IMAGE PROCESSING (RRY025)

STUDIO EXERCISES

Lecture 1, Introduction

On the windows system to use m-files written for the course you must in windows after starting MATLAB type

```
addpath \\mfil.me.chalmers.se\course-err041
```

If you are using the linux operating system after starting MATLAB you should type

```
addpath /chalmers/groups/course-err041
```

EX 1. IMAGE ENHANCEMENT Histograms and Contrast Stretching

Inside MATLAB type **imadjdemo**. Start by loading the *pout* image. The left side shows the original image and the right the enhanced image. The histograms below each image show the fraction of pixels of each grey level, while the top right panel shows the 'transfer function' indicating how a grey level in the input image is mapped to a grey level in the output.

The shape of the transfer function can be changed using the the +/-Brightness and +/- Contrast buttons (experiment). The transfer function can also be adjusted manually by clicking on the top and bottom yellow circles. Experiment with different transform functions. **Questions** Which transform function produces the best image? What does the image histogram look like in this case? Try other images. Finally apply the 'Histogram Equalisation' function by clicking on the 'Operations' box, does this produce a nice looking image? We will discuss more on this subject of transfer functions and histograms at tomorrows lecture... .

EX 2. IMAGE RESTORATIONS

In matlab type *demos* then follow the links to toolboxes, image processing, Deblurring, Wiener filter and follow the example. Alternatively look at the 'Large Scale Constrained Optimization' example under the Optimization toolbox.

EX 3. IMAGE COMPRESSION

In matlab type *dctdemo*. This shows how compression is achieved using the block Discrete Cosine Transform (DCT as used within JPEG compression). The image is first divided into 8x8 pixel blocks, then each block undergoes a DCT (a variant of the Fourier transform) which outputs an 8x8 matrix for each block. In this transforms the largest transform components (due to smooth structure) are in the top left corner and the much lower amplitude high frequency details are in the bottom right corner. Since the human eye cannot usually see such fine details we can compress by discarding these frequency

components before transferring all transform values in the JPEG file over the Internet. At the other end the image is reconstructed on your browser by doing the inverse DCT on each block and assembling the blocks into an image. In this example compression is achieved solely using 'psycho-visual redundancy' (limitations of eye and brain), real JPEG uses other tricks in addition to get more compression.

Play with the slide bar to test different amounts of compression. For a given compression ratio a certain fraction of the whole transform is kept (shown as white in the top right figure), the exact area being chosen on the basis of which transform components have largest amplitude averaged over all the blocks. **Questions** At what degree of compression do you start to see a loss of quality? What happens if we attempt a compression ratio of 64? Which images can be compressed most and which least?