Computer Vision

Howie Choset
http://www.cs.cmu.edu.edu/~choset

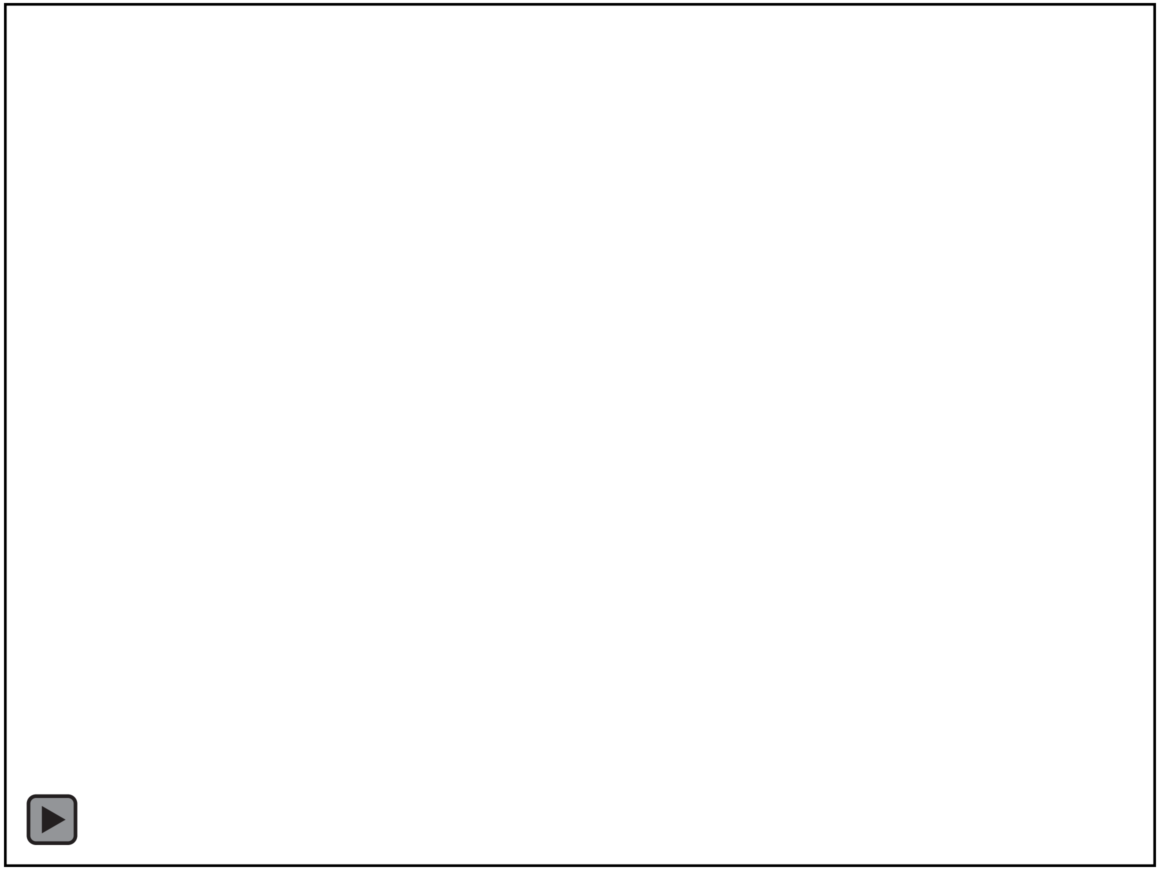
Introduction to Robotics
http://generalrobotics.org

How can we use Computer Vision?

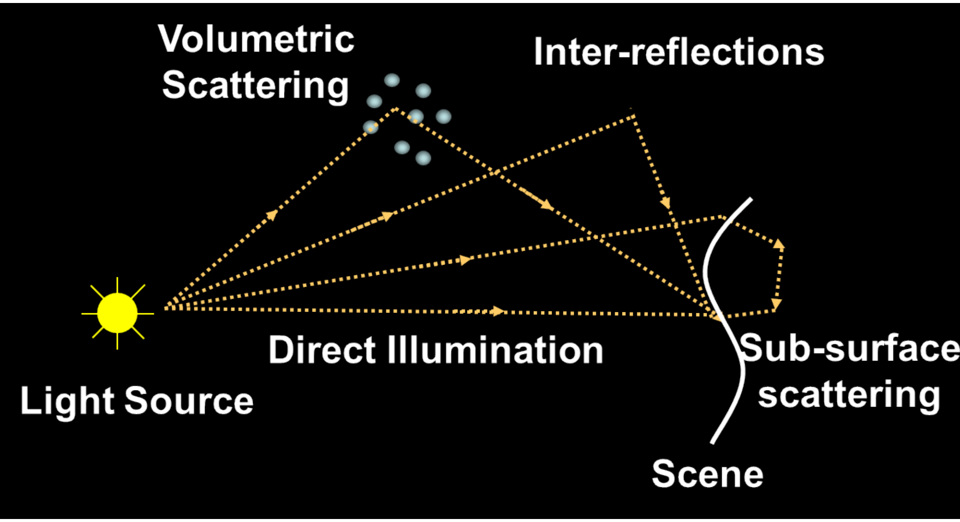
- Understand the world from images!

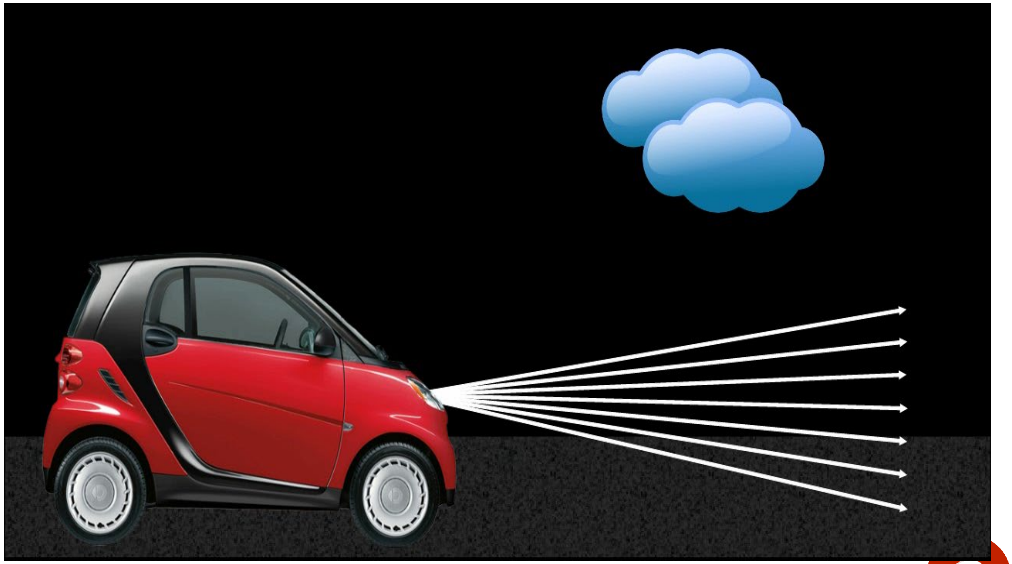
WHO	
WHAT	Facial recognition, object classification, action recognition
WHEN	Object tracking, image labeling, scene reconstruction
WHERE	Scene understanding, image alteration
WHY	

Crowd Tracking



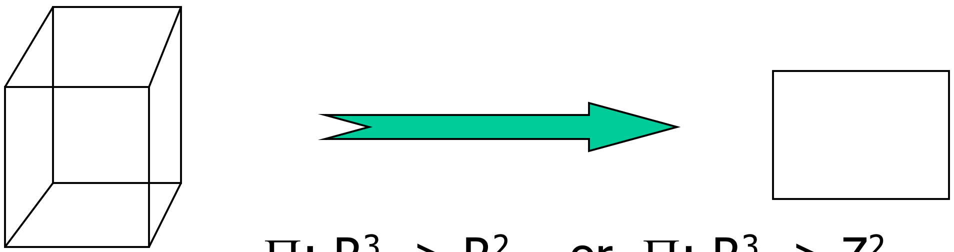
B. Zhou, X. Wang and X. Tang. "Random Field Topic Model for Semantic Region Analysis in Crowded Scenes from Tracklets." in Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2011)





THE ROBOTICS INSTITUTE

Goal of Vision: Recover Projection

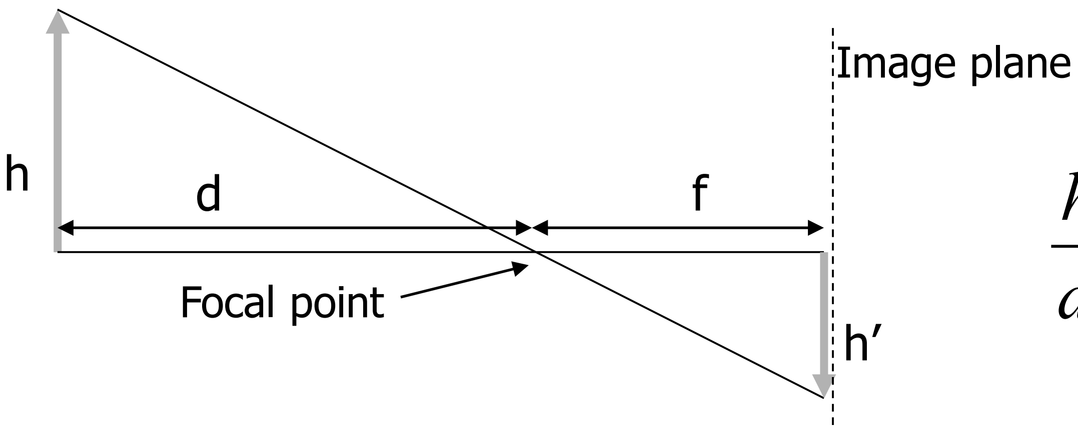


Π: R^3 -> R^2 or Π: R^3 -> Z^2

Recover third dimension or just infer stuff

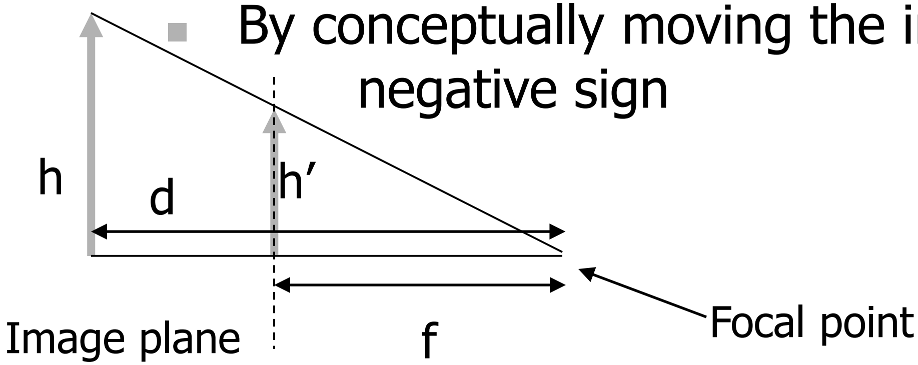
Projection on the image plane

- Size of an image on the image plane is inversely proportional to the distance from the focal point



$$\frac{h}{d} = \frac{-h'}{f}$$

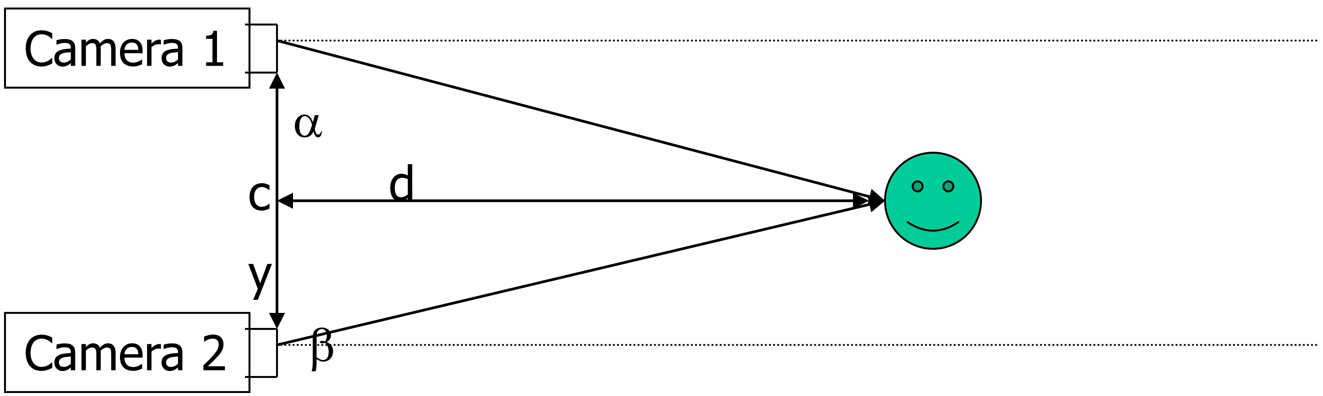
- By conceptually moving the image plane, we can eliminate the negative sign



