Image Processing Projects

1 Initial downloads and installations

- git : `http://git-scm.com/downloads`
- cmake : `http://www.cmake.org`
- qt 4.8.5: `http://qt-project.org/downloads`
- boost: `http://www.boost.org/`
- lapack: `http://www.netlib.org/lapack/`

A C++ compiling environment

Setup an account on github `http://www.github.com`
Create (if you do not have one already) an ssh key and to deposit it on this site.
If you succesfully installed git and setup the git account, you can checkout the project:

```
git clone --recursive git@github.com:CBBProject/CBBProject.git
```

2 Processing of 2D gray-scale images

2.1 Basic Processing

2.1.1 Load an image (already provided)

Input: image path

Output: boolean flag

2.1.2 Save an image (already provided)

Inputs: image path, image

Output: boolean flag

2.1.3 Threshold an image

Inputs: image I, scalar S

Output: thresholded image J such that $\mathtt{J}(\mathtt{i},\mathtt{j}) = \mathtt{I}(\mathtt{i},\mathtt{j})$ if $\mathtt{I}(\mathtt{i},\mathtt{j}) \geq \mathtt{S}$, otherwise $\mathtt{J}(\mathtt{i},\mathtt{j}) = \mathtt{0}$

2.1.4 Display an image

Inputs: image I,
image range (intensities corresponding to min (black) and max (white))

Output: display of image (thresholded to fit the range)

2.1.5 Multiply 2 images pixel-wise

Inputs: images I and J

Output: image K such that K(i,j)=I(i,j)*J(i,j)

2.1.6 Add two images pixel-wise (already provided)

Inputs: images I and J

Output: image K such that K(i,j)=I(i,j)+J(i,j)

2.1.7 Modify the pixel intensities of an image (e.g. by a look-up table)

Inputs: image I
name of function (look-up table) f

Output: image J such that J(i,j) = f(I(i,j))

2.1.8 Create an image defined by a function f(x,y)

Inputs: domain boundaries xmin,xmax,ymin,ymax
image size N,M
name of function f

Output: image J such that J(i,j) = f(i,j)

2.1.9 Calculate the histogram of an image

Inputs: image I
number of bins B

Output: a list of B bin-centers
a list of B pixel counts belonging to each bin

2.1.10 Subsample an image

Input: image I

Output: image J such that J(i,j) = I(2*i,2*j)

2.2 Intermediate

2.2.1 Convolve an image with a mask

`http://en.wikipedia.org/wiki/Kernel_%28image_processing%29`

2.2.2 Compute 1D FFT of an image (in x or in y direction)

use fftw package

2.2.3 Compute 2D FFT of an image

`https://ia700307.us.archive.org/7/items/Lectures_on_Image_Processing/EECE253_06_FourierTransform.pdf`

2.2.4 Convolve an image with a Gaussian, by FFT

`https://ia700307.us.archive.org/7/items/Lectures_on_Image_Processing/EECE253_08_FrequencyFiltering.pdf`

2.2.5 Compute the gradient of an image (in x or in y direction)

`http://en.wikipedia.org/wiki/Image_gradient`

2.2.6 Compute the Laplacian of an image

`http://en.wikipedia.org/wiki/Discrete_Laplace_operator`

2.2.7 Filter the image with a Gaussian filter

`http://en.wikipedia.org/wiki/Gaussian_blur`

2.3 Advanced processing

2.3.1 Compute the contours of an image

by using a Gaussian filter + a Laplacian filter

2.3.2 Subsample an image while avoiding aliasing

by preprocessing with a Gaussian filter, see https://ia600307.us.archive.org/7/items/Lectures_on_Image_Processing/EECE253_11_SamplingAliasing.pdf

2.3.3 Compute the median filter of an image

https://archive.org/download/Lectures_on_Image_Processing/EECE253_16_MedianFilters.pdf