State University of New York at Buffalo

CSE 473/573

Fall 2016 Final Project

Project Title:

Hough Transform to detect circles

Team Members:

Nikhil Prakash (50208057)

Sunil Kunjappan Vasu (50205673)

December 16^{h}, 2016

Literature Review

Identification of an object in an image has been an interesting concept in computer vision and image processing. It can be done by checking for a pattern in the image by simple segmentation but this is not possible in the practical implementations as the image can be distorted by noise, interference and other unwanted factors. To overcome this, the idea of Hough Transform was developed which gave a better ability to detect object in the image. The Hough transform was patented as U.S. Patent 3,069,654 in 1962 and assigned to the U.S. Atomic Energy Commission with the name "Method and Means for Recognizing Complex Patterns"[1]. Hough Transform can even detect overlapped or semi-occluded objects. In short Hough transform is a feature extraction technique used in computer vision, and digital image processing. In this project, we are given an image with coins of different sizes, few are overlapped. Our task is to detect and display circles around the boundary of the coins.

Before the implementation of Hough Transform the image is denoised by Gaussian Filter of 3x3 size. Gaussian filter helps to remove spikes in the image histogram which are the noise in the image. Gaussian performs an averaging process in the neighborhood of image pixels thus bringing an averaging or blurring effect on the image. This operation is also called as Gaussian blur.

After De-noising the image we needs to detect the edges. We have used Canny filter for this purpose. Canny filter is developed by John F. Canny in 1986 and can be used for detection of wide range of edges in the image[2]. The optimality of the detection is based on the three criteria namely – detection, localization and one response criteria. Once the edges are detected by the Canny filter, the image is thresholded into the binary image. Binary image as the name says consist of only two pixels value – black(0) and white(255). We have used Otsu's method for thresholding the image into binary. This method is developed by Nobuyuki Otsu and is helpful for conversion of gray scale image into binary image. Otsu's method works by clustering-based image thresholding[3].

The Circle Hough transform, specialization of Hough Transform, is performed on the binary image after the thresholding to identify the coins in the given image. We start by drawing circles on the edges detected by the canny filter for a specific set of radius in a parametric space names as the accumulator cell. After drawing the circles for each of the radius the accumulator array is scanned for the point with voting greater than a certain threshold. These points will constitute for the center of the circle for that specific radius. This process is repeated for the specific set of radius. And finally all the circles are redrawn on the image.

The mathematical approach for understanding Hough Transform was obtained from the paper "Circle Detection Using Hough Transforms Documentation" by Jaroslav Borovicka { pinus@centrum.cz, jb2383@bris.ac.uk} [4]

Project Final Report

a. Introduction:

Hough transform is a feature extraction technique used in computer vision, and digital image processing. The main advantage of Hough transform is its capability to detect shape of the objects, even if they are occluded. In this project, we are given an image with coins of different sizes, few are overlapped. Our task is to detect and display circles around the boundary of the coins.

Before the implementation of Hough Transform the image is denoised by Gaussian Filter of 3x3 size. Gaussian filter helps to remove spikes in the image histogram which are the noise in the image. This operation is also called as Gaussian blur as it brings a blurring effect to the image.

After De-noising the image we needs to detect the edges. We have used Canny filter for this purpose. Canny filter is developed by John F. Canny in 1986 and can be used for detection of wide range of edges in the image[1]. The optimality of the detection is based on the three criteria namely – detection, localization and one response criteria. Once the edges are detected by the Canny filter, the image is thresholded into the binary image. Binary image as the name says consist of only two pixels value – black(0) and white(255). We have used Otsu's method for thresholding the image into binary. This method is developed by Nobuyuki Otsu and is helpful for conversion of gray scale image into binary image. Otsu's method works by clustering-based image thresholding[2].

