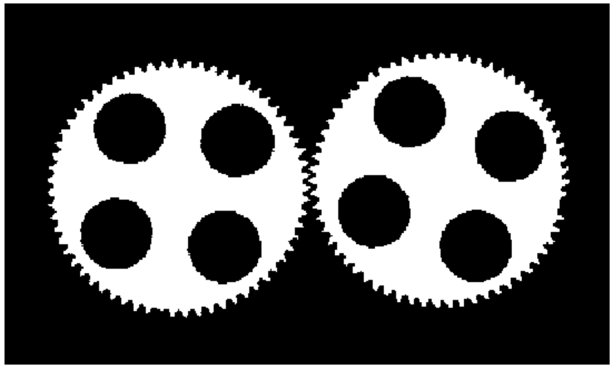
An Introduction to Visual Perception

CPSC 470 – Artificial Intelligence

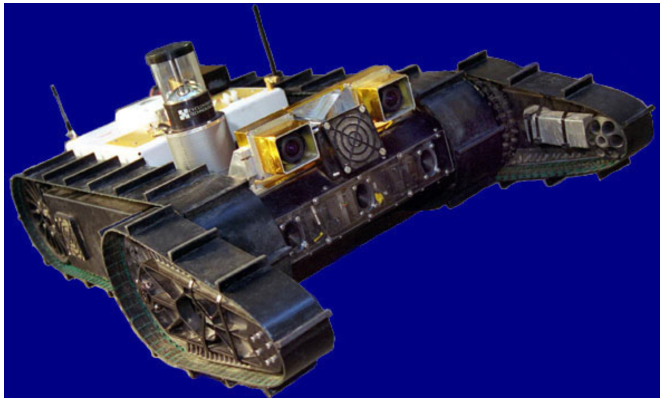
Brian Scassellati

Applications of Machine Vision

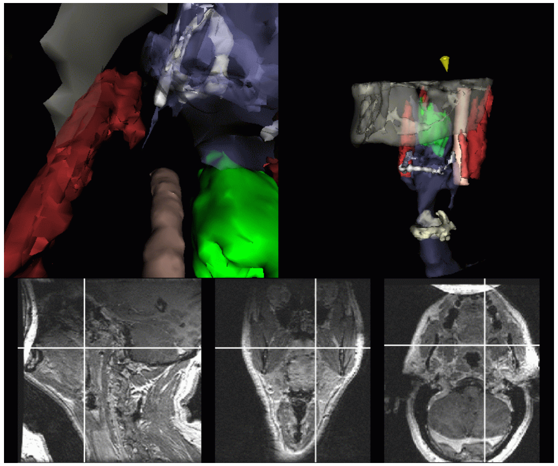
Industrial Inspection



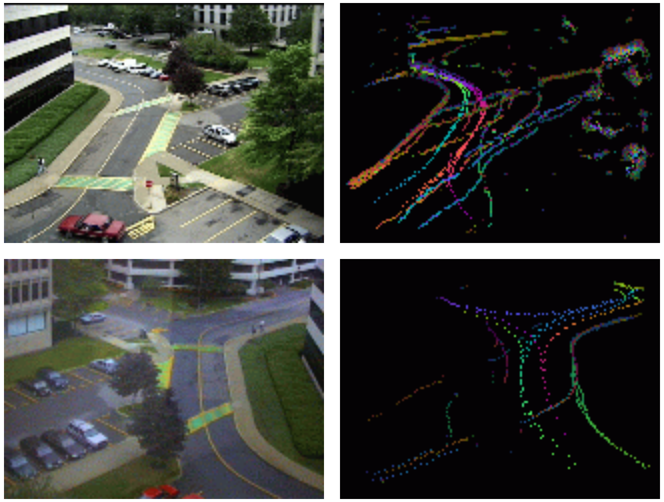
Robotics



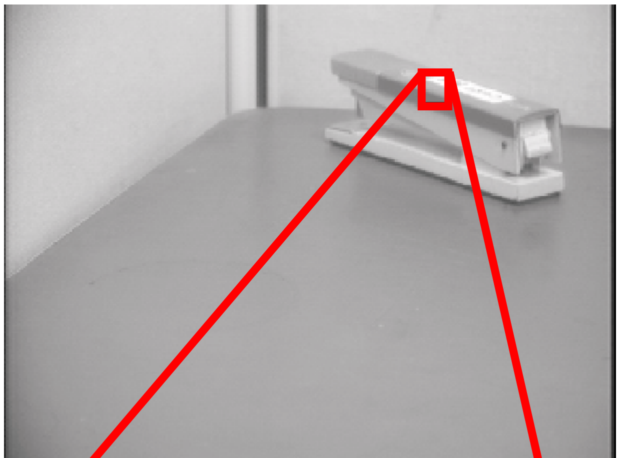
Medical Imaging



Surveillance



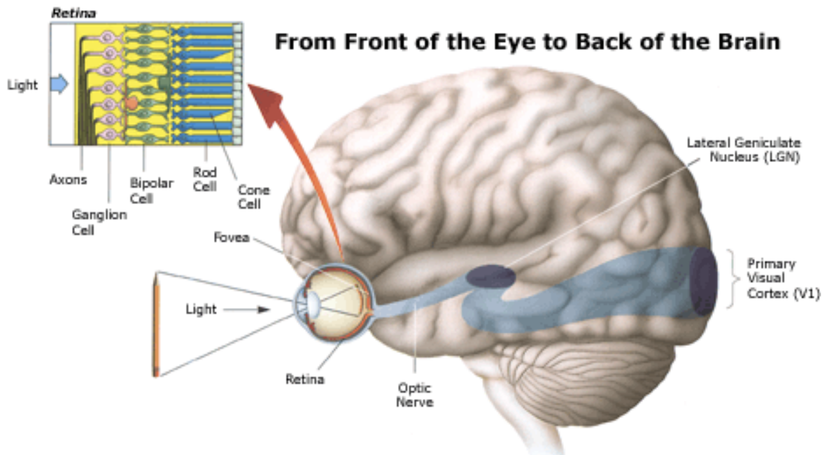
The Central Problem of Vision

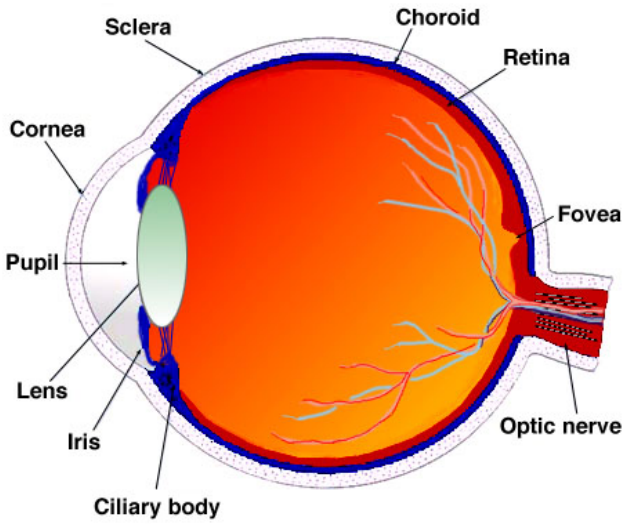


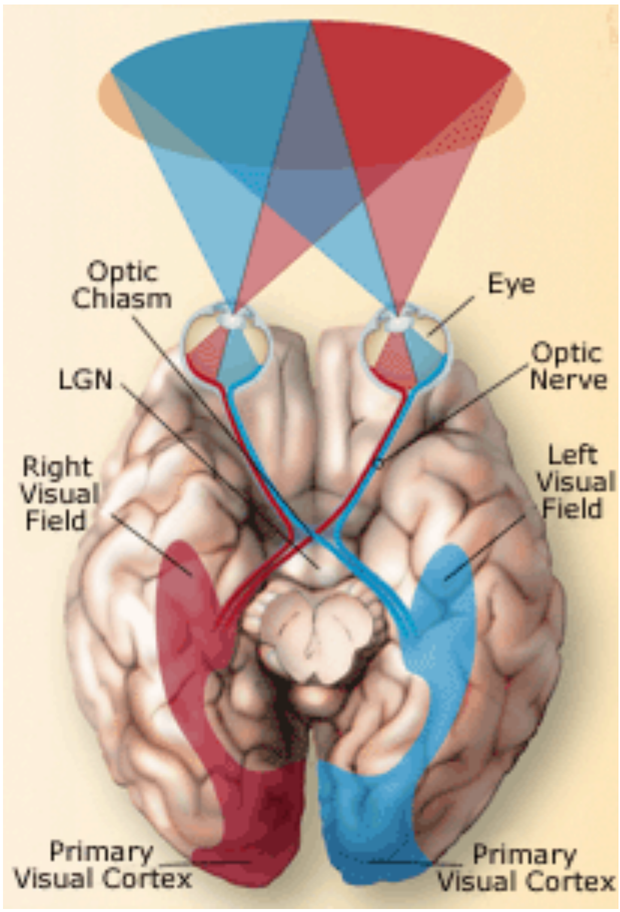
195	209	221	235	249	251	254	255	250	241	247	248
210	236	249	254	255	254	225	226	212	204	236	211
164	172	180	192	241	251	255	255	255	255	235	190
167	164	171	170	179	189	208	244	254	255	251	234
162	167	166	169	169	170	176	185	196	232	249	254
153	157	160	162	169	170	168	169	171	176	185	218
126	135	143	147	156	157	160	166	167	171	168	170
103	107	118	125	133	145	151	156	158	159	163	164
095	095	097	101	115	124	132	142	117	122	124	161
093	093	093	093	095	099	105	118	125	135	143	119
093	093	093	093	093	093	095	097	101	109	119	132
095	093	093	093	093	093	093	093	093	093	093	119

- Visual perception is something that we take for granted
- Extremely hard problems are often present, but difficult to identify
- Perception often does not match our intuitive notions of what we are doing (illusions)

