Report for BBM418-Assignment-1

Ali İhsan ELMAS

April 2020

1 Summary

This document is a report for BBM-418 Computer Vision Laboratory Assignment-1. An implementation of Hough Transform for line detection was done and results are given for this assignment. Sample space consists of bar-code images which has different angles and visibility. Python is used as programming language and OpenCV is used for processing library. Results show that we are able to detect bar-codes with some errors and missing parts.

2 Algorithm

We can divide our algorithm into 8 parts:
1) Prepare image to process: Mask and the bar-code images are both read and then put on each other to crop the region of interest.
2) Apply filters to find edge points: These operations are done in the given order: Convert image to grayscale, apply bilateral filter for smoothing, use canny edge detector to get edge points.
3) Transform found edge points to polar coordinate sinusoidals: rho values are computed for every theta value from 0 to 180 degrees for every x and y edge pair using this formula

$$rho\_val = x * \cos(theta) + y * \sin(theta) \tag{1}$$

Then Hough functions are created using these values
4) Create virtual Hough boxes with an accumulator array: For voting purpose a 2D array is created with dimensions *theta_values* and *max_rho_val-min_rho_val*
5) Fill accumulator with votes: For every theta and rho pair in Hough functions accumulator's value is increased by 1
6) Apply a threshold using maximum number of votes in accumulator and a threshold factor: Elements with larger than a threshold value -which is calculated by multiplying the max vote value with a threshold factor ranging between 0.7 and 0.9- are treated as line candidates.
7) Apply neighborhood suppression to eliminate similar lines: To eliminate very close lines, lines which are closer than a constant to a peak, are set to zero.
8) Plot and save found lines: Found lines are plot on a blank and original image. All images from start of the process are stack in horizontal order and then saved in a folder.

3 Results

Here are the results for given sample images. Threshold factor is selected as 0.9. You can find more results with different Threshold factors in "Results" folder.

Original Im | Smoothed Im | Canny applied Im | Found lines | Lines on Original Im

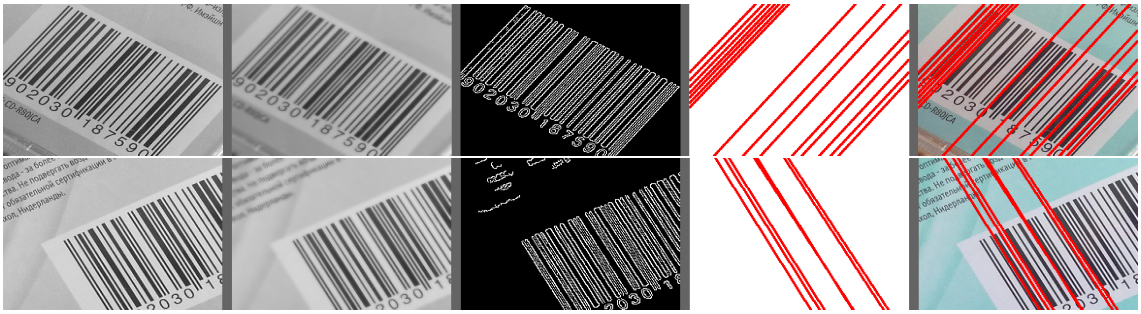

Figure 1: Image 1

Figure 2: Image 2

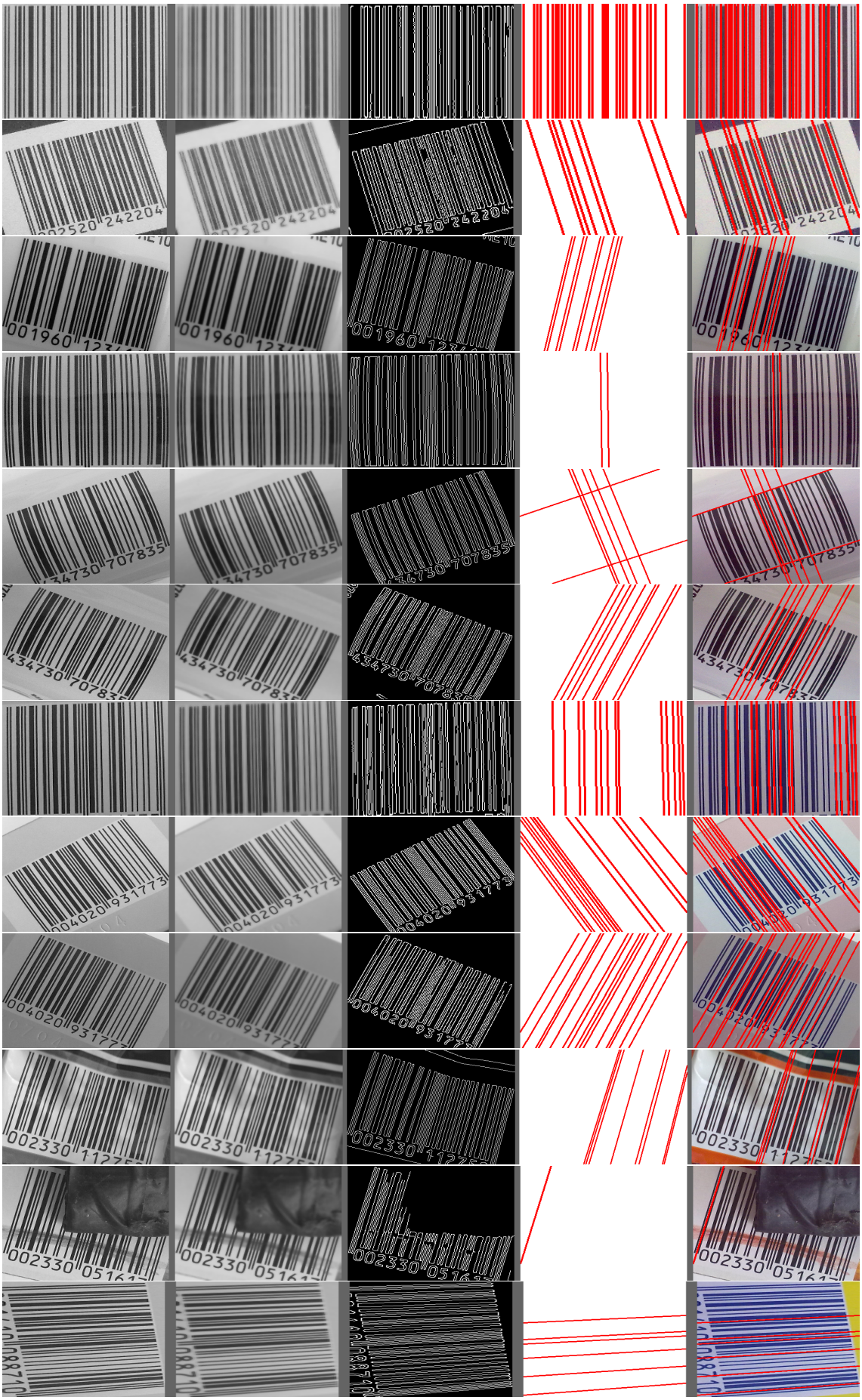

Figure 3: Image 3

Figure 4: Image 4

Figure 5: Image 5

Figure 6: Image 6

Figure 7: Image 7

Figure 8: Image 8

Figure 9: Image 9

Figure 10: Image 10

Figure 11: Image 11

Figure 12: Image 12

Figure 13: Image 13

Figure 14: Image 14