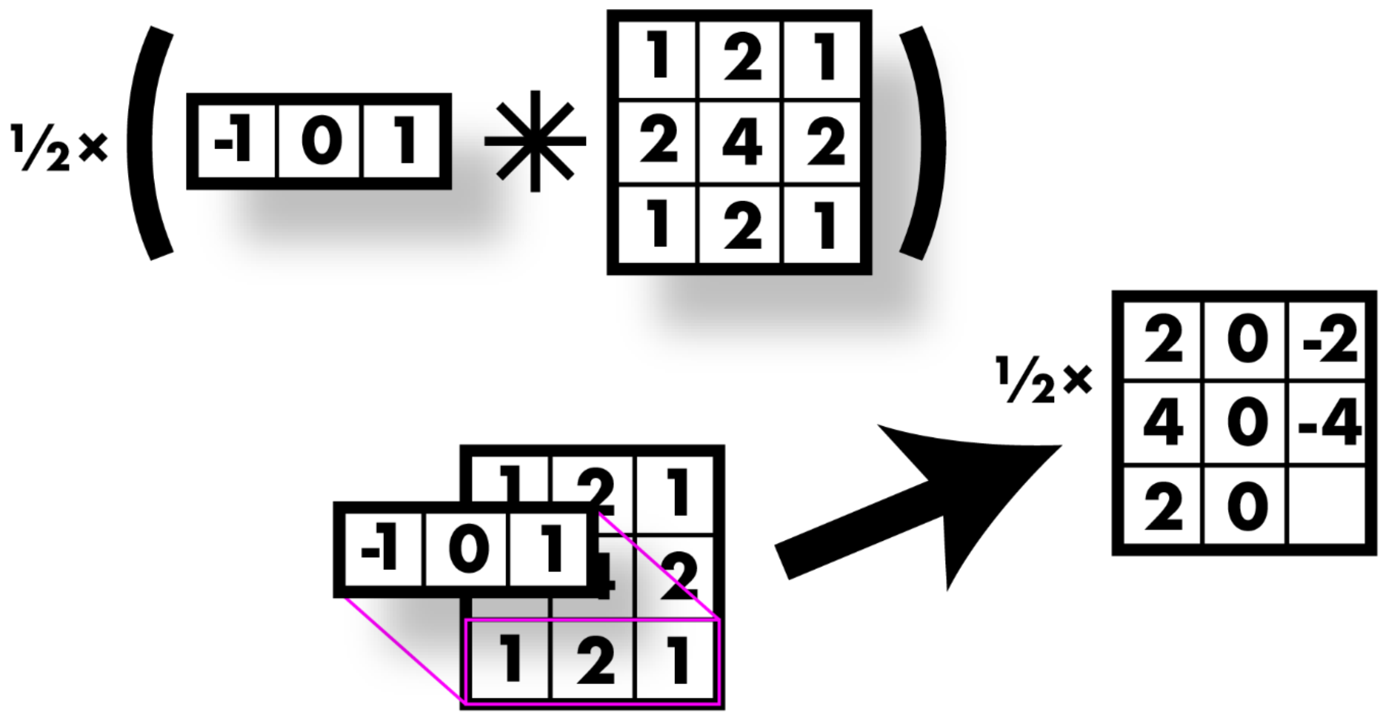
# Smooth first, then derivative



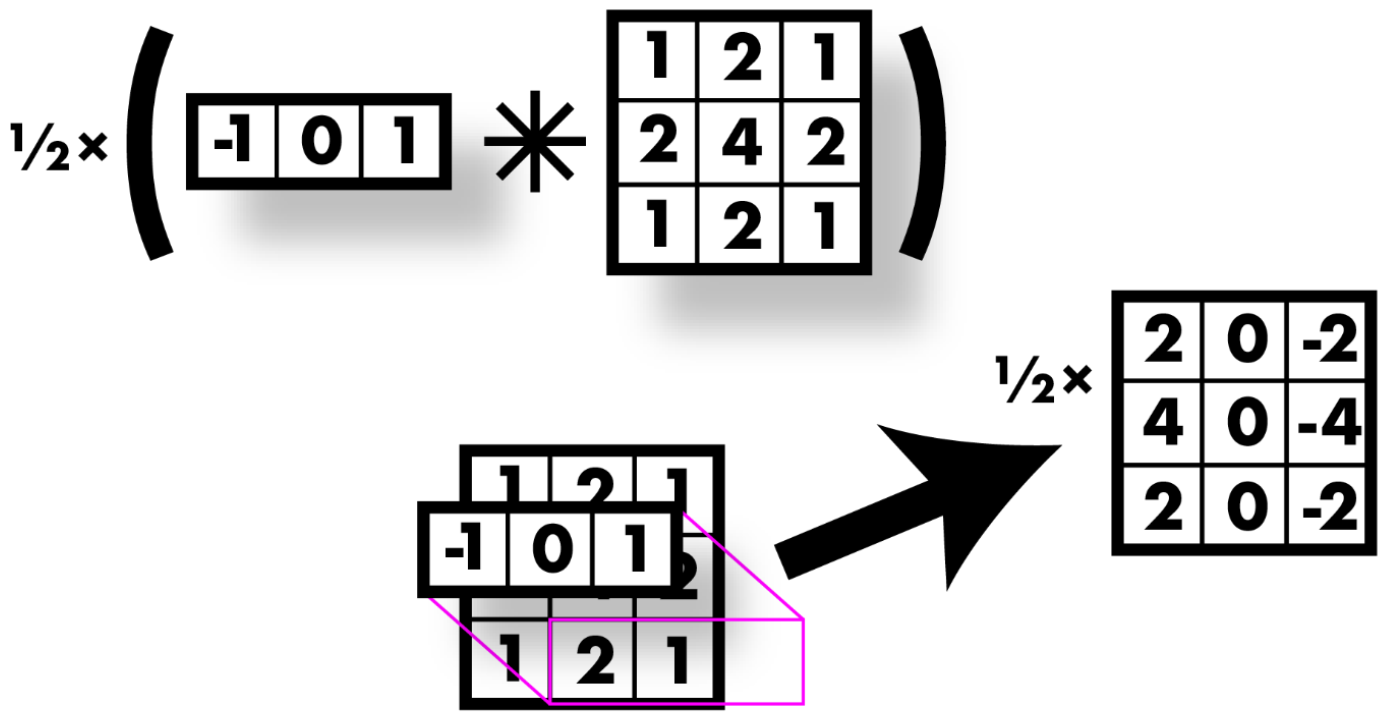
# Smooth first, then derivative



# Smooth first, then derivative

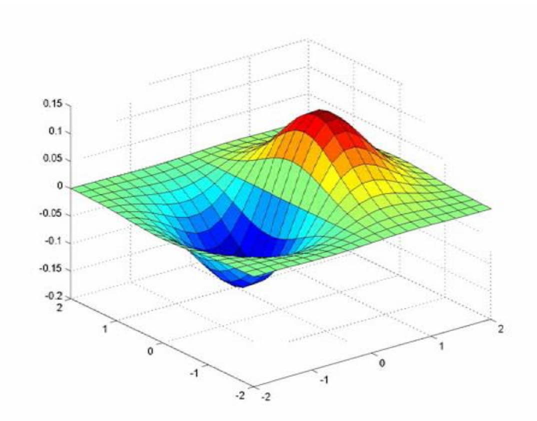


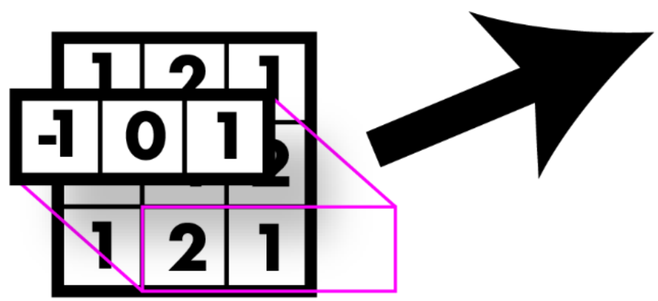
# Smooth first, then derivative



# Sobel filter! Smooth & derivative

$$\tfrac{1}{2}\times\left(\begin{bmatrix}-1 & 0 & 1\end{bmatrix} * \begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}\right)$$