Human Visual System







CCD Cameras

- Small
- Inexpensive (<$100 to $$)
- Common
- Standardized

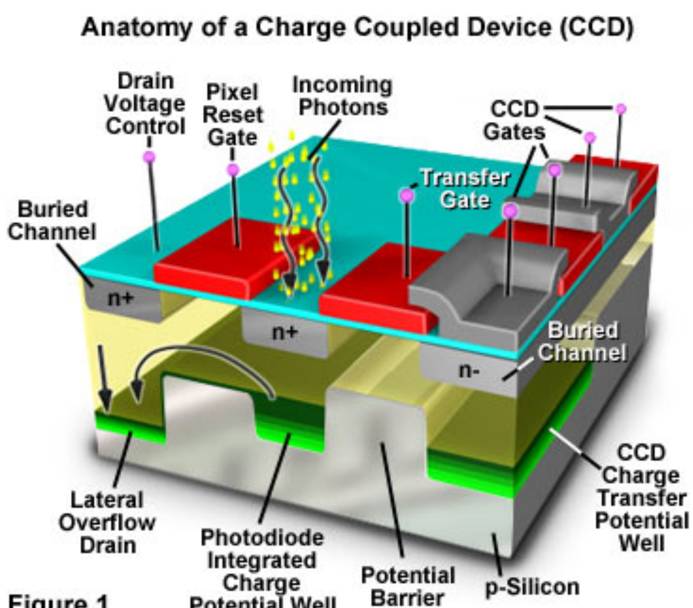


Figure 1

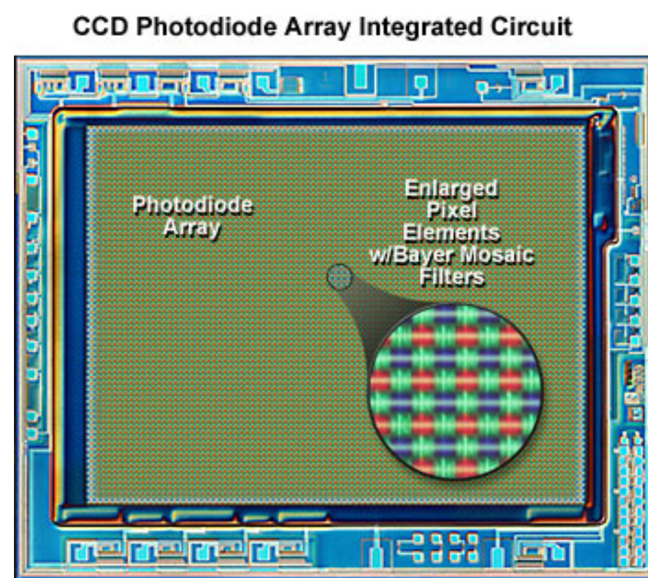
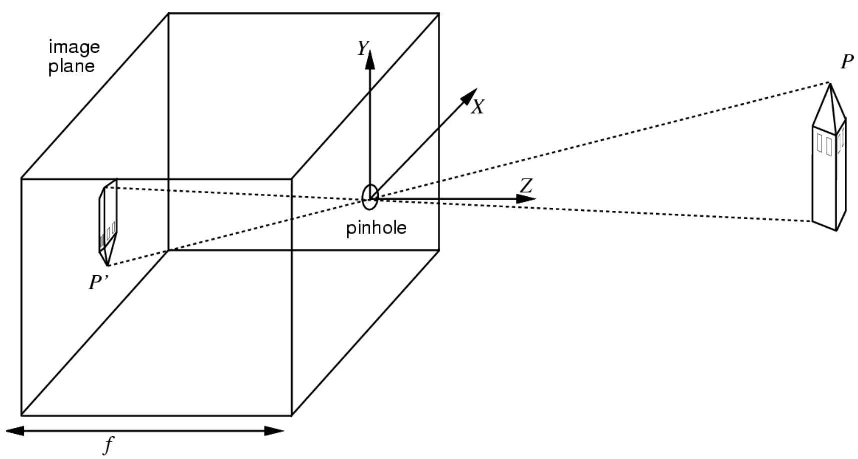


Figure 2

Image Formation: Pinhole Cameras



- If P is a point in the scene with coordinates (X,Y,Z) then the projection on the image plane P' has coordinates (x,y,z)

$$\frac{-x}{f} = \frac{X}{Z} \qquad \frac{-y}{f} = \frac{Y}{Z} \Rightarrow x = \frac{-fX}{Z} \qquad y = \frac{-fY}{Z}$$

Image Formation: Lens Systems

- Most real systems have a lens, which is much wider than a pinhole (and thus admits more light)
- Object at a distance Z from the lens is reproduced at a distance of Z'

$$\frac{1}{Z} + \frac{1}{Z'} = \frac{1}{f}$$

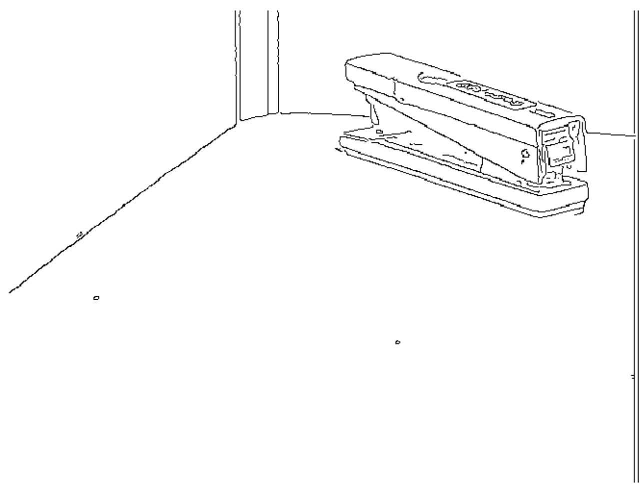
where f is the focal length of the lens

- But because Z >> Z'