The Circle Hough transform, specialization of Hough Transform, is performed on the binary image after the thresholding to identify the coins in the given image. We start by drawing circles on the edges detected by the canny filter for a specific set of radius in a parametric space names as the accumulator cell. After drawing the circles for each of the radius the accumulator array is scanned for the point with voting greater than a certain threshold. We have used the method of average 8 neighbour for finding the maximum voting point. These points will constitute for the center of the circle for that specific radius. This process is repeated for the specific set of radius. And finally all the circles are redrawn on the image.

b. Your Approach:

i. The overall system and how the techniques outlined in your reference paper work.

The overall system is now in capable to detect circles in the test image. The mathematical approach of optimizing the code and faster implementation algorithm were adopted from references paper "Circle Detection Using Hough Transforms Documentation" by Jaroslav Borovicka { pinus@centrum.cz, jb2383@bris.ac.uk} [4] and wikipedia descriptions [1][2][3]

ii. Software and other methodologies employed to implement this project.

We have implemented the project on python platform using openCV library. Python version Python 2.7.12 was used as the development platform. OpenCV inbuilt functions were used for binary thresholding[5], filtering[6], canny edge detection[7] and drawing basic shapes like circles and rectangle[8] in image after recognition.

c. Outcome and Deviations

i. Presentation of the project outcome (algorithmic results)

The algorithmic steps are as follows:

```
1. Read the input image as gray level
2. Perform smoothing by convolving with an 3x3 Gaussian kernel
3. Detect edges in the image using Canny filter
    3.1 Convert the image type to uint8
    3.2 Use OTSU thresholding as parameter and obtain lower and upper thresholds for Canny filter
    3.3 Perform Edge detection and return the edge image
4. Perform Circle detection
    4.1 Repeat the following for each radius r
        4.1.1 Initialize an empty accumulator array acc_cells
        4.1.2 For each row and column in the image do
            4.1.2.1 Compute center(a,b) of the circle using the formula $a = x - rcos(0)$ and $b = y - rsin(0)$
            4.1.2.2 Increment the accumulator cell acc_cell[a][b] by 1

        4.1.3 If the maximum value in acc_cells is greater than a threshold 'max_threshold' then
            4.1.3.1 Perform initial thresholding to discount low voted cells
            4.1.3.2 For each row and column in acc_cells do
                4.1.3.2.1 Using 8x8 neighbors to compute the average sum for acc_cells[row][col]
                4.1.3.2.2 If the average is above a threshold then
                    4.1.3.2.2.1 Save the center point point and the radius to a list 'circles'
5. Draw the circles onto the original image using the 'circles' as center points for the circles
6. Display the circle detected image
```

ii. Discussion of the outcome.

We were able to detect all circles success in the test image and with minimal deviation from the expected results. With the analysis of the algorithm on a set of test images we were able to replicate the detection of circles successfully. We however feels that the circles are not as accurate with respect to the exact shape of the coins in the image. This we think is due to the canny edge detection abnormalities in the pixel space. We believe that a perfect circle detected edge image can produce a better result than the existing results.

iii. Lessons learned from algorithm development

We got a first hand experience in the development of complex image processing and computer vision algorithm into practical working model. This gave us an opportunity to understanding the system limitations that occur in Image processing and computer vision programmings like the computing speed, data structure storage, working on a large set of datas and so on. Optimization of the algorithm is one single task we found very difficult and time consuming. This issue is mainly due to the limitation in the computing power. A better system with a greater computing power would be able to optimize the result further.

d. Explanation of Software and Program Development

i. Presentation of the of software and program development by each partner (based on proposed partition for team project)

We started with the proposed partition of the work load. We were able to be sincere with the partition of the work load and the proposed plan to the extend where we were in a position where the optimization required both of our simultaneous efforts. We however at times had to perform many tasks in collaboration. The final partition of the work was as follows.