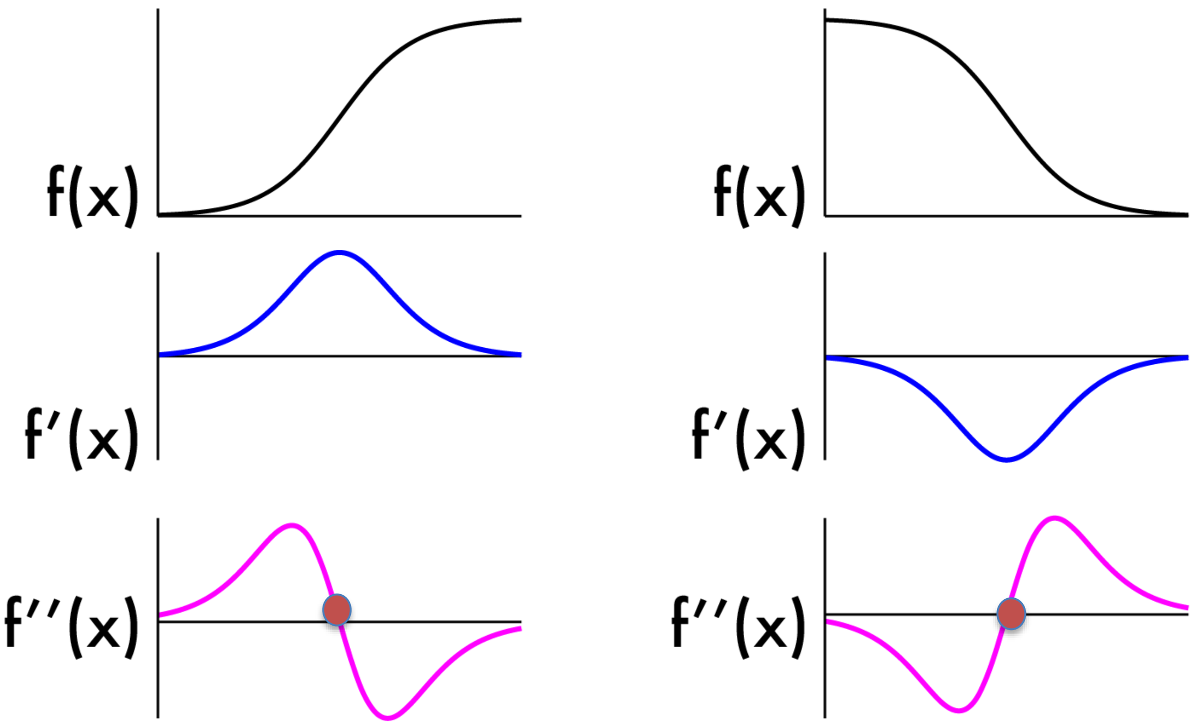



$$\begin{bmatrix}1 & 0 & -1\\ 2 & 0 & -2\\ 1 & 0 & -1\end{bmatrix}$$

# 2nd derivative!

- Crosses zero at extrema



# Canny Edge Detection

- Your first image processing pipeline!
  - Old-school CV is all about pipelines

Algorithm:

- 1. Smooth image (only want “real” edges, not noise)
- 2. Calculate gradient direction and magnitude
- 3. Non-maximum suppression perpendicular to edge
- 4. Threshold into strong, weak, no edge
- 5. Connect together components

http://bigwww.epfl.ch/demo/ip/demos/edgeDetector/

# Canny Characteristics

- The Canny operator gives single-pixel-wide images with good continuation between adjacent pixels

- It is the most widely used edge operator today; no one has done better since it came out in the late 80s. Many implementations are available.

- It is very sensitive to its parameters, which need to be adjusted for different application domains.

# Canny on Kidney

# An edge is not a line...

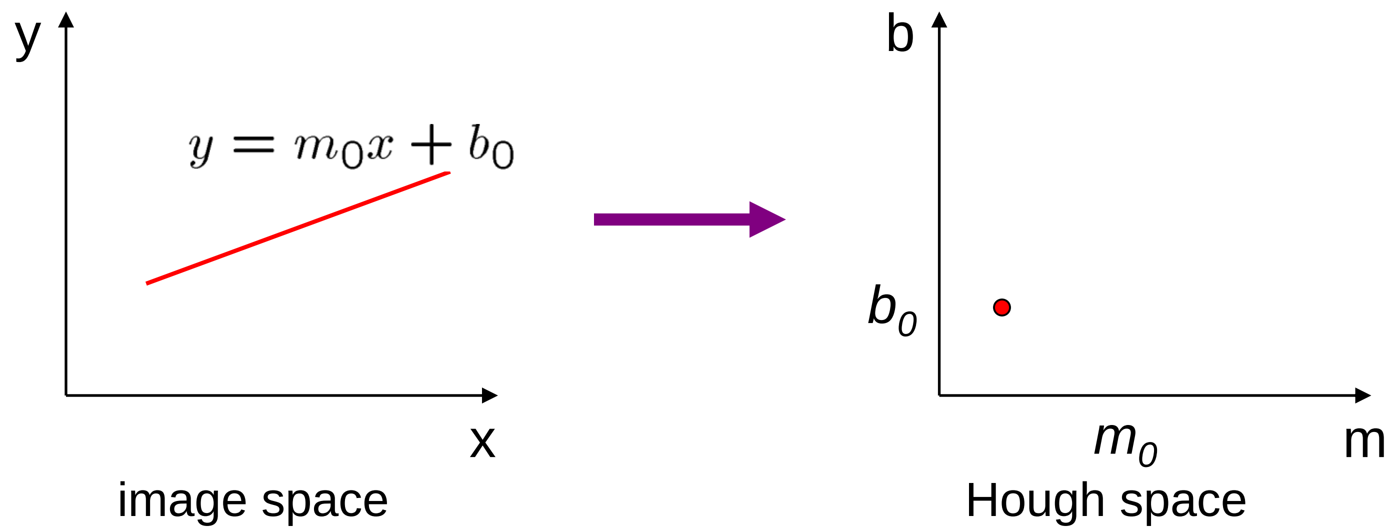
How can we detect ***lines*** ?

# Finding lines in an image

- Option 1:
  - Search for the line at every possible position/orientation
  - What is the cost of this operation?

- Option 2:
  - Use a voting scheme: Hough transform

# Finding lines in an image



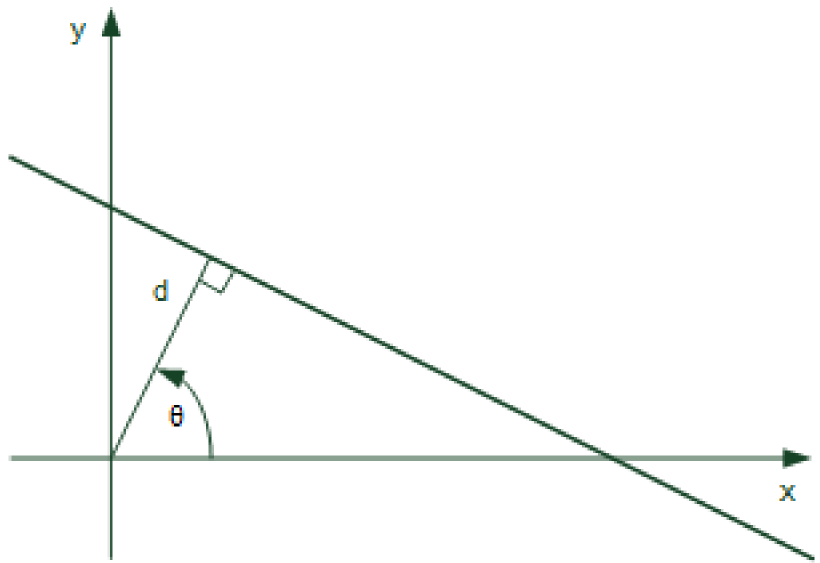

- Connection between image (x,y) and Hough (m,b) spaces
  - A line in the image corresponds to a point in Hough space
  - To go from image space to Hough space:
    - given a set of points (x,y), find all (m,b) such that y = mx + b

# Hough transform algorithm

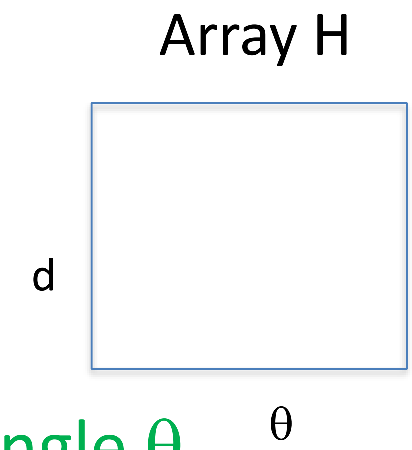
- Typically use a different parameterization

$$d = x cos\theta + y sin\theta$$

  - d is the perpendicular distance from the line to the origin
  - θ is the angle of this perpendicular with the horizontal.



# Hough transform algorithm




- Basic Hough transform algorithm
  1. Initialize H[d, $\theta$]=0
  2. for each edge point I[x,y] in the image

     compute gradient magnitude m and angle $\theta$

     $$d = x cos\theta + y sin\theta$$

     H[d, $\theta$] += 1

  3. Find the value(s) of (d, $\theta$) where H[d, $\theta$] is maximum
  4. The detected line in the image is given by

     $$d = x cos\theta + y sin\theta$$

Complexity? How do you get the lines out of the matrix?

