# Image Segmentation CS828 Spring '12

Angjoo Kanazawa

February 6, 2012

## 1 January 25th 2012 - Lecture 1

**Definition**: Segmentation (for this class):

- About low level vision in general
- Requires a lot of knowledge about th eworld, high level understanding, quite challenging.
- So we're going to focus on simpler segmentation that doesn't require that much knowledge about the world: Uniform surfaces, smooth shape. Still there will be varietion in intensity.
- Want to find uniform region in things (texture, color, motion, smoothness), not necessarily world property. Removed from true segmentation of objects but still useful.
- Image is an 2D geometric structure. Segmentation is clustering that takes advantage of this structure. Based on the assumption that near-by pixels have the same intensity.
-

We're going to look at

1. Diffusion
2. Anisotropic diffusion
3. Graph based algorithms: message passing, thinking of an image as a graph, every pixel is a node in a graph, edges to neighbors → Markov Random FIeld. Gives us a probablistic way to express the state of a node in relation to its neighbors. Usually NP-hard, but graph-cut and belief propagation algorithms still work. The biggest issue is when the number of labels is big.
4. Conditional Random Fields, a general version of MRF
5. Normalized Cut: form a graph
6. Wavelets

| | | | |
|---|---|---|---|
| **Math** | Fourier transforms | Convolution | Diffusion |
| | Wavelets | Level sets | Riemannian Geometry |

| | | |
|---|---|---|
| **Current Research** | Bilateral filtering (by Morel) | Texture Segmentation |
| | Cosegmentation | Affinity propagation |

**Workload**

1. Reports (6 out of 8 papers):Be critical when reading papers, even if the paper is good, what is the really important. Learn to recognize, have a taste. (10%)

2. Presentations: 3 presentations per day, 15 min per paper 10 min each to discuss paper (15%)

3. a take home midterm, Final all on lecture material (50%)

4. Problem set/Project (25%)

# 2 January 30th - Lecture 2

## 2.1 Perceptual Grouping

- Putting pieces to preceive as a whole.
- Depends on the prior knowledge/statistics about the world.

**History**

- Behaviorists dominated in early 20th century, wanted to make psychology scientific, focused on quantifyiable things.
- Rejected anything introspective or mind building internal representations.
- AI, computers, chomsky killed behaviorists.
- Gestalt movement claimed visual system perceived world as a objects and surfaces, as a whole and not as raw atomic stimulus/intensities.

**Classical principles/cues**

- Knowing the role of edges is critical to how we perceive an image
- Similarity, Good continuation, Common Form, Connectivity, Symmetry (seems to jump out), Convexity, Closure, Common Fate, Paraallelism, Collinearity
- convexity beats symmetry? Connectivity also beats symmetry?

**Theories**

- We perceive shapes that are "good form": smooth curves,, pretty abstract
- Bayesian: organizaton that's most likely to be true. Not computationally friendly. Rather than checking all possible options, maybe we look for a certain small set of possiblities. Still doesn't explain everything
-

# 3 February 1st - Lecture 3: Fourier Transform

## 3.1 Mathematical representation

a point in a $\mathbf{R^2}$ can be represented in a coordinate. If $p = (7, 3)$, we really mean $p = 7(1, 0) + 3(0, 1)$. *Any* point can be represented by a linear combination of two vectors. The basis vectors are:

1. Span the entire space: every point in the space can be written by linear combinations of these vectors.
2. Orthogonal: If not, moving in one direction will mean you'll be moving in the another direction
3. Unit: if not, the distance from the origin will not be constant.

We can compute the bases by

1. Linear Projection (inner product with each basis)

$$p = (p \cdot (1, 0))(1, 0) + (p \cdot (0, 1))(0, 1) \tag{1}$$

2. Magnitude of a point $||p||^2 = x^2 + y^2$ v

## 3.2 Functions in $\mathbf{R}^1$

The domain of the function is $[0, 2\pi]$, and we'll deal with functions in $\mathbf{R}^1$.

**Def:** a delta function:

$$\delta_s(t) = \begin{cases} 0 & s \neq t \\ \infty & s = t \end{cases}, \int_0^{2\pi} \delta_s(t) dt = 1$$

We'll write functions by using delta functions as a basis.

In infinite dimensions,

$$f(t) = \int f_s(\delta_s(t) ds$$

is the same as (**??**) but in infinite dimensions. Tw basis are orthogonal if their inner products are 0, in infinite dimensions, this is taking the integral. So delta functions are orthogonal.

This is a bad representation in some ways. It doesn't converge to the right representation (the function) quickly: using countable number of delta functions will not be a good representation of the function because it will only be correct in those places. We also need a lot of co-efficients.

**Differen Representation** Divide the interval $[0, 2\pi]$ into short $k$ intervals with width $\frac{2\pi}{k}$. Use a rectangle in a interval as basis. They are orthogonal, so we can scale these rectangles and set it to a height that is equal to the average of the function in that interval. We have a piece-wise representation of a function using a finite basis. As $k \to \infty$, the approximation gets better. The *Reimann integral.* Here, we're stuck with a cetain level of accuracy as we fix $k$.

