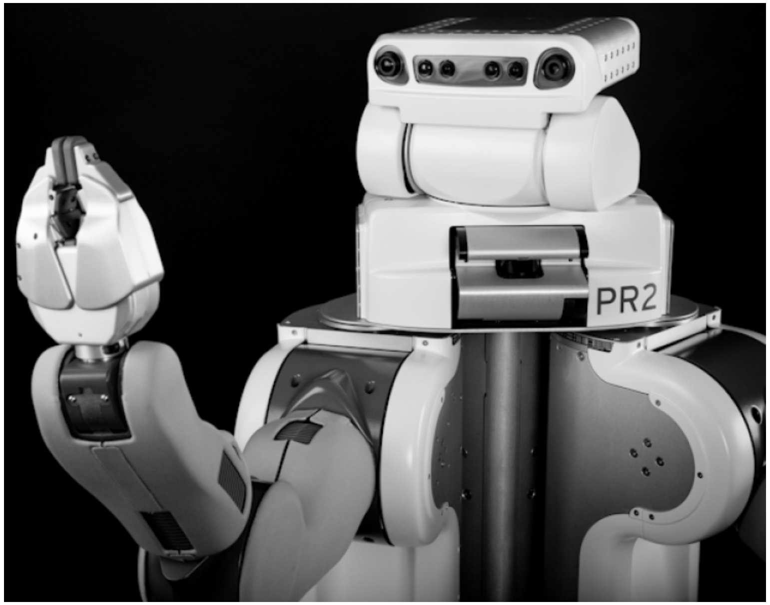
Chapter 8
Perception



Part 1

8.1 Image Processing Operators and Algorithms

Outline

- 8.1 Image Processing Operators and Algorithms
  - 8.1.1 Taxonomy of Computer Vision Algorithms
  - 8.1.2 High Pass Filtering Operators
  - 8.1.3 Low Pass Operators
  - 8.1.4 Matching Signals and Images
  - 8.1.5 Feature Detection
  - 8.1.6 Region Processing
  - Summary

Outline

- 8.1 Image Processing Operators and Algorithms
  - <u>8.1.1 Taxonomy of Computer Vision Algorithms</u>
  - 8.1.2 High Pass Filtering Operators
  - 8.1.3 Low Pass Operators
  - 8.1.4 Matching Signals and Images
  - 8.1.5 Feature Detection
  - 8.1.6 Region Processing
  - Summary

Introduction

- Mobile robotics needs these forms of mathematics:
  - kinematics: for relationships between robot and things
  - probability and statistics: for likelihood in absence of info
  - moving reference frames: for inertial sensors
- Perception also uses:
  - signal processing:
    - Suppress noise
    - Enhance edges
    - Match signals.
- Data can be
  - Range or appearance
  - 1D or 2D

Introduction

- Mobile robotics needs these forms of mathematics:
  - kinematics: for relationships between robot and things
  - probability and statistics: for likelihood in absence of info
  - moving reference frames: for inertial sensors
- Perception also uses:
  - signal processing:
    - Suppress noise
    - Enhance edges
    - Match signals.
- Data can be
  - Range or appearance
  - 1D or 2D

8.1.1 Taxonomy

- Image Processing
  - Operates on pixels without regard for what they represent.
  - Operates on raw input data
- Geometric Computer Vision
  - Infers shape or motion or both
  - Focus on spatial relationships
- Semantic Computer Vision
  - Recognize, reason about, interpret the nature of the scene

8.1.1.1 Image Processing Algorithms

- Edge Detection
- Smoothing
- Segmentation
- Feature Detection
- Optical Flow

8.1.1.2 Geometric Computer Vision

- Shape Inference
- Feature Tracking
- Visual Odometry
- Structure from Motion

8.1.1.2 Semantic Computer Vision

- Pixel Classification
- Object Detection
- Object Recognition
- Obstacle Detection
- Scene Understading

Outline

- 8.1 Image Processing Operators and Algorithms
  - 8.1.1 Taxonomy of Computer Vision Algorithms
  - <u>8.1.2 High Pass Operators</u>
  - 8.1.3 Low Pass Operators
  - 8.1.4 Matching Signals and Images
  - 8.1.5 Feature Detection
  - 8.1.6 Region Processing
  - Summary

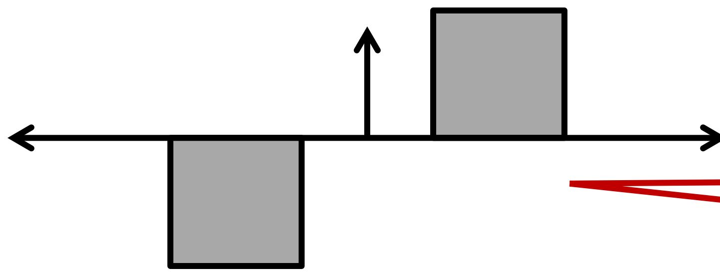
8.1.2 High Pass Operators

- Various operations (e.g. derivatives) applied to signals can …
  – enhance the high frequency information
  – and suppress the low frequency information.
- Good when high frequencies are the signal.
- Bad when the high frequencies are noise

8.1.2.1 First Derivatives in 1D

(Central Difference Template)

- Can visualize as a “template” (aka stencil, kernel, mask) which is applied everywhere in an image.



Looks like a sine curve

-1	0	1

- Approximations of arbitrary complexity can be obtained by:
  - writing Taylor series approximations and …
  - solving for the derivatives that appear in terms of the function values that appear.

8.1.2.2 Image Operators as Masks

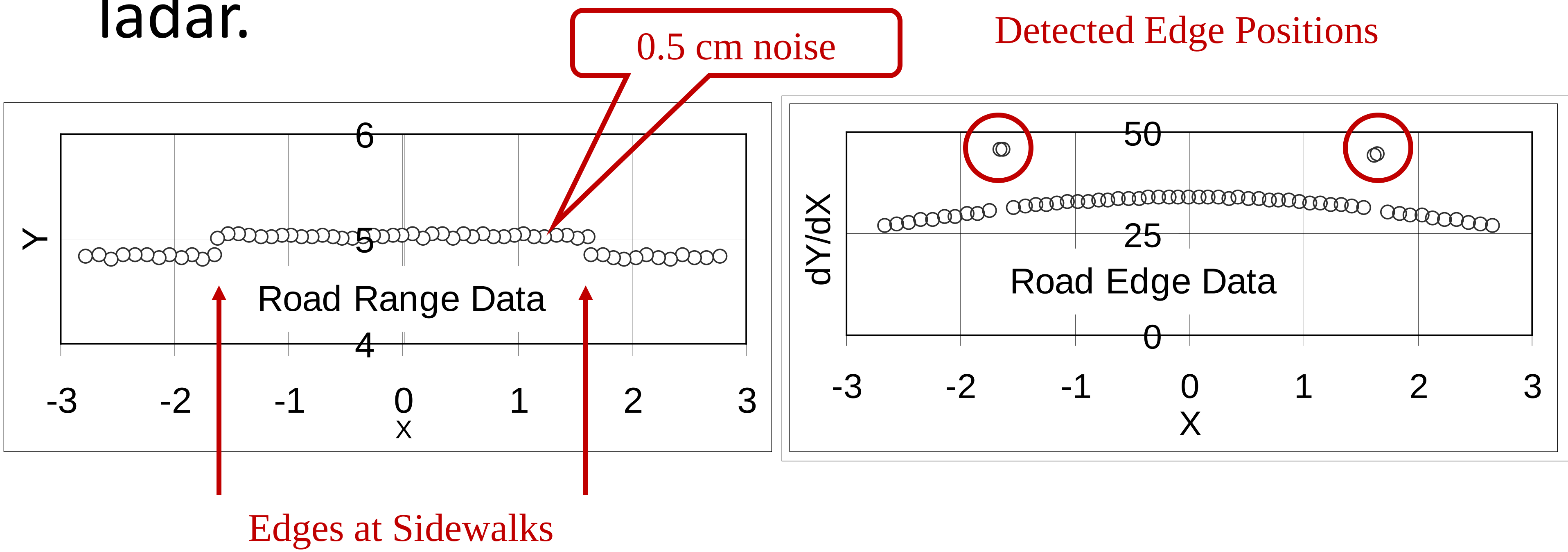
- To apply the operator (tempalte) at a given pixel....
  - Position the template
  - Perform a vector dot product