# Line segments from Hough Transform

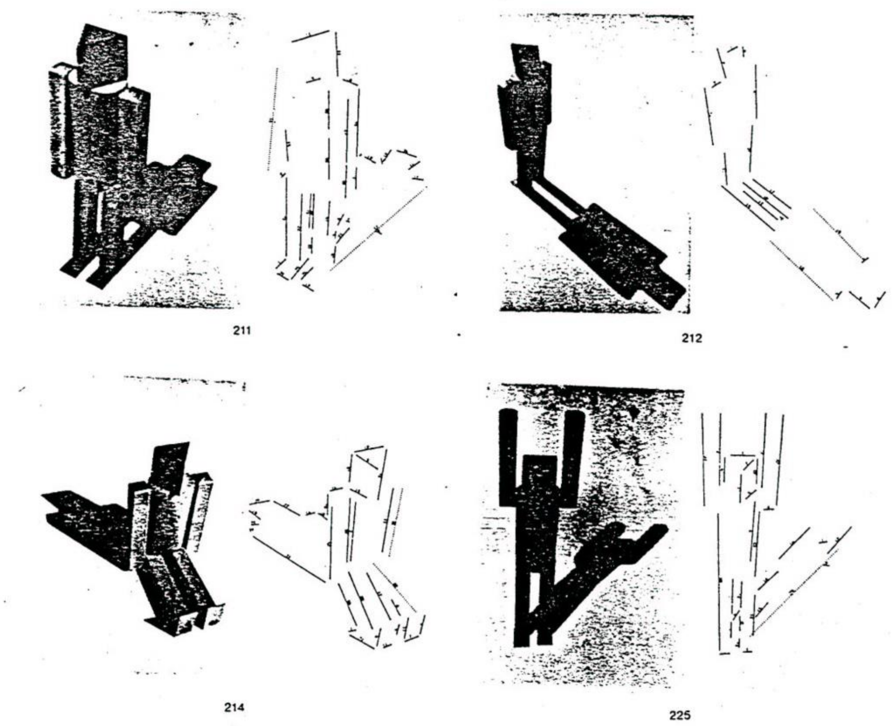



Fig.7. Puppet scenes 211, 212, 214, 225 and the edges recovered by the algorithm.

# Extensions

- Extension 1: Use the image gradient (we just did that)

- Extension 2
  - give more votes for stronger edges
- Extension 3
  - change the sampling of (d, $\theta$) to give more/less resolution
- Extension 4
  - The same procedure can be used with **circles,** squares, or any other shape, How?
- Extension 5; the Burns procedure. Uses only angle, two different quantifications, and connected components with votes for larger one.

# Finding lung nodules (Kimme & Ballard)

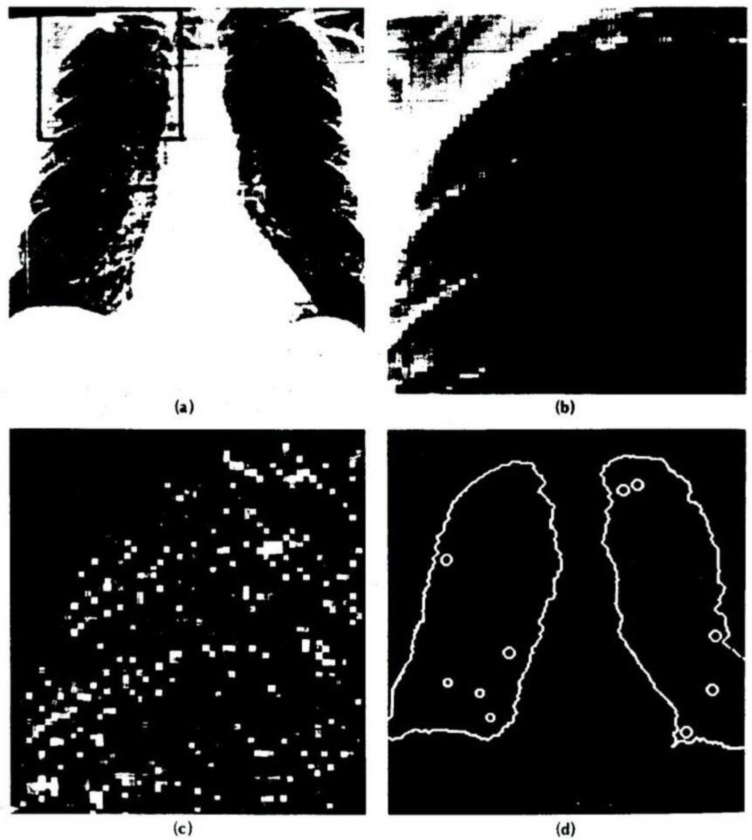



**Fig. 4.7** Using the Hough technique for circular shapes. (a) Radiograph. (b) Window. (c) Accumulator array for $r = 3$. (d) Results of maxima detection.

# K-Means Clustering

Form K-means clusters from a set of n-dimensional vectors

1. Set ic (iteration count) to 1

2. Choose randomly a set of K means $m_1(1)$, …, $m_K(1)$.

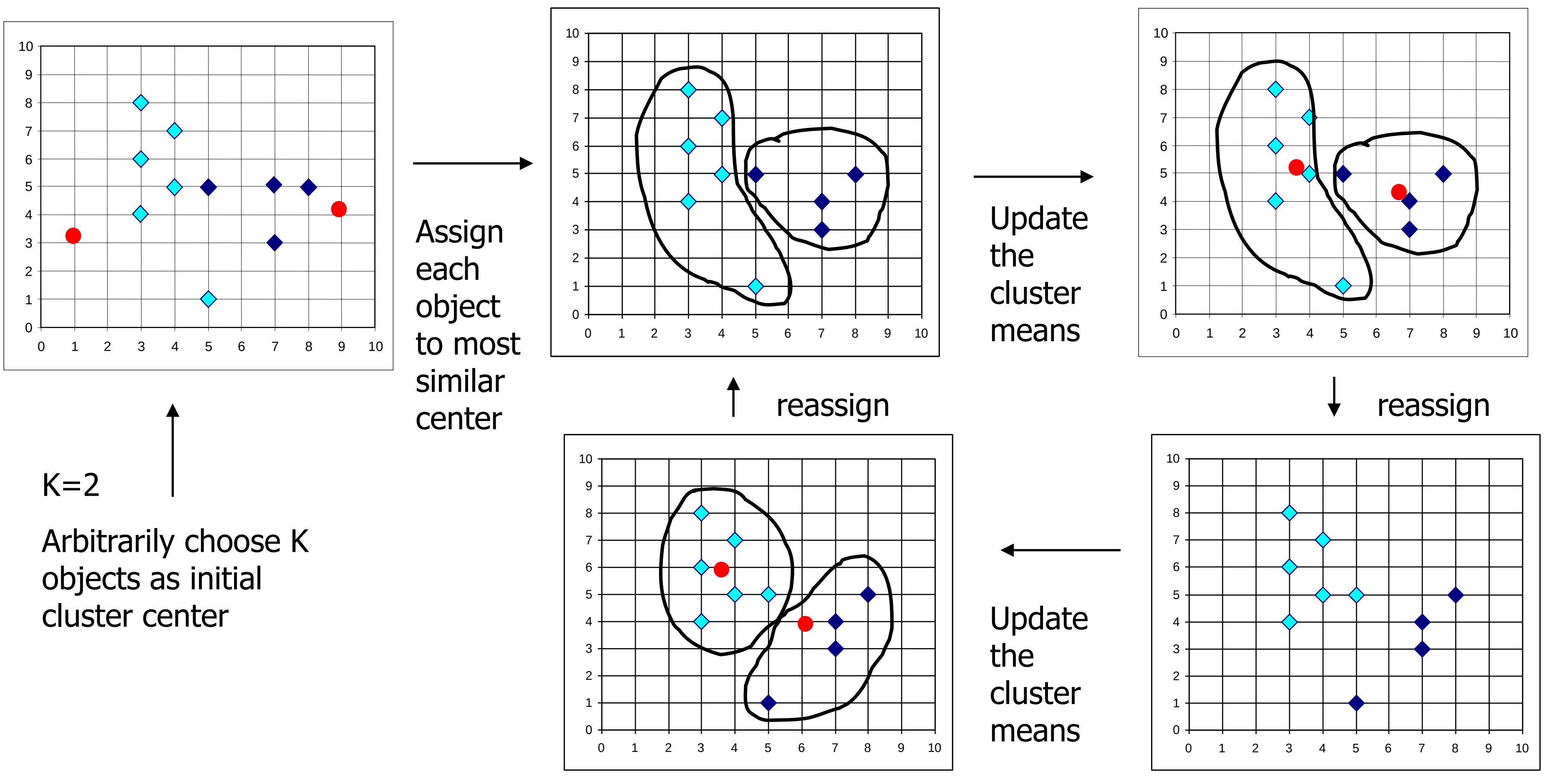
3. For each vector $x_i$ compute $D(x_i , m_k(ic))$, $k=1,\ldots K$
   and assign $x_i$ to the cluster $C_j$ with nearest mean.

