Image Matching, Edges & Cross Correlation

Lecture #08

February 13th, 2013

Colorado State University

Similarity Transforms

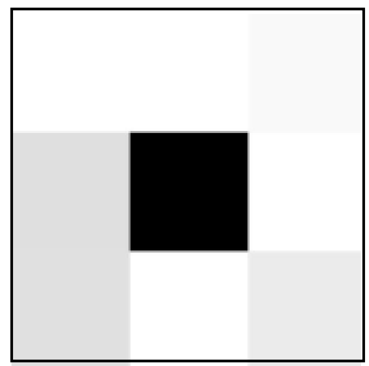
- So we can compensate for image translation using the shift theorem
  – O(n log(n)) correlation in Frequency Space
- Can we compensate for
  – image rotation? Yes
  – image scaling? Yes
  – perspective distortion? No
- How? By using the shift theorem again...

Rotation →Translation

- Our goal is to apply a coordinate transformation to the original spatial domain image such that a change in rotation becomes a change in translation.
- How? By converting from Cartesian Coordinates to Polar Coordinates.
  - Remember that the Polar Coordinates of a point in 2D are its distance from the origin d and the angle of the vector from the origin to the point ϕ

Polar Coordinates

Image (3x3)



<u>Cartesian</u>

[0,0]=255
[0,1]=250
[0,2]=224
[1,0]=225
[1,1]=0
[1,2]=255
[2,0]=249
[2,1]=255
[2,2]=235

<u>Polar</u>

[0,0]=0	[1,0]=255
[0,45]=0	[1,45]=224
[0,90]=0	[1,90]=250
[0,135]=0	[1,135]=255
[0,180]=0	[1,180]=225
[0,225]=0	[1,225]=249
[0,270]=0	[1,270]=255
[0,315]=0	[1,315]=235

Polar coordinates imply an image resampling - use bilinear interpolation

Polar Coordinates (cont.)

- Why convert into polar coordinates?
  - Because a cartesian rotation becomes a polar translation
    - 2nd coordinate -- angle -- advances…
  - Apply Fourier transform to polar image, find maximum shift
    - Shift in distance coordinate must be zero!
  - Shift implies rotation in cartesian space.

Scale →Shift

- What if one image is scaled relative to another?
- Convert image to LogPolar Coordinates
  - Just like Polar Coordinates, except that the distance coordinate is expressed by its logarithm
- Now a shift in the distance coordinate is equivalent to a scale change
- Note that the shift theorem finds shifts in both dimensions, so you can match images that are both rotated and scaled.

Limitations

- We can only match rotated and/or scaled image if we know the point they are rotated and/or scaled about!
- We can match translated images or rotated & scaled images, but not both.
- There is an informal, iterative way to handle small changes in translation, rotation & scale
  – Fix translation, adjust rotation & scale shift,
  – Fix rotation/scale, adjust translation
  – Repeat until peak in shift theorem reaches a threshold, or fails to improve

Gross Translation & Rotation

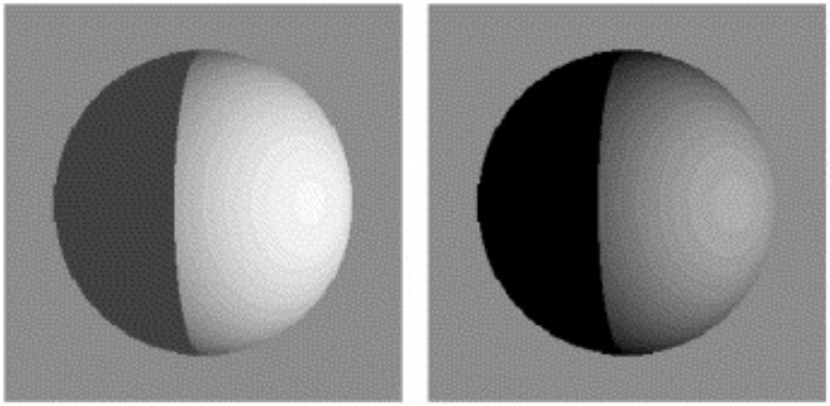
- The brute force approach to making correlation insensitive to gross translations and rotations is to generate R templates at R different angles.
  – Increase complexity by the factor R
  – Only accurate if R is fairly large
- Can one guess orientation and only consider one template per pixel location?
  – measure the orientation of a pixel?
  – What does it even mean?
- One of many ways to motivate <u>**Edges**</u>

