Edge detection

//Input: resampled image, IR

//Output: image with edges detected, IRESULT

create 2-D filter for horizontal edge emphasizing, F1

IP 🡨 IR filtered using F1

create 2-D filter for contrast enhancement, F2

IU 🡨 IR filtered using F2

I1 🡨 gabor\_filter\_func( IP )

I2 🡨 gabor\_filter\_func( IU )

IRESULT 🡨 superimpose I1 and I2

return IRESULT

**function:**

**gabor\_filter\_func ( image )**

lambda [ ] 🡨 [ 0, 2, 4, 8, 16, 32] // different scales

theta [ ] 🡨 [ 00, 600, 1200 ] // different orientations

[ M, N ] 🡨 size of image

Fedgemapk1 [ M ][ N ] 🡨 { 0,.. .., 0 }

Fedgemapk2 [ M ][ N ] 🡨 { 0,.. .., 0 }

for i 🡨 1 to length of lambda

L 🡨 lambda[ i ]

for j 🡨 1 to length of theta

t 🡨 theta[ j ]

// apply gabor filter

gabout [ ] 🡨 gabor ( image, 1, t, L, 0, 0.5 )

//calculate mean and std deviation for each gabor responses

std\_mean 🡨 mean of gabout

std\_deviation 🡨 standard deviation of gabout

difference 🡨 absolute value of (mean of gabout - standard deviation of gabout)

if difference < 50 then

//create a edge map using threshold k1

k1 🡨 2.5

tmpI1 🡨 gabout < (-1 \* std\_deviation \* k1) | gabout > ( std\_deviation \* k1 )

edgemap1 🡨 tmpI1

Fedgemapk1 🡨 Fedgemapk1 | edgemap1

end

end

end

end

gabor ( img, sigma, theta, lambda, psi, gamma )

// Input:

// img : Input image

// sigma: Variance

// theta : orientation

// lamda : wavelength of sinusoidal factor

// psi : phase offset

// gamma : spatial aspect ratio

//Output:

//gb: gabor filtered image

sigma\_x 🡨 sigma;

sigma\_y 🡨 sigma/gamma;

//Bounding box

nstds 🡨 3

xmax 🡨 maximum of(absolute value of (nstds\*sigma\_x\*cos(theta)),

absolute value of (nstds\*sigma\_y\*sin(theta)))

xmax 🡨 upper bound of (maximum of ( 1, xmax ) )

ymax 🡨 maximum of(absolute value of (nstds\*sigma\_x\*sin(theta)),

absolute value of (nstds\*sigma\_y\*cos(theta)))

ymax 🡨 upper bound of (maximum of ( 1, ymax ) )

xmin 🡨 -xmax

ymin 🡨 -ymax

[x,y] 🡨 replicates the grid vectors ( cmin to xmax ) and ( ymin to ymax ) to produce

the coordinates of a rectangular grid (X, Y).

// Rotation

x\_theta 🡨 x\*cos(theta)+y\*sin(theta);

y\_theta 🡨 -x\*sin(theta)+y\*cos(theta);

gb 🡨 e(0.5\*(x\_theta^2/sigma\_x^2+y\_theta^2/sigma\_y^2))

\* cos(2\*pi/lambda\*x\_theta+psi)

Imgabout 🡨 convolution of (img and imaginary part of (gb))

Regabout 🡨 convolution of (img and real part of (gb))

gb 🡨 square root of (Imgabout \* Imgabout + Regabout \* Regabout)