4. Increment ic by 1, update the means to get $m_1(ic),\ldots,m_K(ic)$.

5. Repeat steps 3 and 4 until $C_k(ic) = C_k(ic+1)$ for all k.

# Simple Example

## INIT.

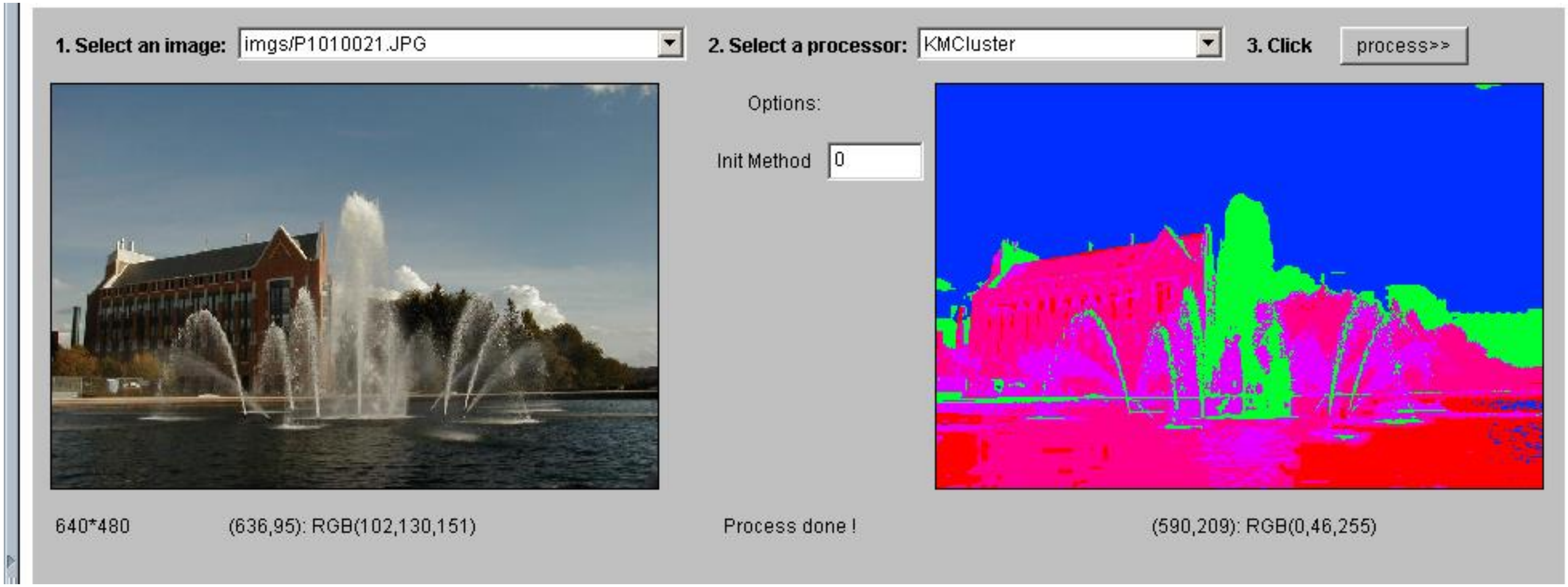


# Space for K-Means

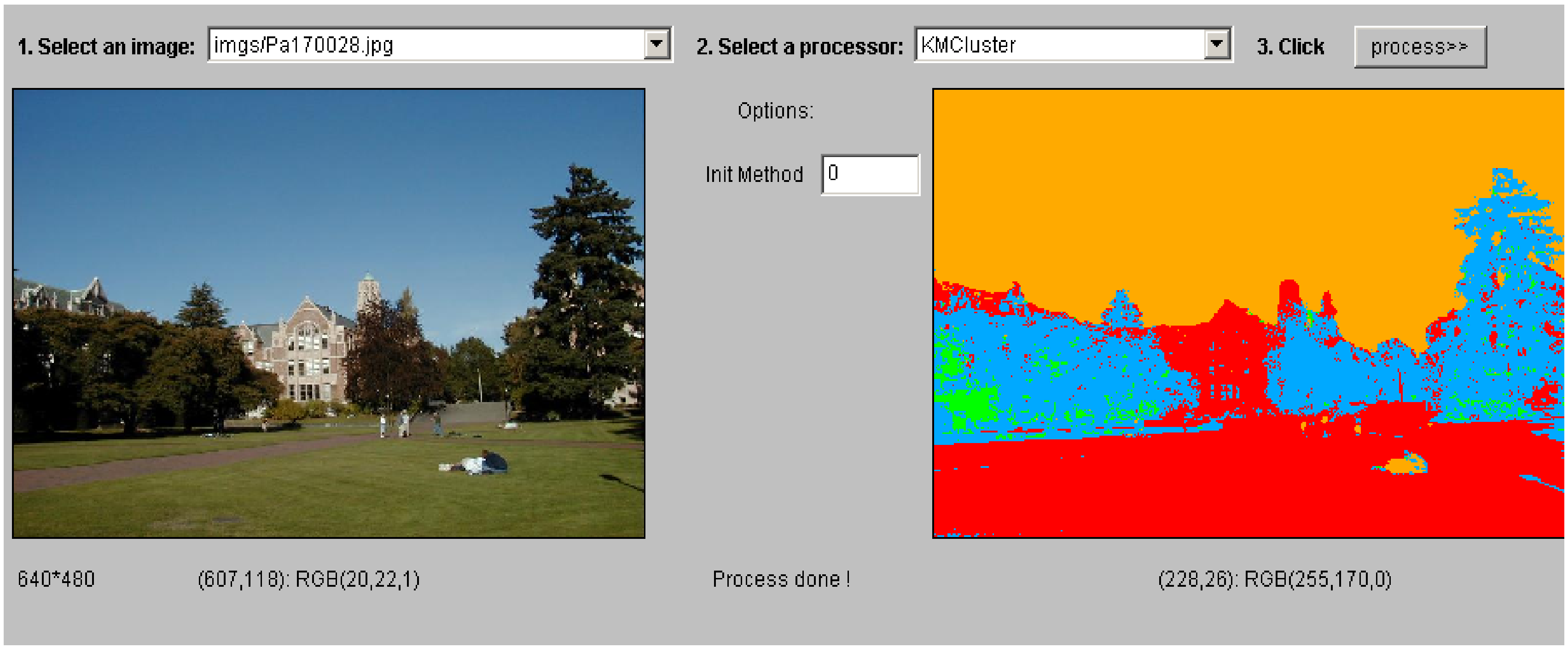
- The example was in some arbitrary 2D space
- We don't want to cluster in that space.
- We will be clustering in gray-scale space or color space.
- K-means can be used to cluster in any n-dimensional space.