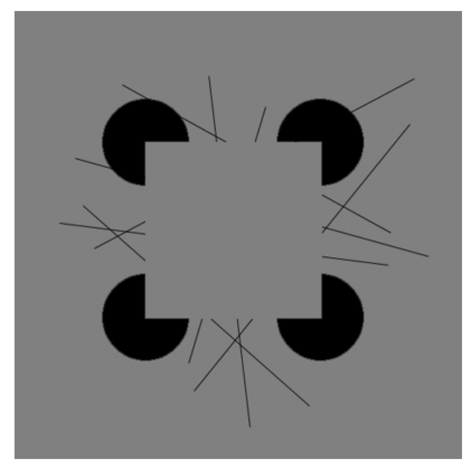
$$\frac{1}{Z} + \frac{1}{Z'} \approx \frac{1}{Z'} \Rightarrow \frac{1}{Z'} \approx \frac{1}{f}$$

- Therefore, we often use the pinhole projection as an approximation of image formation

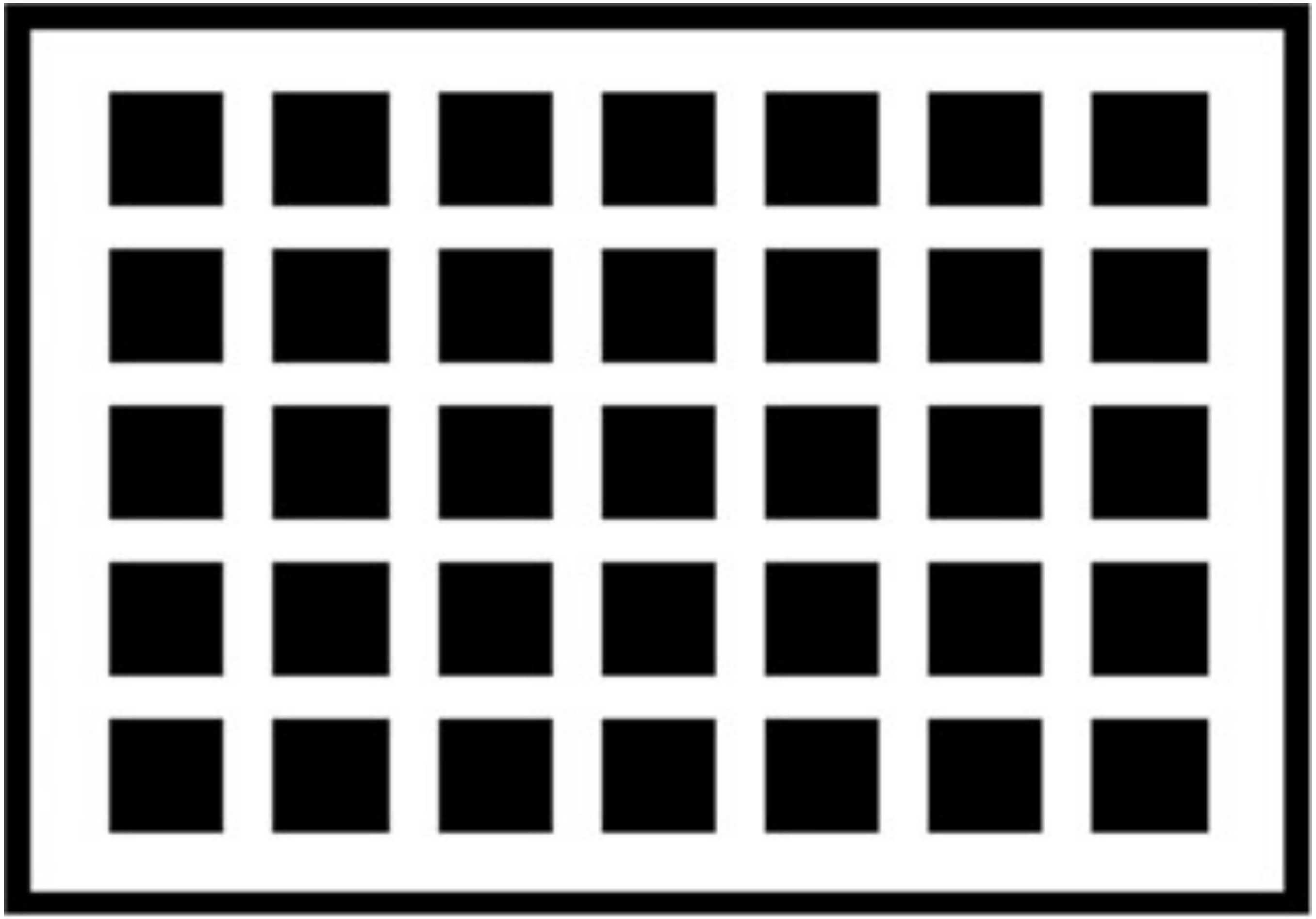
Early Vision Operations: Edge Detection



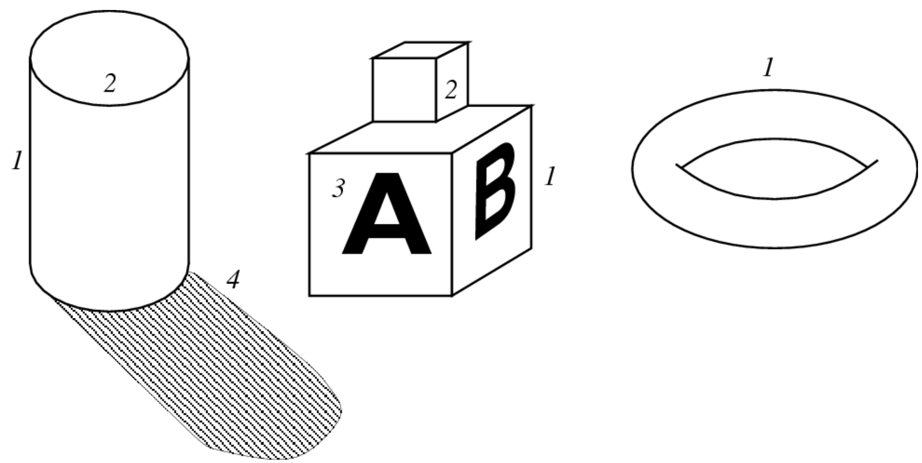
- Want to find the boundaries of objects
- Initially seen as a low-level vision problem
- Real edges can be much more complex



Hermann Grid Illusion



Early Vision Operations: Different Kinds of Edges



1. Depth discontinuity
2. Surface orientation discontinuity
3. Reflectance discontinuity
4. Illumination discontinuity (shadows)

Mathematical Tools: Convolution

- The result of convolving two functions *f* and *g* is another function *h*

$$h(x) = \int_{-\infty}^{+\infty} f(u)g(x-u)du \qquad h(x) = \sum_{u=-\infty}^{+\infty} f(u)g(x-u)$$

- This generalizes to two dimensions

$$h(x,y) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(u,v)g(x-u,y-v)\,du\,dv$$

$$h(x,y) = \sum_{-\infty}^{+\infty}\sum_{-\infty}^{+\infty} f(u,v)g(x-u,y-v)$$

1-D Convolution
Demo Applets

http://www.jhu.edu/~signals/

http://www.cs.brown.edu/exploratories/freeSoftware/catalogs/signal_processing.html