$$\frac{h}{d} = \frac{h'}{f}$$

Stereo Vision

- Way of calculating depth from two dimensional images using two cameras

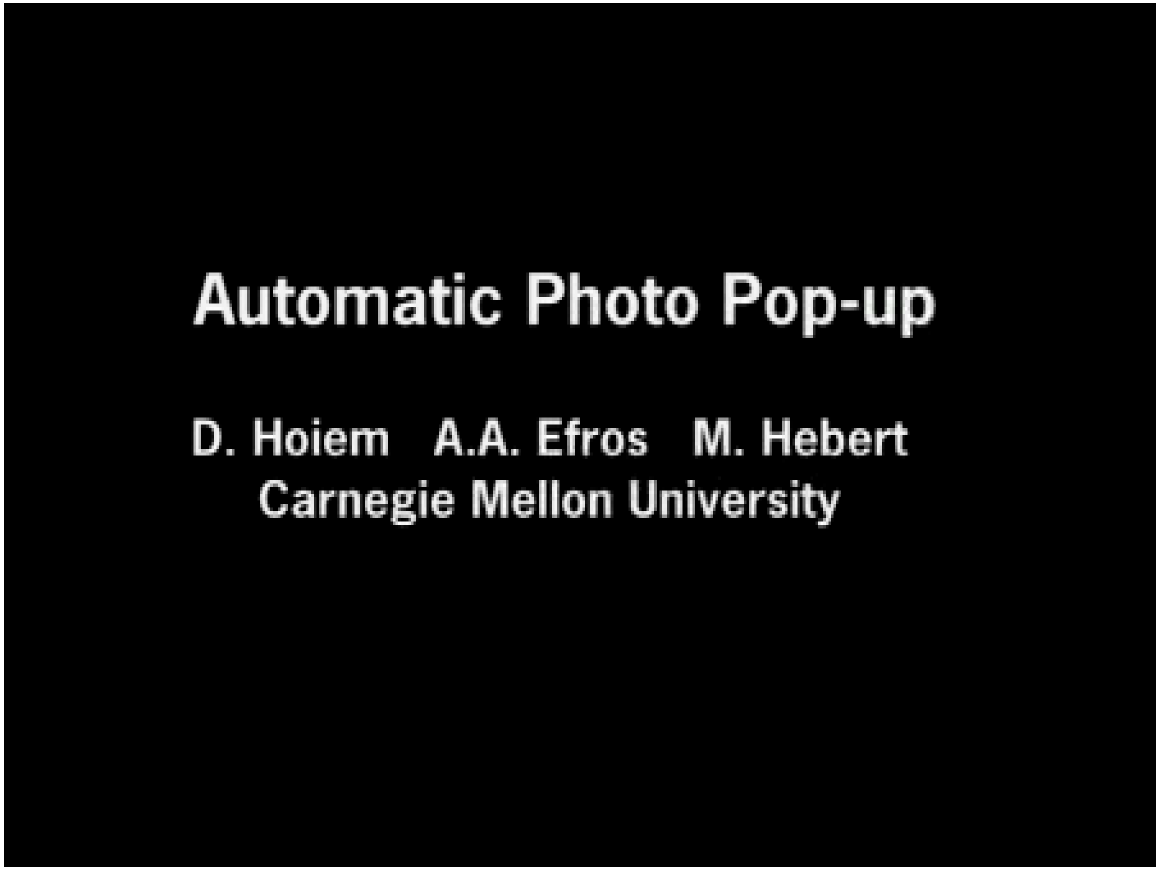


- d and y are unknowns, α and β can be determined processing and c is known

$$\tan\alpha = \frac{d}{c-y} \qquad y = \frac{c\tan\alpha}{\tan\alpha - \tan\beta}$$

$$\tan\beta = \frac{d}{y} \qquad d = y\tan\beta$$

Scene Reconstruction

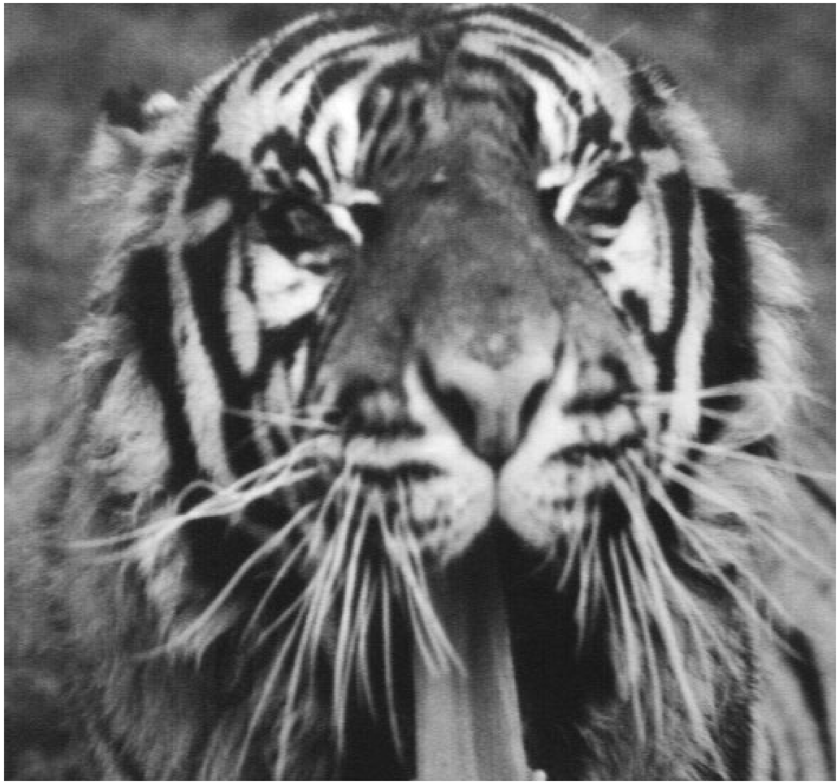


Hoiem, Derek, Alexei A. Efros, and Martial Hebert. "Automatic photo pop-up."*ACM Transactions on Graphics (TOG)*. Vol. 24. No. 3. ACM, 2005.

Scene Understanding

Gupta, Abhinav, et al. "From 3d scene geometry to human workspace."*Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011.

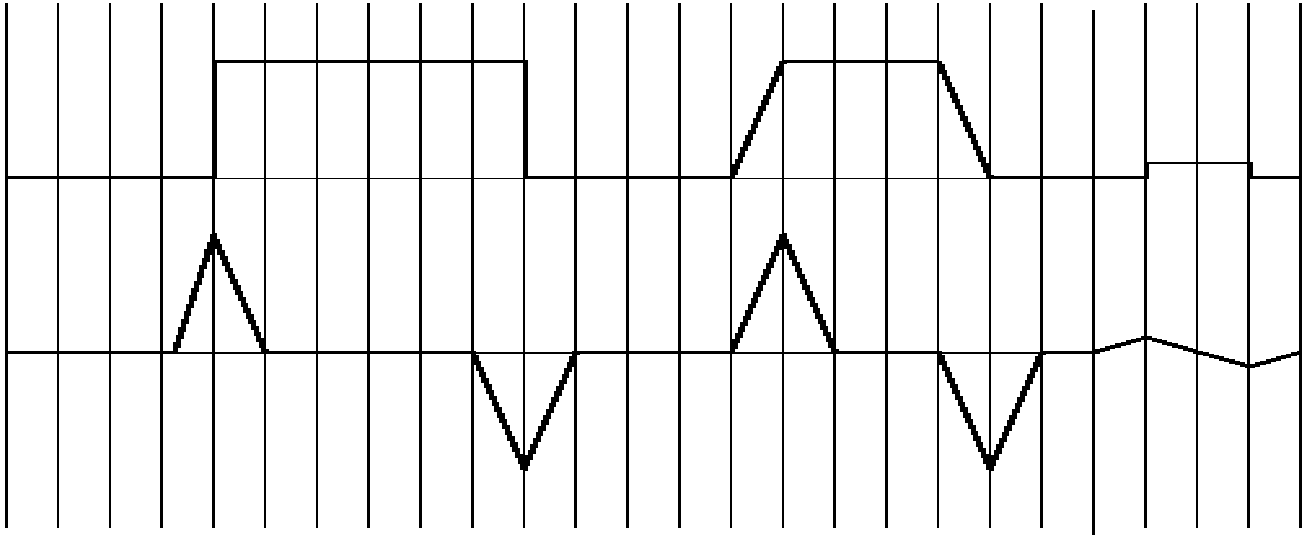
Edge Detection



Edge Detection

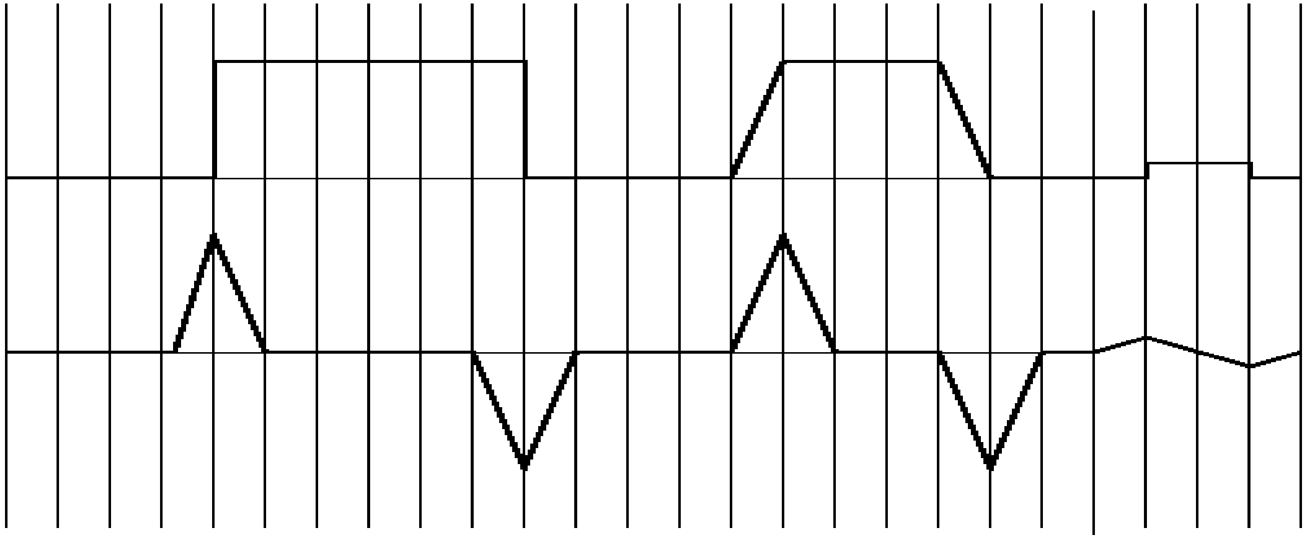
Edge detection

- Scanline: one row of pixels in an image
- Compute B[m+1] – B[m]



Edge detection

- Scanline: one row of pixels in an image
- Take the first derivative of a scanline



- The derivative becomes nonzero when an edge (pixels change values) is encountered

Implementing 1st derivative edge detection digitally

- Derivative is defined as $\lim_{x \to c} \frac{f(x) - f(c)}{x - c}$

Implementing 1st derivative edge detection digitally

- Derivative is defined as $\lim_{x \to c} \frac{f(x) - f(c)}{x - c}$
- With a scan line, the run (x – c) is 1, and the rise (f(x) – f(c)) is B[m+1] – B[m]

Implementing 1st derivative edge detection digitally

- Derivative is defined as $\lim_{x \to c} \frac{f(x) - f(c)}{x - c}$
- With a scan line, the run (x – c) is 1, and the rise (f(x) – f(c)) is B[m+1] – B[m]
- This becomes

$$I[m] = 1 \cdot B[m+1] + -1 B[m]$$

Implementing 1st derivative edge detection digitally

- Derivative is defined as $\lim_{x \to c} \frac{f(x) - f(c)}{x - c}$
- With a scan line, the run (x – c) is 1, and the rise (f(x) – f(c)) is B[m+1] – B[m]
- This becomes

$$I[m] = 1 \cdot B[m+1] + -1 B[m]$$

- This is really just a dot product of the vector [-1 1] repeated each pixel in the resulting image

$$I[m] = \begin{bmatrix} B[m] & B[m+1] \end{bmatrix} \bullet \begin{bmatrix} -1 & 1 \end{bmatrix}$$

Implementing 1st derivative edge detection digitally

- Derivative is defined as $\lim_{x \to c} \frac{f(x) - f(c)}{x - c}$
- With a scan line, the run (x – c) is 1, and the rise (f(x) – f(c)) is B[m+1] – B[m]
- This becomes

$$I[m] = 1 \cdot B[m+1] + -1 B[m]$$

- This is really just a dot product of the vector [-1 1] repeated each pixel in the resulting image

MASK

$$I[m] = \begin{bmatrix} B[m] & B[m+1] \end{bmatrix} \bullet \begin{bmatrix} -1 & 1 \end{bmatrix}$$

