Solution ELE 510

Image proc. with robotic vision

Exam 07.12.2016

Exercise 1

a) (1) Image coding and compression.

This refers to how a digital image is represented on a computer. Im. coding can refer to how each pixel is assigned a number or a vector of numbers to describe a color or grayscale.

Im. compression refers to methods that can code an image effectively so to use as little space on a disk or time to transfer as possible.

1a)

2 image enhancement refers to the process of improving or changing an image so that it looks subjectively better.

ex. increase contrast
remove noise etc.

3 image segmentation refers to the process of extracting the outlines of different regions in the image.

For ex. segmenting a moving foreground object from a background, segmenting suspicious regions in medical images for further (manual or automatic) investigation etc.

1a)

(4) image description refers to the prosess of extracting features from images (or from regions in images) These features can be significant points, or amount of edges, frequency content, texture descriptors etc. Features can be used for registration or classification etc.

1b) i) a greylevel image is represented as a matrix of integers (or numbers) where for example the values between 0 and 255 range from black (=0) to white (=255) [0,255] gives 256 different values that can be represented with 8 bit, thus we say that such an image is represented with 8 bpp (bit pr. pixel) where pixel means picture element.

A color image consist of three such matrices to be able to represent all colors. One way of defining the color space is by constructing each color by Red, Green and Blue components, we call this an RGB image where one matrix defines the amount of red in each pixel, the second matrix the amount of green etc. There are other ways of representing a color image as well: Luminance, Chrominance 1 and Chrominance 2 where Luminance represent the grey level image information an Chrominance the color informa Or HSV - Hue, Saturation and Value.

ii) R: b)

G: e)

B: d)

Luminance: c)

1c) $\hat{g}(m,n) = \frac{1}{MN} \sum_k \sum_l g(k,l)\, e^{-j2\pi\left(\frac{km}{M} + \frac{ln}{N}\right)}$

$$\hat{g}(0,0) = \frac{1}{M \cdot N} \sum_k \sum_l g(k,l)\, \underbrace{e^{-j2\pi(0+0)}}_{e^0 = 1}$$

$$\hat{g}(0,0) = \frac{1}{M \cdot N} \sum_k \sum_l g(k,l)$$

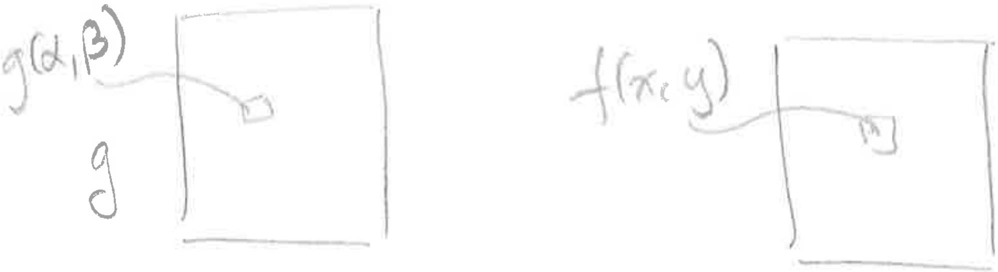
$$\hat{g}(0,0) = \text{mean}(g(k,l))$$

The Fourier coefficient $\hat{g}(0,0)$ is found as the mean of all the pixel values in the whole image. This is also known as the DC-component (direct current) and shows the energy of the lowest frequency (with freq = 0).

d)

$$f \rightarrow \boxed{O} \rightarrow g$$

$$h(x,\alpha,y,\beta) = h_c(x-\alpha)h_r(y-\beta)$$



x, y refers to row and column position in input image, f.

f

α, β refers to row and column position in output image g

in general:

$$g(\alpha,\beta) = \sum_x \sum_y h(x,\alpha,y,\beta)\cdot f(x,y)$$

Thus the output pixel at position α, β is a linear combination of (in general) all pixels in f.