# K-Means Example 1

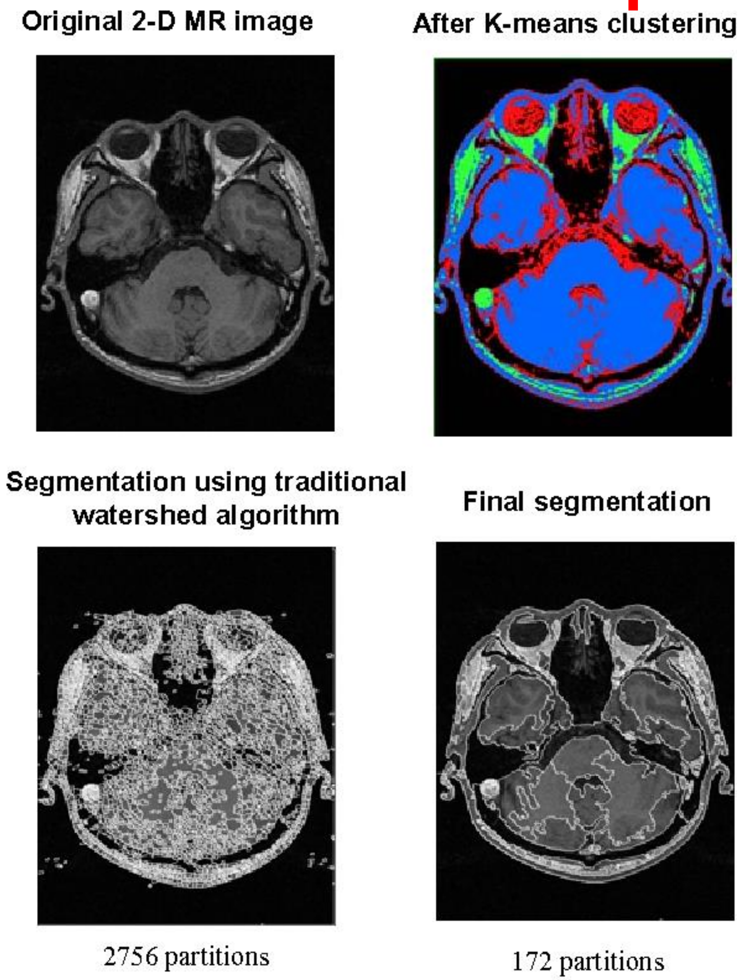
# K-Means Example 2



# K-Means Example 3



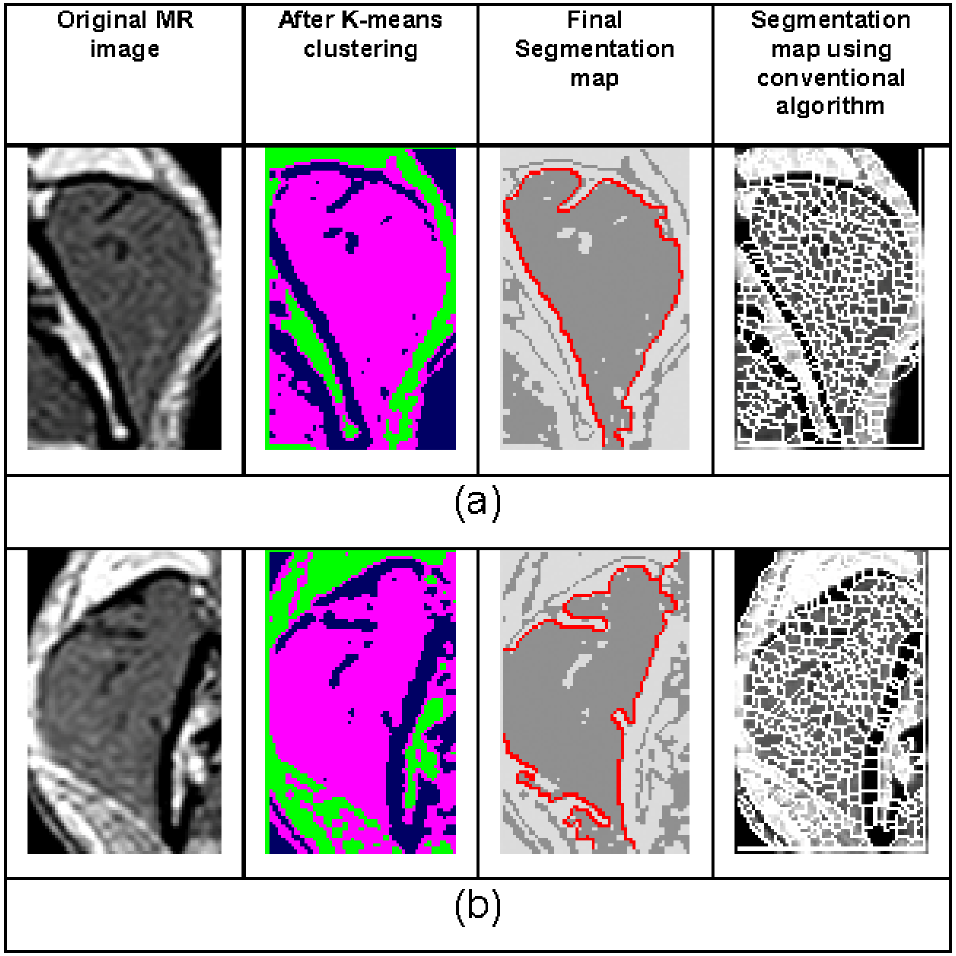
# K-Means Example 4




(b)

Published in 2006 IEEE Southwest Symposium on Image Analysis and Interpretation 2006
Medical Image Segmentation Using K-Means Clustering and Improved Watershed Algorithm
H. P. Ng, S. Ong, K. Foong, P. Goh, W. Nowinski

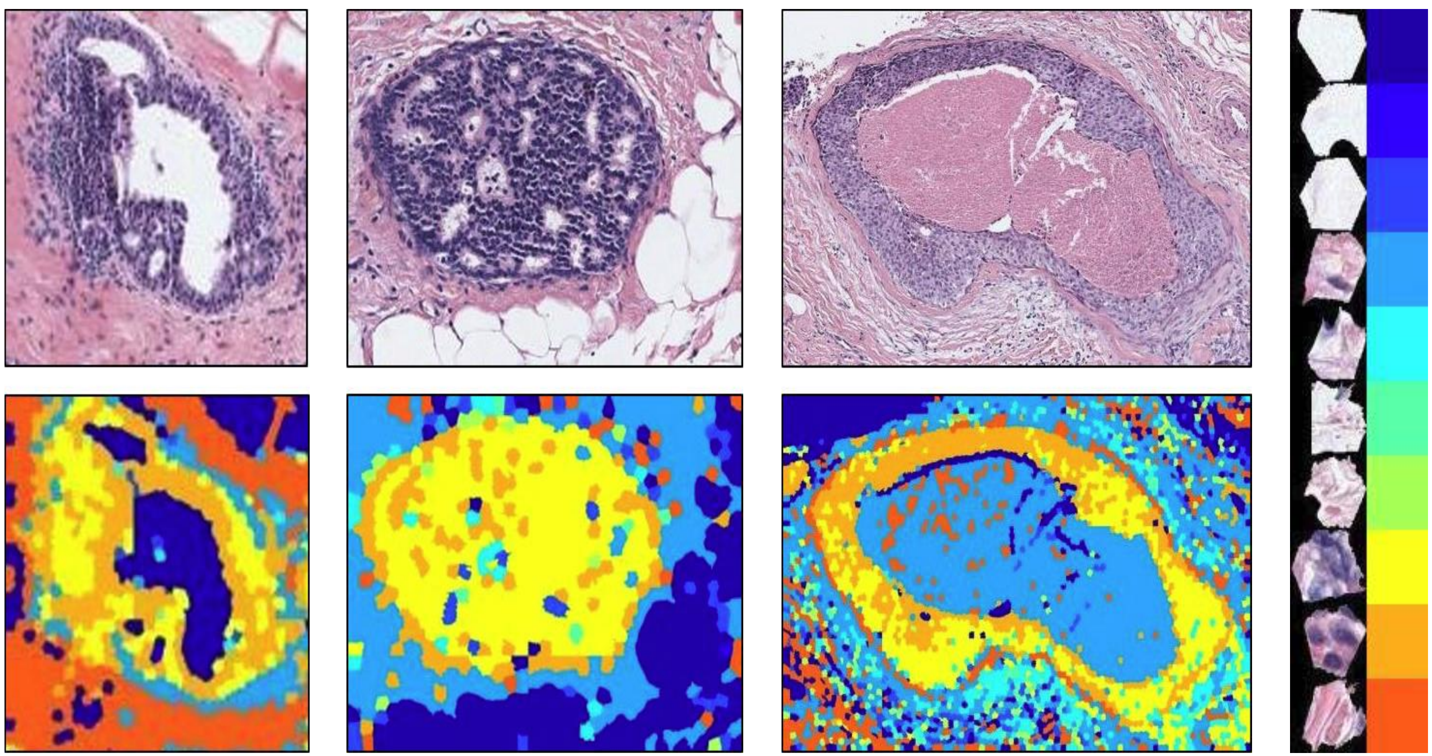
# K-Means Example 5




Published in 2006 IEEE Southwest Symposium on Image Analysis and Interpretation 2006
Medical Image Segmentation Using K-Means Clustering and Improved Watershed Algorithm
H. P. Ng, S. Ong, K. Foong, P. Goh, W. Nowinski