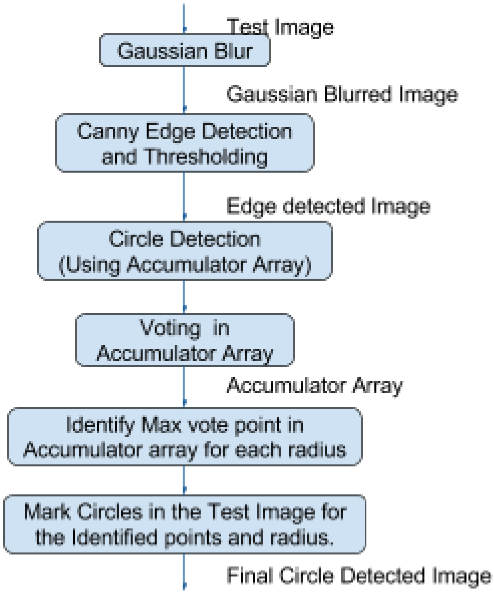
Sunil Kunjappan Vasu:

Gaussian Blur
Canny Edge Detection
Thresholding
Detecting circle radii
Optimization
Report + Status Report

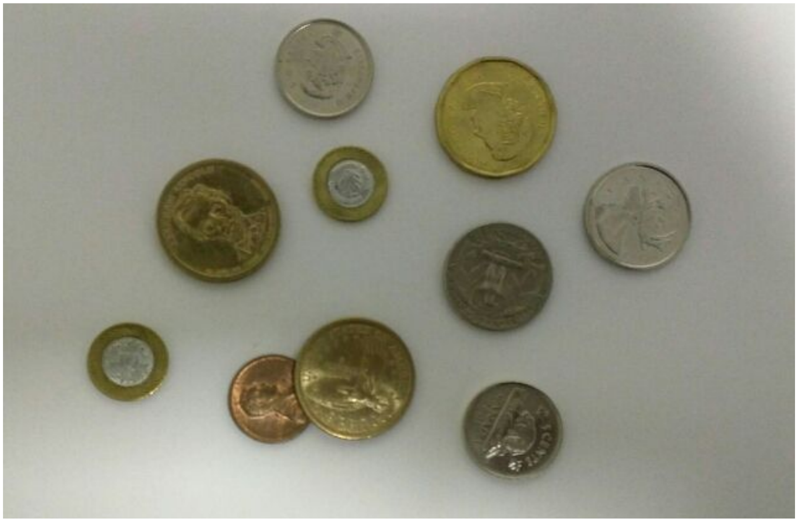
Nikhil Prakash:

Circle Detection
Marking circles in test image
Final image
Testing for other images
Optimization
Report + Status Report

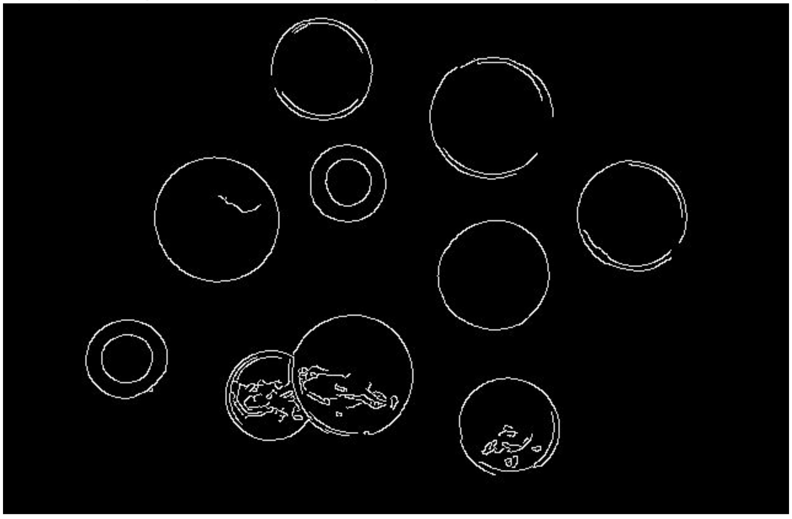
The programmatic flow chart of the project is as follows.