Convolution

- This operation of moving a mask across an image has a name, called convolution

- In order to mathematically apply a filter to a signal, we must use convolution
  - If you know Laplace transforms, this is a multiplication in the Laplace domain

Convolution: Analog

$$y(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau$$

Convolution: Analog

$$y(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau$$

Given a symmetric h (common in image processing), simplifies to

$$y(t) = \int_{-\infty}^{\infty} x(\tau)h(\tau)d\tau$$

Convolution: Analog

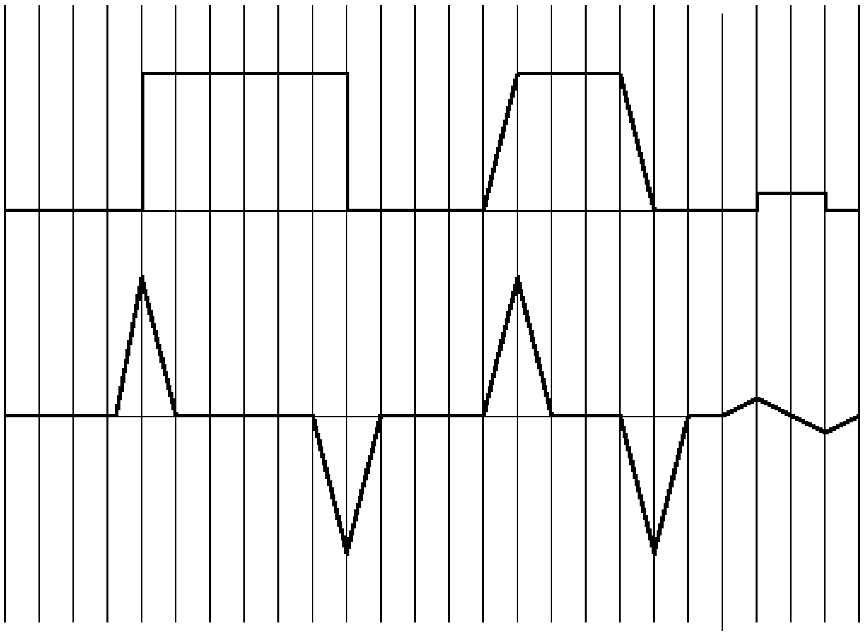
$$y(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau$$

Given a symmetric h (common in image processing), simplifies to

$$y(t) = \int_{-\infty}^{\infty} x(\tau)h(\tau)d\tau$$

h(t) = [-1 1]

Move across the signal x (possibly a scanline in an image)

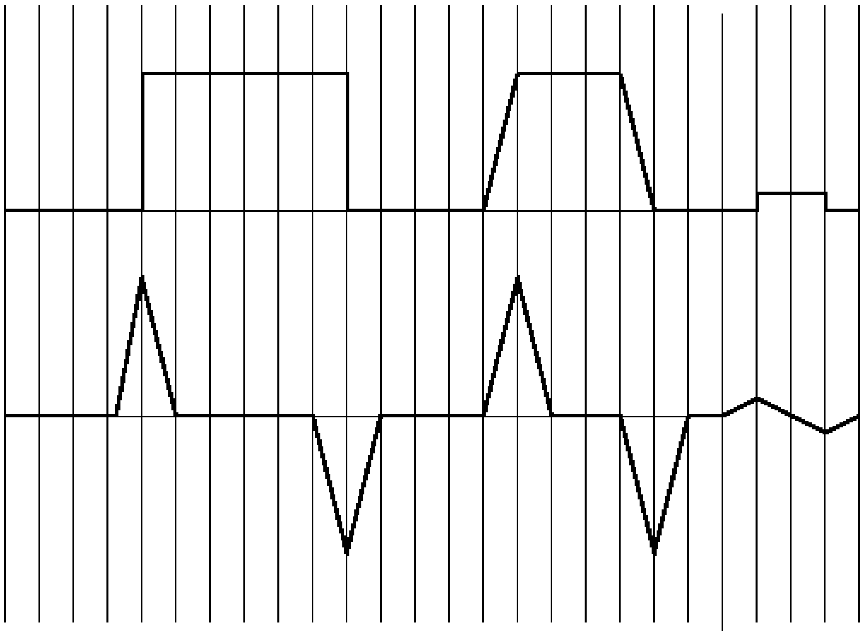


Convolution: Digital

$$y[n] = \sum_{k=-\infty}^{\infty} x[n+k]h[k]$$

h(t) = [-1 1]

Move across the signal x (possibly a scanline in an image)



Convolution: Digital

$$y[n] = \sum_{k=-\infty}^{\infty} x[n+k]h[k]$$

More useful in image processing on a digital computer

x[n] is a pixel in an image, y[n] is the resulting pixel

0	2	2	0	1	1	3

0	1	1

1.

0	1	1

0	2	2	0	1	1	3

4				

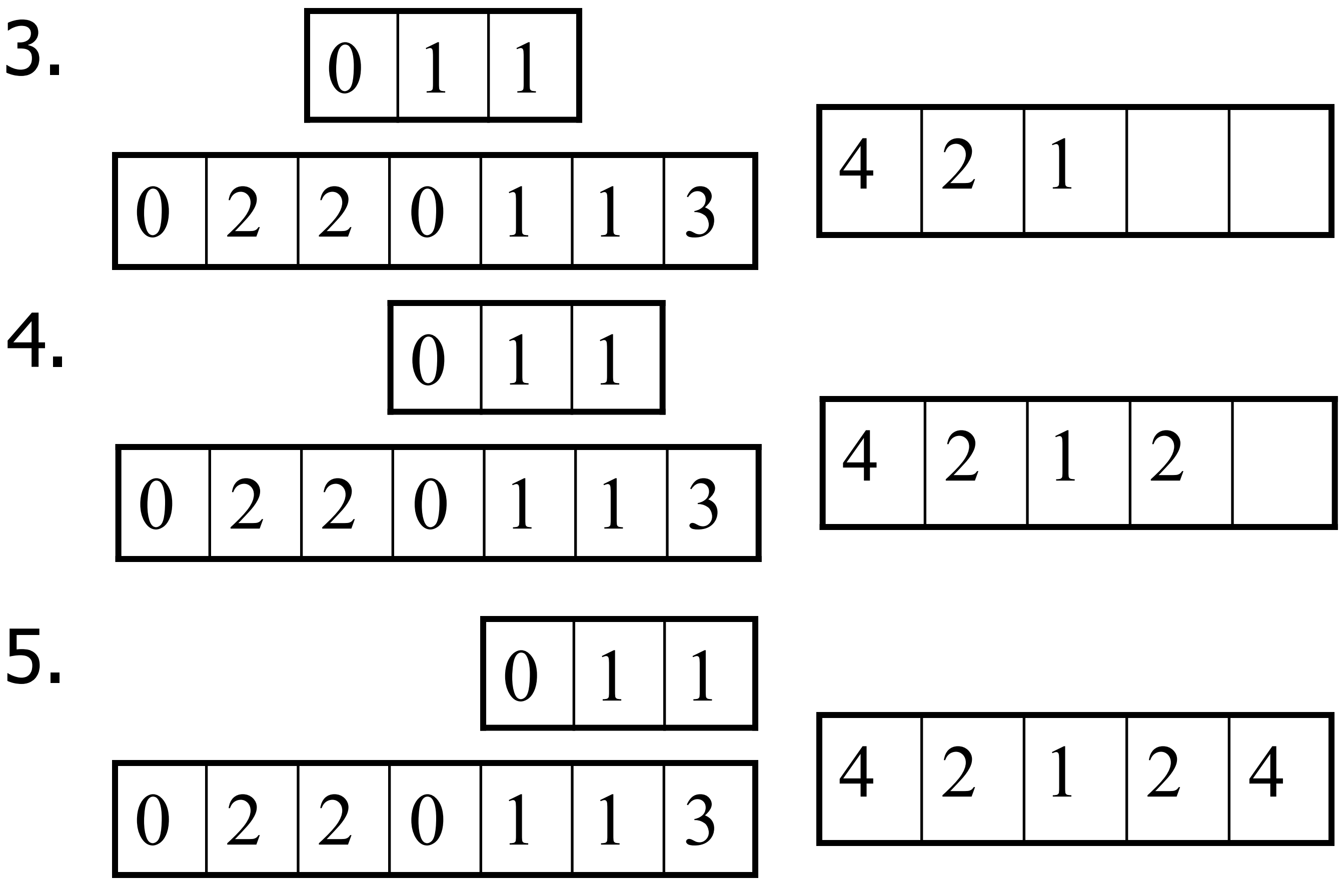
2.

	0	1	1

0	2	2	0	1	1	3

4	2			

Convolution example, cont



Second Derivative

0	-1	1	*	0	-1	1	=	-1	2	-1

Convolution: Two-dimensional

$$y(m,n) = \sum_{m_0} \sum_{n_0} x(m_0 + m, n_0 + n) h(m_0, n_0)$$

- Rotate your mask 180 degrees about the origin (if you were doing "correct" convolution, but since we are doing the "other" convolution, you can skip this step.
- Do the same dot product operation, this time using matrices instead of vectors
- Repeat the dot product for every pixel in the resulting image
- In the boundary case around the edges of the image there are two options
  - extend the original image out using the pixel values at the edge
  - Make the resulting image y smaller than the original and don't compute pixels where the mask would extend beyond the edge of the original

Convolution: Old and New

- Analog

$$y(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau \qquad y(t) = \int_{-\infty}^{\infty} x(\tau+t)h(\tau)d\tau$$

- Digital

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k] \qquad y[n] = \sum_{k=-\infty}^{\infty} x[n+k]h[k]$$

- Two dimensional, digital

$$y(m,n) = \sum_{m_0}\sum_{n_0} x(m_0,n_0)h(m-m_0,n-n_0)$$

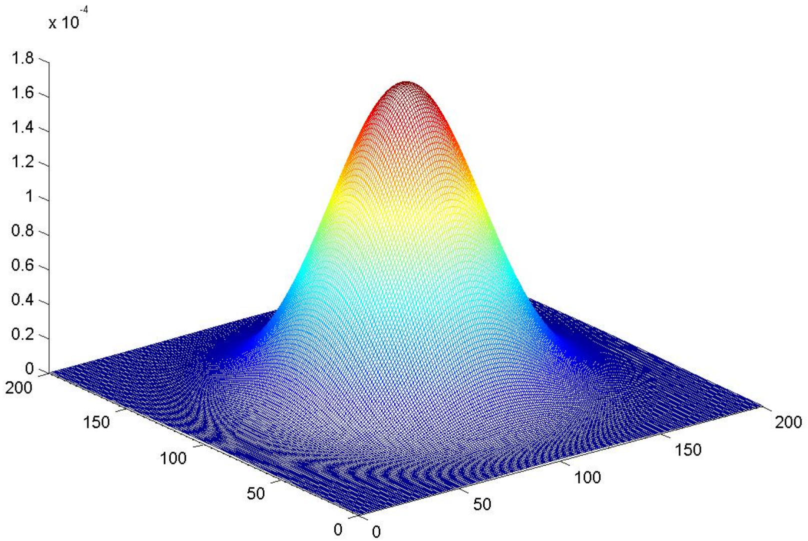
$$y(m,n) = \sum_{m_0}\sum_{n_0} x(m+m_0,n+n_0)h(m_0,n_0)$$

Filters, Masks, Transforms

- Smoothing / Averaging / Blurring
- Edge detection
  - Wide masks
- Object detection

Gaussian Masks

- Used to smooth images and for noise reduction
- Use before edge detection to avoid spurious edges

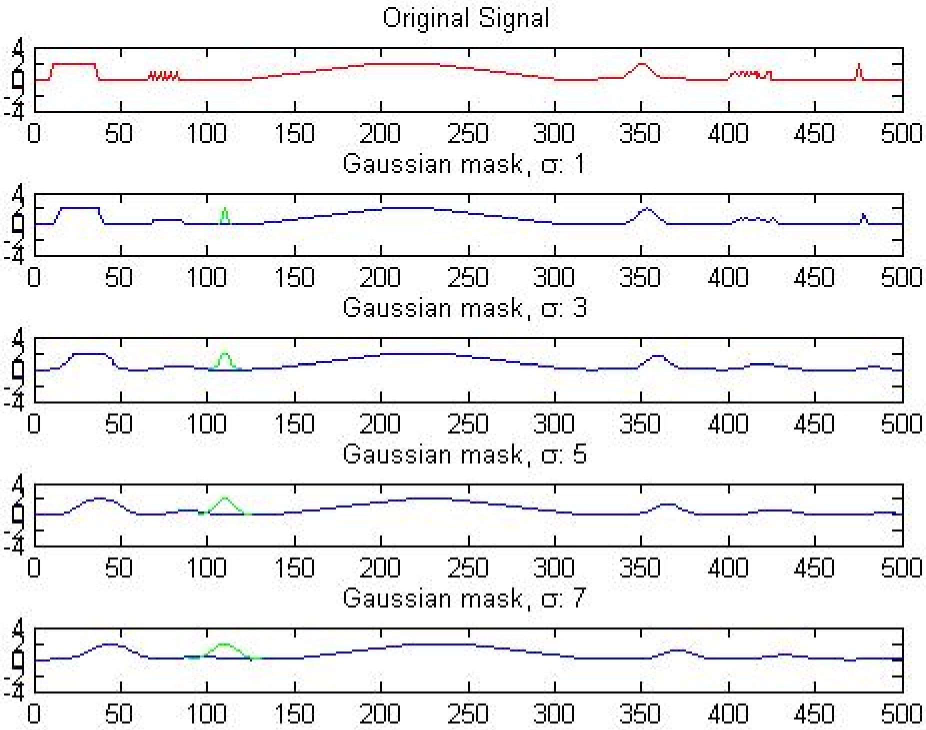


Johann Carl Friedrich Gauss

April 30, 1777 – Feb 23, 1855
number theory, statistics, analysis, differential geometry, geodesy, electrostatics, astronomy, and optics.