$$y'(i) = \sum_{k \in \{-1, 0, 1\}} m(k)y(i+k)$$

- Typically both the signal and the template are discrete signals.

8.1.2.3 1st Derivatives of 1D Range Data

- The central difference can be used to detect edges in range data.
- Consider data produced by a downward looking ladar.



8.1.2.4 First Derivatives of 2D Intensity Data

- In 2D, a famous central difference, the "Sobel" operator, looks like:

–1	0	1
–2	0	2
–1	0	1

$$\frac{\partial}{\partial x} I(x, y)$$

1	2	1
0	0	0
–1	–2	–1

$$\frac{\partial}{\partial y} I(x, y)$$

8.1.2.4 First Derivatives in 2D

(Sobel Operator Result)

- Each output pixel is an approximation of the gradient magnitude at the corresponding place in the input image.

8.1.2.4 First Derivatives in 2D

(Sobel Operator Result)

8.1.2.5 Second Derivatives in 1D

- Compute second derivatives as second differences - differences of first differences.
- Based on earlier definitions:

$$\frac{d^2y}{dx^2} \sim \frac{1}{\Delta x}\left[\left.\frac{dy}{dx}\right|_{fwd} - \left.\frac{dy}{dx}\right|_{bwd}\right]$$

- This expands to:

$$\frac{d^2y}{dx^2} \sim \frac{1}{\Delta x}\left[\frac{y(x+\Delta x) - y(x)}{\Delta x} - \frac{y(x) - y(x-\Delta x)}{\Delta x}\right]$$

$$\frac{d^2y}{dx^2} \sim \frac{y(x+\Delta x) - 2y(x) + y(x-\Delta x)}{(\Delta x)^2}$$

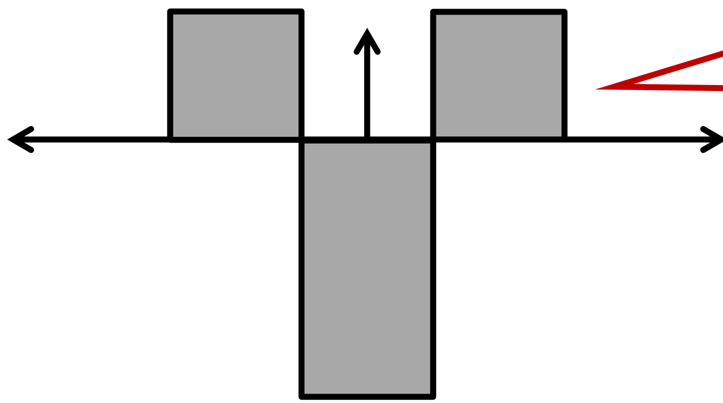
Derivatives Amplify Noise.

THE ROB

Second Derivative Template

- Can visualize as a template which is applied everywhere in an image.

Looks like an upside down sombrero

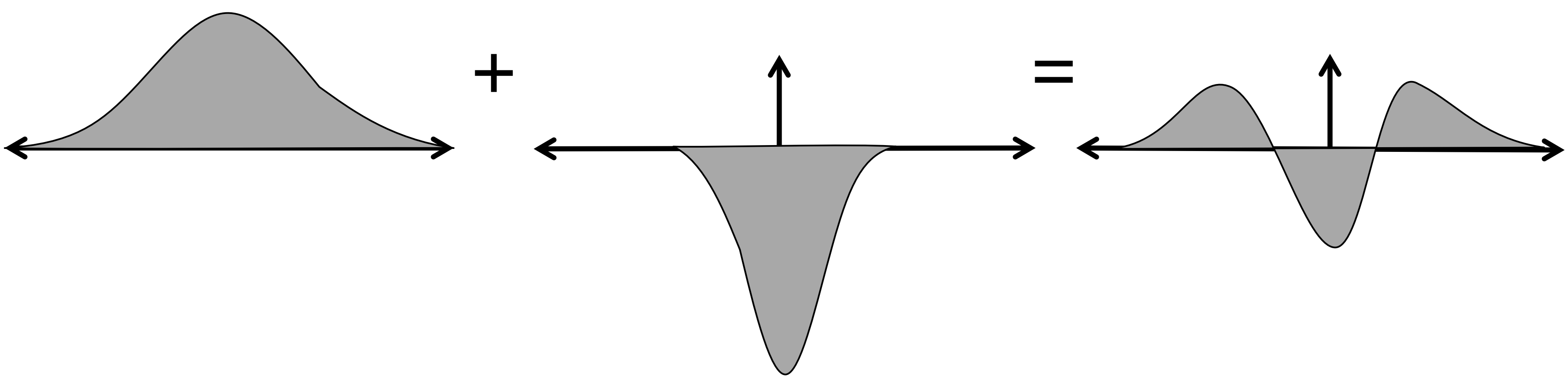


1	-2	1

- Often, this is approximated by subtracting a thin Gaussian from a wide one.