Test Image
Gaussian Blur
Gaussian Blurred Image
Canny Edge Detection and Thresholding
Edge detected Image
Circle Detection (Using Accumulator Array)
Voting in Accumulator Array
Accumulator Array
Identify Max vote point in Accumulator array for each radius
Mark Circles in the Test Image for the Identified points and radius.
Final Circle Detected Image

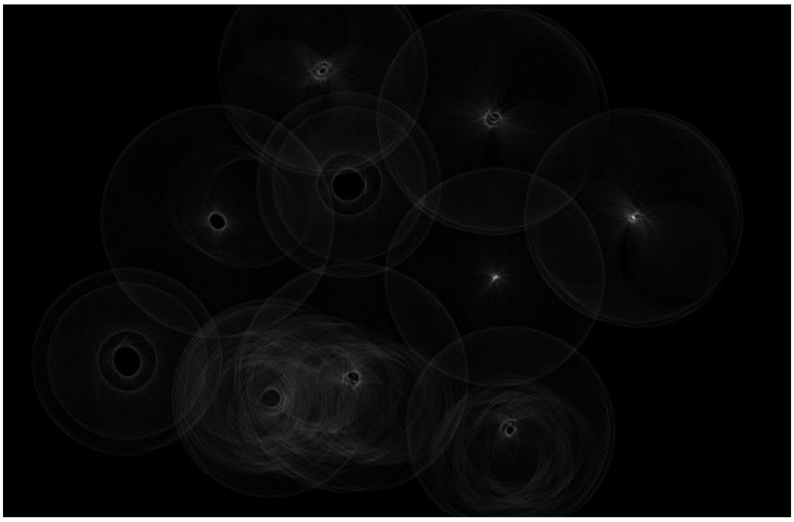
The intermediate results in the programmatic flow is as follows:
Test Image:



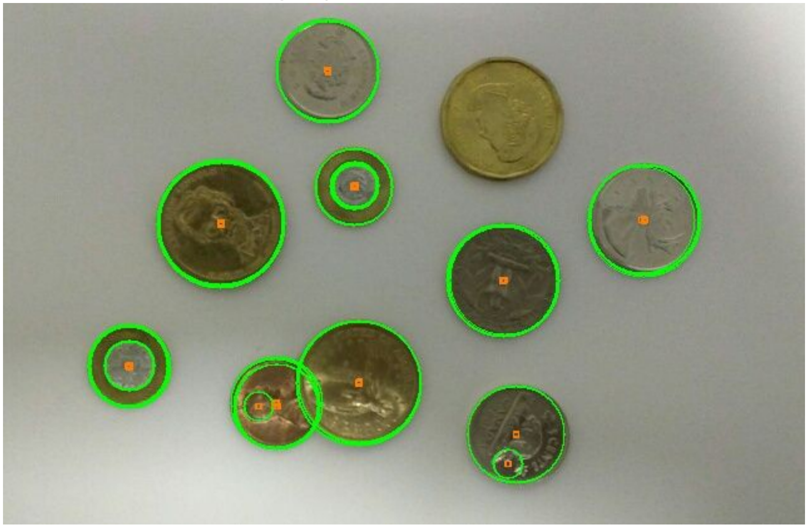
Canny Edge detected Image:



Accumulator Matrix for one of the radius:



Circle detected Image (intermediate detection for small radius range):



ii. Discussion of the code development lessons. Clear explanation is needed when he programing codes are adopted from online sources or from standard and advanced software routines.

We started the development after a thorough literature survey on Hough Transform and how it can be used to implement the circle detecting in the image. We however faced no issues where we would need to consult online for codes for the project. However we have adopted the inbuilt functions from openCV library[5]-[8] to perform basic task.

iii. Lessons learned from programming codes development.

