CS664 Computer Vision

2. Filtering

Dan Huttenlocher

Announcements

- No class on Tuesday (Jan 29)
- Meet in 315 Upson starting Thursday (Jan 31)
- Reading: Edges (and filtering) handout
- Handouts and lecture slides on web at www.cs.cornell.edu/courses/cs664/

What's An Image?

- Think of as function f from R^2 to R, mapping (x,y) location to intensity
  - Or vector valued for color image
- Only defined over finite domain and range

Digital Images

- Generally work with discrete images
  - Sample 2D space on a regular grid
  - Quantize each value
    - Round to nearest integer
- Represented as matrix of integer values

$j \rightarrow$

$i \downarrow$

62	79	23	119	120	105	4	0
10	10	9	62	12	78	34	0
10	58	197	46	46	0	0	48
176	135	5	188	191	68	0	49
2	1	1	29	26	37	0	77
0	89	144	147	187	102	62	208
255	252	0	166	123	62	0	31
166	63	127	17	1	0	99	30

Filtering

- Compute new image by combining pixel values from an input image
  - Generally spatially local over some region of support
  - E.g., mitigating effects of noise/error

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

3x3 (kxk) Mean Filter Example

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

?	?	?	?	?	?	?	?	?	?
?	0	10	20	30	30	30	20	10	?
?	0	20	40	60	60	60	40	20	?
?	0	30	60	90	90	90	60	30	?
?	0	30	50	80	80	90	60	30	?
?	0	30	50	80	80	90	60	30	?
?	0	20	30	50	50	60	40	20	?
?	10	20	30	30	30	30	20	10	?
?	10	10	10	0	0	0	0	0	?
?	?	?	?	?	?	?	?	?	?

$$G[i,j] = \frac{1}{(2k+1)^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} F[i+u, j+v]$$

Border Pixels

- Border cases need to be handled somehow
  - Produce smaller image by summing only when entire w by h window fits inside image
  - Sum only value inside image but produce full size image
    - In effect summing zeroes outside image
  - Assume value outside image some non-zero value
    - E.g., reflected copy of the image
- No right answer, reflection often least bad

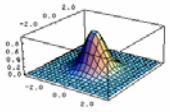
Cross Correlation Filtering

- Generalize to weight at each location in window $G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i+u,j+v]$
- Cross correlation written as G = H⊗F
  - Not always consistent sometimes written as ⋆
- H is called kernel, filter or mask
  - What sum to?
- Mean filtering – uniform kernel values
- Implementation note: use H[u+k,v+k]
  - Non-negative array indices

Gaussian Filter

- Gaussian in two-dimensions

$$h(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{\sigma^2}}$$



- Weights center pixels more
- Falls off smoothly
- Integrates to 1
- Larger σ produces more equal weights (blurs more)
- Normal distribution

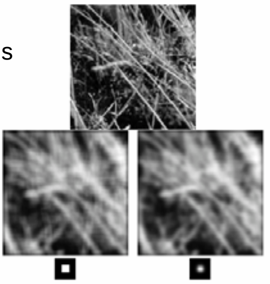
$\frac{1}{16}$

1	2	1
2	4	2
1	2	1

$H[u,v]$

Gaussian Versus Mean Filter

- Mean filter blurs but sharp changes remain as well
  - “Blocky”
- Gaussian not blocky looking
- Same area masks
  - But Gaussian small at borders



Cross Correlation Examples

H

0	0	0
0	1	0
0	0	0

⊗

F

0	0	0	0	0
0	a	b	c	0
0	d	e	f	0
0	g	h	i	0
0	0	0	0	0

=

G

?	?	?	?	?
?	a	b	c	?
?	d	e	f	?
?	g	h	i	?
?	?	?	?	?

H

a	b	c
d	e	f
g	h	i

⊗

F

0	0	0	0	0
0	0	0	0	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0

=

G

?	?	?	?	?
?	i	h	g	?
?	f	e	d	?
?	c	b	a	?
?	?	?	?	?