8.1.2.6 2nd Derivatives of Large Support

- Notice that
  - 1st derivatives are even functions of 2 humps
  - 2nd derivatives are odd functions of 3 humps
- When Δx is large relative to the pixel size, a <u>difference of Gaussians</u> is a good way to do a 2nd derivative:



8.1.2.7 Derivatives as Robust Comparisons

- Suppose y(x) has the Taylor series…

$$y(x) = a + bx + \frac{1}{2}cx^2 + \frac{1}{3!}dx^3 + \ldots$$

- Then its first and second derivatives are:
-

$$y'(x) = b + cx + \frac{1}{2}dx^2 + \ldots$$

Same, regardless of bias (a)

$$y''(x) = c + dx + \ldots$$

Same, regardless of bias (a) and scale (b)

8.1.2.8 Second Derivatives in 2D

- The Hessian matrix of a scalar spatial signal z(x,y) is:

$$\frac{\partial^2 z}{\partial \underline{x}^2} = \begin{bmatrix} \partial^2 z/\partial x^2 & \partial^2 z/\partial x \partial y \\ \partial^2 z/\partial x \partial y & \partial^2 z/\partial y^2 \end{bmatrix}$$

Associates a matrix with every point in an image

- Its trace is a scalar called the Laplacian:

$$\nabla^2 z = \frac{\partial^2 z}{\partial x^2} + \frac{\partial^2 z}{\partial y^2}$$

A measure of the "magnitude" of the Hessian.

8.1.2.8 Second Derivatives in 2D

(Laplacian Kernel)

- Looks like so:

0	1	0
1	-4	1
0	1	0

- Just the sum of two second derivatives at right angles.

8.1.2.8 Second Derivatives in 2D

(Uses of 2nd Derivatives)

- 1) Maximal Edge Detection: First derivatives (edges) are locally highest (or lowest) when the second derivatives are zero. “Zero crossings”
- 2) Normalization: The second derivative of a signal contains all information except:
  - the mean (bias) and
  - the linear deviation from the mean (scale).

8.1.2.9 Statistical Normalization in 1D

- The mean of a signal at time t, computed over an interval T is:

$$f_{mean}(t) = \frac{1}{T} \int_{(t-T/2)}^{(t+T/2)} f(\tau) d\tau$$

- The rms value is: $f_{rms}(t) = \sqrt{\frac{1}{T} \int_{(t-T/2)}^{(t+T/2)} [f(\tau)]^2 d\tau}$

- Lets similarly define the standard deviation as:

$$f_{std}(t) = \sqrt{\frac{1}{T} \int_{(t-T/2)}^{(t+T/2)} [f(\tau) - \bar{f}(t)]^2 d\tau}$$

- The normalized signal (at interval T) can be defined as:

$$\tilde{f}(t) = \frac{f(t) - f_{mean}(t)}{f_{std}(t)}$$

8.1.2.10 Image Sum Notation

- To avoid messy looking sums.
- Let index i vary symmetrically of a window of width h

$$\sum_{i \in h} f = \sum_{i=-h/2}^{i=h/2} f$$

8.1.2.11 Statistical Normalization – 2D

- In discrete 2D imagery:
  - Local mean:
  $$\mu(x,y) = \frac{1}{wh}\sum_{i \in w}\sum_{j \in h} I(x+i, y+j)$$
  - Variance:
  $$\sigma^2(x,y) = \frac{1}{(wh-1)}\sum_{i \in w}\sum_{j \in h}\{I(x+i, y+j) - \mu(x,y)\}^2$$
  - Standard Deviation:
  $$\sigma(x,y) = \sqrt{\sigma^2(x,y)}$$
  - Normalized Image:

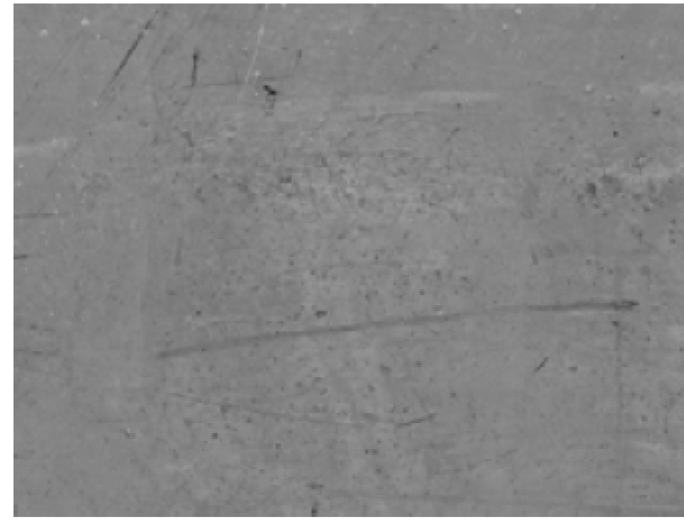
Removes bias and then reports deviation in units of σ.

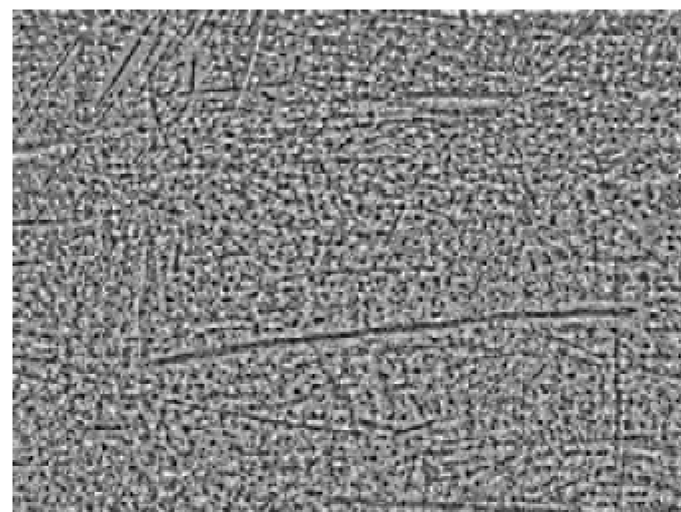
$$\tilde{I}(x,y) = \frac{I(x,y) - \mu(x,y)}{\sigma(x,y)}$$

Contains normalized deviation from the mean.

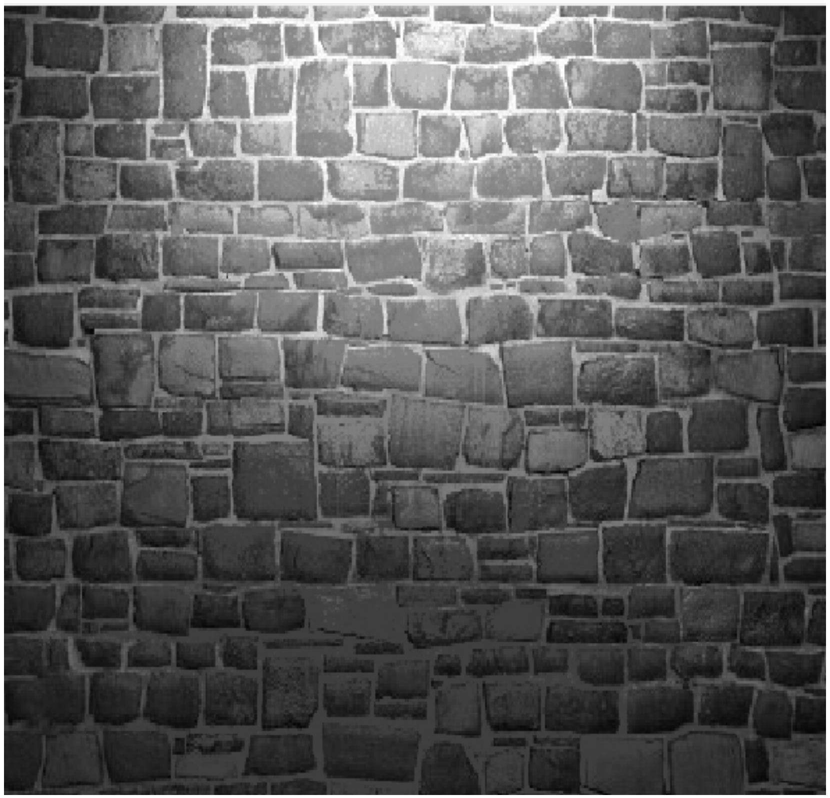
8.1.2.11 Statistical Normalization – 2D
(Enhancing Texture)

