Image Processing and Computer Vision, Dept of Computer Science, University of Bristol

COMS30121: Assignment 01 ('Coin Counter')

Introduction

This assignment consists of two subtasks. As with the previous weeks' lab exercise, you will be provided with a Visual Studio Solution with working code. The code will load the images below and display them. Expand on this solution to solve the tasks of this assignment.

Task One: Edge Detection

As detailed in this week's lecture, the detection of edges is pivotal in the task of distinguishing shapes. Your first task is to implement the Sobel edge detector by convolving an image with the Sobel kernels and calculate the gradient information. *(Without using the convolution and sobel functions offered by OpenCV.)*

Task Breakdown

1. Implement a function `sobel`.

2. This function should take an image as an input and compute the following

   - Image containing the derivative in the x direction $\frac{\partial f}{\partial x}$
   - Image containing the derivative in the y direction $\frac{\partial f}{\partial y}$
   - Image containing magnitude of the gradient $\nabla f(x, y)$
   - Image containing the direction of the gradient ρ

3. Be able to display the four images above and discuss their appearance in relation to the original image.

Implementation Details

- Implementation of a Sobel filter requires convolution. You have already used convolution in OpenCV for last week's task – so feel free to look back at those labsheets and review how convolution can be implemented.

- Similar to the sharpening filter from last week, convolution with the Sobel filter can result in a negative response value and generally in a wider range of values. Think carefully about the quantisation of these results, the handling of the image boundary and the data type of the matrix in which you will hold them.

- When displaying images, make sure the data is in a format that OpenCV can display. The two main types for displaying are `unsigned short` between 0-255 and `float` or `double` data in the range 0-1.

Task Two: Hough Circle Detection

In this task, you will be using the gradient images generated from task one and perform a Hough Circle transform to detect the location of the coins in the provided images (..and one coin image of your own choice).

Task Breakdown

1. Take the gradient magnitude image you generated for task one and apply a thresholding operation (e.g. set all pixels larger than a threshold value T to 255 and all others to 0) to determine the set of pixels with the strongest gradient magnitudes to be considered for circle detection.

2. Implement a function `hough` that calculates the (3D) Hough Space (x_0, y_0, r) from the thresholded gradient magnitude image and the gradient orientation image. You need to make a decision about the size (i.e. number of cells) in your Hough Space. The function should at least have the following input parameters:

   - The thresholded gradient magnitude image and the gradient orientation image
   - Threshold by which a peak in the Hough Space is actually considered a circle
   - You may also wish to consider minimum / maximum radius and distance between circle centres.

3. Display the Hough Space for each image. As the Hough Space for circles is 3D, you can create a 2D image by summing the values of the radius dimension. This will highlight circle centres.

4. Threshold the Hough Space and display your set of found circles on the original images.

Implementation Details

- Consider representing your Hough Space as a 3D array.

- When you are displaying the Hough Space, you might want to take the logarithm of the image to make the image values more descriptive.

Formative Assessment (marks do not count towards unit)

- ***For a mark above 40*** *you need to have implemented task one and be able to answer basic questions about your program and the images that your Sobel filter produces*

- ***For a mark above 50*** *you also need to implement the basics of task two and be able to answer questions about your program and the Hough Space you have generated*

- ***For a mark above 60*** *you also need to implement all of task two and be able to answer questions about the Hough Space you have generated and its performance on the sample images*

- ***For a mark above 70*** *you also need to be able to explain in detail the concepts behind convolution, the Hough Transform and all choices you have made in your implementation, as well as all specific properties of the results you have achieved*