Ph 21 Assignment 3: Image Processing

This assignment covers some of the basic parts of image processing, a new(ish) and growing field. If you don't already have it, obtain the Python Imaging Library (PIL). It is already installed on all the lab computers. Find the documentation, skim it to get an idea of what might be possible.

1. Import an image into your local directory, either by copy pasting or by writing a script that takes an image from the internet. If you really liked assignment 1, try taking an image result from a search engine.

2. Write a python script that imports the image into python. This can be done using Image.open . Have it export some basic info about the image to the command line, like its size!

3. We are going to write an algorithm for edge detection. This is an important aspect of machine vision, and a good introduction to image processing algorithms. First, skim the wikipedia page. The best general algorithm is the Canny edge detector, which we will implement. You are encouraged to try various ideas you might have before spoiling the fun by reading how the pros do it. However, to get an idea of the effect you can achieve, try the existing Image Filter called `FIND_EDGES`.

4. First, noise might prove to be a problem. Most edges are more obvious than individual pixel values, which are subject to jitter. Therefore, convolve your image with a blurring filter. Common choices are Gaussian profile, 5x5 stencil. How will you treat the edge pixels?

5. Next, edge detection on the slightly blurred image. An edge is a place where the derivative of the pixel brightness function (red+green+blue, with or without colour filtering) is quite large. Therefore you want some way to measure the local derivative, or Del, of the image. This will take the form of a stencil. The midpoint method is the least accurate way of doing this. Some algorithms measure horizontally and vertically, then take the square sum. Others measure in 45 degree increments, and sum accordingly. The best stencil is the first derivative of a Gaussian. There is a parameter left over in the width of the function, which has to be chosen, normally with an idea of the scale of the edges or objects you want to resolve in mind. To demonstrate this, produce a series of edge detection images with different values of the width parameter.

6. Optimise the system to the extent of your patience. Try edge detection on different colours, with different stencils, with or without blurring, and compare the "signal-to-noise" ratio obtained. How do you avoid double detections? Do you think you could use an edge detector to defeat CAPTCHAs? What other uses can you imagine for edge detection algorithms?

7. Write a sentence or two on edge detectors and this assignment.