2D Convolution Example

Starting Image

18	64	32	10	9	14	14
10	20	40	60	20	40	10
39	56	24	25	83	20	55
23	57	85	94	39	5	60
23	64	46	83	7	24	73
52	35	31	55	63	35	92
48	56	83	65	93	20	11

$\circledast$

Convolution Kernel

-1	**0**	**+1**
-2	0	+2
-1	0	+1

=

Resulting Image

2D Convolution Example

Starting Image

18	64	32	10	9	14	14
10	20	40	60	20	40	10
39	56	24	25	83	20	55
23	57	85	94	39	5	60
23	64	46	83	7	24	73
52	35	31	55	63	35	92
48	56	83	65	93	20	11

$\circledast$

Convolution Kernel

-1	**0**	**+1**
-2	0	+2
-1	0	+1

=

Resulting Image

	59					

-1*18 + 0*64 + 1*32
-2*10 + 0*20 + 2*40
-1*39 + 0*56 + 1*24 = 59

2D Convolution Example

Starting Image

18	64	32	10	9	14	14
10	20	40	60	20	40	10
39	56	24	25	83	20	55
23	57	85	94	39	5	60
23	64	46	83	7	24	73
52	35	31	55	63	35	92
48	56	83	65	93	20	11

$\circledast$

Convolution Kernel

-1	**0**	**+1**
-2	0	+2
-1	0	+1

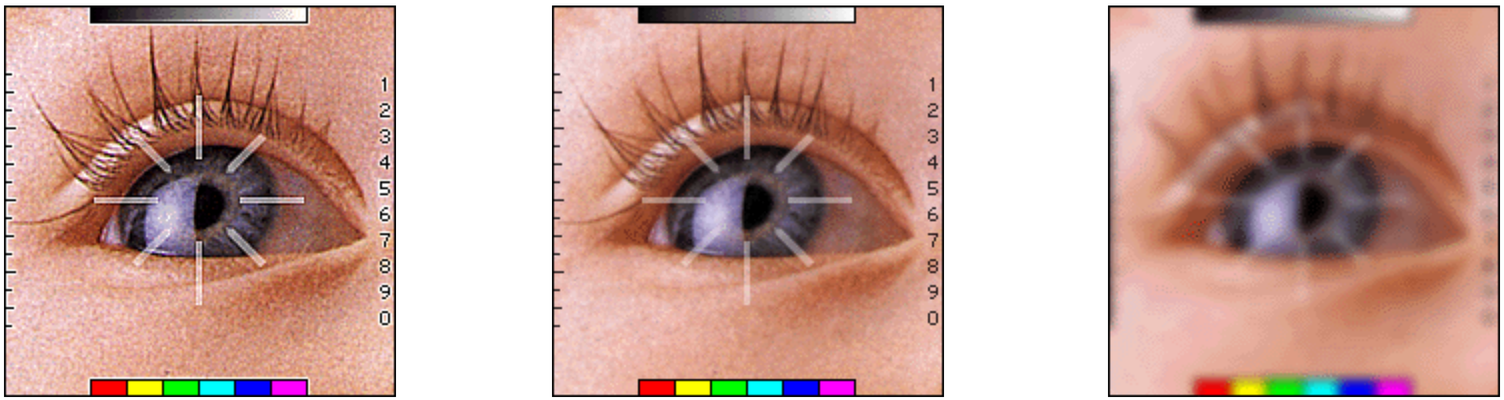
=

Resulting Image

	59	-5				

$$-1*64 + 0*32 + 1*10$$
$$-2*20 + 0*40 + 2*60$$
$$-1*56 + 0*24 + 1*25 =$$

Applications of Convolution: Smoothing



Sharpen Blur

- Convolving an image with a small rectangular or Gaussian filter can sharpen/blur the image