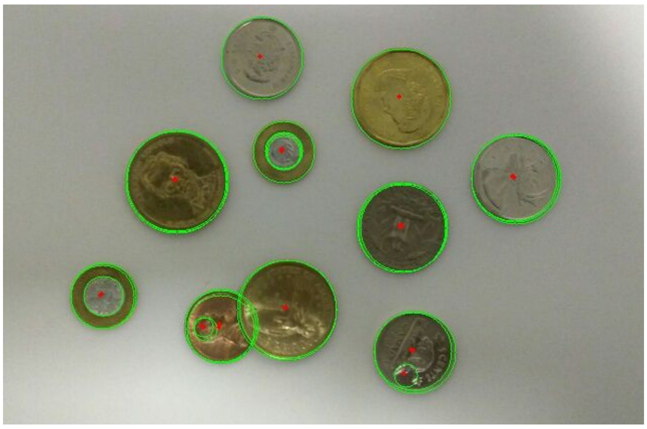
When we started with the programming code development we faced issues in the the representation of data structure and their reliability to handle such a huge amount of pixel datas. Optimization of programming code is one single task we found very difficult and time consuming. This issue is mainly due to the limitation in the computing power. A better system with a greater computing power would be able to optimize the result further.

e. Summary and Discussion
i. Summary of the project

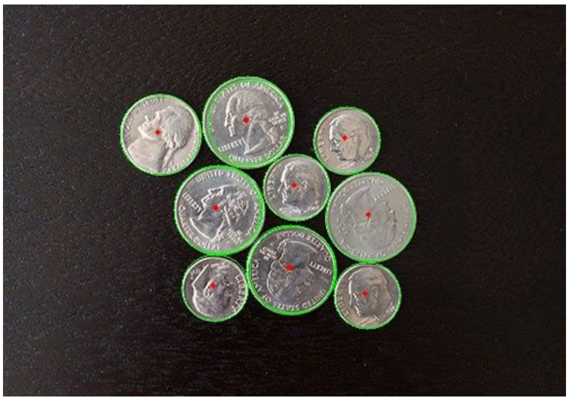
Hough transform is a feature extraction technique used in computer vision, and digital image processing. The main advantage of Hough transform is its capability to detect shape of the objects, even if they are occluded. In this project, we are given an image with coins of different sizes, few are overlapped. Our task is to detect and display circles around the boundary of the coins.

We were able to detect all circles success in the test image and with minimal deviation from the expected results. With the analysis of the algorithm on a set of test images we were able to replicate the detection of circles successfully. Some of the circle detected image results are as follows:

Circle Detection on given Test Image:



Other Image we tested:



ii. Lessons learned in this course, including classroom, homework sets and project

The course CSE 573 – Computer Vision and Image Processing was indeed a demanding course. We have to put in a lots of our efforts and time for the timely completion of the projects, Homeworks and exam preparations. However in the end we feel that the basic concept for this diverse field of study has been deeply ingrained in us by the instructor and teaching assistants. We learnt concepts like signal representation which includes Fourier series, Fourier Transform, Cosine signal representation, wavelet transform. We were introduced to basic technique for image manipulation like edge detection, segmentation, region merging, data structures in image processing and so on. We had opportunities to develop our programming skills to implement image processing algorithms in software platform like python, matlab using openCV. Extraction of low level features from images where also implemented in our homeworks. The project helped us in realizing how a complex algorithm like the Hough transform can be implemented in the python and understand the practical issues regarding the optimization that can occur in image processing.

h. Reference List

[1] https://en.wikipedia.org/wiki/Hough_transform
[2] https://en.wikipedia.org/wiki/Otsu's_method
[3] https://en.wikipedia.org/wiki/Canny_edge_detector
[4] http://www.borovicka.org/files/research/bristol/hough-report.pdf
[5] http://docs.opencv.org/trunk/d7/d4d/tutorial_py_thresholding.html
[6] http://docs.opencv.org/2.4/modules/imgproc/doc/filtering.html
[7] http://docs.opencv.org/trunk/da/d22/tutorial_py_canny.html
[8] http://docs.opencv.org/2.4/modules/core/doc/drawing_functions.html