Wide Masks

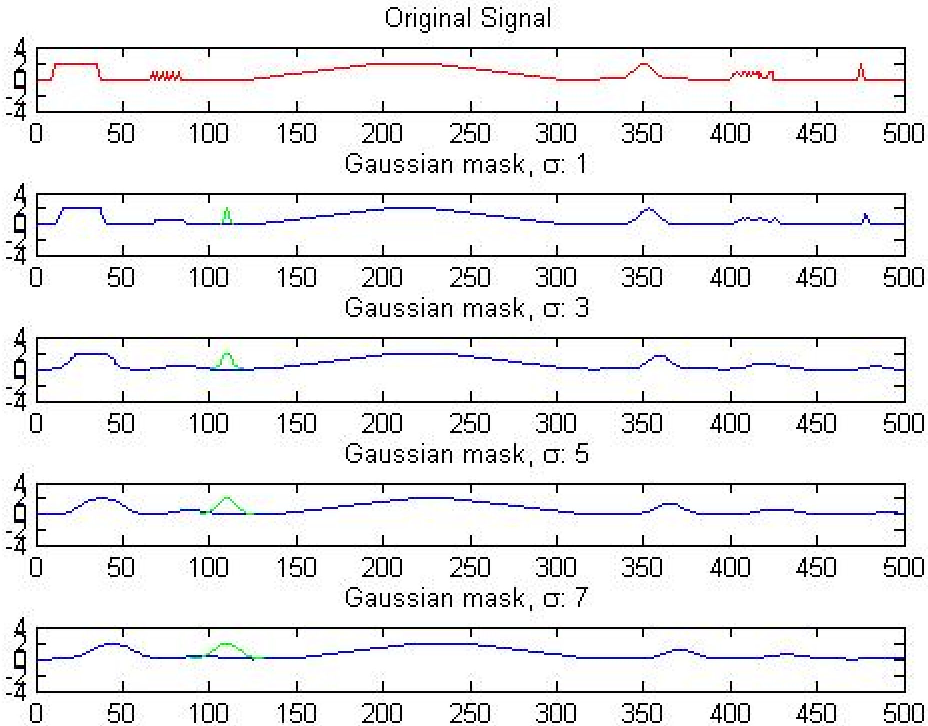
- Wider masks lead to uncertainty about location of edge
- Can detect more gradual edges



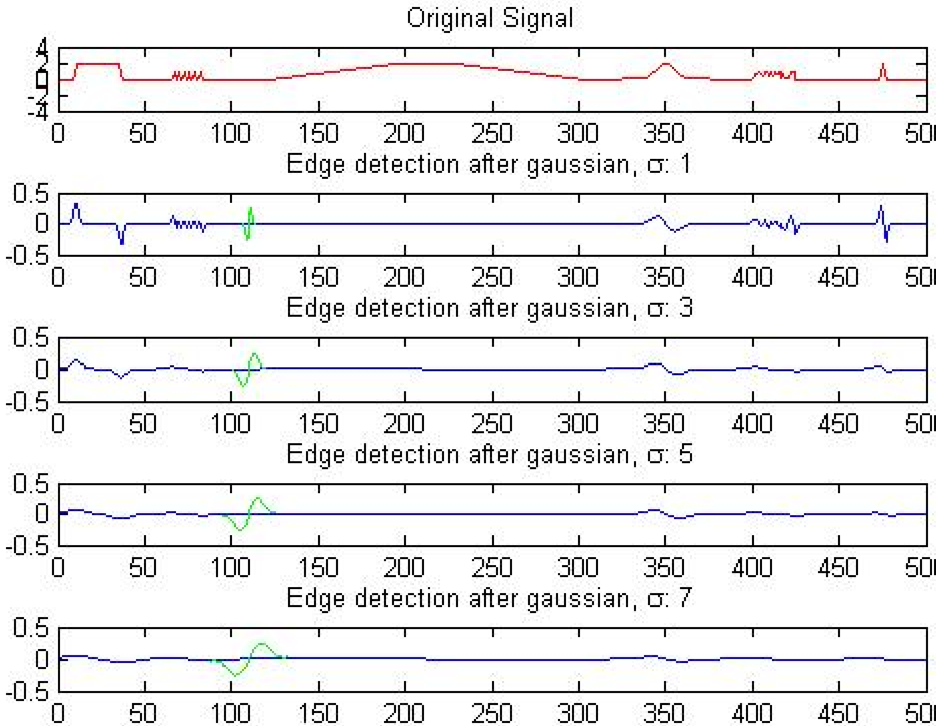
Convolution with Gaussian distribution

Wide Masks

- Wider masks lead to uncertainty about location of edge
- Can detect more gradual edges

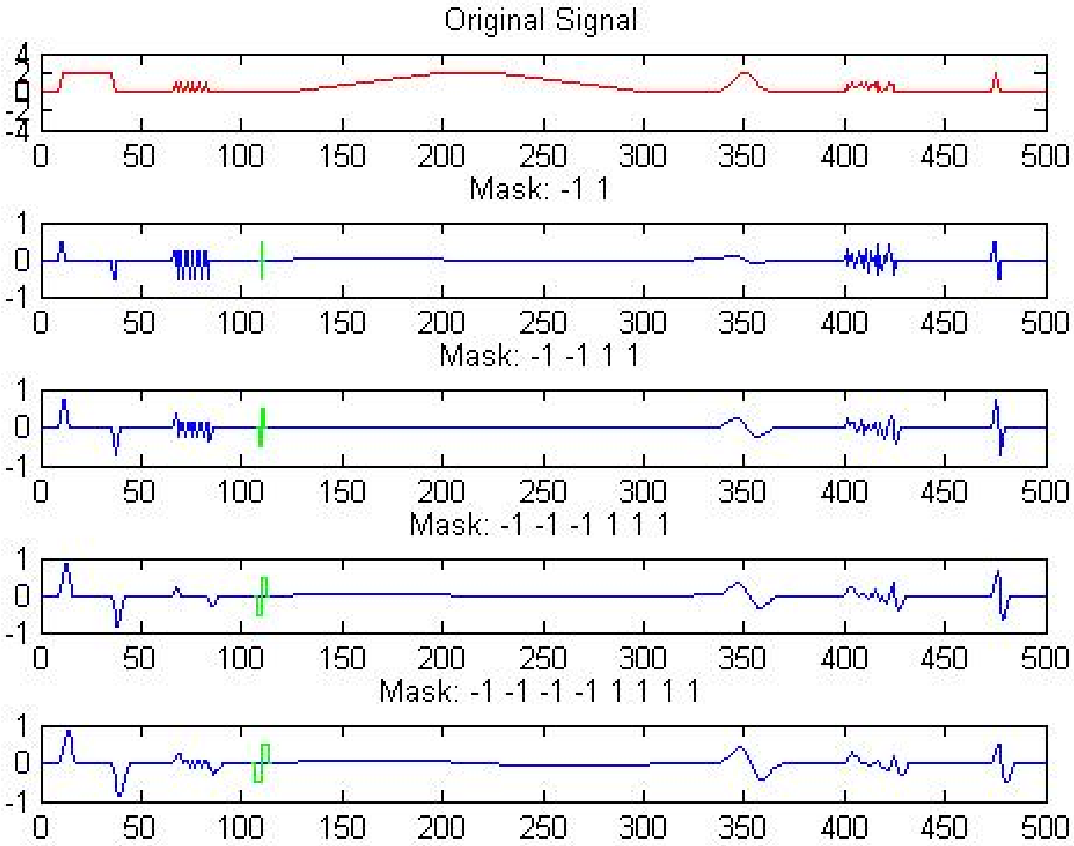


Convolution with Gaussian distribution

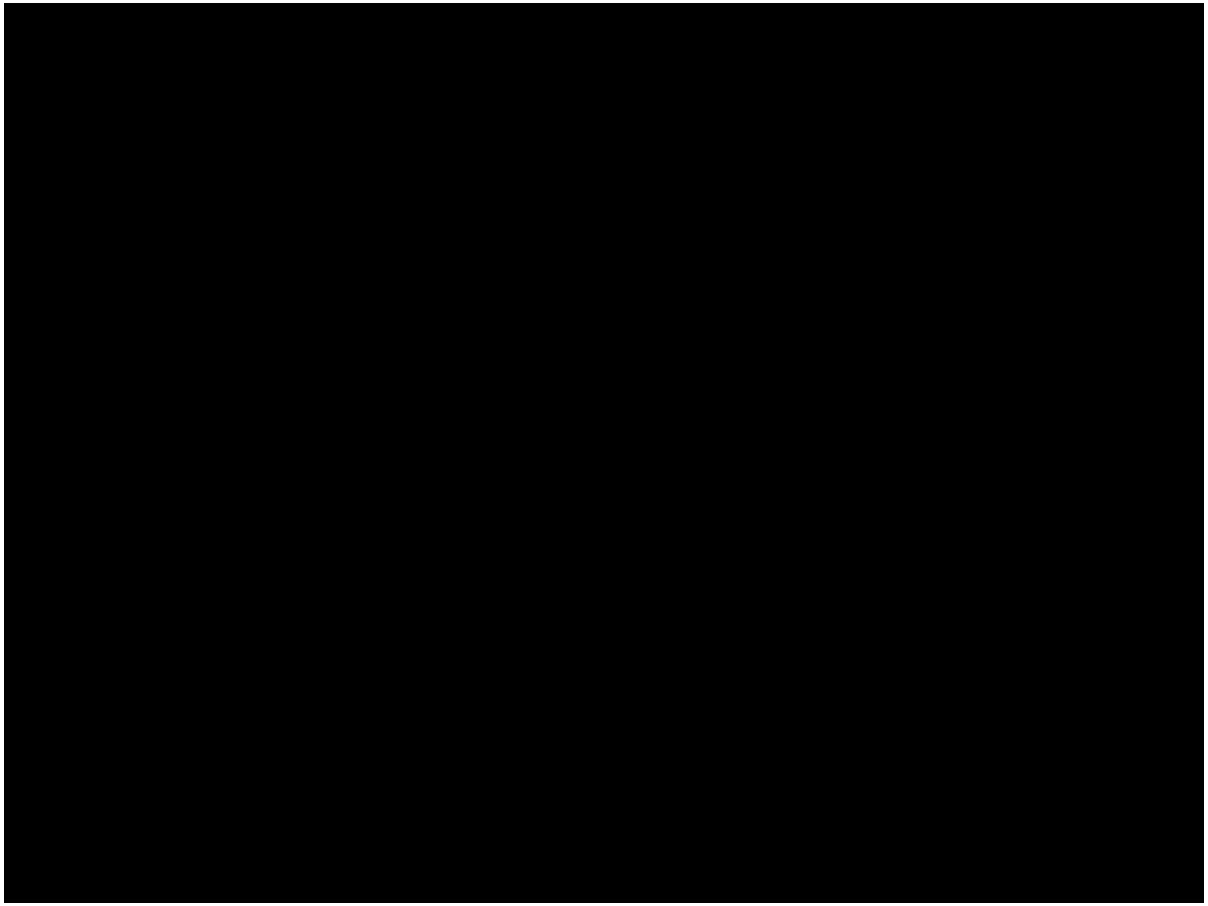


Edge detection on a signal that was convolved with a Gaussian

Wide Masks


Original Signal
4
2
0
-2
-4
0 50 100 150 200 250 300 350 400 450 500
Mask: -1 1
1
0
-1
0 50 100 150 200 250 300 350 400 450 500
Mask: -1 -1 1 1
1
0
-1
0 50 100 150 200 250 300 350 400 450 500
Mask: -1 -1 -1 1 1 1
1
0
-1
0 50 100 150 200 250 300 350 400 450 500
Mask: -1 -1 -1 -1 1 1 1 1
1
0
-1
0 50 100 150 200 250 300 350 400 450 500

