More Fourier Matching

CS 510 Lecture #17
3/20/02

Fourier Matching

- Up until Monday, we were matching images in the spatial domain
  - Why? If two images match, their frequencies and phases match, too.
  - If h(x) is the cross correlation of f(x) and g(x), then Fourier(h(x)) = F(x)G*(x)
    - F(x) = Fourier(f(x)), same with G
    - G*(x) is the complex conjugate of G(x)

Compensating for Shift

Go back to the shifting relation. The phase difference between two identical images shifted by (x_0,y_0) is:

$$e^{i2\pi(ux_0, vy_0)}$$

Why is this interesting? Because:

$$\frac{F_1(u,v)F_2^*(u,v)}{\left|F_1(u,v)F_2^*(u,v)\right|} = e^{i2\pi(ux_0, vy_0)}$$

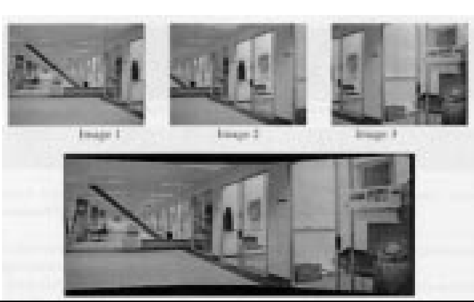
Shift (cont.)

- To compute the best shifted match, we compute the shift (x_0, y_0) for each frequency (u,v):

$$\frac{F_1(u,v)F_2^*(u,v)}{\left|F_1(u,v)F_2^*(u,v)\right|} = e^{i2\pi(ux_0, vy_0)}$$

- We sum the magnitudes of $|F_1F_2^*|$ at every shift (x_0, y_0), and find the peak; this corresponds to the shift with the highest correlation.

Example



Back to Correlation

- I said that if two images are similar, then phase correlation says that the maximum phase value corresponds to the best shift x_0, y_0.
- What is the relationship between the height of the peak and the correlation score of the images under that shift?
- Parseval's theorem says that: $\int_{-\infty}^{\infty} h^2(x)dx = \int_{-\infty}^{\infty} \left|H(f)\right|^2 df$
- So normalizing source image normalizes frequency space, and the height of the peak is the correlation score

Similarity Transforms

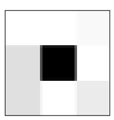
- So we can compensate for image translation using the shift theorem
- Can we compensate for
  - image rotation? *Yes*
  - image scaling? *Yes*
  - perspective distortion? *No*
- How? By using the shift theorem again...

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)



Cartesian	Polar	
[0,0]=255	[0,0]=0	[1,0]=255
[0,1]=250	[0,45]=0	[1,45]=224
[0,2]=224	[0,90]=0	[1,90]=250
[1,0]=225	[0,135]=0	[1,135]=255
[1,1]=0	[0,180]=0	[1,180]=225
[1,2]=255	[0,225]=0	[1,225]=249
[2,0]=249	[0,270]=0	[1,270]=255
[2,1]=255	[0,315]=0	[1,315]=235
[2,2]=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 technique for handling small changes in translation, rotation & scale
  - Assume no translation, find rotation & scale shift, apply it
  - Assume no rotation/scale, find translation, apply it
  - Repeat until peak in shift theorem reaches a threshold, or fails to improve