Computer Vision

CSC 548, Artificial Intelligence II

Vision

- Computer vision is concerned with the automatic extraction, analysis, and understanding of useful information from a single image or a sequence of images.
- Some applications of computer vision
  - Optical character recognition (OCR)
  - Medical imaging
  - Surveillance
  - 3D model building (photogrammetry)
  - Fingerprint recognition and biometrics

Digital Image Representation

- Consider a black and white image, also referred to as grayscale or gray level images.
- Each pixel (picture element) corresponds to achromatic or monochromatic light, that is, light devoid of color.
- Digital images are simply contiguous blocks of numbers in computer memory.
- Digital images are discrete functions that correspond to the average scene luminance as perceived by the camera over time.

Digital Image Representation

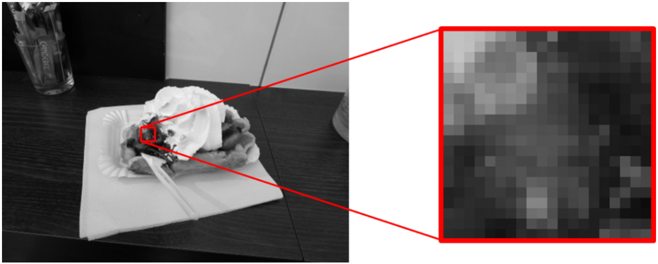

Figure 1: image

Digital Image Formats

- Properties that define an image format
  - Pixel resolution (e.g. 640×480 pixels)
  - Pixel bit depth (e.g. 8 bit signed, 16 bit signed, etc.)
  - Number of planes (e.g. 1 for grayscale, 3 for color)
  - Colorspace (e.g. RGB, YUV, etc.)
  - Pixel format (e.g. planar vs. packed)

Representing Color Images

- A pixel in a color image can be represented as a vector in a three dimensional color space.

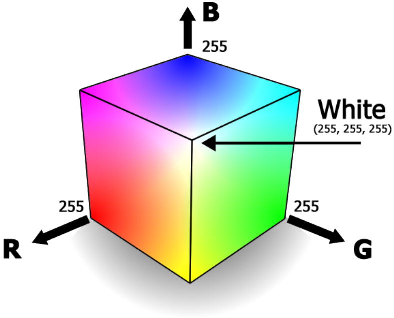

Figure 2: image

Pixel Transformations

- A general image processing *operator* is a function that takes one or more input images and produced an output image.
- In the continuous domain:

$$g(x) = h(f(x)) \text{ or } g(x) = h(f_0(x), \ldots, f_n(x))$$

  where x is in the D-dimensional *domain* of the functions (usually D = 2) and the functions f and g operate over some *range*, which can be scalar or vector-valued.

- For discrete images, the domain consists of a finite number of pixel locations, $x = (i, j)$ and we can write:

$$g(i, j) = h(f(i, j)$$

Pixel Transformations

- Two common point processes are multiplication and addition by a constant.

$$g(x) = af(x) + b$$

  where the parameters $a > 0$ and b are often called the *gain* and *bias* parameters.

- The gain is said to control *contrast* and the bias is said to control *brightness*.
- The gain and bias can also vary spatially

$$g(i,j) = a(i,y)f(i,j) + b(i,j)$$

Linear Filtering

- A *local operator* or *neighborhood operator* uses a collection of pixels in the vicinity of a given pixel to determine its final output.
- The most common type of of neighborhood operator is a *linear filter* where an output pixel's value is determined by a weighted sum of input pixel values.

$$g(i,j) = \sum_{k,l} f(i+k, j+l)h(k,l)$$

where the values in the weight *kernel* $h(k,l)$ are often called *filter coefficients*.

Convolution

- Defintion:

$$h(t) = f * g \equiv \int_{\infty}^{\infty} f(\tau)g(t-\tau)d\tau$$

- Breaking it down; for a given value of τ:
  - Take the mirror of g – $g(-\tau)$
  - Shift it by a given value of t – $g(t-\tau)$
  - Multiply by $f(\tau)$ – $f(\tau)g(t-\tau)$
  - Integrate from $[-\infty, \infty]$
  - Repeat for every value of t from $[-\infty, \infty]$

Discrete Convolution

- Definition

$$h(t) = f * g \equiv \sum_{\tau} f(\tau)g(t-\tau)d$$

- Integration is replaced by a summation
- The function g is a *kernel*.
- In image processing, the image and the kernel are discrete functions.

Image Convolution Example

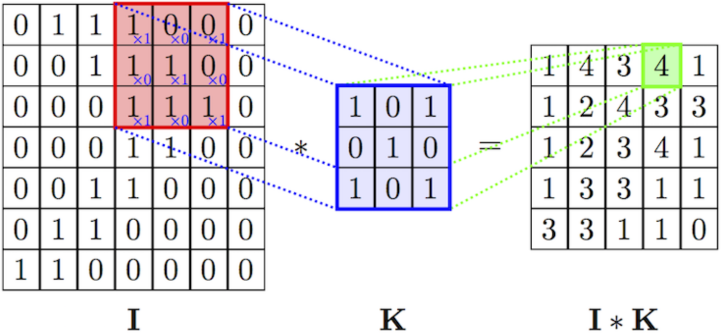

Figure 3: image

Border Effects

- Kernel convolution usually requires values from pixels outside of the image boundaries.
- To compensate for this, a number of *padding* modes can be used
  - Zero: set all pixels outside the source image to zero
  - Constant: set all pixels outside the source image to a specific value
  - Clamp: repeat edge pixels indefinitely
  - Wrap: the image is conceptually wrapped (tiled)
  - Mirror: reflect pixels across the image edge

Gaussian Kernel

- A Gaussian kernel is based on the Gaussian distribution and is used to smooth an image.
- The discrete 1D kernel coefficients can be generated from Pascal's triangle

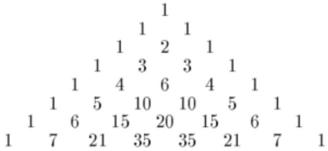

Figure 4: image

- The 2D coefficients can be obtained by convolving two 1D kernels

$$\begin{bmatrix} 1 & 2 & 1 \end{bmatrix} * \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} = \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Binary Images

- A binary image is a digital image that only has two possible values for each pixel.
- Binary images are often created by a *thresholding* operation

$$\theta(f, t) = \begin{cases} 1 & \text{iff} \geq t, \\ 0 & \text{otherwise} \end{cases}$$

Morphological Operators

- The most common binary image operations are called *morphological operations* because they change the *shape* of the binary objects.
- To perform a morphological operator, the binary image is convolved with a binary *structuring element*.
- The matrix dimensions specify the *size* of the structuring element.
- The pattern of ones and zeros specifices the *shape* of the structure element.

Morphological Operators

Let $c = f \otimes s$ be the integer-valued count of the number of ones inside each structuring element as it is scanned over the image and S be the size of the structuring element in pixels.

- Dilation: $\texttt{dilate}(f, s) = \theta(c, 1)$
- Erosion: $\texttt{erode}(f, s) = \theta(c, S)$
- Majority: $\texttt{maj}(f, s) = \theta(c, S/2)$
- Opening: $\texttt{open}(f, s) = \texttt{dilate}(\texttt{erode}(f, s), s)$
- Closing: $\texttt{close}(f, s) = \texttt{erode}(\texttt{dilate}(f, s), s)$