ISTA 352
Lecture 36
Image Analysis (III, mostly about filters)

Administrivia

• Schedule for rest of term
  – Today and Wednesday, finish up image processing
  – Friday, Leonard will be back (stereo perception)
  – Monday, review, quiz hints, course evaluation
    • Monday lecture will not be recorded
  – Next Wednesday, quiz 4
  – Monday, Dec 10, project presentations (optional)

Administrivia

• Quiz 3 grades were generally good
• Worst question was #3

Linear Filtering (2D)

Gray scale image (matrix)

Compute product of the weights in the mask with corresponding image ones, and sum up (dot product)

Result goes into a new image at the same place as the mask location

Then slide mask over one pixel and do it again (etc.)
3. Consider the filter on the left, and the image on the right. Compute the filter response at the image point emphasized. [Hint. The result is a single number] Show your work! (4 marks)

We center the filter so the 4 is over the 2, multiply the numbers that are matched, and sum up. Skipping the ones that are zero in the mask, we have five terms:

\((-1)(1)+(-1)(3)+(4)(2)+(-1)(2)+(-1)(9)\) = \(-1 - 3 + 8 - 2 - 9\) = \(-7\)

Or more quickly (or to check) by

\(4*2 - (1 + 3 + 2 + 9)\) = \(8 - 15\) = \(-7\)

**An Isotropic Gaussian Filter**

- The picture shows a smoothing kernel proportional to

\[
\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)
\]

(a reasonable model of a circularly symmetric fuzzy blob)

- The Gaussian filter is the standard way to smooth images

**Block Averaging**

**Gaussian**

**Image Scale**

- The difference between a tree in the distance, and its leaves up close, is one of image scale

- An arbitrary image will have multiple arbitrary scales

- Typically we analyze images at various scales

- A good way to think of rescaling an image is to smooth with a Gaussian and sub sample the results.
Linear Filtering as Functions

• Because the fundamental operation is a dot product, the filtering method just described is linear.

• Specifically, given the filtering operation defined by the mask $M$, denoted by $f_M()$, we have

$$f_M(a I_1 + b I_2) = a f_M(I_1) + b f_M(I_2)$$

• Exercises
  – Verify this is true for one of the linear function examples
  – Verify this is not always true for max() and median()

Correlation and Convolution

• Notice that the mask (kernel) is just another image.

• We denote the operation just described as the correlation between the signal $g()$ and the kernel $h()$

$$f(I) = g \odot h$$

• A similar, but more important operator is convolution

$$f(I) = g \otimes h$$

• Operationally, convolution is correlation by a mask (kernel) that is flipped over each axis (for 2D, X and Y)
  – If the mask is symmetric, then $\otimes$ and $\odot$ are the same.

Correlation and Convolution (II)

• Both convolution and correlation are associative

$$ (A \otimes B) \otimes C = A \otimes (B \otimes C) $$

$$ (A \odot B) \odot C = A \odot (B \odot C) $$

• This can save CPU time!

• Interestingly, convolution is commutative, but correlation is not commutative.
Correlation and Convolution (III)

- Other notations for convolution are * (1D) and ** (2D).
- In Matlab to implement convolution, use the function conv () (for 1D) and conv2() (for 2D).
- For correlation, see filter() and filter2().
- For images, see also imfilter (does both, depending on options, and has some extra options).

Correlation and Convolution (IV)

- One final complexity is what happens as the mask gets to the edge of the image.
  - One choice is to simply stop, but then your output is smaller than the original image.
    - In Matlab this is the flag 'valid' (this is not the default).
  - A second choice is to pad the image to make it big enough by a variety of means.
    - Just add zeros (this is the default for Matlab).
    - Repeat the pixel at the edge.
    - Consider the image as periodic pattern.
    - Periodic, but reflect the image.

2D convolution example (from MathWorks website)

For example, suppose the image is:

\[
A = \begin{bmatrix}
17 & 24 &  1 &  8 & 15 \\
23 &  5 &  7 & 14 & 16 \\
 4 &  6 & 13 & 20 & 22 \\
10 & 12 & 19 &  3 &  9 \\
11 & 18 & 25 & 2 &  4 \\
\end{bmatrix}
\]

and the convolution kernel is:

\[
h = \begin{bmatrix}
 8 & 1 & 6 \\
 3 & 5 & 7 \\
 4 & 9 & 2 \\
\end{bmatrix}
\]

To do the complete convolution, set A and h as above in Matlab, and do conv2(A,h,'same'). Try also conv2(A,h) — make sure you understand the difference!

Filters are templates

- Applying a filter at some point can be seen as taking a dot-product between the image and some vector.
- Filtering the image yields a set of dot products.
- Useful intuition:
  - Filters look like the effects they are intended to find.
  - Filters find effects that look like them.
  - Remember to flip your filter if you are implementing correlation using convolution.

Filters for steps in X (left) and Y (right). The step in X goes from high-to-low. Convolving with it finds high-to-low steps due to the flip.
<table>
<thead>
<tr>
<th>Filter for a dark spot</th>
<th>Filter for a dark bar at 135°</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image3.png" alt="Image" /></td>
<td><img src="image4.png" alt="Image" /></td>
</tr>
</tbody>
</table>