EECS 207A notes by Nasser Abbasi

Edge Detection Algorithms

Gradient based (first derivative)

Rober Cross operator (uses 2x2 filter)

Sobel operator (extension of Robert cross operator to 3x3 filter. Approximates first derivative MAGNITUDE

Kirsh operator (another 3x3 filter extension)

$$\begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}$$)etects IORIZONT ιL lines

$$\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$$ Detects VERTICAL lines

How it works:
Filter the image with the above filters. These filters compute the magnitude of the gradient (approximation). Each center pixel will contain a value which is the magnitude of the gradient. Use another pass to threshold. What is left are the set of edge pixels. Connected edge pixels are the edges.

Definitions

Edge: change in gray level at one specific location.
Position of edge: center of ramp connection low gray level with high gray level.
Gradient: This is just the first derivative, but in vector form. Used for 2 and 3D. Magnitude of gradient is the magnitude of the steepest slope. There is ONE unique value for the gradient magnitude.

Gaussian in 2D= $\sigma^2 e^{-\frac{x^2+y^2}{2\sigma^2}}$

In 2D, possible Gaussian filter $\frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$

$$f = \frac{a_0}{2} + \sum_{n=1}^{N} a_n \mathrm{Cos}[nx] + \sum_{n=1}^{N} a_n \mathrm{Sin}[nx]$$

$$a_n = \frac{1}{\pi}\int_{-Pi}^{Pi} f(x)\, \mathrm{Cos}[nx]\, dx$$

$$b_n = \frac{1}{\pi}\int_{-Pi}^{Pi} f(x)\, \mathrm{Sin}[nx]\, dx$$

MARR-HILDRETH (or Laplacian of Gaussian, LoG)
Based on what is known about biological vision)

1. in natural images, features occur at different scales
2. in nature some local averaging occur (we see no diffraction patterns)
3. Gaussian best matches human vision (smooth in spatial and band-limited in frequency)
4. the maximum value of an intensity change occur where the second derivative is zero
5. orientation independent differential operator of lowest order is Laplacian

How it works:
1. Convolve image with 2D Gaussian filter (Gaussian is an averaging filter which better handles noise)
2. Convolve the resulting image with the Laplacian filter.
3. edge pixels are those for which there is a zero crossing in resulting image

Basic digital Laplacian $\begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$

Extended Laplacian which includes derivatives along diagonals as well $\begin{pmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{pmatrix}$

Gaussian averaging filter provides gentler smoothing and preserves edges better than a similarly sized mean filter. Smoothing depends on standard deviation. Gaussian can be approximated digitally with a 5x5 filter

Features:
1. Locality is not very good.
2. Edges not always thin.
3. edge lines always closed.
4. retain noise more.

Canny edge detector
An optimal mathematical based algorithm

Issues edge detector must address:
1. Should detect only edges, should detect them all
2. localization: distance between edge pixels and edge lines should be minimum
3. Should not select multiple edge pixels for same edge at same location, only one. (not clear)

Features:
1. Canny assume noise is white Gaussian noise.
2. Uses hystresis thresholding rather than fixed thresholding.
3. Maximize SNR and localization. Hence must maximize SNR*Localization. Localization is the reciprocal of the distance on an edge pixel to the edge line.
4. more robust to noise
5. avoids multiple edge lines
6. thinner edge lines

DCT $$T(i) = c(i)\sum_{n=0}^{N-1} V(n) cos\frac{(2n+1)i\pi}{2N}$$

where:
$c(0) = \sqrt{\frac{1}{N}}$
$c(k) = \sqrt{\frac{2}{N}}$ where k != 0.

IDCT $$V(i) = \sum_{n=0}^{N-1} c(n)T(n) cos\frac{(2i+1)n\pi}{2N}$$

where:
$c(0) = \sqrt{\frac{1}{N}}$
$c(k) = \sqrt{\frac{2}{N}}$ where k != 0.

Compression

Loosless

Hufmann coding

LZW(a form of hufman coding) used by zip?)

GIF , 8 bit color (index to color tables)

TIFF

Loosy

JPEG (24 bit color)

Sequential mode | Lossless mode | Progressive mode | Hierarchical mode

Firstly, a conversion of an RGB signal (such as the one used on your monitor) to YUV requires just a linear transform, which is really easy to do with analogue circuitry and it is cheap to compute numerically.

Secondly, YUV allows separating the colour information from the luminance component (which we perceive as brightness). For this reason YUV is used worldwide for television and motion picture
In a TV signal the luminance channel is modulated separately: this makes it possible for old TV sets to work even though there is no colour support (that is, the “colour” information is sent separately). Since the human eye is also much more responsive to luminance, JPEG compresses more heavily the chroma channels which are less likely to cause perceptible differences in the resulting image.

Quantization error is the main source of the Lossy Compression.

DCT is like FFT, but can approximate linear signals well with few coefficients.

Takes advantage of limitations in the human vision system to achieve high rates of compression
works by encoding in the frequency domain.
Human eyes does not notice loss of noise data in image (high frequency data)
steps:
1. convert to YCbCr (same as YUV) color model
2. Divid image to 8x8 blocks
2. perform DCTon 8x8 blocks
3. apply quantization tables every value is to be multiplied with a certain constant depending on it's position in DCT-matrix
4. apply ZigZag to group low frequency coefficients in top of vector.
5. DPCM on DC component , RLE on AC Components
5. Entropy coding