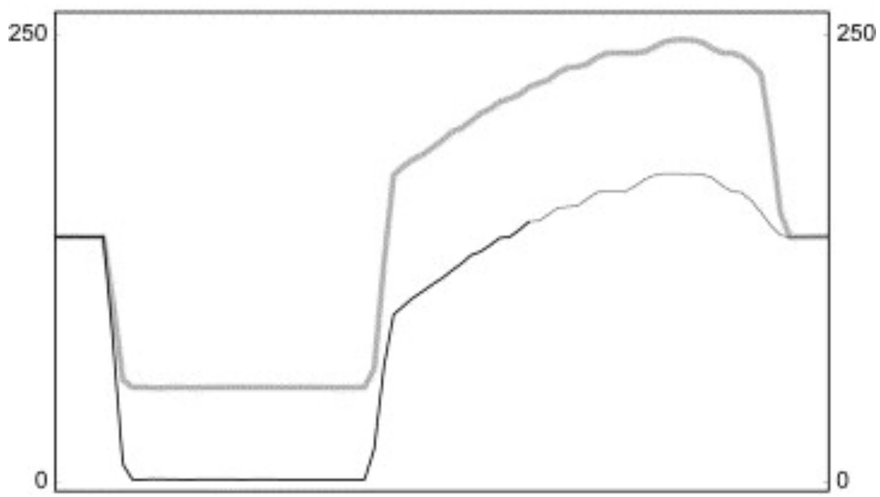
Image as Surface

- View the image as a 3D surface
  - For every (x,y) pixel location, the intensity can be thought of as the z (height) value.
    - Color images are 5D surfaces hard to think about.
    - Color can also be thought of as 3 3D surfaces
  - Pretend the surface is continuous

- Every point on the image surface has a direction of maximum change (remember your multivariate calculus?), and a magnitude of change in that direction

Surface in 1D





1D cross-section of simple image surface

Image from http://ars.els-cdn.com/content/image/1-s2.0-S1077314204001675-gr1.jpg

Image Edges

- To find direction and magnitude of change, compute the mag. in any 2 orthogonal directions and interpolate
  – Again, this assumes a continuous surface
- WLOG choose the X & Y directions:
  – dx(x,y) = I(x,y) - I(x-1,y)
  – dy(x,y) = I(x,y) - I(x,y-1)
- The edge magnitude and orientation is:

$$|\Delta| = \sqrt{dx^2 + dy^2} \qquad \theta = \cos^{-1}\left(\frac{dy}{|\Delta|}\right)$$

Estimating Edge Orientation

- Problem: images are not continuous surfaces
  - estimates of dx, dy based on grid sampling
  - note that if accurate,

$$I(x,y) - I(x+1,y) = I(x-1,y) - I(x,y)$$

  - estimating derivatives from two values is highly error prone.

Accurate Edge Estimation

- We want to compute a real-valued function
  - The partial derivatives dx & dy
- All we have to work with are samples at equidistant points
- So model the function in terms of its Taylor series expansion:

$$f(x+h) = f(x) + \frac{h^1}{1!} f'(x) + \frac{h^2}{2!} f''(x) + \cdots$$

Accurate (II)

- Look at equations for f(x+h) and f(x-h):

$$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2} f''(x) + \cdots \qquad (1)$$

$$f(x-h) = f(x) - hf'(x) + \frac{h^2}{2} f''(x) - \cdots \qquad (2)$$

subtract equation 2 from 1

$$f(x+h) - f(x-h) = 2hf'(x) + \cdots$$

and solve for f'