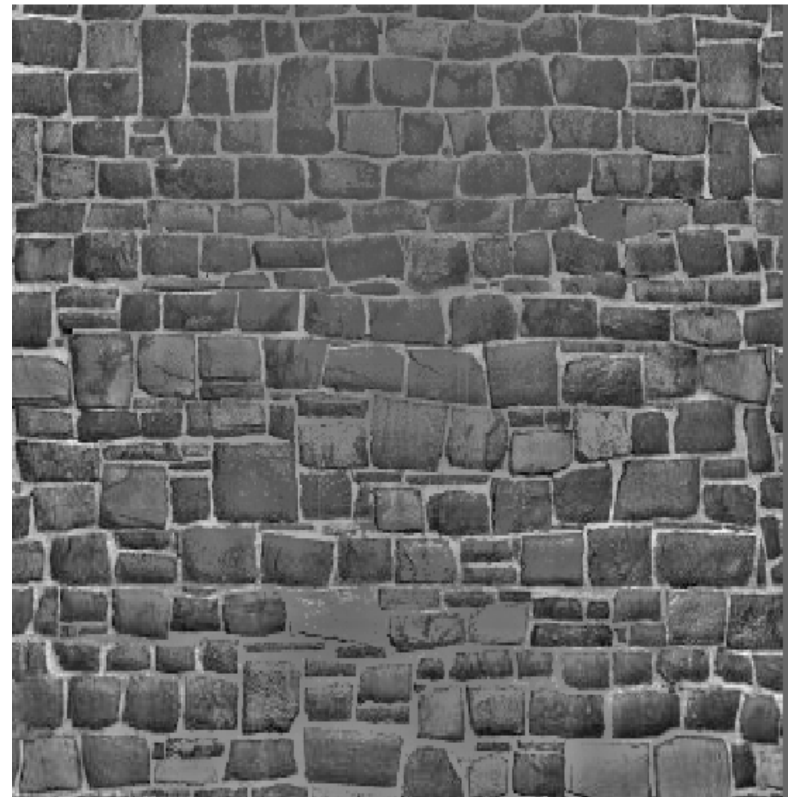
- Normalization:
  - not well behaved if the denominator is small
  - often makes no sense to match flat signals anyway
- Use after a 2nd derivative has removed the local plane fit.
- Result is the “texture” or “edginess” of the signal





8.1.2.11 Statistical Normalization – 2D
(Enhancing Texture)





Outline

- 8.1 Image Processing Operators and Algorithms
  - 8.1.1 Taxonomy of Computer Vision Algorithms
  - 8.1.2 High Pass Filtering Operators
  - <u>8.1.3 Low Pass Operators</u>
  - 8.1.4 Matching Signals and Images
  - 8.1.5 Feature Detection
  - 8.1.6 Region Processing
  - Summary

8.1.3 Low Pass Operators

- Various operations (e.g. integrals) applied to signals can
  - enhance the low frequency information
  - and suppress the high frequency information.
- Good when low frequencies are the signal.
- Bad when the low frequencies are not useful.

8.1.3.1 Average Filtering

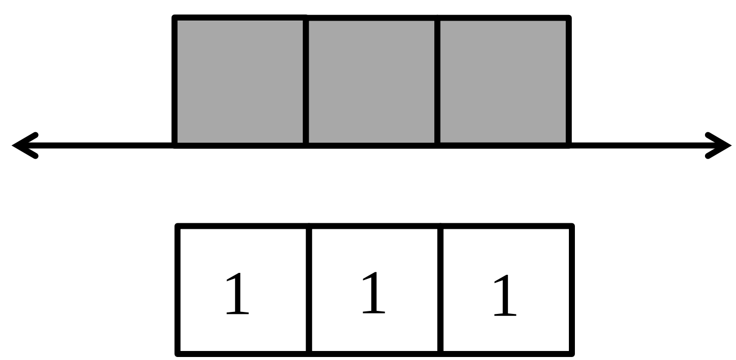
- Replace every signal value by the average of the neighborhood around it.
- In 1D, this is:

$$\bar{f}(t) = \frac{1}{T} \int_{(t-T/2)}^{(t+T/2)} f(\tau)d\tau$$

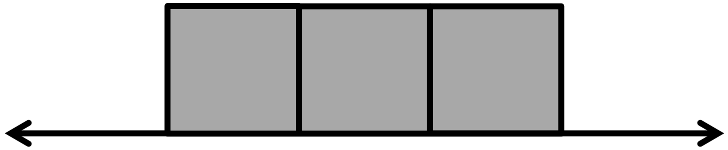
- T is called the "support" of the operator.

8.1.3.1 Average Filtering (Efficiency of Box Filter)

- Efficient ways to compute:
  - subtract the last value and add the next as window moves.
- Image "pyramids" can be defined where each layer is half as large as layer below.