# K-Means Example 5

- Superpixel clustering in breast biopsy images



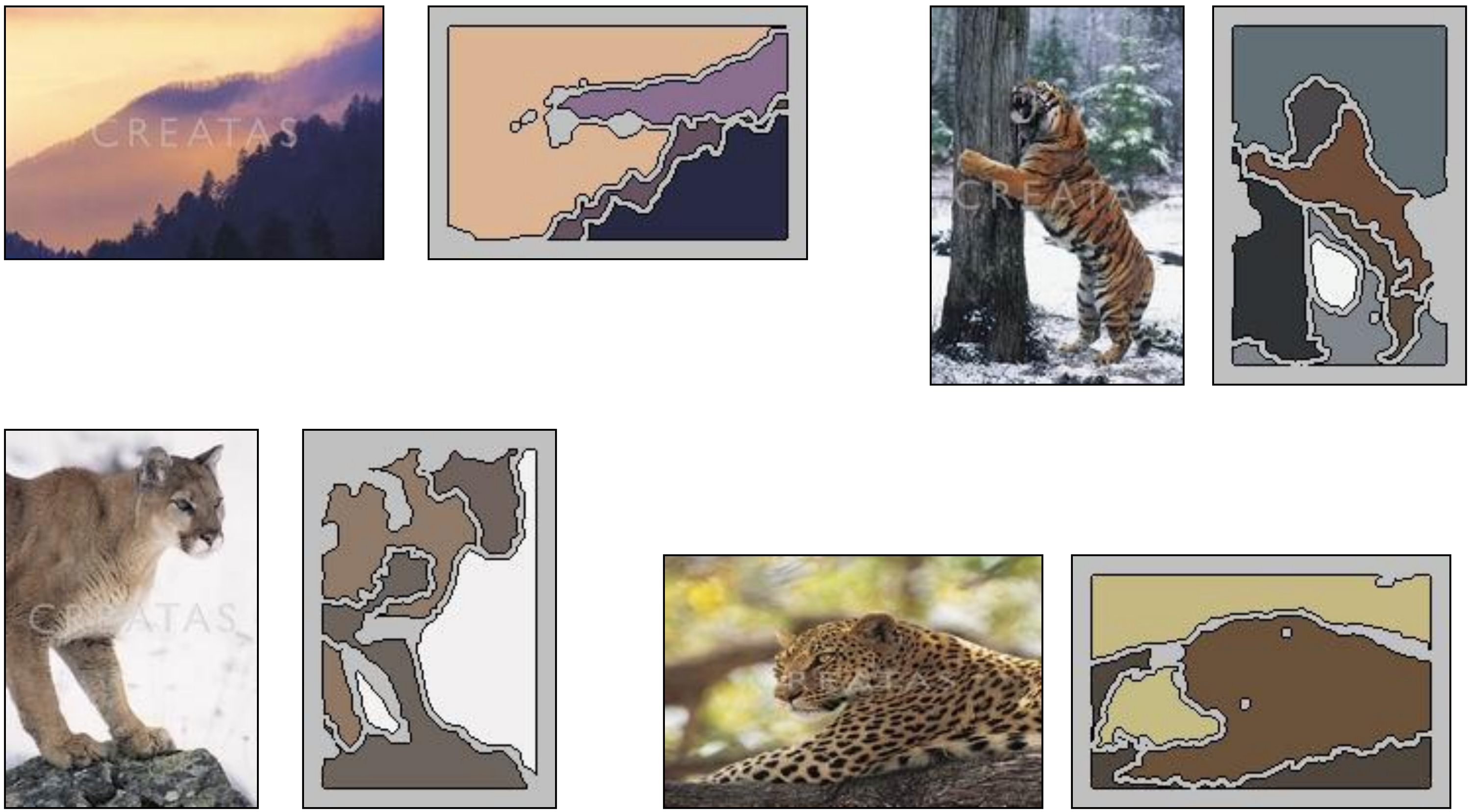
*benign* *atypia* *DCIS*

# K-means Variants

- Different ways to initialize the means
- Different stopping criteria
- Dynamic methods for determining the right number of clusters (K) for a given image

- The EM Algorithm: a probabilistic formulation of K-means

# Blobworld: Sample Results using color, texture, and EM

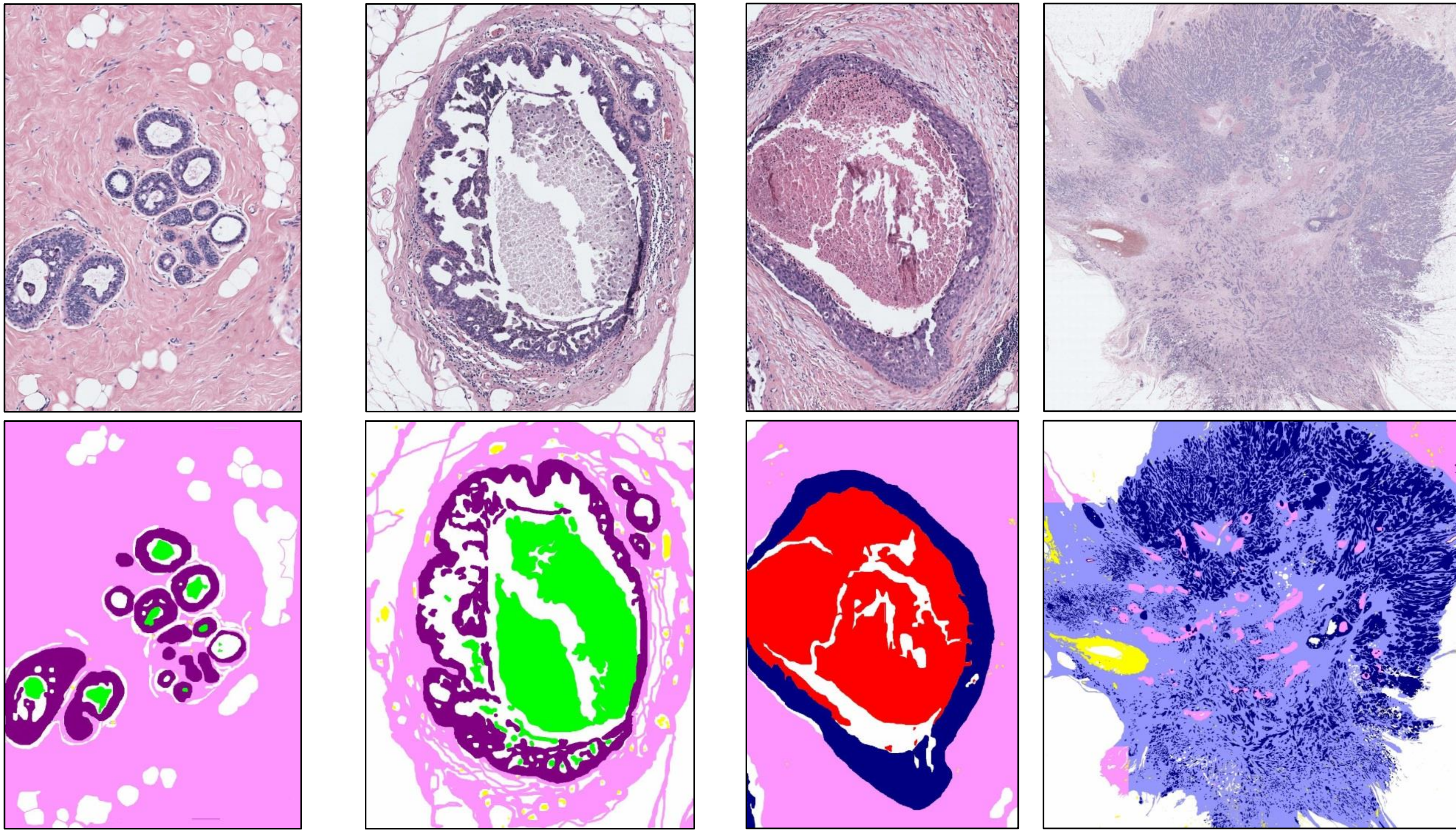


# Semantic Segmentation

- Instead of grouping pixels based on color, texture or whatever properties
- Teach a classifier what important regions look like, so it can find them.
- This is usually done via deep learning, which we will discuss later in the course.
- But here’s a preview.

# Training Labels

background · benign epithelium · normal stroma · secretion · necrosis
malignant epithelium · desmoplastic stroma · blood