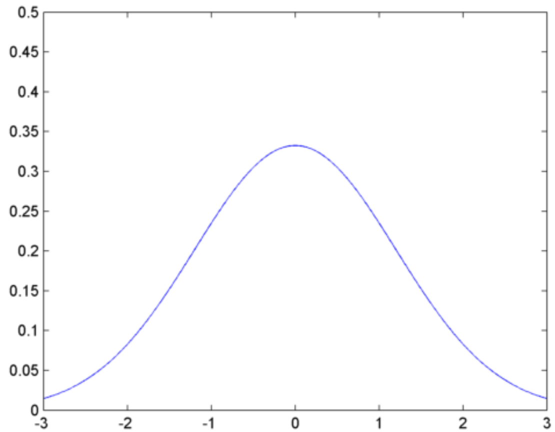
Cat Video



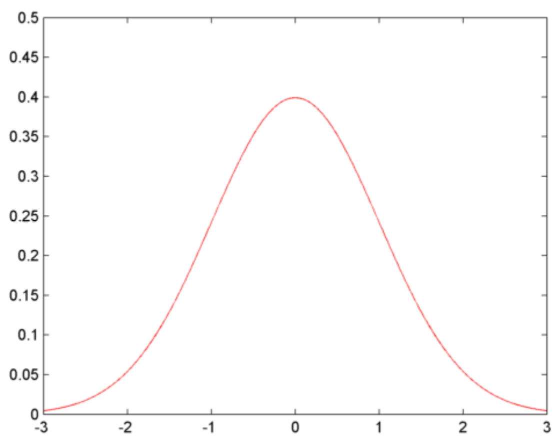
Interest Points

- How would we find "interesting" parts of the image?
- Let's say corners are "interesting"
- Convolve with Gaussians, take difference:

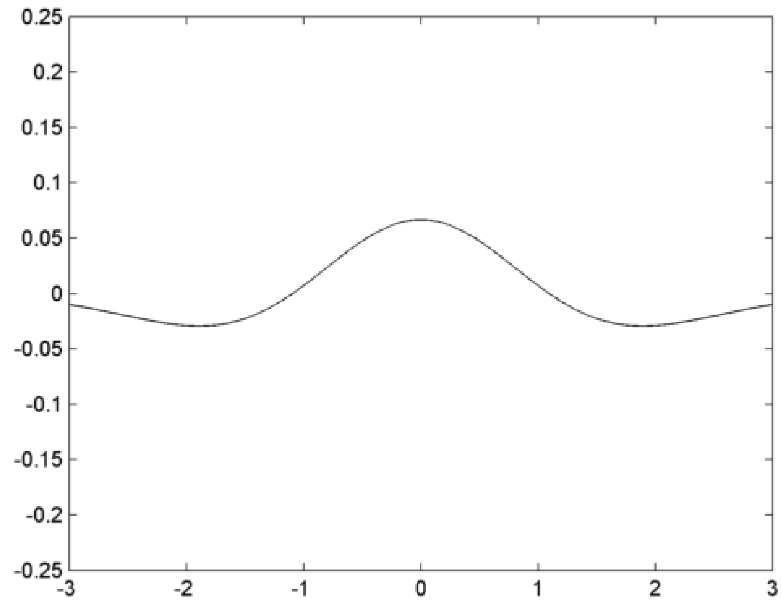
G1
$\sigma = 1.2$



G2
$\sigma = 1.0$



G1 – G2



Difference of Gaussians (DoG)

Difference of Gaussians

Original Howie

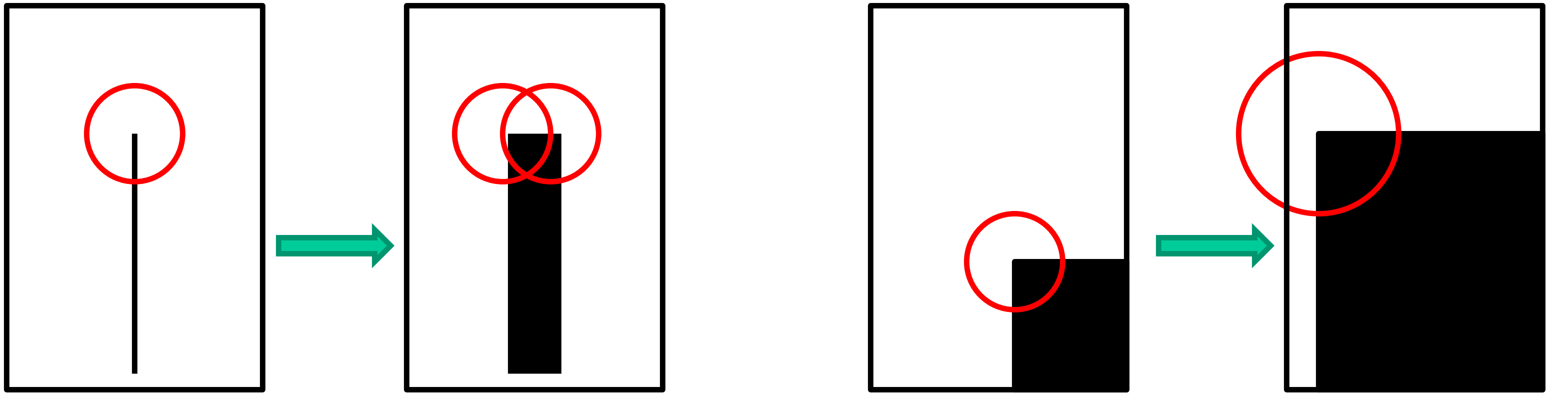
Howie with less blur

Howie with more blur

Difference of Howies

Scale Stability

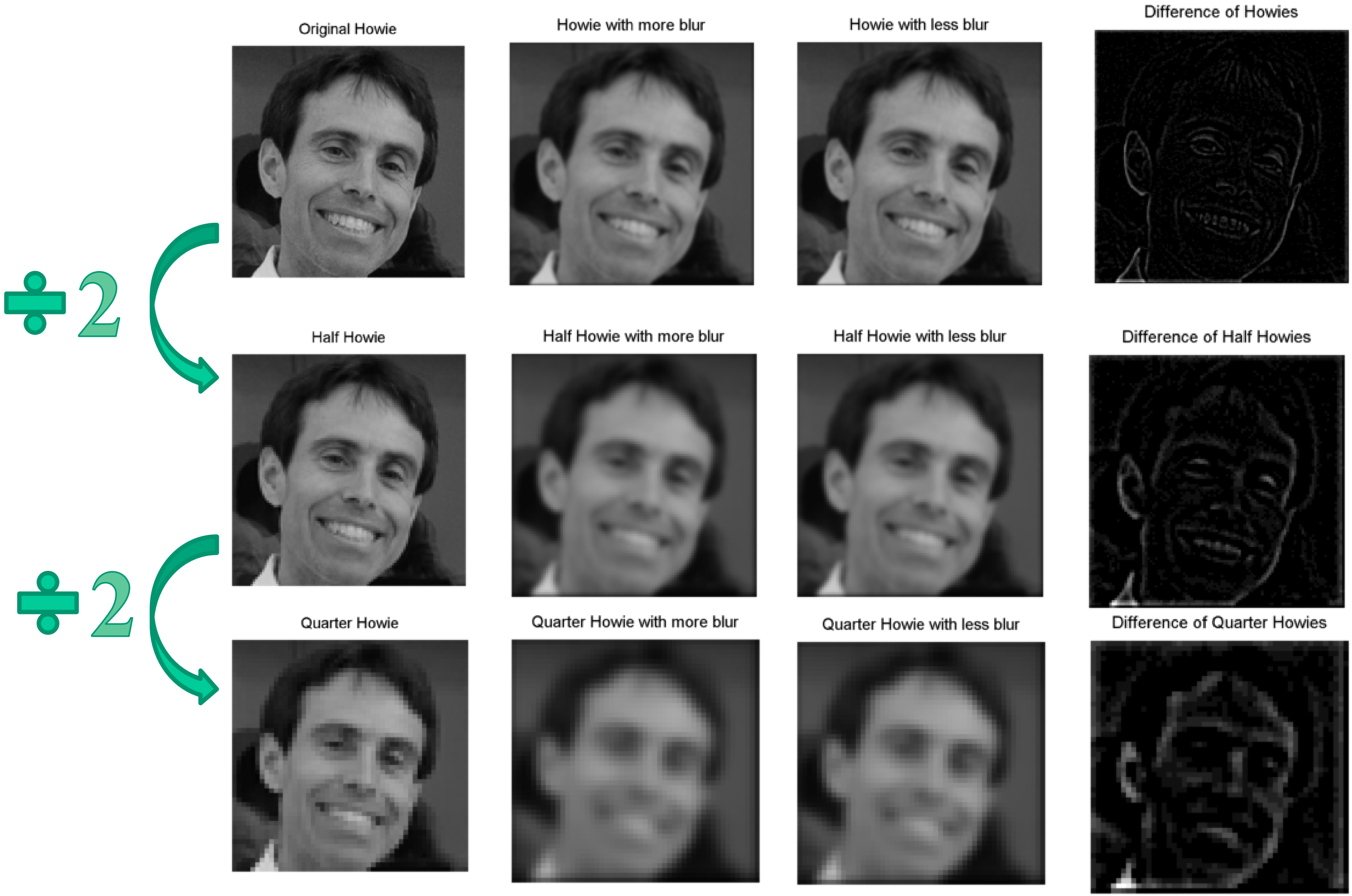
- Interest points should be stable to scaling
- Can calculate DoG at many scales, find strong edges in <u>space</u> and <u>scale</u>



Poorly localized

Well localized

Scale Space Example



Keypoint Example

Original Keypoints

Scaled 110%

Keypoint Orientation

- Local gradient magnitude, orientation calculated
- Keypoint scale affects blur, sampling used

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

- Magnitude and Gaussian weighted orientations binned, max taken as keypoint orientation

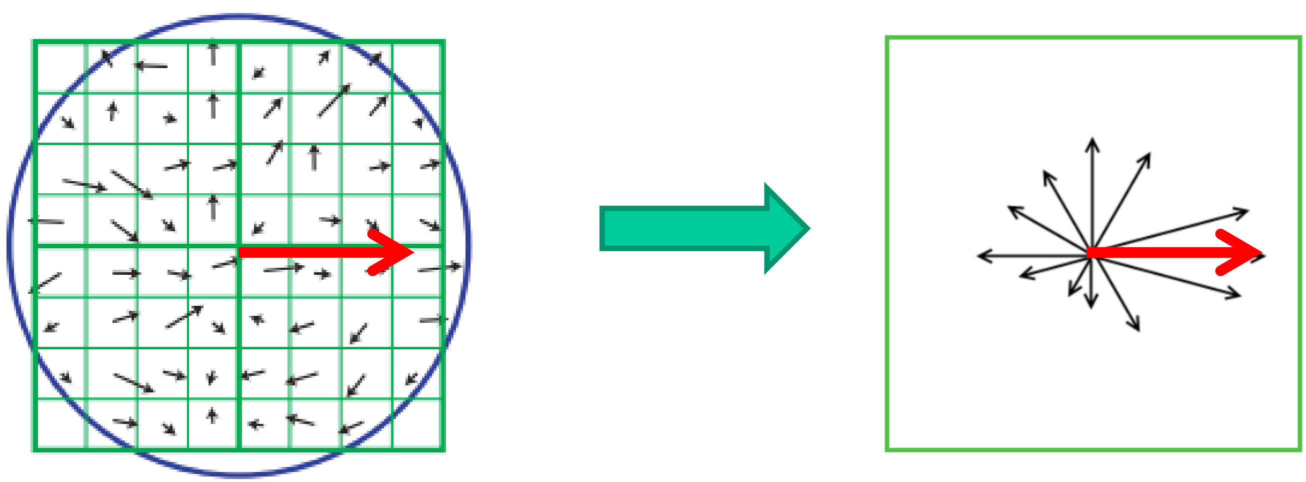


Image gradients

Figure from David G. Lowe

Local Descriptor

- Surrounding gradients summarized by smaller regional histograms
- Descriptor normalized to account for illumination