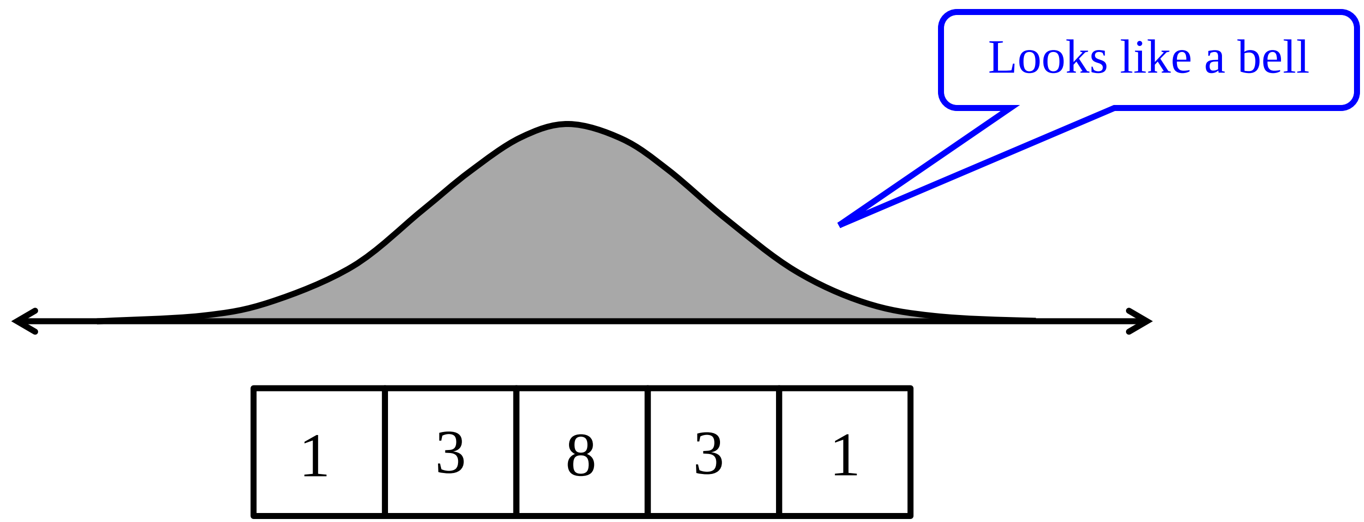


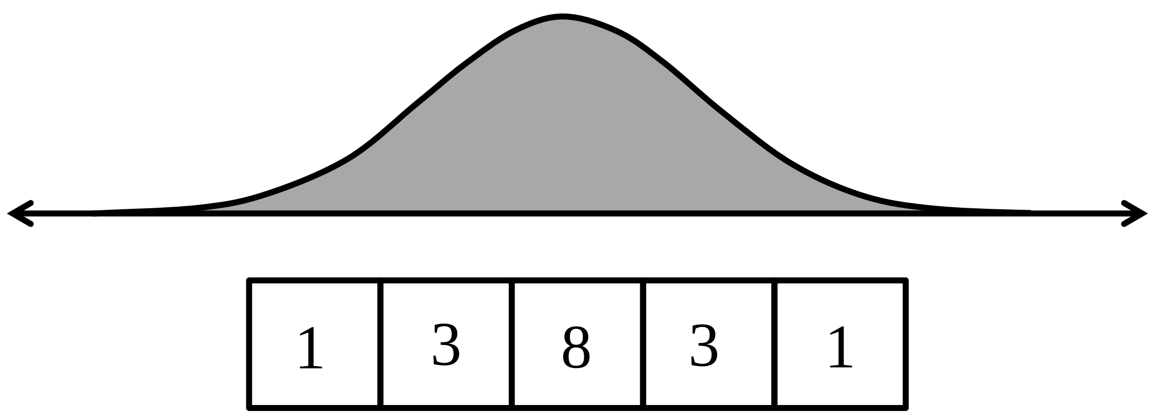
1	1	1



8.1.3.2 Gaussian Filtering

- Could use a Gaussian shaped kernel:




1
3
8
3
1

Template Properties

- Notice that the operators covered so far have these properties:
  - integrals: even and unimodal (bell)
  - 1st derivative: odd and bimodal (sine)
  - 2nd derivative: even and trimodal (sombrero)

Outline

- 8.1 Image Processing Operators and Algorithms
  - 8.1.1 Taxonomy of Computer Vision Algorithms
  - 8.1.2 High Pass Filtering Operators
  - 8.1.3 Low Pass Operators
  - <u>8.1.4 Matching Signals and Images</u>
  - 8.1.5 Feature Detection
  - 8.1.6 Region Processing
  - Summary

8.1.4 Matching Signals and Images

- Some motivating uses are:
  - recognition. Determining if an instance of an object appears in the image.
  - registration/mosaicking. Joining together two partial views to produce a larger view.
  - tracking. determining the displacement that a known region has undergone as a result of parallax or motion.

8.1.4.1 Convolution

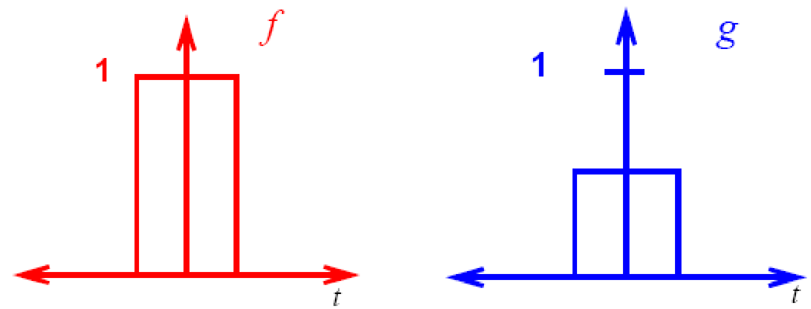
- Formally, the convolution of two signals f(t) and g(t) is the:
  - integral of the continuous product of the two.
  - computed as a function of their relative position, as they are slid over each other.
- That is:

$$f^*g = \int_0^t f(\tau)g(t-\tau)d\tau$$

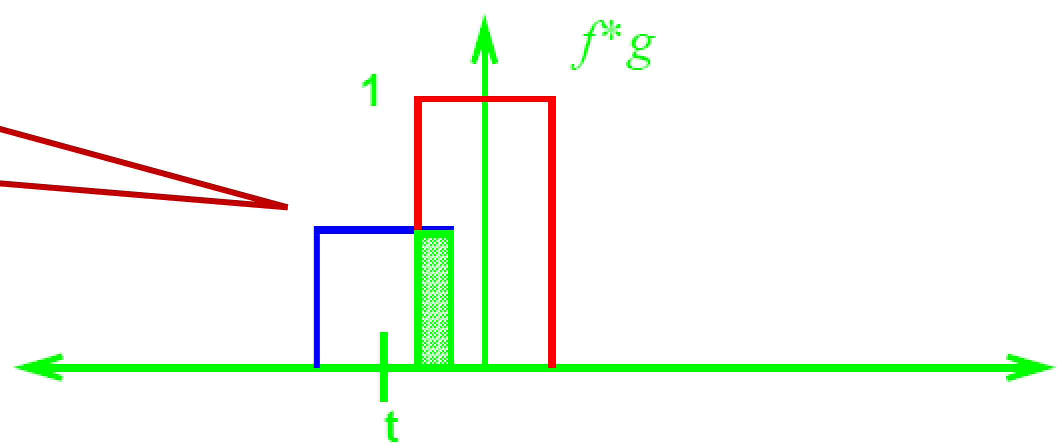
8.1.4.1 Convolution

$$f^*g = \int_0^t f(\tau)g(t-\tau)d\tau$$

- τ is a dummy variable
- when $\tau = t$ integrand contains g(0)
- when $\tau = t + e$ integrand contains g(-e)
- Hence g() is reflected about the origin.



Output is area under their product in the region of overlap.



8.1.4.1 Convolution

(Convolution as Image Processing)

- The relationship to vision is this.
  - g() can be thought of as an <u>operator</u> which is to be applied at every pixel in the image.
  - f() can be thought of as the <u>image</u> to be operated upon.
  - Changes in the variable t (or x or y in the spatial domain) correspond to moving the operator over the image.
- The region over which g() is nonzero is often called the <u>support</u> of the operator.