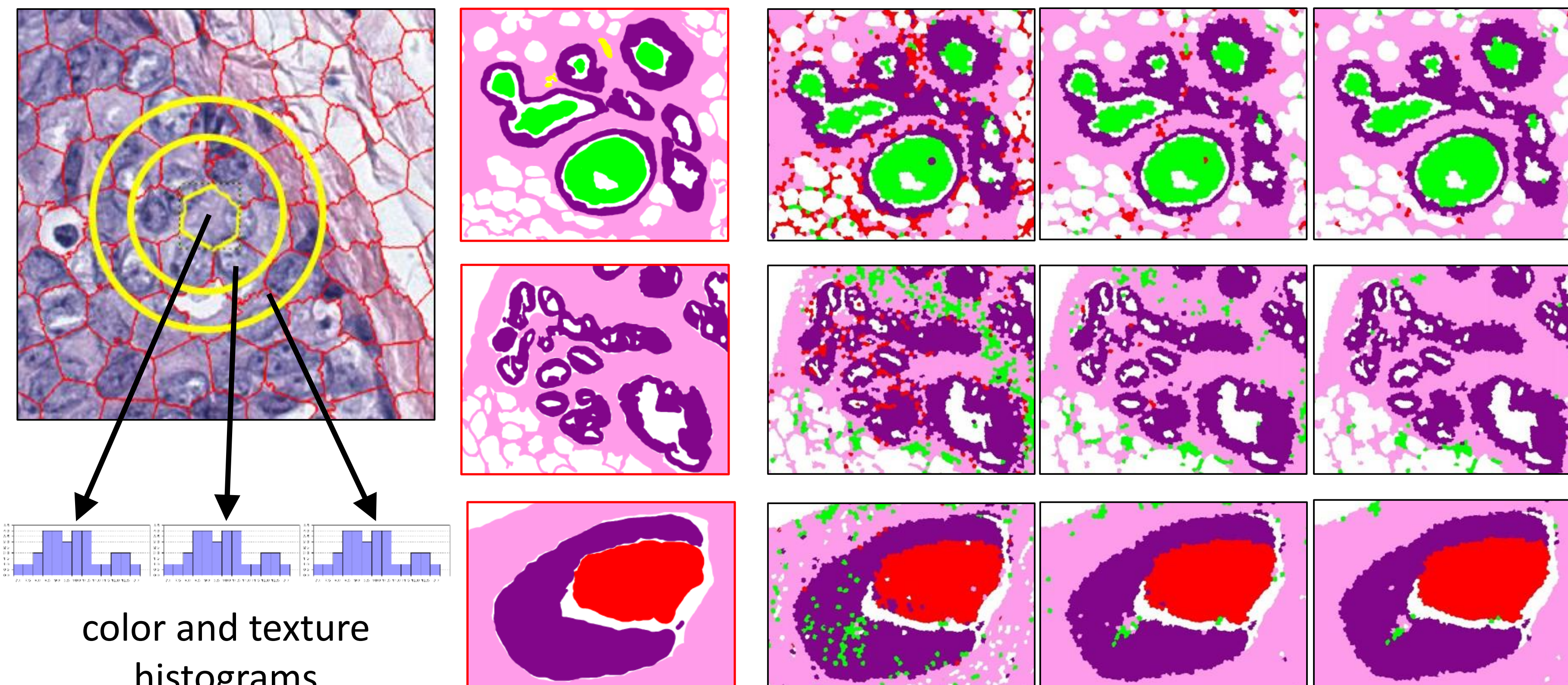


# Meaning of Labels

- Benign Epithelium: epithelial cells from the benign and atypia categories
- Malignant Epithelium: epithelial cells from DCIS and invasive cancer
- Normal Stroma: normal connective tissue
- Desmoplastic Stroma: stroma associated with a tumor
- Secretion: benign substance filling the ducts
- Necrosis: dead cells at the center of the ducts in DCIS and invasive cases
- Blood: blood cells
- Background: empty areas inside ducts

# Superpixel + SVM-based Segmentation

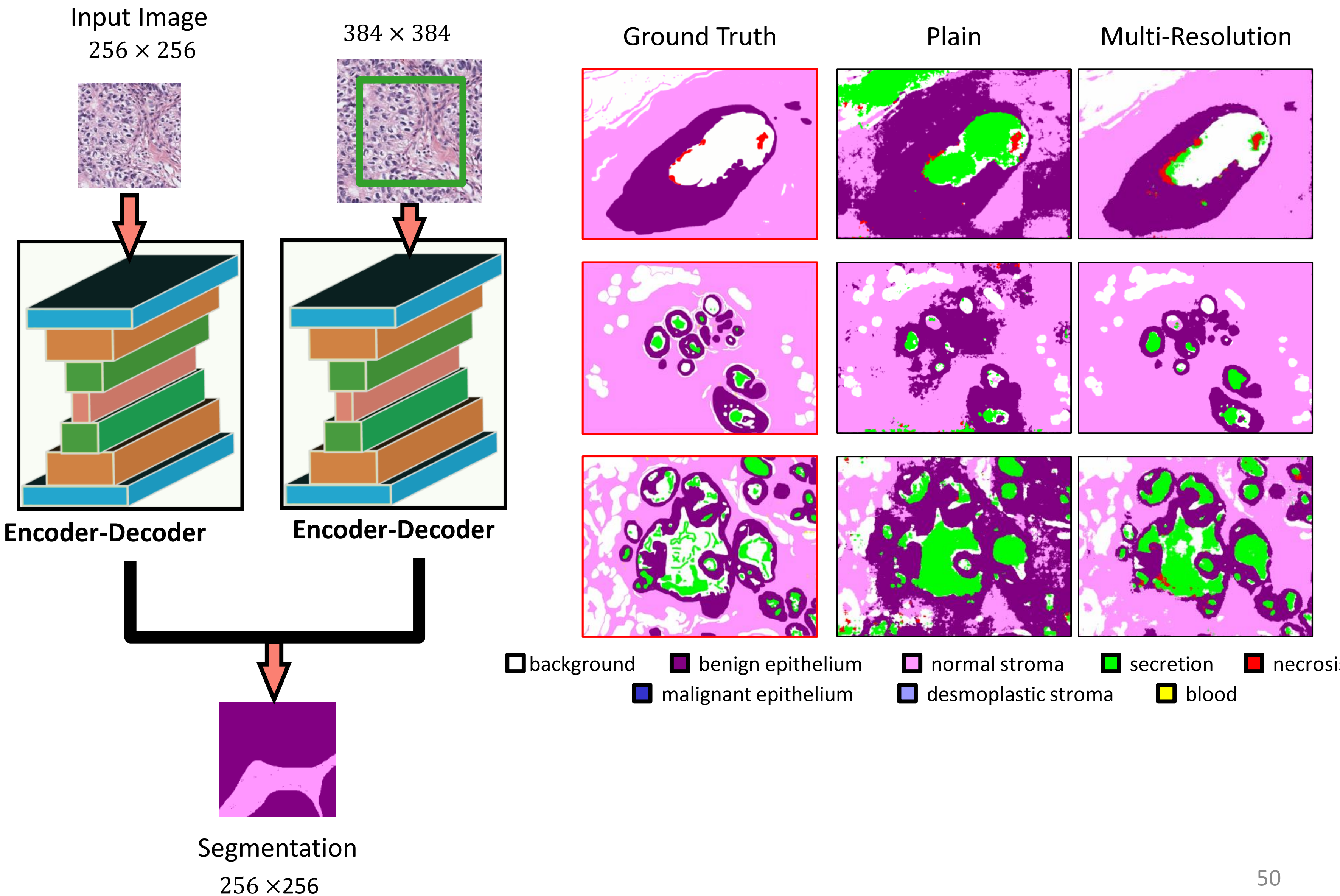
Ground Truth | no neighborhood | 1 neighborhood | 2 neighborhoods



color and texture histograms

background | benign epithelium | normal stroma | secretion | necrosis | malignant epithelium | desmoplastic stroma | blood

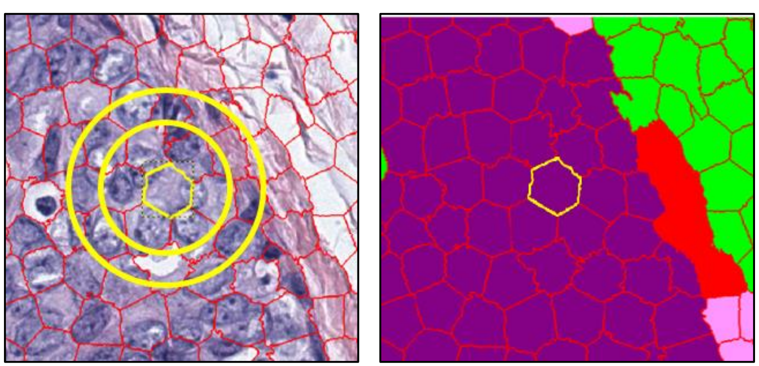
# CNN-based Segmentation



# Supervised Tissue Label Segmentation

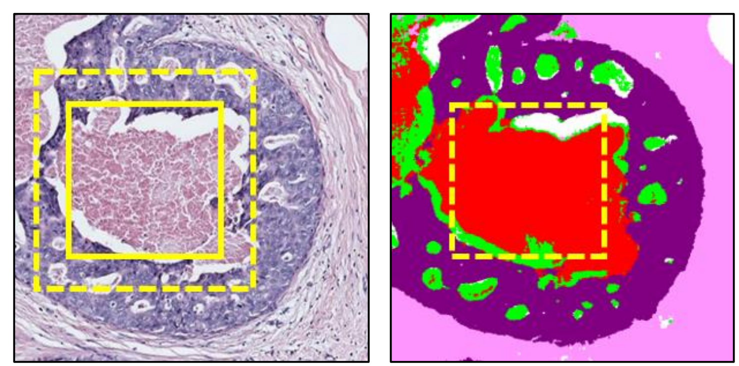
**Superpixel + SVM**

- Each superpixel is assigned a class label.
- Context: Two circular neighborhoods
- Relatively simple model
- Faster to train (~3 hours)