To get an arbitrary accuracy, we can reuse basis from multiple $k$s. i.e. if we divide the interval in 2, then 4, etc, then we'll get many rectangles or infinite bases that are *not* orthogonal, but can represent any function with finite pieces.

Functions are uncountable, but we're trying to represent it as a countable set of bases. But this is okay because we enforce the functions to be continuous.

### 3.3 Fourier Series

The basis elements:

- Height of $\sqrt{\frac{1}{2\pi}}$
- $\frac{\cos(t)}{\sqrt{n}}$ all are multiplied by a constant so when integrated it is 1.
- $\frac{\sin(t)}{\sqrt{n}}$
- $\cos(2t)$, $\sin(2t)$

They are unit vectors (normalized) and they are orthonormal i.e. $\int \sin(t)\cos(t)dt = 0$. But better, draw them around $\pi$. sin is symmetric around $\pi$, cos is negative symmetric. So if they are multiplied together, the signs are different so they cancel and gives you 0.

Now, we can write any function as an infinite sum of these basis elements:

$$f(t) = a_0 + \sum_{k=1}^{\infty} a_k \cos kt + \sum_{k=1}^{\infty} b_k \sin kt \tag{2}$$

If the sums were finite upto $N$, then $\lim N \to \infty ||f(t)|| = 0$. This is a better representation then the delta functions because if we use enough co-efficients we will get really good approximation to the function.

$\cos^{2n}(t/2)$: Look at waht $\cos(t/2)$ look like, then raise it to a higher power. Really quicly, it will peak and look more like a delta function. By adding a constant in, $\cos(t/2 + a)$, we can shift the peaks.

Becasue we know that we can approximate any function with infinite delta functions, this means we can also do it with these basis. There are couple of identities by trigonometry to write higher power trig functions as a single power functions. i.e. trig functions with different frequences: $sin^2(t/2) = \frac{1-\cos(t)}{2}$, $sin^2(t) = \frac{1-\cos 2t}{2}$

**Intuition:** In practice, functions are smooth and with very small coefficients we can get a very good approximations.

#### Notation

$$\cos kt + i \sin kt = e^{ikt} \tag{3}$$

There are simple ways of computing these coefficients $a_k, b_k$. If we want $a_k$, we **take the inner product** of the function and $\cos kt$ i.e. $\int f(t) \cos ktdt$.

**Complex case** Given

$$c_k = \langle f, e^{ikt}, \rangle = \langle f, \cos kt \rangle + i \langle f, \sin kt \rangle,$$

$c_{-k} = \langle f, e^{i-kt}, \rangle = \langle f, \cos kt \rangle - i \langle f, \sin kt \rangle$

Then

$$c_k e^{ikt} + c_{-k} e^{-ikt} = a_k \cos kt + b_k \sin kt \tag{4}$$

We get back to the fourier representation.

Following from $a \sin t + b \cos t = c \cos(t + k)$, $k$ is the phase, or the shift of functions.

**Parsevaal's Theorem:** Same as the pythagorean theorem:

$$\int f^2(t) dt = \frac{\pi}{2} a_0^2 + \pi \sum (a_k^2 + b_k^2)$$

This is good to use to measure how good our approximation is. So We can do

$$||(\int f(t) - a_0 - \sum_{k=1}^{N} a_k \cos kt - \sum_{k=1}^{N} b_k)^2|| = ||(\sum_{N+1}^{\infty} a_k \cos kt - \sum_{N+1}^{\infty} b_k)^2||$$

## 3.4 Fourier Transform

Let $f(t)$ is periodic going from $[0, 2\pi l]$. Then, we can represent $f(t)$ by

$$f(t) = \sum c_k e^{ikt/l}$$

(By dividing with $l$, we're stretching the basis element in $[0, 2\pi]$.) As $l \to \infty$, this gives us every possible fraction, all of $\mathbf{Q}$. Which mean we write this as:

$$f(t) = \int_{-\infty}^{\infty} F(k) e^{ikt} dk \tag{5}$$

Remember: $e^{ikt}$ carries the orthonromal basis, now extending to all of $\mathbf{R}$, this means the coefficients are now in the $\infty$ domain so we write coefficients as $F(k)$, and call this the **Fourier transform** of $f(k)$.

(**??**) is the approximation of $f(t)$, the inverse operation to get the fourier transform is;

$$F(k) = \int_{-\infty}^{\infty} f(k) e^{-ikt} dk \tag{6}$$

$e^{-ikt}$ is negative because it's the complex conjugate of $e^{ikt}$, (square it we multiply it with the complex conjugate.)

# 4 February 6th - Lecture 4: Smoothing & Convolution

Why do we *smooth* images? It's a way of passing information around, also it connects it more to segmentation (looking for a uniform property). When we smooth, we can take things that are similar and make them more similar. It also allows us to represent images in multiple scales, it helps us get rid of fine details, giving us coarser representations of an image. That is we want to remove high frequency portion and analyze the low frequency part.

Smoothing can be done by *convolution*.