$$\text{Rectangular} = \frac{1}{49}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \qquad \text{Gaussian} = \frac{1}{138}\begin{bmatrix} 1 & 1 & 2 & 2 & 2 & 1 & 1 \\ 1 & 2 & 2 & 4 & 2 & 2 & 1 \\ 2 & 2 & 4 & 8 & 4 & 2 & 2 \\ 2 & 4 & 8 & 16 & 8 & 4 & 2 \\ 2 & 2 & 4 & 8 & 4 & 2 & 2 \\ 1 & 2 & 2 & 4 & 2 & 2 & 1 \\ 1 & 1 & 2 & 2 & 2 & 1 & 1 \end{bmatrix}$$

Simple Edge Detectors

Roberts Cross

- Uses two 2*x*2 convolution kernels

$$Gx = \begin{bmatrix} +1 & 0 \\ 0 & -1 \end{bmatrix} \quad Gy = \begin{bmatrix} 0 & +1 \\ -1 & 0 \end{bmatrix}$$

- Magnitude

$$|G| = \sqrt{Gx^2 + Gy^2}$$

- Direction

$$\theta = \arctan\left(\frac{Gy}{Gx}\right) - \frac{3\pi}{4}$$

Sobel Operator

- Uses two 3x3 convolution kernels

$$Gx = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \quad Gy = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

- Magnitude

$$|G| = \sqrt{Gx^2 + Gy^2}$$

- Direction

$$\theta = \arctan\left(\frac{Gy}{Gx}\right)$$

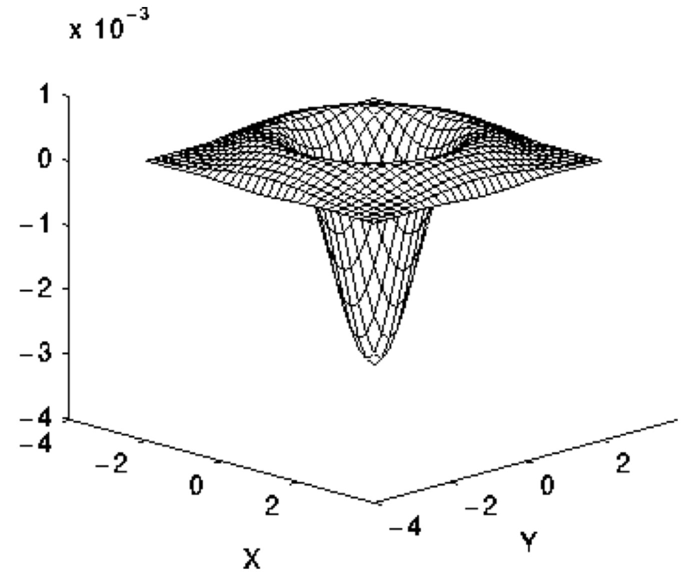
More Complex Edge Detectors

Canny Edge Detector

- Smooth (convolve with a Gaussian)
- Apply a 1st-derivative operator (similar to Roberts Cross).
- Keep all pixels where the gradient is a local maximum along the gradient direction are detected. This produces lines with a width of one pixel.
- Thresholding: Edges are traced starting at height T1 and stopping when the height drops below T2. This way weak edges are included if they are connected to strong edges.

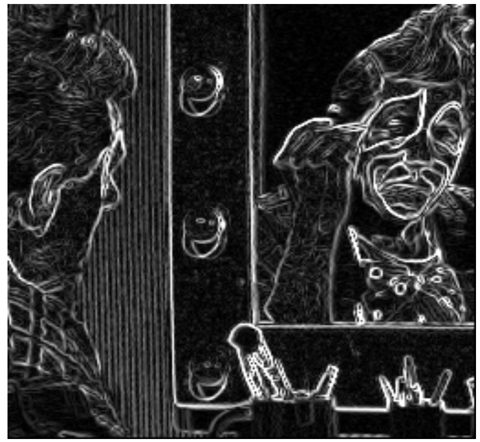
Marr Edge Detector

- a.k.a. Zero crossing detector
- Finds the zero crossings of the second derivative.
- Smooth with a Gaussian
- Find 2nd derivatives with a Laplacian