$h(x,\alpha,y,\beta)$ gives the coefficents (many can be zero.)

separable but not shift invariant:

$$h(x,\alpha,y,\beta) = h_c(x,\alpha)\cdot h_r(y,\beta)$$

shift invariant but not separable:

$$h(x\alpha,y,\beta) = h(x-\alpha, y-\beta)$$

Exercise 2

a) $$h_r(\beta - y) = \begin{cases} 1 & \text{if } \beta - y = -1 \\ 2 & \text{if } \beta - y = 0 \\ 3 & \text{if } \beta - y = 1 \end{cases}$$

$$h_c(\alpha - x) = \begin{cases} 1 & \text{if } \alpha - x = 0 \\ 1 & \text{if } \alpha - x = 1 \end{cases}$$

$$f = \begin{bmatrix} 1 & 1 & 2 \\ 2 & 2 & 4 \\ 1 & 3 & 5 \end{bmatrix}$$

matrix form:

$$h_r = \begin{bmatrix} 2 & 3 & 0 \\ 1 & 2 & 3 \\ 0 & 1 & 2 \end{bmatrix} \qquad h_c = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

$$g = h_c \cdot f \cdot h_r = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 2 \\ 2 & 2 & 4 \\ 1 & 3 & 5 \end{bmatrix}} \cdot \begin{bmatrix} 2 & 3 & 0 \\ 1 & 2 & 3 \\ 0 & 1 & 2 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 2 \\ 3 & 3 & 6 \\ 3 & 5 & 9 \end{bmatrix} \cdot \begin{bmatrix} 2 & 3 & 0 \\ 1 & 2 & 3 \\ 0 & 1 & 2 \end{bmatrix} = \begin{bmatrix} 3 & 7 & 7 \\ 9 & 21 & 21 \\ 11 & 28 & 33 \end{bmatrix}$$

b) $h_1 = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$ $h_2 = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$

i) image b) corresponds to filtermask h_2 which is an approximation of the Laplacian (i.e. the double derivative)

This can be seen by the double edge, i.e. the edges goes from black to white and the zero crossing corresponds to the edge.

h_1: smooths over rows and take the 1.st. derivative over columns. this is one of the Prewit filtermask, emphesizing edges in the vertical direction, i.e. horizontal lines. This effect is seen in image e)

ii) I would use filtermask h_2 for two reasons.

① h_1 only finds vertical edges

② slow rising edges gives a blurry edge detection using 1st derivative, whereas h_2 (Laplacian = 2nd derivative) gives a double edge image with the zero crossing at the peak of the edge.

c) Homomorphic filtering:

To deal with multiplicative noise, or uneven illumination this can be useful.

$f(x,y) = i(x,y) \cdot r(x,y)$ i.e. the pixel values is a product of the illumination function $i(x,y)$ and the reflectance $r(x,y)$.

9

$$f(x,y) = i(x,y) \cdot r(x,y)$$

$$\log(f(x,y)) = \log(i(x,y) \cdot r(x,y))$$

$$= \underbrace{\log(i(x,y)) + \log(r(x,y))}$$

additive noise problem!
can use linear filters.

$$\left[\log(f(x,y))\right]' = O\left[\log(i(x,y)) + \log(r(x,y))\right]$$

$$\downarrow$$

$$g(x,y) = e^{\left[\log(f(x,y))\right]'} = \hat{f}(x,y)$$

d) $$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u, y+v) - I(x,y)\right]^2$$

$$E(u,v) \approx [u \; v] \, H \begin{bmatrix} u \\ v \end{bmatrix}$$

$E(u,v)$ gives a measure of the energy of of the change of values in all directions within a small weighted window around u,v

Lines:

no change

large change

Corners:

large change in all directions.

This energy can be approximated by the second eq. where

$$H = \begin{bmatrix} I_x I_x & I_x I_y \\ I_y I_x & I_y I_y \end{bmatrix}$$

where I_x and I_y are gradient image matrix, i.e. 1st. derivative in x and y directions

The $E(u,v)$ values are not found, instead the H matrix is found as

$$H = \sum_{\text{window}} \begin{bmatrix} I_x I_x & I_x I_y \\ I_y I_x & I_y I_y \end{bmatrix}$$

the properties of this matrix makes it possible to evaluate pot. corner points by finding

$$R = \det(H) - k\left(\frac{\text{trace}(H)}{2}\right)^2$$

R is an image with values in every pixel. high value → more probably a corner point.

Finally the X number of pixels with largest values $R(x,y)$ are picked out as the X most likely corner points.