**CNN**

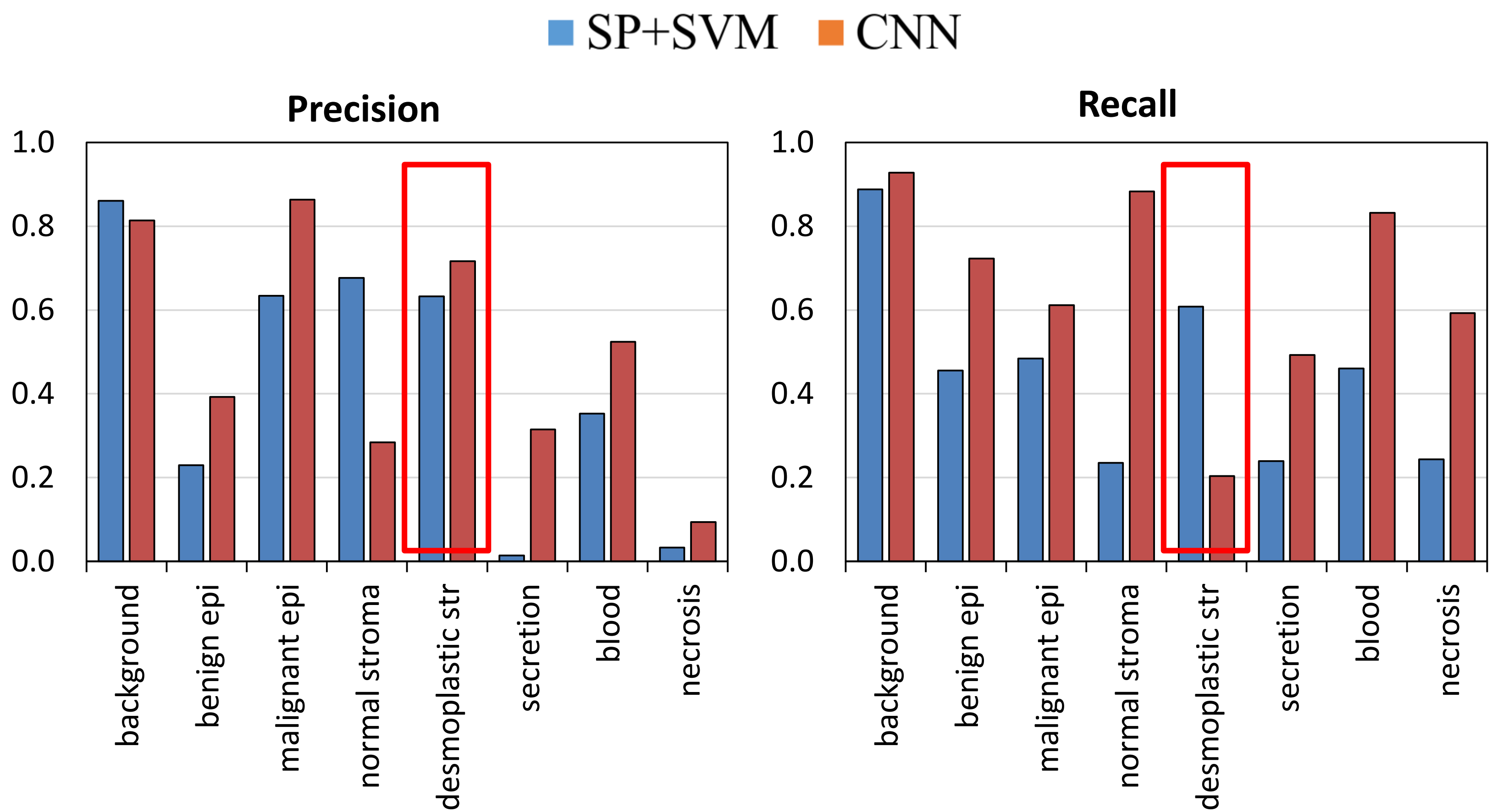
- Each pixel is assigned a class label.
- Context: 256x256 and 384x384 pixel patches
- More complex model
- ~1 week to train on special hardware



# Results

**Mean $F_1$-score**

| | |
|---|---|
| SP+SVM | 0.40 |
| CNN | 0.50 |



# Confusion Matrices

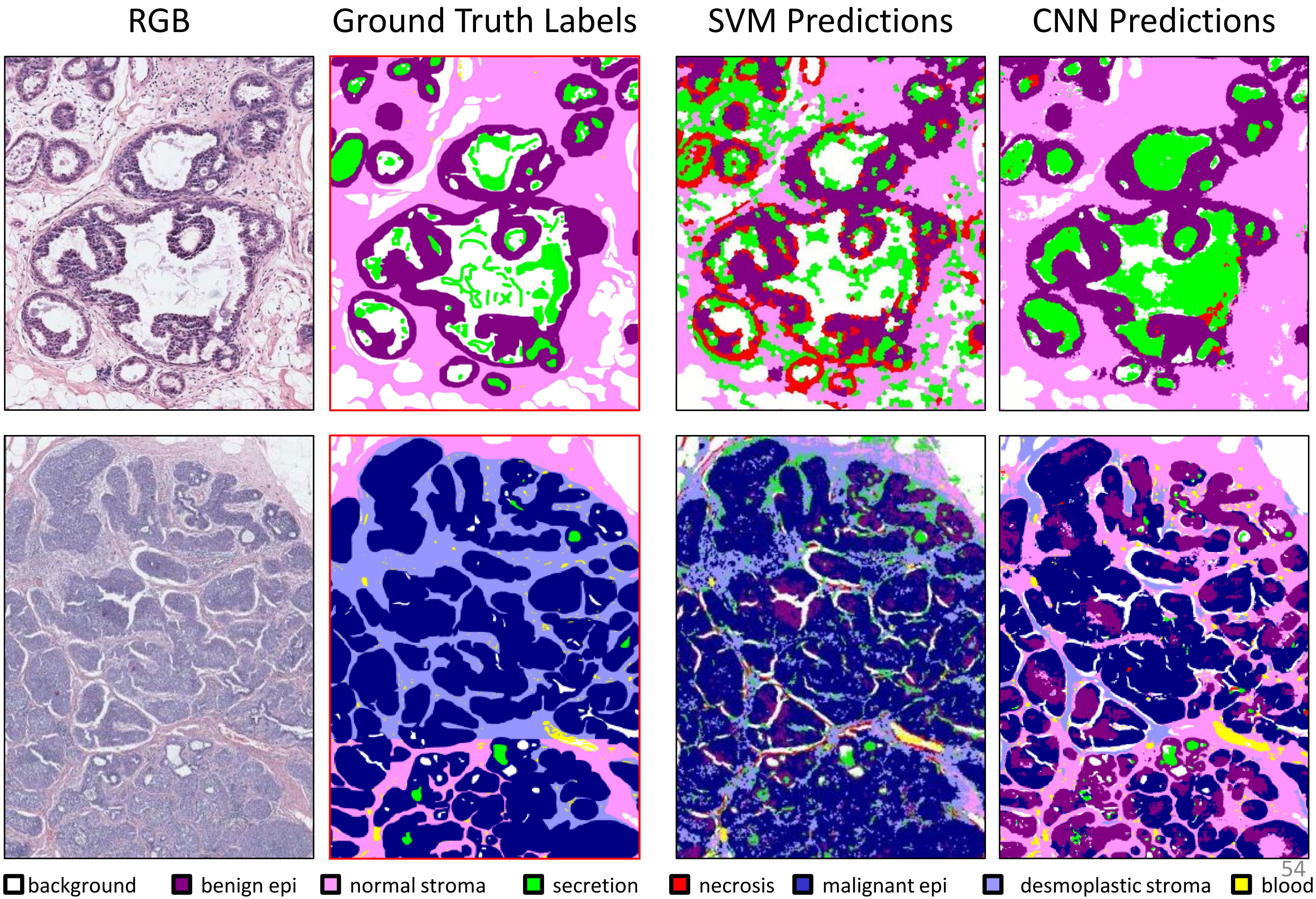
| Ground Truth \ Prediction | background | benign epithelium | malignant epithelium | normal stroma | desmoplastic stroma | secretion | blood | necrosis |
|---|---|---|---|---|---|---|---|---|
| background | .89 | .01 | .01 | .00 | .02 | .07 | .00 | .01 |
| benign epithelium | .02 | .27 | .31 | .03 | .27 | .08 | .00 | .02 |
| malignant epithelium | .02 | .10 | .47 | .00 | .24 | .16 | .00 | .00 |
| normal stroma | .05 | .01 | .06 | .28 | .35 | .19 | .04 | .03 |
| desmoplastic stroma | .04 | .03 | .17 | .03 | .61 | .11 | .00 | .01 |
| secretion | .04 | .02 | .07 | .27 | .14 | .20 | .21 | .05 |
| blood | .03 | .01 | .05 | .13 | .23 | .05 | .46 | .04 |
| necrosis | .12 | .04 | .13 | .00 | .26 | .20 | .01 | .24 |

Superpixels + SVM

| Ground Truth \ Prediction | background | benign epithelium | malignant epithelium | normal stroma | desmoplastic stroma | secretion | blood | necrosis |
|---|---|---|---|---|---|---|---|---|
| background | .93 | .01 | .01 | .03 | .01 | .01 | .00 | .00 |
| benign epithelium | .01 | .72 | .16 | .09 | .01 | .01 | .00 | .00 |
| malignant epithelium | .06 | .09 | .61 | .13 | .07 | .02 | .00 | .03 |
| normal stroma | .03 | .02 | .01 | .88 | .05 | .00 | .01 | .00 |
| desmoplastic stroma | .06 | .02 | .05 | .66 | .20 | .00 | .01 | .00 |
| secretion | .15 | .06 | .05 | .07 | .02 | .49 | .07 | .09 |
| blood | .01 | .01 | .01 | .10 | .04 | .00 | .83 | .00 |
| necrosis | .10 | .01 | .11 | .01 | .01 | .15 | .01 | .59 |

CNN

# Segmentation Results



# Segmentation Summary

- Tissue-label segmentation is a useful abstraction.
- We developed a set of 8 tissue labels and collected pixel-label data from a pathologist on 58 ROIs.
- We trained two models: SVM and CNN
- CNNs performed significantly better than SVMs both quantitatively and qualitatively.