Image Processing

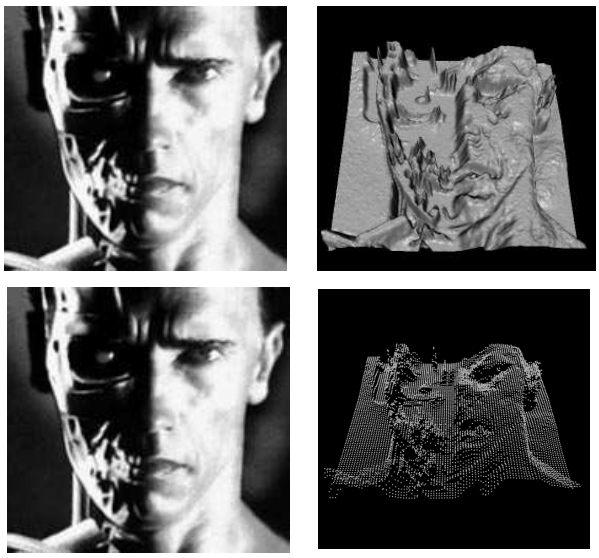
Forget Object Space

- So far, most of what we've done assumes we have some underlying abstract representation of what's in a scene.
- But there are many graphics techniques that operate only on images
- Image processing: operations that take images as input, produce images as output
- Is 3D graphics more important than image processing? *No way!*

Images

- In its most general form, an image is a function f from R^2 to R
  - $f(x, y)$ gives the intensity of a channel at position (x, y)
  - Realistically, we expect the image only to be defined over a rectangle, with a finite range:
    - $f: [a,b]\mathrm{x}[c,d] \rightarrow [0,1]$
  - A colour image is just three functions pasted together:
    - $F(x, y) = (f_r(x, y), f_g(x, y), f_b(x, y))$
- In computer graphics, we usually operate on digital (discrete) images
  - Quantize space into units (pixels)
  - Image is constant over each unit
  - A kind of step function
  - $f: \{0 \ldots m\text{-}1\}\mathrm{x}\{0 \ldots n\text{-}1\} \rightarrow [0,1]$
- An image processing operation typically defines a new image f' in terms of an existing image f

Images as Functions



Pixel-to-pixel Operations

- The simplest operations are those that transform each pixel in isolation
  - $f'(x, y) = g(f(x,y))$
- Example: threshold, RGB ➙ greyscale

Pixel Movement

- Some operations preserve intensities, but move pixels around in the image
  - $f'(x, y) = f(g(x,y), h(x,y))$
- Examples: many amusing warps of images

Noise

Original

Salt and pepper noise

Impulse noise

Gaussian noise

- Common types of noise:
  - Salt and pepper noise: contains random occurences of black and white pixels
  - Impulse noise: contains random occurences of white pixels
  - Gaussian noise: variations in intensity drawn from a Gaussian normal distribution

Noise Reduction

- How can we “smooth” away noise?

- Is there a more abstract way to represent this sort of operation? *O*f course there is*!*

Convolution

- Convolution is a fancy way to combine two functions.
  - Think of f as an image and g as a "smear" operator
  - g determines a new intensity at each point in terms of intensities of a neighbourhood of that point

$$\begin{aligned} h(x,y) &= f(x,y) * g(x,y) \\ &= \int_{-\infty}^{\infty} f(x',y')g(x-x',y-y')dx'dy' \end{aligned}$$

- The computation at each point (x,y) is like the computation of cone responses

Discrete Convolution

- Since digital images are like step functions, integration becomes summation. We can express convolution as a two-dimensional sum:

$$\begin{aligned} h[i,j] &= f[i,j] * g[i,j] \\ &= \sum_k \sum_l f[k,l]g[i-k,j-l] \end{aligned}$$

Convolution Representation

- Since f and g are defined over finite regions, we can write them out in two-dimensional arrays:

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

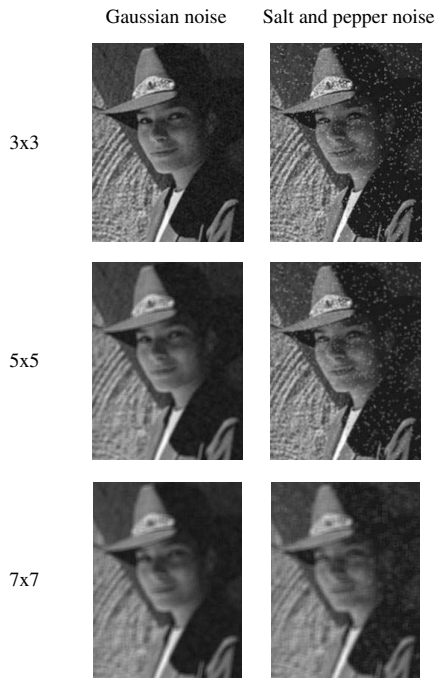
X .2	X 0	X .2
X 0	X .2	X 0
X .2	X 0	X .2

- *Note*: This is not matrix multiplication*!*

Mean Filters

- How can we represent our noise-reducing averaging filter as a convolution diagram?