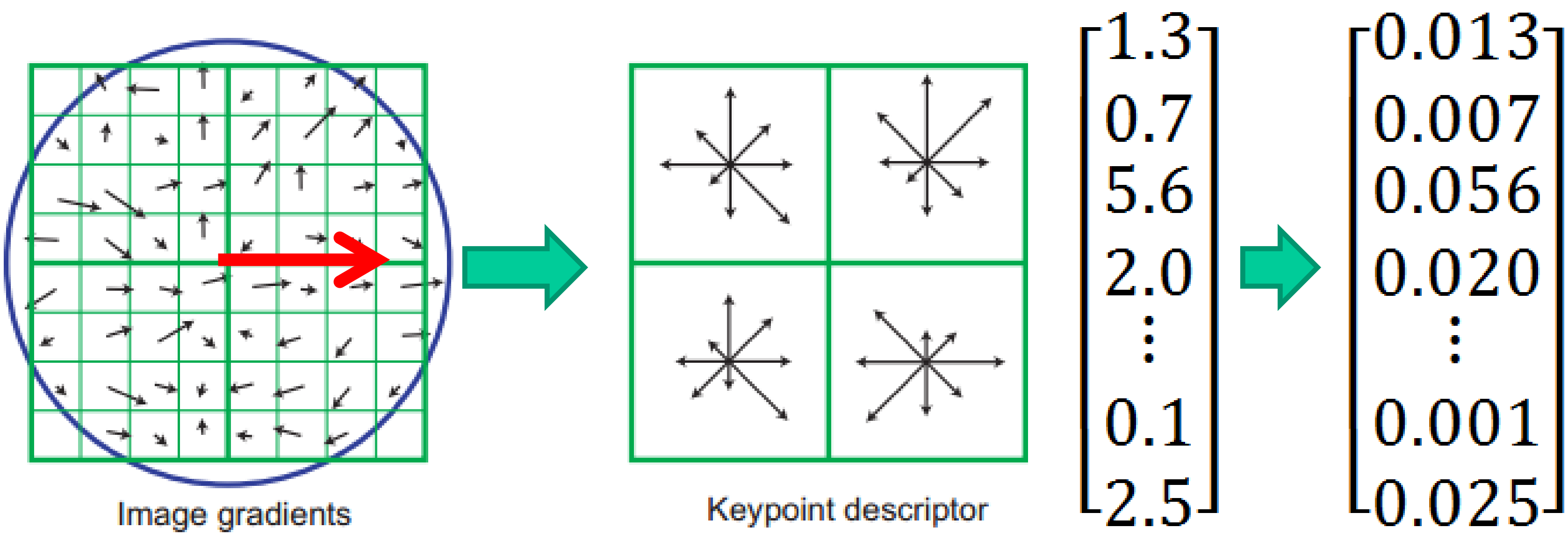
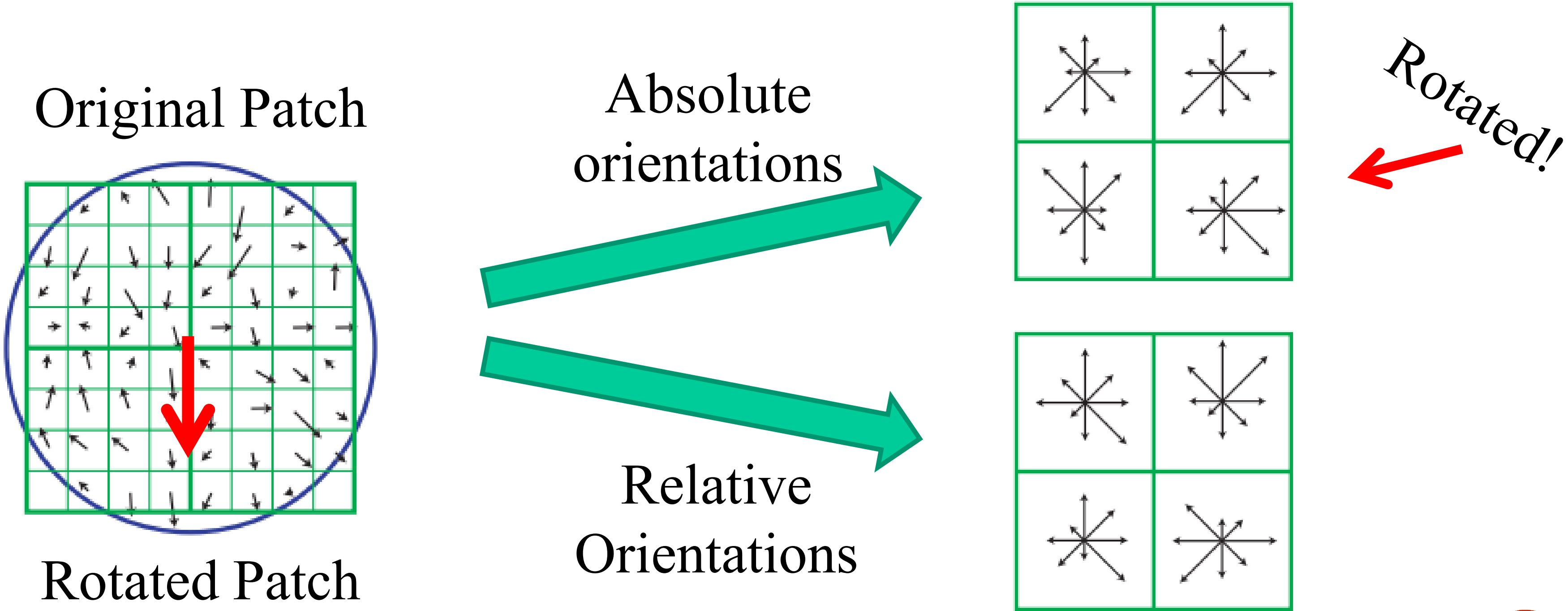


Figure from David G. Lowe

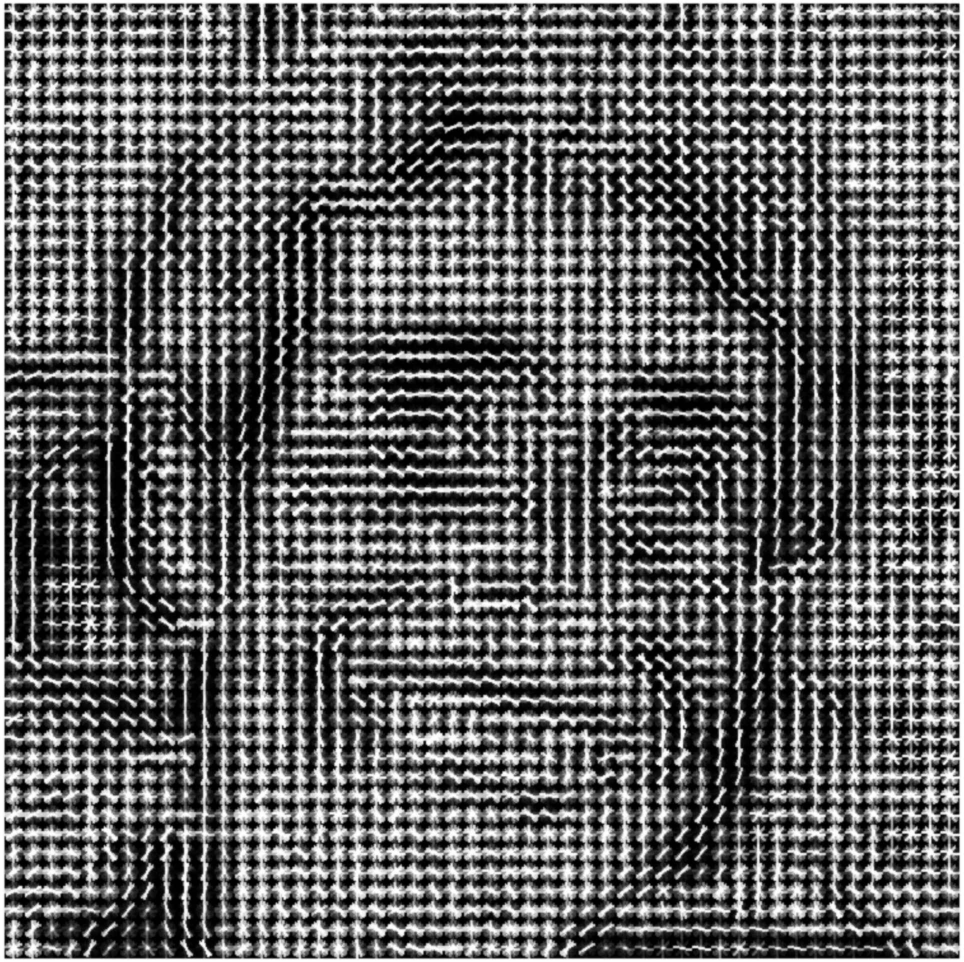
Rotational Invariance

- Descriptor orientations calculated relative to keypoint orientation



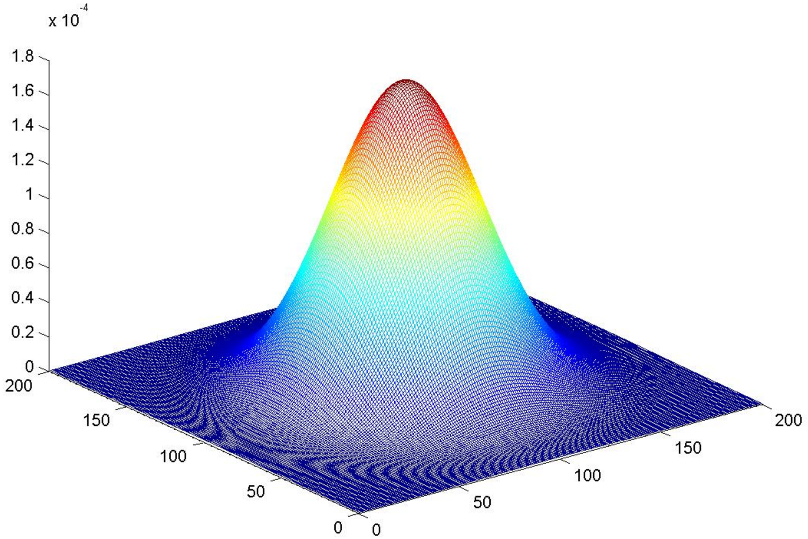
Histogram of Oriented Gradients (HOG)

- What if we calculated descriptors at every pixel?
  - "Dense features"
- Can be used as different "image"with many channels



Gaussian Masks

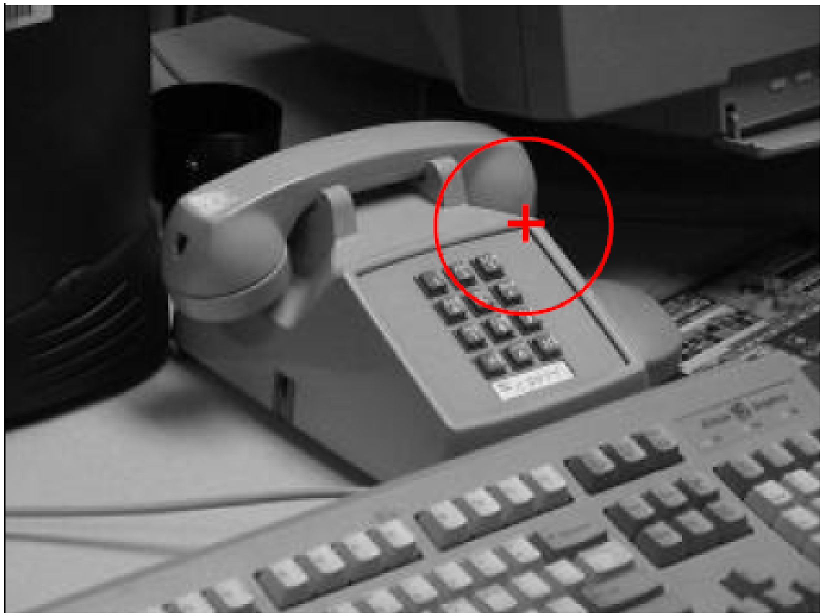
- Used to smooth images and for noise reduction
- Use before edge detection to avoid spurious edges



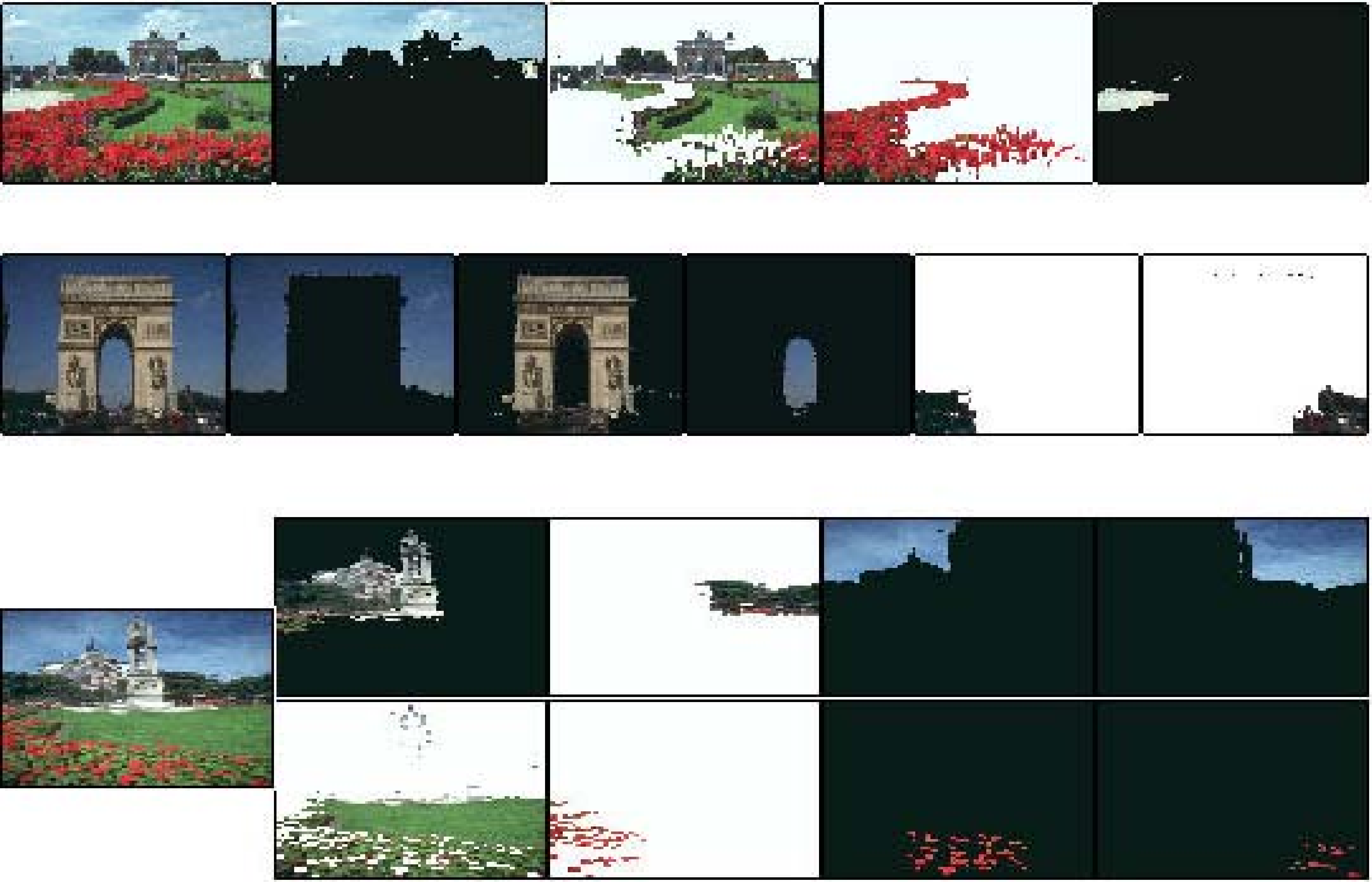
Johann Carl Friedrich Gauss

April 30, 1777 – Feb 23, 1855
number theory, statistics, analysis, differential geometry, geodesy, electrostatics, astronomy, and optics.