Convolution

- Closely related operation that “flips” indices

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i-u,j-v]$$

- Written as G = H⋆F
  - Again, notation not always consistent
- Note ⋆ and ⊗ same when H or F symmetric
  - Generally termed isotropic
- Convolution has nice properties
  - Commutative: A⋆B=B⋆A
  - Associative: A⋆(B⋆C)=(A⋆B)⋆C
  - Distributive: A⋆(B+C)=(A⋆B)+(A⋆C)

Convolution Examples

0	0	0
0	1	0
0	0	0

H

$\star$

0	0	0	0	0
0	a	b	c	0
0	d	e	f	0
0	g	h	i	0
0	0	0	0	0

F

=

?	?	?	?	?
?	a	b	c	?
?	d	e	f	?
?	g	h	i	?
?	?	?	?	?

G

a	b	c
d	e	f
g	h	i

H

$\star$

0	0	0	0	0
0	0	0	0	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0

F

=

?	?	?	?	?
?	a	b	c	?
?	d	e	f	?
?	g	h	i	?
?	?	?	?	?

G

Identity for Convolution

- Unit impulse: one at origin, zero elsewhere
- Suggests why simple averaging produces "blocky" results
  - Consider a=b=… =i=K

a	b	c
d	e	f
g	h	i

$\star$

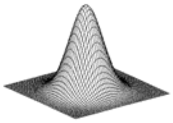
0	0	0	0	0
0	0	0	0	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0

=

?	?	?	?	?
?	a	b	c	?
?	d	e	f	?
?	g	h	i	?
?	?	?	?	?

Efficient Gaussian Smoothing

- The 2D Gaussian is decomposable into separate 1D convolutions in x and y
- First note that product of two one-dimensional Gaussians



$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$

$$\frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-1}{2}\left(\frac{x^2}{\sigma^2}\right)} \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-1}{2}\left(\frac{y^2}{\sigma^2}\right)}$$

- Can view as product of two 1d vectors
  - Column times row vectors, each 1d Gaussian

Expressing as 1D Convolutions

- Use unit impulse as a notational trick
  - Continuous case: δ(x) = ∞ when x is 0, else 0
  - Discrete case: δ[x] = 1 when x is 0, else 0
  - $f \star \delta = f$
- $h_\sigma = h_{\sigma x} \star h_{\sigma y}$

$$\frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-1}{2}\left(\frac{x^2}{\sigma^2}\right)} \delta(y)$$

$$\frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-1}{2}\left(\frac{y^2}{\sigma^2}\right)} \delta(x)$$

- $h_\sigma \star I = (h_{\sigma x} \star h_{\sigma y}) \star I = h_{\sigma x} \star (h_{\sigma y} \star I)$
  - Two 1D convolutions, don't sum all the zeroes

2D Gaussian as 1D Convolutions

0	0	0	0	0
0	16	16	16	0
0	16	16	16	0
0	16	16	16	0
0	0	0	0	0

$\star$

1/4
1/2
1/4

=

0	4	4	4	0
0	12	12	12	0
0	16	16	16	0
0	12	12	12	0
0	4	4	4	0

$\star$

1/4	1/2	1/4

=

0	0	0	0	0
0	16	16	16	0
0	16	16	16	0
0	16	16	16	0
0	0	0	0	0

$\star$

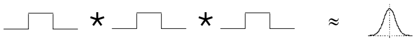
1/16	1/8	1/16
1/8	1/4	1/8
1/16	1/8	1/16

=

1	3	4	3	1
3	9	12	9	3
4	12	16	12	4
3	9	12	9	3
1	3	4	3	1

Fast 1D Gaussian Convolution

- Repeated convolution of box filters approximates a Gaussian
  - Application of central limit theorem, convolution of pdf's tends towards normal distr.