Using Mean Filters



Gaussian Filters

- Gaussian filters weigh pixels based on their distance to the location of convolution.

$$g[i,j] = e^{-(i^2+j^2)/(2\sigma^2)}$$

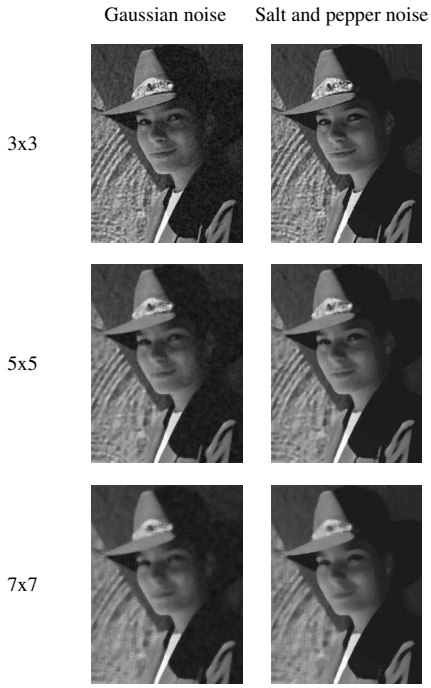
- This does a good job of blurring noise while preserving features of the image.

Using Gaussian Filters

Median Filters

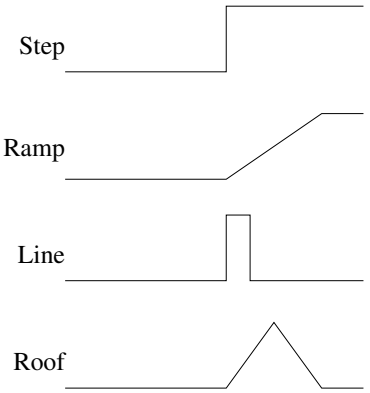
- A Median Filter operates over a kxk region by selecting the median intensity in the region.
- What advantage does a median filter have over a mean filter?
- Is a median filter a kind of convolution?

Using Median Filters



Edge Detection

- One of the most important uses of image processing is edge detection
  - Really easy for humans
  - Really difficult for computers
  - Fundamental in computer vision
  - Important in many graphics applications
- What defines an edge?



Gradient

- The gradient is the 2D equivalent of the derivative:

$$\nabla f(x,y) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$$

- Properties of the gradient
  - It's a vector
  - Points in the direction of maximum increase of f
  - Magnitude is rate of increase
- How can we approximate the gradient in a discrete image?

Edge Detection Algorithms

- Edge detection algorithms typically proceed in three or four steps:
  - Filtering: cut down on noise
  - Enhancement: amplify the difference between edges and non-edges
  - Detection: use a threshold operation
  - Localization (optional): estimate geometry of edges beyond pixels

Edge Enhancement

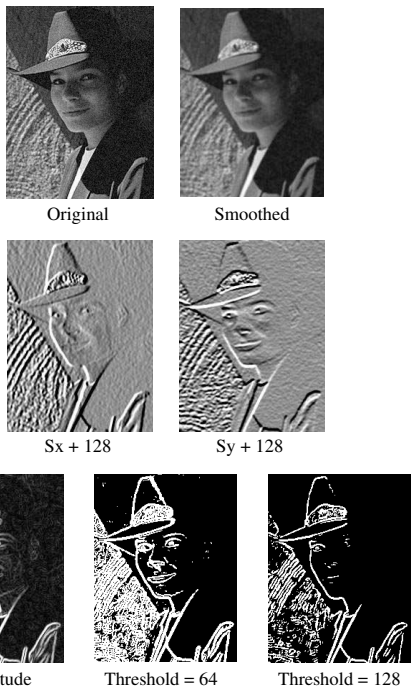
- A popular gradient magnitude computation is the Sobel operator:

$$s_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

$$s_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

- We can then compute the magnitude of the vector (s_x, s_y)

Using Sobel Operators



Second Derivative Operators

- The Sobel operator can produce thick edges. Ideally, we're looking for infinitely thin boundaries.
- An alternative approach is to look for local extrema in the first derivative: places where the change in the gradient is highest.
- We can find these by looking for zeroes in the second derivative
- Using similar reasoning as above, we can derive a Laplacian filter, which approximates the second derivative:

$$\Delta^2 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

- Zero values in the convoluted image correspond to extreme gradients, i.e. edges.

Summary

- Formal definitions of image and image processing
- Kinds of image processing: pixel-to-pixel, pixel movement, convolution, others
- Types of noise and strategies for noise reduction
- Definition of convolution and how discrete convolution works
- The effects of mean, median and Gaussian filtering
- How edge detection is done
- Gradients and discrete approximations