Computer Vision

ECE/CSE 576
Filters

Linda Shapiro
Professor of Computer Science & Engineering
Professor of Electrical & Computer Engineering
Let’s do something interesting already!!
Want to make image smaller
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
448x448 -> 64x64
IS THIS ALL THERE IS??
THERE IS A BETTER WAY!
LOOK AT HOW MUCH BETTER
How do?
How do? Averaging!
How do? Averaging!

“interpolation”

averaging
What is averaging?

AVG
What is averaging? A weighted sum

\[
\text{AVG} = \frac{\text{sum}(\text{pix})}{7 \times 7}
\]
What is averaging? A weighted sum

$$\text{sum} \left[ \frac{1}{49} \right]$$
Call this operation “convolution”

Note: multiplying an image section by a filter is actually called “correlation” and convolution involves inverting the filter first, but since our filters are generally symmetric, we call everything convolution. This is what all computer vision people do.
Convolutions on larger images
Kernel slides across image
Kernel slides across image

What happens at the edges of the images?

1. Zero padding: easiest but can give bad results
2. Duplicate the outermost row or column
3. Use wraparound
Convolutions on larger images
This is called box filter

Box filters
Box filters smooth image

\[
\frac{1}{49} \quad \ast 
\]

Box filters

\[
\frac{1}{N \times M} 
\]
Box filters smooth image

The figure shows a box filter represented by a grid of 1s and a central 0. The filter is applied to an image of a dog and a bicycle. The filter kernel is applied with a weight of 1/49, leading to a smooth image of the scene.
Now we resize our smoothed image
So much better!
Box filters have artifacts
Box filters have artifacts
We want a smoothly weighted kernel
Gaussians

\[ G(x) = \frac{1}{\sqrt{2\pi \sigma^2}} e^{-\frac{x^2}{2\sigma^2}} \]
2d Gaussian

\[ g(x, y) = \frac{1}{2\pi\sigma^2} \cdot e^{-\frac{x^2 + y^2}{2\sigma^2}} \]
Better smoothing with Gaussians
Better smoothing with Gaussians
Better smoothing with Gaussians

Box Filtered

Gaussian Filtered
Wow, so what was that convolution thing??

\[ q = a \times r + b \times s + c \times t + d \times u + e \times v + f \times w + g \times x + h \times y + i \times z \]
Wow, so what was that convolution thing??

\[ q = a \times r + b \times s + c \times t + d \times u + e \times v + f \times w + g \times x + h \times y + i \times z \]
Calculate it, go!
Calculate it, go!
Guess that kernel!
Highpass Kernel: finds edges
(applied to the graytone image!)

```
0  -1  0
-1  4  -1
0  -1  0
```

![Example of image processing with highpass kernel](image_url)
Guess that kernel!
Identity Kernel: Does nothing!
Guess that kernel!
Sharpen Kernel: sharpens!
(applied to all three bands)

Note: sharpen = highpass + identity!
Guess that kernel!
Emboss Kernel: stylin’ (applied to all three bands)
Guess those kernels!
Sobel Kernels: edges (applied to a graytone image and thresholded)
Sobel Kernels: edges and gradient!
Sobel Kernels: edges and gradient!

This visualization is showing the magnitude and direction of the gradient. We will talk further about this when we discuss edges.
And so much more!!
Assignment 2

Image resizing and filtering
First things first!

• First, you need to run `git pull` from inside your homeworks folder to get the latest changes from GitHub.

• Remember that you might need some of your code from the previous hw (e.g. `set_pixel`) for this hw as well. Have your code from hw1 in your `src` folder.

• Then run:
  – `make clean`
  – `make`
Assignment 2

Nearest Neighbor Interpolation and Resizing

• float nn_interpolate(image im, float x, float y, int c); in src/modify_image.c

• image nn_resize(image im, int w, int h);

Bilinear Interpolation and Resizing

• float bilinear_interpolate(image im, float x, float y, int c);

• image bilinear_resize(image im, int w, int h);
Assignment 2

Box Filter

• void l1_normalize(image im)
• image make_box_filter(int w)

Convolution

• image convolve_image(image im, image filter, int preserve)
• image make_highpass_filter()
• image make_sharpen_filter()
• image make_emboss_filter()
Assignment 2

Gaussian
• image make_gaussian_filter(float sigma)

Hybrid Images
• image add_image(image a, image b)
• image sub_image(image a, image b)
Assignment 2

Sobel Operator (next lecture)

• image make_gx_filter()
• image make_gy_filter()
• void feature_normalize(image im)
• image *sobel_image(image im)
• image colorize_sobel(image im)