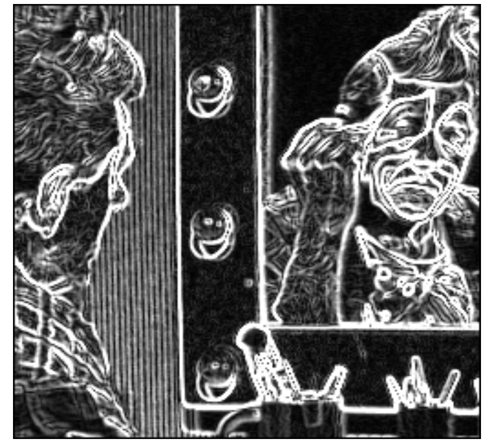


Comparison of Edge Detectors

Original image



Results using Roberts Cross



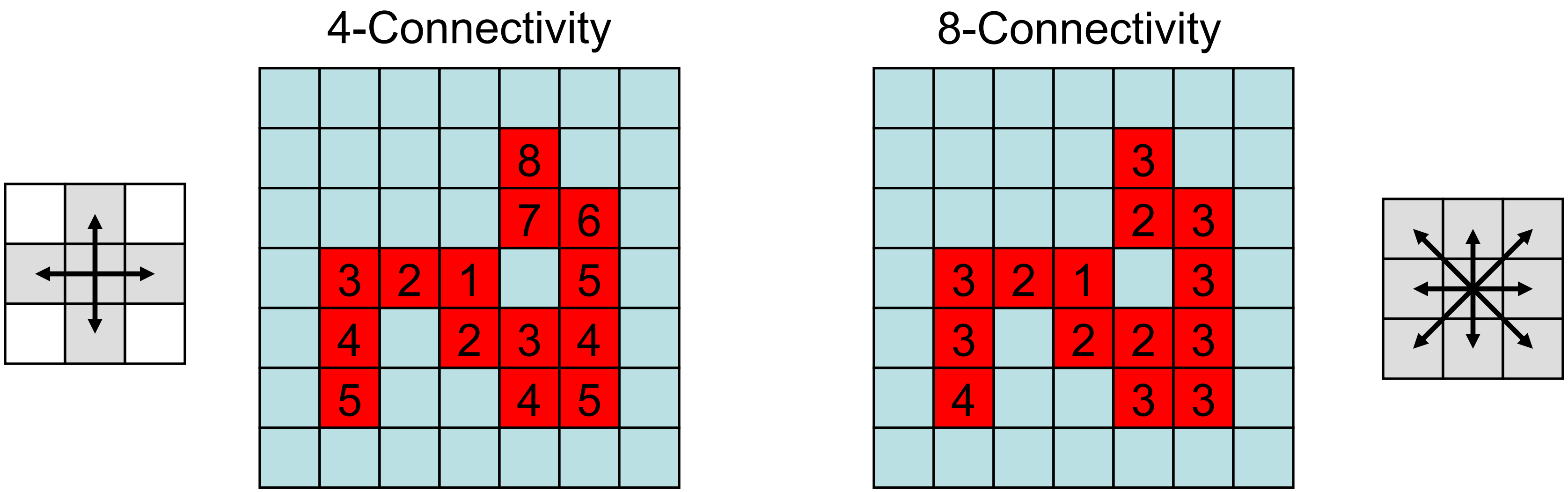
Results using Sobel

Zero crossings with σ = 2.0

Canny with σ = 1.0, T1 = 255, T2 = 1

Canny with σ = 2.0, T1 = 128, T2 = 1

Segmentation via Region Growing



- Region growing techniques start with one pixel of a potential region and try to grow it by adding adjacent pixels until the pixels being compared are too dissimilar.
- The first pixel selected can be just the first unlabeled pixel in the image or a set of seed pixels can be chosen from the image.
  - Usually a statistical test is used to decide which pixels can be added to a region.

Overview: *K*-Means

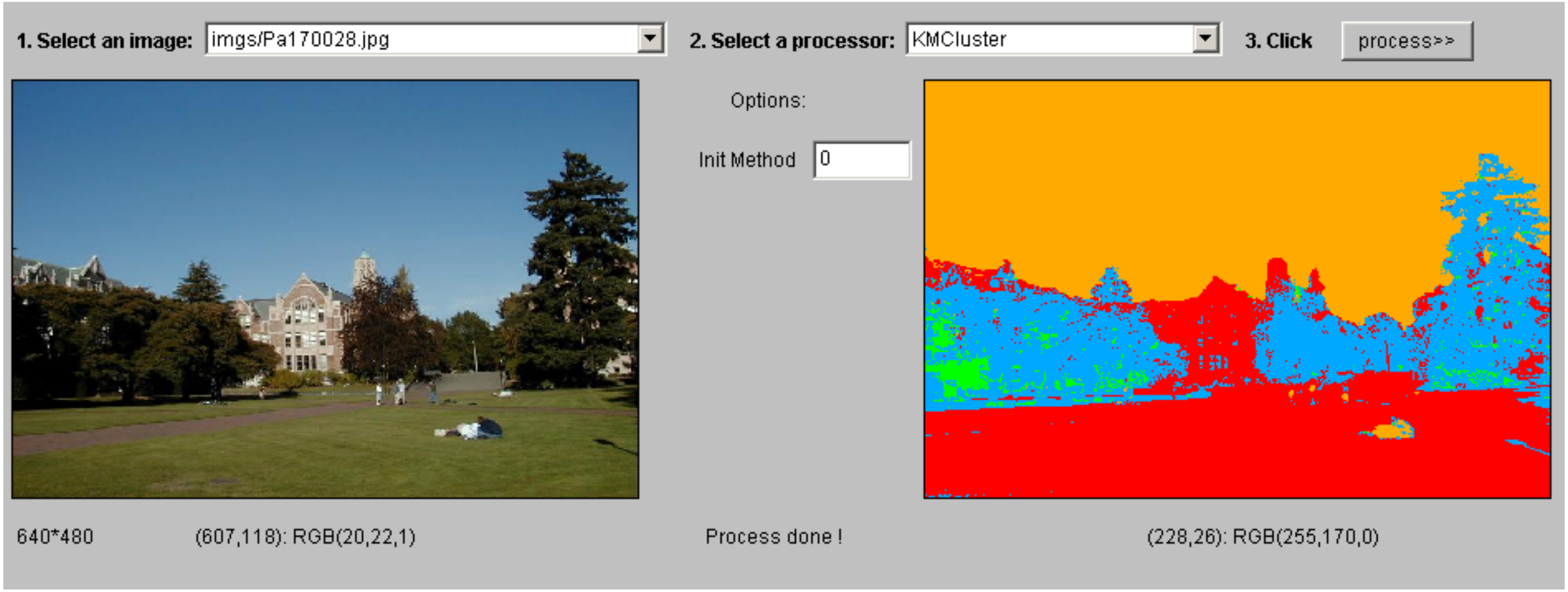
- Clustering is the process of partitioning a group of data points into a small number of clusters.
- In general, we have *n* data points $\mathbf{x}_i$, i=1…n to partition into *k* clusters.
- *K*-means aims to find the positions u_i, i=1…k that minimize the distance from the data points to the cluster, where c_i is the set of points belonging to to cluster *i*