1 1 1	$\star$	1	1	1	=	0	0	1	2	3	2	1	0	0	
	$\star$	1	1	1	=	0	1	3	6	7	6	3	1	0	
	$\star$	1	1	1	=	1	4	10	16	19	16	10	4	1	

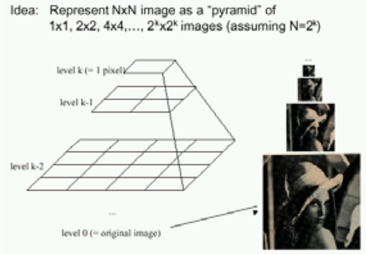
Good Approximation to Gaussian

- Convolution of 4 unit height box filters of different widths yields low error
  - Wells, PAMI Mar 1986
- Simply apply each box filter separately
  - Also separate horizontal and vertical passes
  - Each box filter constant time per pixel
    - Running sum
- For Gaussian of given σ
  - Choose widths w_i such that $\Sigma_i (w_i^2-1)/12 \approx \sigma^2$
- In practice faster than explicit G_σ for $\sigma \approx 2$

Gaussian Filter and Sub-Sample

Gaussian Pyramid

- Filter and subsample at each level
  - Uses only 1/3 more storage than original



Idea: Represent NxN image as a "pyramid" of 1x1, 2x2, 4x4,..., 2^kx2^k images (assuming N=2^k)

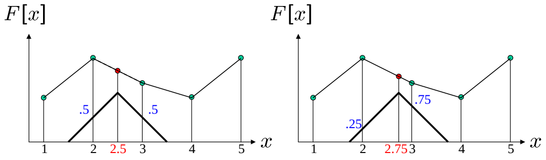
Sampling and Interpolation

- What if scale is not halving of the image
- What if want to upsample not downsample
- More general issue of constructing best samples on one grid given another grid
  - Often referred to as resampling
- If scaling down, first lowpass filter
- In both cases then map from one grid to another
  - Bilinear interpolation (2 by 2)
  - Bicubic interpolation (usually 4 by 4)

1D Linear Interpolation

- Compute intermediate values by weighted combination of neighboring values
  - Can view as convolution with "hat" on the original grid
    - E.g., equal spacing yields mask

.5	.5



Linear Interpolation by Convolution

- Implement by convolution with mask based on grid shift
  - If grid shifted to right by amount 0<a<1 then use mask [(1-a) a]
- For example grid shifted halfway between

.5	.5

$*$

0	2	4	4	8	6	6	4	2

1	3	4	6	7	6	5	3

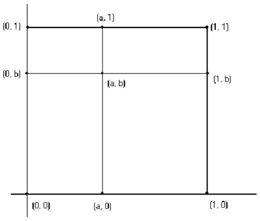
- Upsampled - combine

0	1	2	3	4	4	4	6	8	7	6	6	6	5	4	3	2

Bilinear Interpolation

- Value at (a,b) based on four neighbors

  $(1 - b)\ (1 - a)\ F_{0,0} + (1 - b)\ a\ F_{1,\ 0}$
  $+\ b\ (1 - a)\ F_{0,1} + b\ a\ F_{1,\ 0}$



Bilinear Interpolation by Convolution

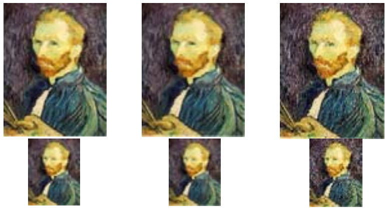
- Convolution with two-dimensional function

$h(x, y)$

- Perform two one-dimensional convolutions
  - Separable; simple to verify
  - New grid shifted down and to right by (a,b)
    - Where (as standard) origin of grid in upper left
  - Convolve horizontally with [(1-a) a)] then vertically with $[(1\text{-}b)\ b]^{\mathbf{T}}$

Comparing Sampling Methods

- Bilinear filter and subsample, Gaussian filter and subsample, straight subsample



1/2

1/4

- Bicubic generally works better