8.1.4.2 Correlation

- The (cross) correlation is defined as the integral of the product over a region:

$$f \times g = \int_0^t f(\tau) g(t+\tau) d\tau$$

- Same as convolution but without the flip of g().

8.1.4.3 Correlation in 2D

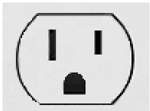
- In discrete 2D imagery, the (double) integral becomes a double sum:

$$(\tilde{F} \times \tilde{G})(x, y) = \frac{1}{wh} \sum_{i \in h} \sum_{j \in w} \tilde{F}(x+i, y+j)\tilde{G}(x+i, y+j)$$

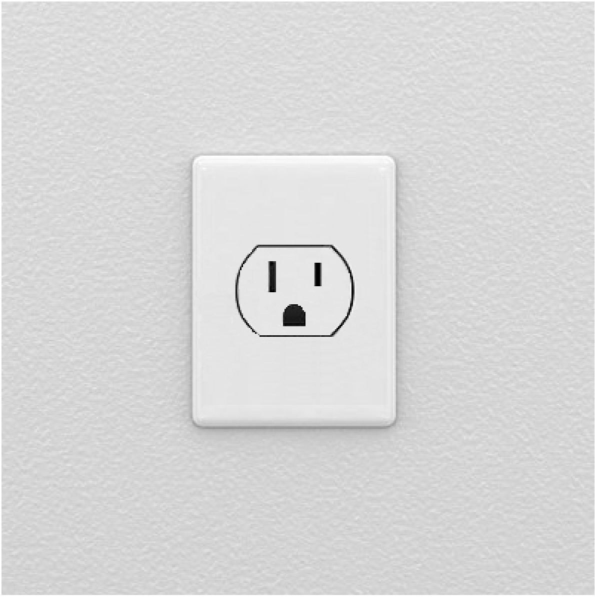
- Often used to match regions in two images against each other by searching a region of (x,y) for the best match.

8.1.4.3 Correlation in 2D : Example

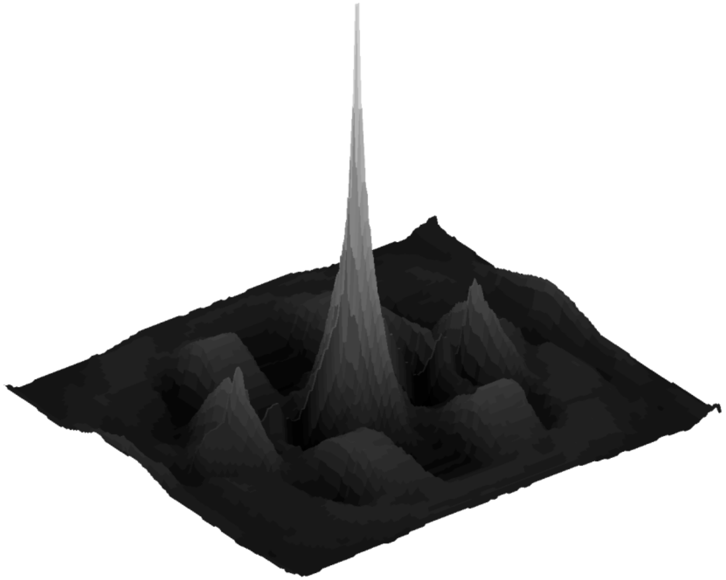
The peak in the correlation signal occurs at the location of the closest match.



Find This Plug



In This Image



8.1.4.4 Sums of Differences

- An alternative view of matching is minimizing differences.
- The sum of squared differences of two signals is defined as:

$$SSD(f,g)(t) = \int_0^t [f(\tau) - g(t+\tau)]^2 d\tau$$

- For an image, this becomes

$$SSD(F,G)(x,y) = \frac{1}{wh} \sum_{i \in w} \sum_{j \in h} [\tilde{F}(i,j) - \tilde{G}(x+i, y+j)]^2$$

8.1.4.4 Sums of Differences

(Absolute Differences)

- An alterative to squared differences is <u>absolute</u> differences:

$$SAD(f, g) = \int_0^t |f(\tau) - g(\tau)| d\tau$$

- For an image, this becomes

$$SAD(F, G) = \frac{1}{wh} \sum_{i \in h} \sum_{j \in w} |\tilde{F}(x+i, y+j) - \tilde{G}(x+i, y+j)|$$

Outline

- 8.1 Image Processing Operators and Algorithms
  - 8.1.1 Taxonomy of Computer Vision Algorithms
  - 8.1.2 High Pass Filtering Operators
  - 8.1.3 Low Pass Operators
  - 8.1.4 Matching Signals and Images
  - <u>8.1.5 Feature Detection</u>
  - 8.1.6 Region Processing
  - Summary

8.1.5 Feature Detection

- Features == “interesting” points
- Are points, curves, or regions in an image which are distinguished in some useful way.
- The word “feature” is used in a broad range of contexts in vision.
  - Points with high texture or where lines intersect in imagery.
  - Regions like edges, lines, shapes (e.g. blobs of specified moments) in images.
  - Points of high curvature in range imagery.
  - Regions of constant curvature in range imagery.
  - Regions of constant depth in sonar data.

8.1.5 Feature Detection

(Good Features Are…)

- Persistent from image to image and hence trackable
- Relatively rare in the image and hence a good way to distill the scene to a few pieces of data
- Known to be well distributed providing a good basis for triangulation
- Surrounded by relatively distinct neighborhoods which creates a potential for recognition.

- Normally, we care mostly about their position and perhaps about an attribute or two (like length).

8.1.5.1 Detecting Features to Track in Imagery

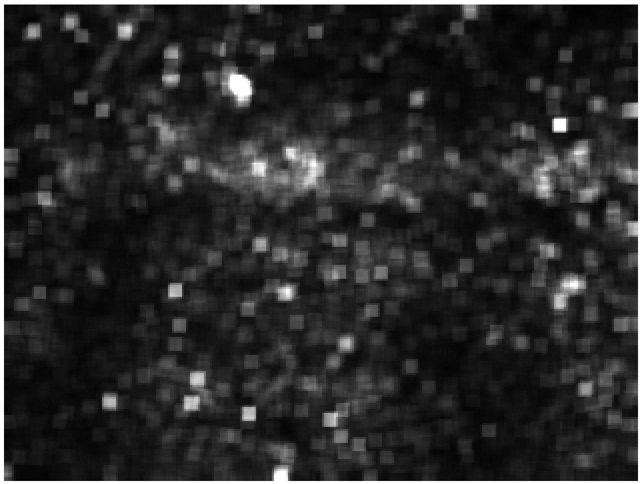
- A minimal assumption is that the environment is "textured".
- Assume also that textures are not repetitive - different places look somewhat different.
- Harris Detector considers Eigenvalues of:

$$H = \begin{bmatrix} \sum w(x,y)\nabla x \nabla x & \sum w(x,y)\nabla x \nabla y \\ \sum w(x,y)\nabla x \nabla y & \sum w(x,y)\nabla y \nabla y \end{bmatrix}$$