$$\arg\min_{c} \sum_{i=1}^{k} \sum_{\mathrm{x} \in c_i} d(\mathrm{x}, u_i) = \arg\min_{c} \sum_{i=1}^{k} \sum_{\mathrm{x} \in c_i} \left\| \mathrm{x} - u_i \right\|_2^2$$

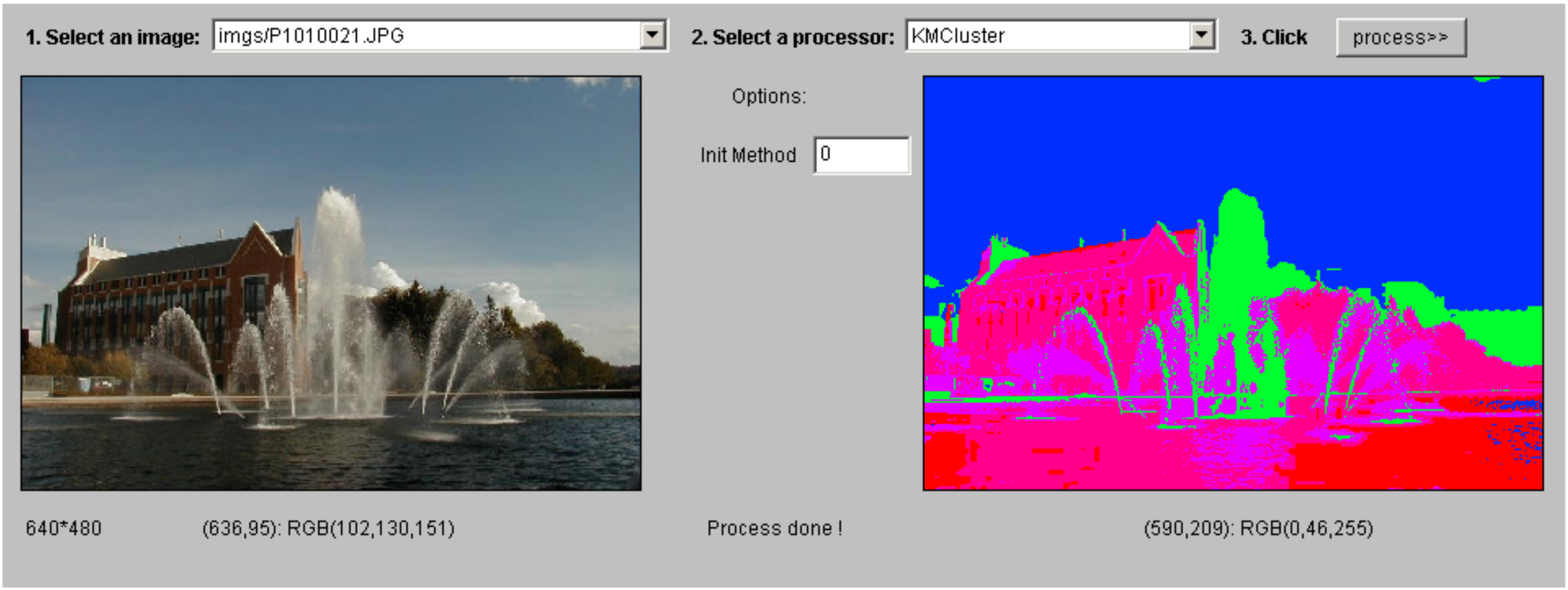
d= *s*quared Euclidian distance

- This is NP hard; *K*-means hopes to find global minimum

K-Means Example 1



K-Means Example 2



Administrivia

- PS 6 due on Wednesday
  - PS 7 – Reinforcement learning
  - PS 8 – Vision and robotics