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} + \cdots$$

Accurate (III)

- So the best ±1 mask is [-1,0,1], from

$$\frac{d}{dx} f(x,y) = \frac{f(x+1,y) - f(x-1,y)}{2}$$

- As an exercise, the best ±2 mask is [1,-8,0,8,-1]

$$\frac{d}{dx} f(x,y) = \frac{-f(x+2,y) + 8f(x+1,y) - 8f(x-1,y) + f(x-2,y)}{12}$$

Stable Edges (III)

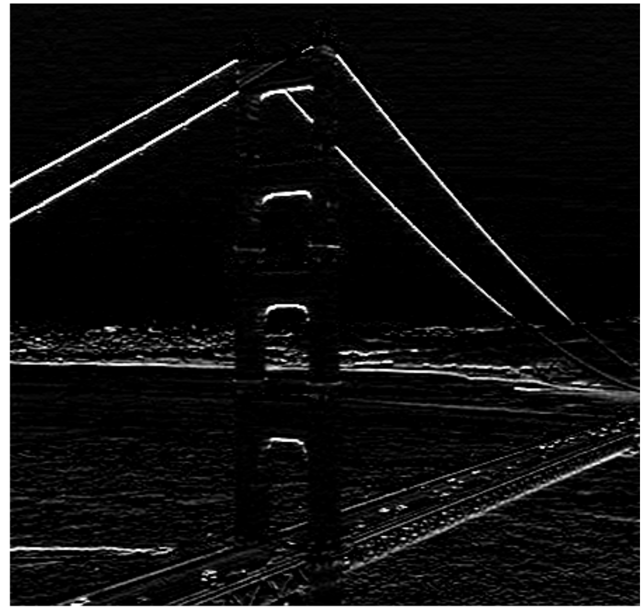
- Of course, pixels are still noisy and pixels are related to adjacent rows.
- The Sobel Edge Masks

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

Dx Dy

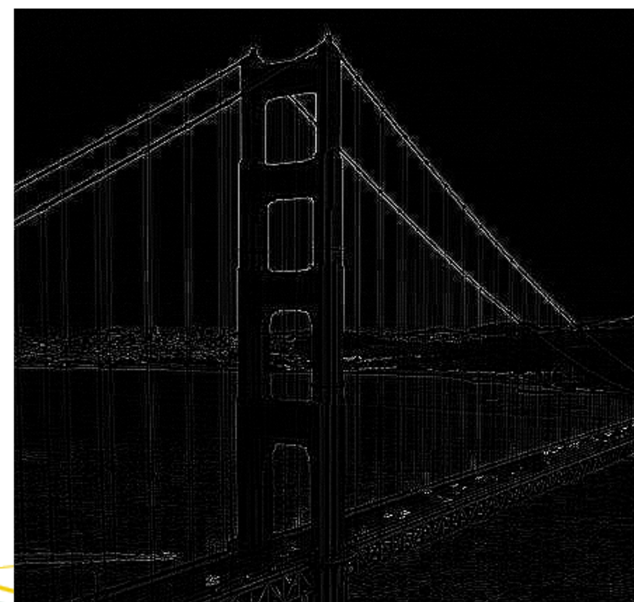
Sobel Explanation

- In any row (dx) or column (dy), this is a [-1,0,1] mask to estimate the derivative
- [1,2,1] weights approximate a $\sigma=1$ Gaussian.
  - Over-constrained fit of a plane to 9 points
  - Minimizes the sum-of-squared error
- Multiply result by ¼ to keep results < 255
- Multiply results by 1/8 to generate 8-bit response



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

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



$$\begin{vmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{vmatrix}$$

Rotation-Free Cross Correlation

- Make sure template is centered on an edge
  – Measure it's orientation
- For every pixel location:
  – compute edge orientation at pixel
  – rotate template until pixel edge matches the orientation of the template edge
    - rotate using bilinear interpolation
  – correlate template with image
- Makes correlation insensitive to rotation
  – *If edge direction estimates are accurate*
  – We will find ways to make them more accurate…