Covariance of Image Gradient Components

8.1.5.1 Detecting Features to Track in Imagery
(Texture Scores)

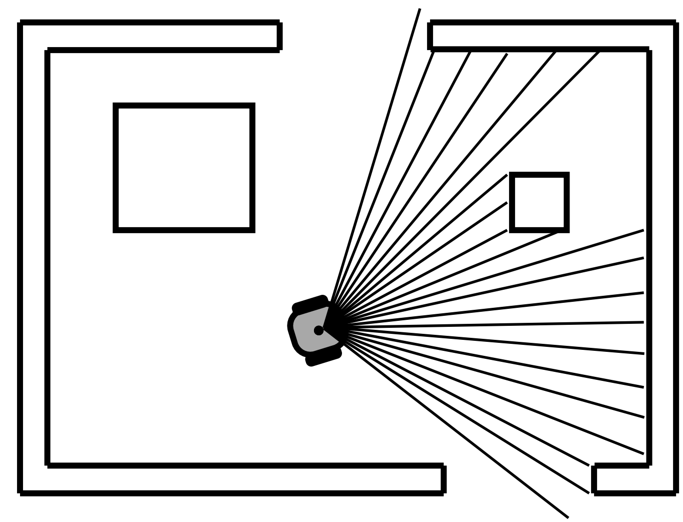
- Bright spots in the right image are regions of high texture in the left image.

8.1.5.1 Detecting Features to Track in Imagery
(Harris Corners)

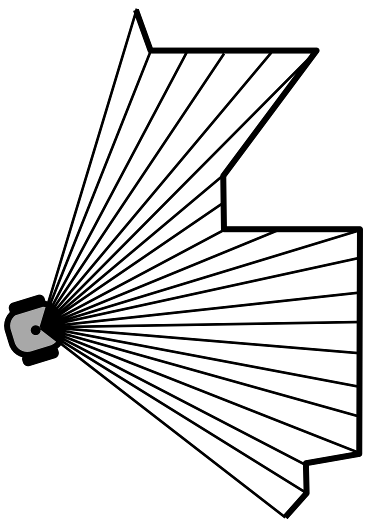
8.1.5.2 Finding Corners in Range Data

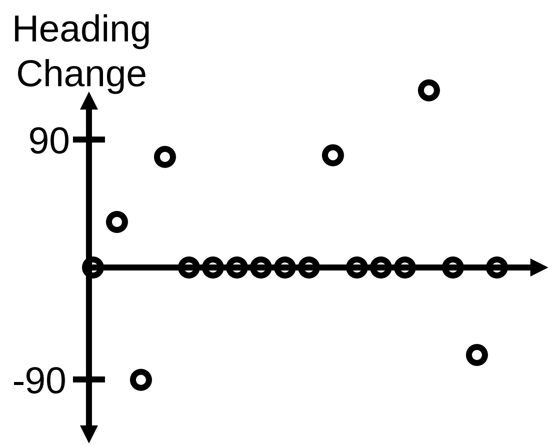
- Ladar scans of indoor scenes have a lot of right angles in them.
- These are useful for computing ego motion (visual odometry) or for map-based guidance.
- Beware occluding edges that masquerade as real surfaces.
- While sparsely separated endpoints may be an occluding edge, closely separated ones are probably not.



8.1.5.2 Finding Corners in Range Data

- Curvature-based edge finding is particularly effective in environments which are composed mostly of lines.
- Points of infinite curvature are corners
- Points of high curvature are likely to be corners that got smoothed by aliasing





Outline

- 8.1 Image Processing Operators and Algorithms
  - 8.1.1 Taxonomy of Computer Vision Algorithms
  - 8.1.2 High Pass Filtering Operators
  - 8.1.3 Low Pass Operators
  - 8.1.4 Matching Signals and Images
  - 8.1.5 Feature Detection
  - 8.1.6 Region Processing
  - Summary

8.1.6 Region Processing

- These create and process arbitrary shapes in an image. Only some will be covered here.
- Segmentation
  - extracts regions of pixels that are similar in some way.
  - In many cases (in both intensity and range imagery), these regions will correspond to objects.
- Growing and Thinning
  - shrink and expand regions.
  - useful for cleanup of small errors.
- Splitting and Merging
  - of lines and regions can be used to find canonical descriptions of scenes in terms of natural objects.
  - a good way to find the largest object possible of some type (e.g. line).

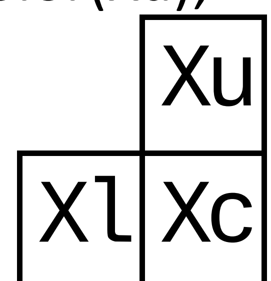
8.1.6 Region Processing

- Medial Axis and Grassfire Transforms
  - extremely efficient for finding the skeleton and range contours of an arbitrary shape.
- Moment and Invariant Computations
  - abstract regions to a few numbers that are often invariant to scale and perspective transformation.
  - provide convenient metrics to compare shapes for recognition purposes.
- Histogramming and Thresholding
  - Finds natural boundaries of pixel classes
  - Shrink images to a small number of "colors"

8.1.6.1 Segmentation : Appearance Imagery

- Want to group groups pixels which are similar and adjacent into regions.
- Similiarity measure can be anything.
- A very fast one-pass algorithm exists.
- A second pass though equivalence classes produces unique ids for each “blob”.
- Generalizes readily to nonbinary imagery.

```
if (f(Xc)==0) then continue
else {
  if( f(Xu) ==1 && f(Xl) ==0)
    color(Xc) = color(Xu);
  if( f(Xl) ==1 && f(Xu) ==0)
    color(Xc) = color(Xl);
  if( f(Xl) ==1 && f(Xu) ==1)
    color(Xc) = color(Xu);
    color(Xl) equiv color(Xu);
}
```

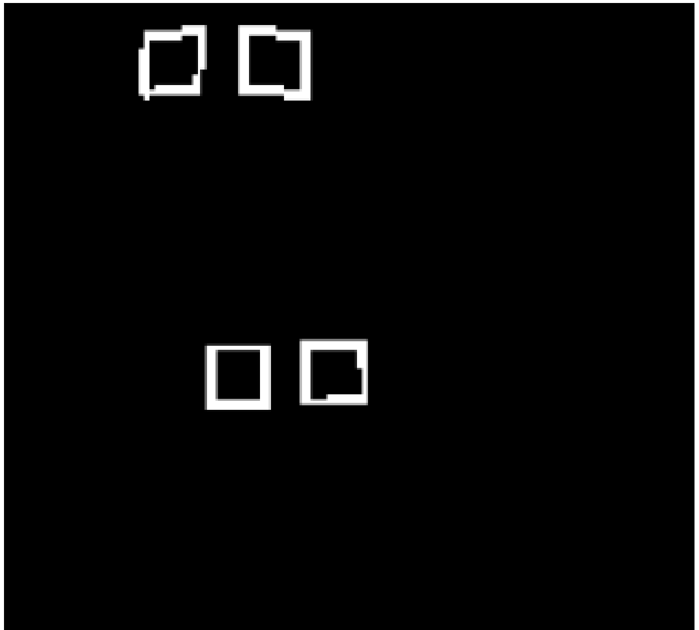


8.1.6.2 Detecting Shapes

- Eg Fiducials - deliberately place markers when you can.
- Once you have a region, <u>moments</u> are a compact encoding of shape.