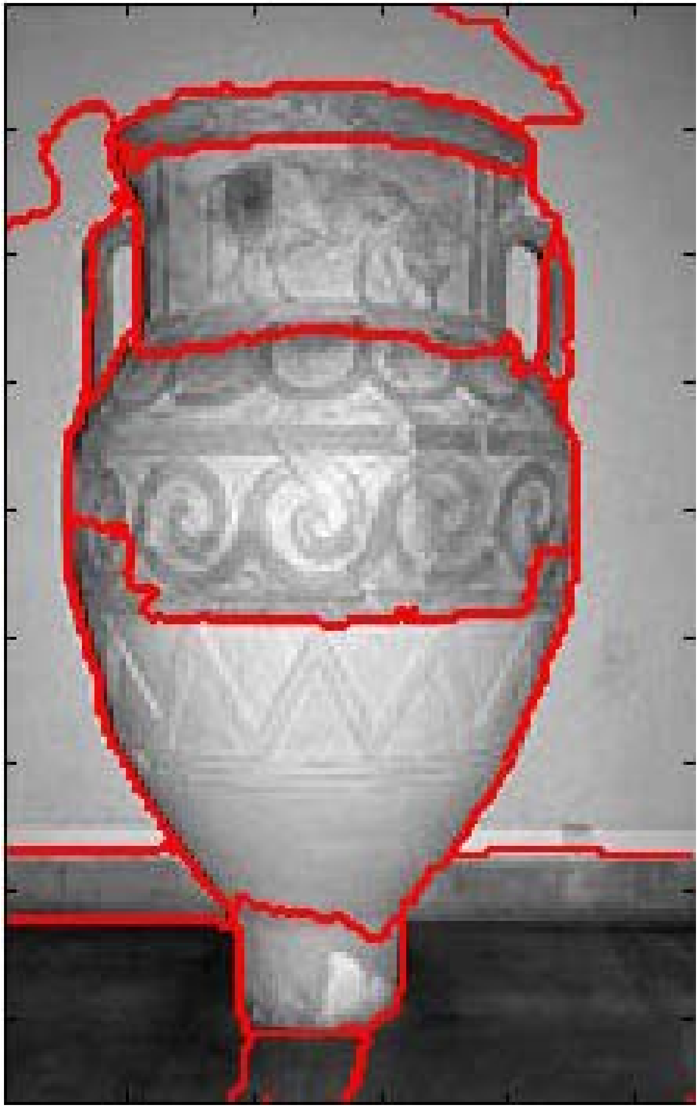
Interest points and descriptors

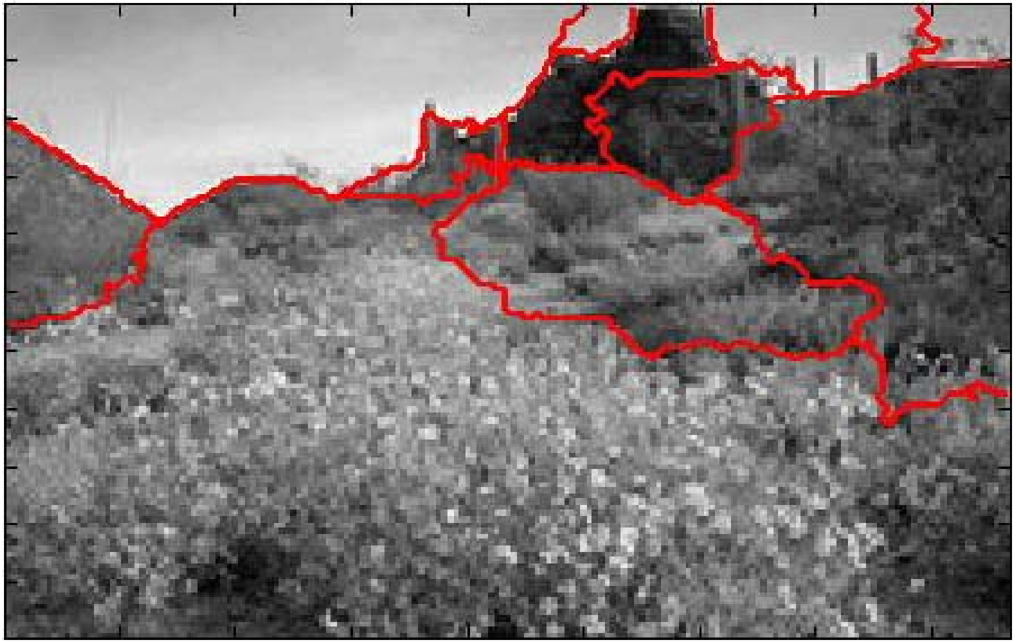


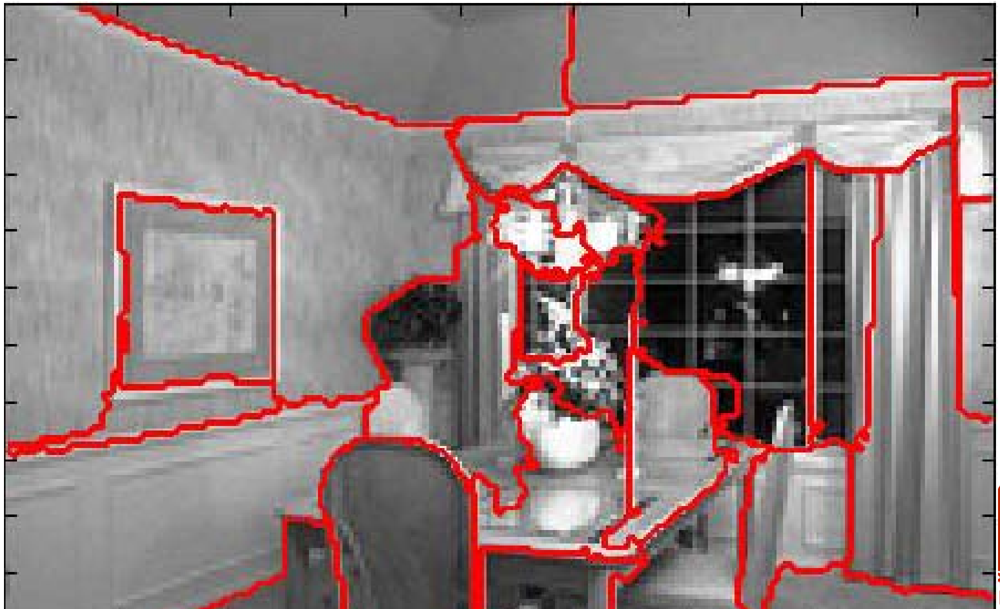
Segmentation



Segmentation





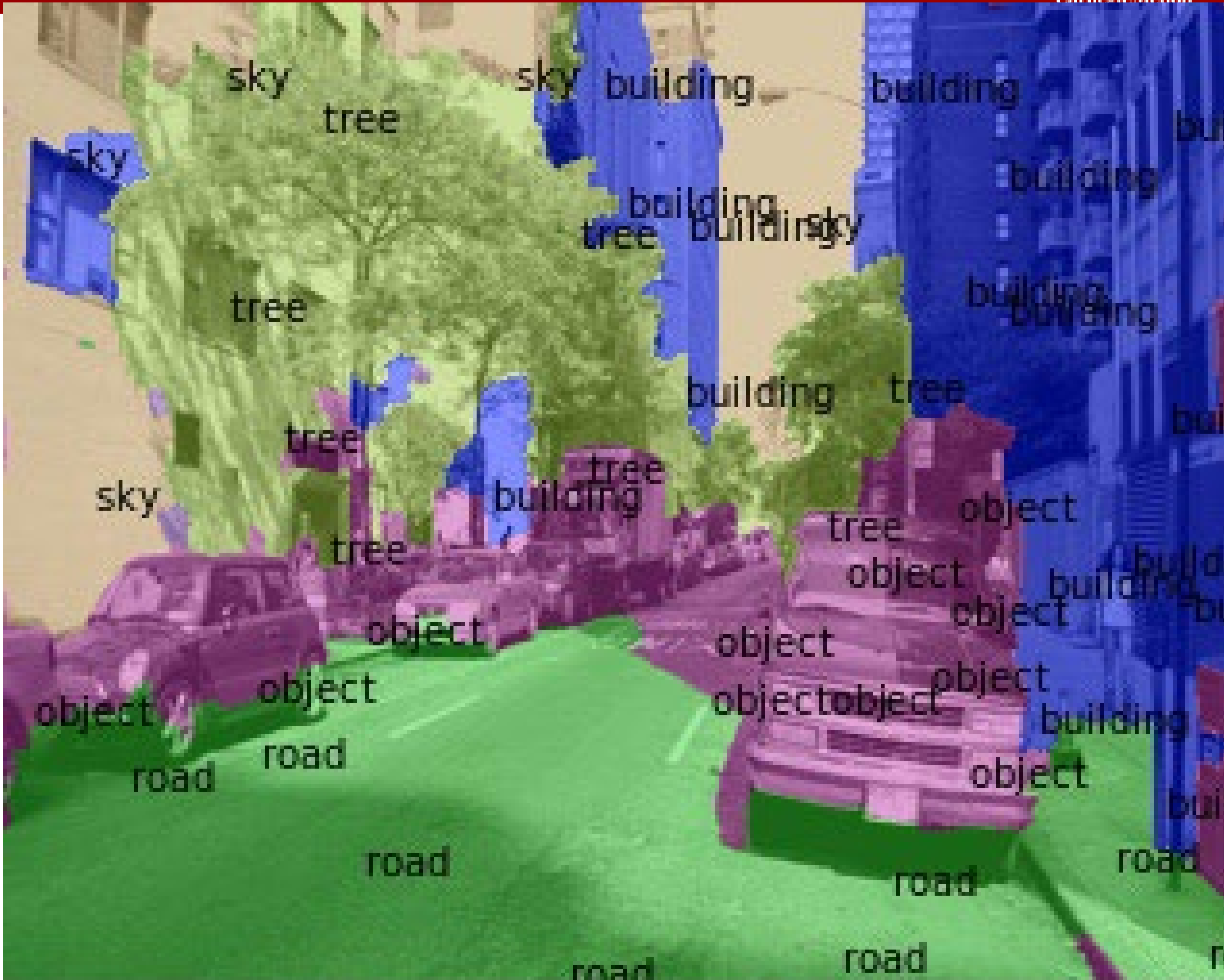


Recognizing objects and understanding scenes

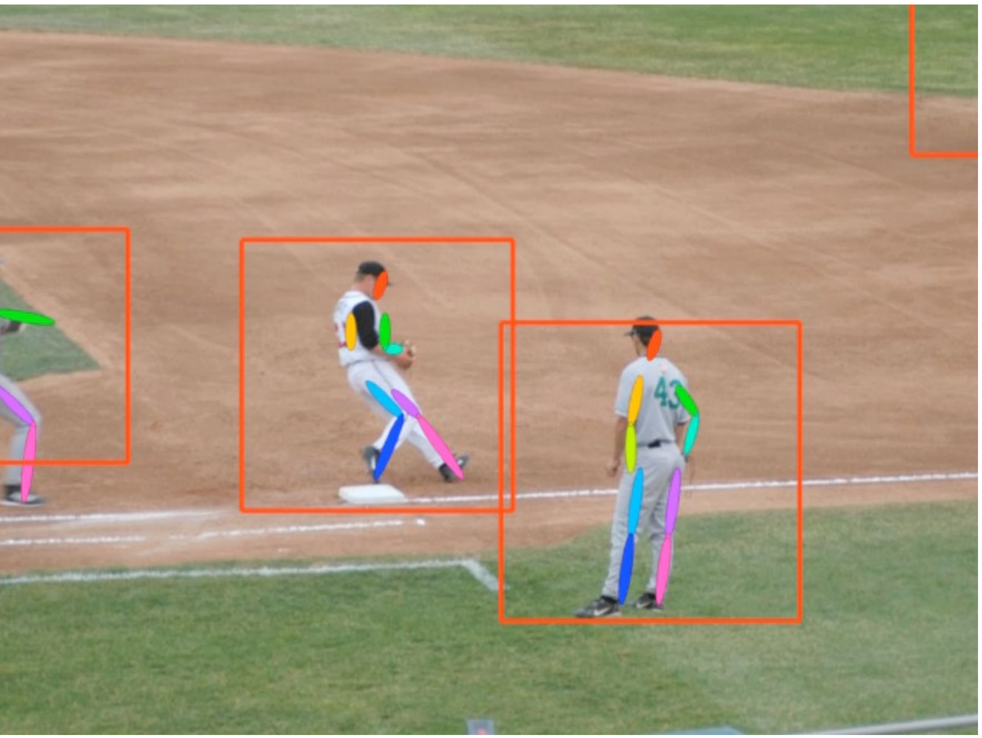

thermos : 35%

Recognition

Classification


sky
sky
building
building
tree
sky
building
tree
tree
building
tree
tree
building
sky
tree
object
tree
object
object
object
object
object
object
road
road
building
object
road
road
road
road

