Filtering

CSE P 576
Larry Zitnick (larryz@microsoft.com)

What do computers see?
Images can be viewed as a 2D function.

What do computers see?

Linear filtering = Applying a local function to the image using a sum of weighted neighboring pixels.

Image filtering

Such as blurring:

\[
g(x, y) = \sum_{x'} \sum_{y'} f(x + x', y + y') h(x', y')
\]

Image filtering
Image filtering

- Linear filters can have arbitrary weights.
- Typically they sum to 0 or 1, but not always.
- Weights may be positive or negative.
- Many filters aren’t linear (median filter.)

What does this filter do?

Gaussian filter

\[ G_\sigma(x, y) = \frac{1}{Z} e^{-\frac{(x^2 + y^2)}{2\sigma^2}} \]

What does real blur look like?

Gaussian vs. mean filters

First and second derivatives
First and second derivatives

What are these good for?

- Subtracting filters
  \[\text{Sharpen}(x,y) = f(x,y) - \alpha (f * \nabla^2 G_\sigma(x,y))\]

- Combining filters
  \[f * g * g' = f * h\]
  for some \(h\)

- Combining Gaussian filters
  \[f * G_\sigma * G_{\sigma'} = f * G_{\sigma''}\]
  \[\sigma'' = \sqrt{\sigma^2 + \sigma'^2}\]

More blur than either individually (but less than \(\sigma'' = \sigma + \sigma'\))
Separable filters

\[ G^\sigma = G^\sigma_x \ast G^\sigma_y \]

Compute Gaussian in horizontal direction, followed by the vertical direction. **Much faster!**

\[
G^\sigma_x(x, y) = \frac{1}{\sqrt{2\pi \sigma^2}} e^{-\frac{x^2}{2\sigma^2}}
\]

\[
G^\sigma_y(x, y) = \frac{1}{\sqrt{2\pi \sigma^2}} e^{-\frac{y^2}{2\sigma^2}}
\]

Not all filters are separable.

---

Sums of rectangular regions

How do we compute the sum of the pixels in the red box?

After some pre-computation, this can be done in constant time for any box.

This "trick" is commonly used for computing Haar wavelets (a fundamental building block of many object recognition approaches.)

---

The trick is to compute an "integral image." Every pixel is the sum of its neighbors to the upper left.

Sequentially compute using:

\[
I(x, y) = I(x-1, y) + I(x, y-1) - I(x-1, y-1)
\]
Spatially varying filters

Some filters vary spatially.

\[ \sum_{x'} \sum_{y'} G_{x'}(x', y') G_{y'}(f(x, y) - f(x + x', y + y')) = g(x, y) \]

Useful for deblurring.

Constant blur

Same Gaussian kernel everywhere.

Slides courtesy of Sylvian Paris

Bilateral filter

Maintains edges when blurring!

The kernel shape depends on the image content.

Slides courtesy of Sylvian Paris

Borders

What to do about image borders:

- black
- fixed
- periodic
- reflected
Sampling

CSE P 576
Larry Zitnick (larryz@microsoft.com)

Up-sampling

How do we compute the values of pixels at fractional positions?

Bilinear sampling:

\[
\begin{align*}
  f(x + a, y + b) &= (1 - a)(1 - b)f(x, y) + \\
  &\quad a(1 - b)f(x + 1, y) + \\
  &\quad (1 - a)b f(x, y + 1) + \\
  &\quad abf(x + 1, y + 1)
\end{align*}
\]

Bicubic sampling fits a higher order function using a larger area of support.
Down-sampling

If you do it incorrectly your images could look like this:

Check out Moire patterns on the web.

Solution

Filter before sampling, i.e. blur the image first.

With blur

Without blur

• **Aliasing** can arise when you sample a continuous signal or image
  - occurs when your sampling rate is not high enough to capture the amount of detail in your image
  - Can give you the wrong signal/image—an alias
  - formally, the image contains structure at different scales
    - called “frequencies” in the Fourier domain
  - the sampling rate must be high enough to capture the highest frequency in the image