$$I_x = \sum x \qquad I_y = \sum y \qquad I_{yy} = \sum y^2$$

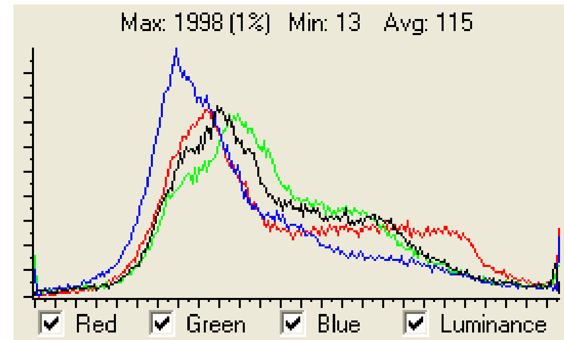
$$I_{xx} = \sum x^2 \qquad I_{xy} = \sum xy$$



- Compare moments of detected regions to those of prototypes.

8.1.6.3 Histogramming

- Basically, the PDF of colors and intensities.
- Orange, black, white, and green below.



Also, an excellent way to compare images. See SIFT features, for example.

8.1.6.4 Segmentation: 2D Range Imagery

- Curvature is invariant to viewpoint so it's a good feature for recognition.
- For a surface of the form: $z = z(x, y)$
- The Hessian matrix is:

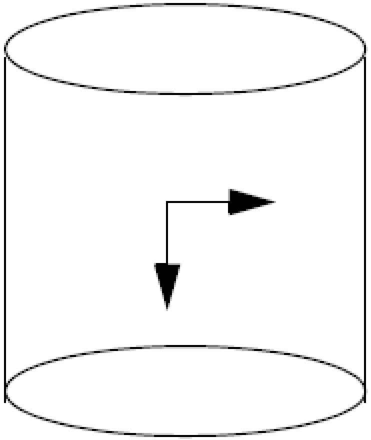
$$H = \begin{bmatrix} \dfrac{\partial^2 z}{\partial x^2} & \dfrac{\partial^2 z}{\partial x \partial y} \\ \dfrac{\partial^2 z}{\partial x \partial y} & \dfrac{\partial^2 z}{\partial x^2} \end{bmatrix}$$

8.1.6.4 Segmentation: 2D Range Imagery

- Define Mean and Gaussian curvatures:

$$H = \frac{\kappa_1 + \kappa_2}{2} \qquad G = \kappa_1 \kappa_2$$

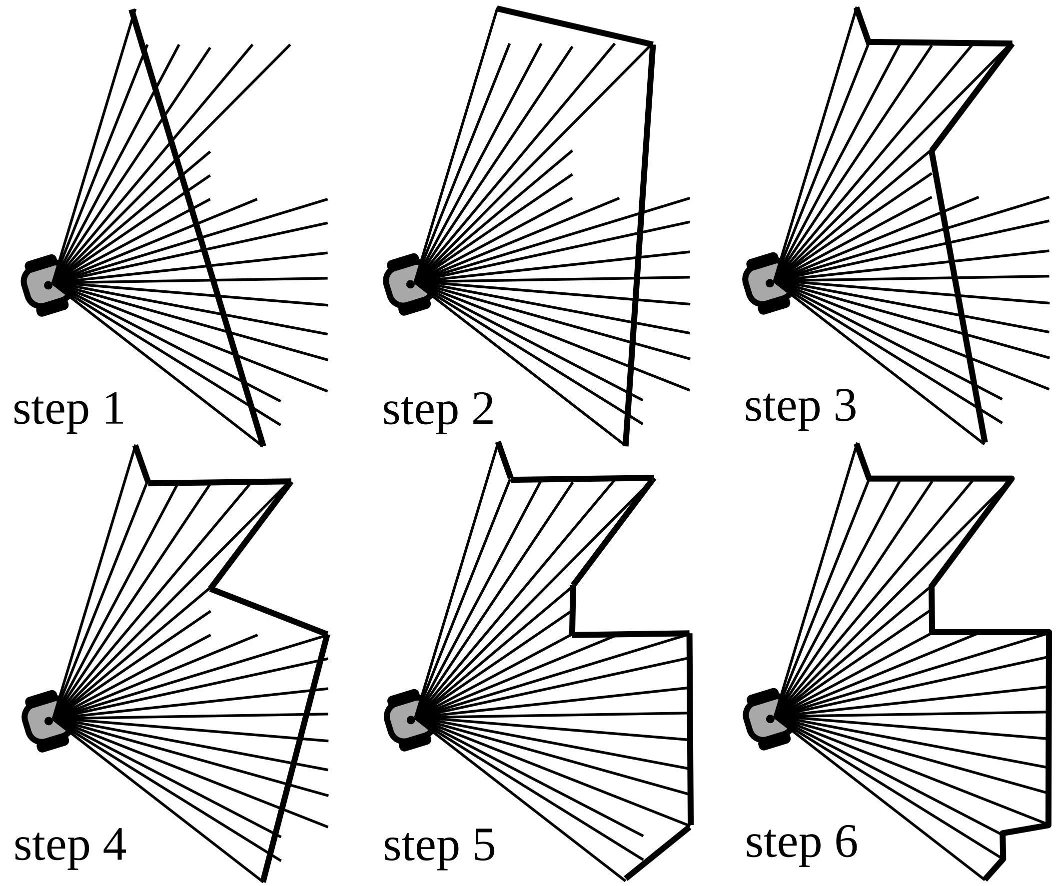
- The 9 possible pairings of (0,+,-) and (0,+,-) lead to 9 classes of local shape.
  - Spherical, cylindrical, saddle, etc.



Cylinder: The maximum curvature is oriented axially and the minimum is oriented longitudinally.

8.1.6.5 Segmentation: 1D Range Imagery

- Split and Merge is one of many good ideas applicable to this problem.
- Start with one line segment joining start and end.
- In each iteration, for each line segment, find the point of largest deviation and split there.
- Check if two lines should merge.
- Repeat.



Carnegie Mellon THE ROBOTICS INSTITUTE

Outline

- 8.1 Image Processing Operators and Algorithms
  - 8.1.1 Taxonomy of Computer Vision Algorithms
  - 8.1.2 High Pass Filtering Operators
  - 8.1.3 Low Pass Operators
  - 8.1.4 Matching Signals and Images
  - 8.1.5 Feature Detection
  - 8.1.6 Region Processing
  - <u>Summary</u>

Summary

- Image processing and signal processing are close cousins.
- Discrete mathematics provides the tools for filtering images to enhance and suppress high and low frequencies.
- Correlation measures similarity and can be used to find things.