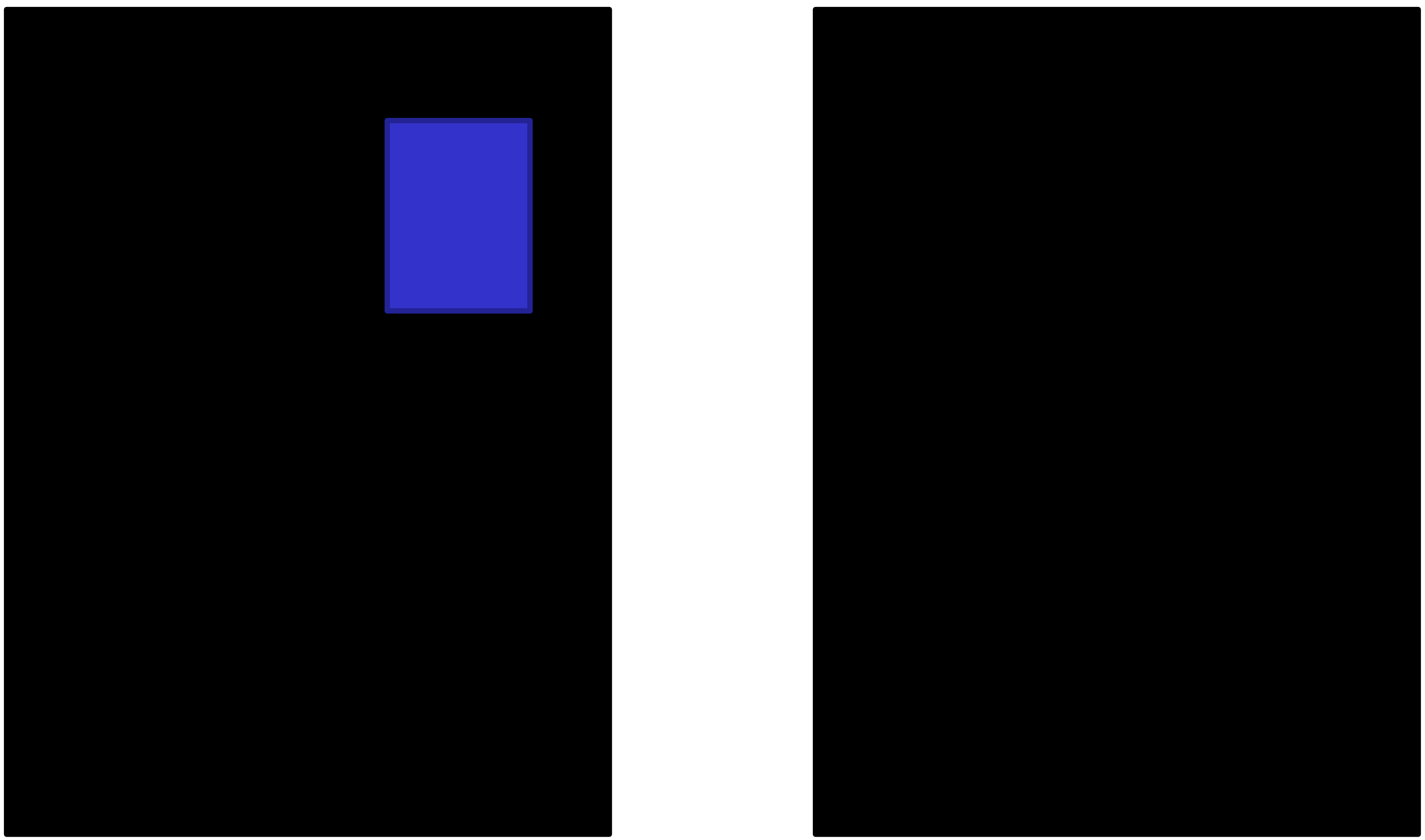
Using context



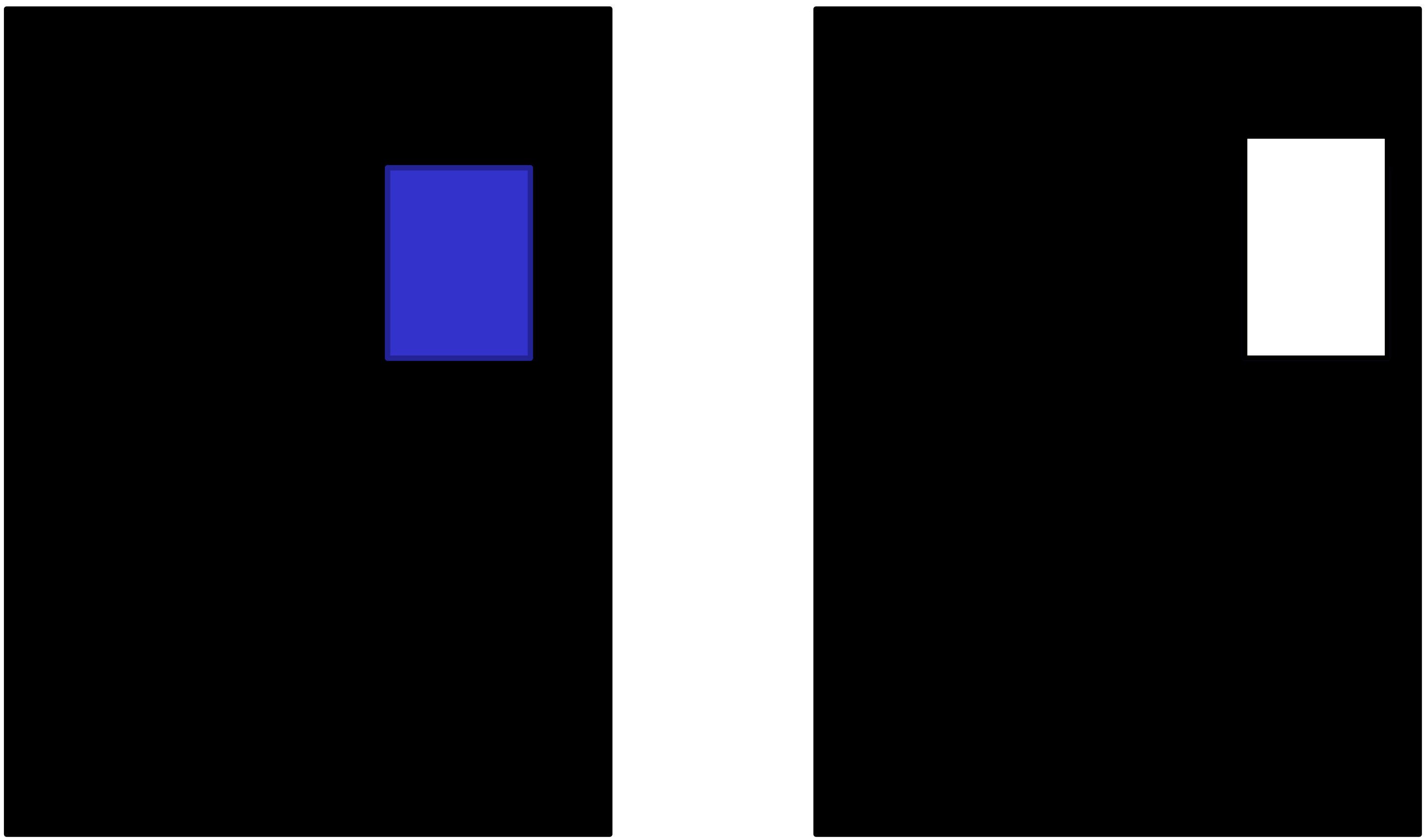
Understanding videos

A series of images

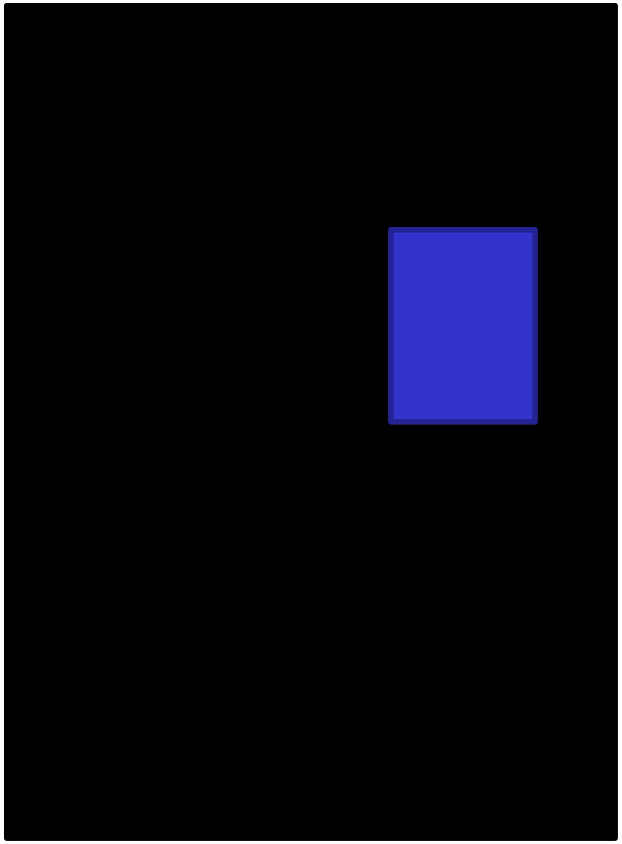
Using difference in pixel values to find objects

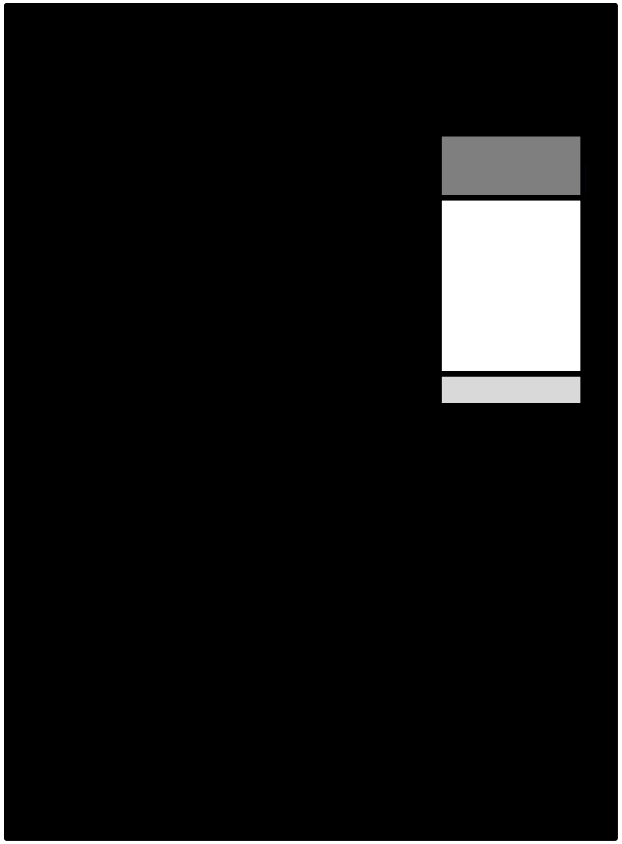


Using difference in pixel values to find objects



Using difference in pixel values to find objects





Motion