In vision we always assume vision. Given a noisy input, the true intensity + noise, say $P_i = 100 + n_i$, smoothing takes the average of all pixels, we'll have

$$\begin{aligned} &= \frac{1}{M}\sum P_i \\ &= \frac{1}{M}\sum 100 + n_i \\ &= 100 + \frac{1}{M}\sum n_i \end{aligned}$$

A simple example of smoothing, the average of a lot of random variables makes the std of noise smaller.

## 4.1 1-D image

Think of 1-D images as a function: $f(t)$. We want to replace a pixel by the average of its neighbors. We write this as:

$$h(t) = \frac{1}{2\delta}\int_{t-\delta}^{t+\delta} f(t^{'})dt^{'} \tag{7}$$

(Where $t^{'}$ is just another points, not derivatives)

Let us define,

$$g(t) = \begin{cases} \frac{1}{2\delta} & \text{for } -\delta \leqslant t \leqslant \delta \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

Note that $g(t)$ sums to 1. (Like a pdf of $U(-\delta, \delta)$.)

Now, we can write (7) as

$$h(t) = \int_{-\infty}^{\infty} f(t-u)g(u)du \tag{9}$$

It's as if we take the function $g$ and winding it up so that it's centered around $t$ and taking the inner product to get $h$. It flips, the left side of the filter is applied to the right side of point $t$. The resulted $h$ is just shifting $g$ at everypoint and taking the inner product.

We write this as

$$h(t) = g(t) * f(t) \tag{10}$$

It's natural to take the weighted average of your neighbors, because it's more likely that people around you have more information. So we use a gaussian filter.

## 4.2 Convolution

Two properties:

1. Linear: $g * kf = k(g * f)$, $g * (f_1 + f_2) = g * f_1 + g * f_2$
2. Shit-invariant: Take my function and translate, then convolve with a filter is same as convolving with a filter and shifting it. i.e. if $f'(t) = f(t + k)$, $h = g * f$, and $h^{'} = g * f^{'}$, then $h^{'} = h(t + k)$

**Convolution Theorem** Given $F$ as the fourier transform of $f$, ($F$ is the function of frequency), and the same for $G$, $g$, and $H$, $h$.

$$f * g = h \Leftrightarrow FG = H \tag{11}$$

*Proof*: $f * g = \int f(t-u)g(u)du$, call this $h$. To take the fourier transform of this, we take the inner product of $h$ and $e^{-itw}$. So

$$H(w) = \int e^{-itw}(\int f(t-u)g(u)du)dt$$

Define $v = t - u$. Now,

$$\begin{aligned} H(w) &= \int e^{-itw}(\int f(t-u)g(u)du)dt \\ &= \int e^{-i(v+u)w}(\int f(v)g(u)du)dv \\ &= \int e^{-iuw}g(u)du \int e^{-ivw}f(v)dv \\ &= GF \end{aligned}$$

The sines and cosines are eigenvectors of functions because when you convolve it with any filter it just scales it.

The narrower the gaussian, the broader the fourier transform, The broader my gaussian, the narrower my fourier transform. So the lower frequency part gets preserved and the higher frequency (the edge of gaussian) gets reduced more.

*Intuition*: If the gaussian is so sharp that it's like a delta function, it'll only scale the function at that point $t$. Then, the fourier transform of a delta function is uniformly 1 at all frequencies. Because convolving with a delta function doesn't change anything, so $f * \delta = f$, bug $F * G = H = F$, so $G$ is a uniformly 1 that doesn't change anything.

Similarly, if the gaussian is so broad that it's like a uniform function, then the fourier transform is like a delta function.

*example* A sinc function: $G(w) = \int_{-\delta}^{\delta} \frac{1}{2\delta} e^{-iwt}dt = \frac{2\sin(\delta w}{w}$. Plot it, bad fourier transform because the high frequency components go in and out. Compared, the gaussian filter provides us a very good fourier transform.

**Why remove higher frequency components?** If we assume the noise is i.i.d., we can show that the fourier transform of the noise is uniform. i.i.d. noise has equal energy. Bc this noise has the same energy everywhere, it's called the *white noise*. If we think of our image as some smooth pixels with a white noise. Images tend to have much more low frequencies than high frequencies. Noise is equal in low and high frequency, so if you reduce the high frequency components, it significantly reduces the noise.

**Band-pass filter**: Looks like $U(a, b)$, it perfectly preserves the low frequency component. It's fourier transform is the sinc function. So there's limitation to use these perfect filters.

**High-pass filter**: inverse of $U(a, b)$.

Fourier series:

$$f(t) = a_0 + \sum a_k \cos(kt) + \sum b_k \sin(kt)$$

($K$ is the frequency) The derivative:

$$f'(t) = \sum -a_k \sin(kt) + \sum b_k \cos(kt)$$

This is also a fourier series, *taking the derivative has the effect of scaling the coefficients by the frequency.* As frequency gets higher, it amplifies the coefficients.

This is why it's dangerous to take the derivative of a noisy image, because the derivative amplifies the high frequency components with a lot of noise.

Taking the derivative is like convolution.

**Gaussian Filter** For any function, the more spatially localized (peaked) it is, the broader it is in frequency. Vice versa.