**Documentation for Problem 1**

**By Manish Kundu**

The problem statement asked us to highlight the medial axis of the moving needle in the video clips provided.

The solution had various parts in it-

1. The first task was to extract the frames from the video. The was already covered in the Winter School lectures.
2. From the frame, we had to subtract the foreground (i.e. the needle) from the background. I used cv2.createBackgroundSubtractorKNN(dist2Threshold=800, detectShadows=False) to get the foreground. I found the image containing considerably less noise as compared to another option available, cv2.createBackgroundGOG2().

KNN (K-Nearest Neighbours) is a machine learning algorithm. It basically compares a new data entry from the existing data entries and accordingly, puts it in a category or class of the nearest neighbour.

1. The next step was to clean the image to get rid of the noise and other imperfections. I used cv2.MORPH\_OPEN followed by cv2.MORPH\_CLOSE as I found this to be the cleanest procedure. I used cv2.getStructuringElement(cv2.MORPH\_ELLIPSE,(3,3)) as the kernel, which I found [here](https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_imgproc/py_morphological_ops/py_morphological_ops.html).
2. To obtain the edges from the cleaned image, I used GaussianBlur followed by the cv2.Sobel() operator. The Sobel Operator is a discrete differentiation operator. It computes an approximation of the gradient of an image intensity function.
3. After getting the edges, I used Hough Lines Transform to detect the lines. I implemented Hough Lines Transform from scratch, as advised.
4. To implement the Hough Lines Transform, I first defined the Hough space range. The range is the discrete possible values of rho and theta which are the voting candidates. The vote bank array to store the votes was then initialised. Then we looped through all the edge points(the pixels with non-zero values in the image after applying Sobel operator.

For each theta in the theta range, we calculated the rho value for the current point. Then we found the closest rho to the current rho value we calculated and increased the vote for the (current\_rho, current\_theta) in the vote bank by 1.

After this, all the lines (rho, theta) above the threshold were identified. I used np.where() to do this easily. Then I used np.vstack().T to finally get the final array of lines to be returned by the Hough Lines function.

1. The next part was to get the average of all the lines obtained by Hough Lines as a single line being displayed is desired. To do this, I basically obtained the average slope and average intercept of the lines. I had two separate cases for negative slope and positive slope as I found this to give more accurate average line then simply averaging all the available lines. I displayed the line with that final slope that had a greater number of lines obtained using Hough Lines.
2. The final step was to simple display the line using cv2.line().

**Challenges Faced –**

1. The image even after cleaning had some noise in it. Especially the part where the needle was moving rapidly. In this part, motion blur was visible in the cleaned image. There was also noise present when the needle was picking up the rings from there respective places.
2. The edge detection omitted some edges.
3. I implemented the Hough Lines Transform from scratch using python. I found my implementation to be slow compared to opencv’s implementation. I believe that the speed difference is due to the opencv implementation being programmed in c++, which allows faster running speed.
4. The hough lines was sometimes showing unnecessary lines, especially the rotating part of the needle was being detected sometimes.
5. The average of the lines was not always optimal.
6. The end point the needle cannot be found by standard hough transform.

**General Takeaways –**

The problem was fun, simple yet challenging. I learnt Hough Lines Transform properly, now I feel I can teach it to anyone easily. Coding Hough lines transform from scratch cleared the doubts I had about its functioning. I wish to improve the part where I took the average of the lines. I need to explore some alternative ways I can get the best line. Also to find the end point of the line, I found out that I need to use probabilistic hough transform. I tried reading a few papers, but most had mathematics involved which I do not know yet. I will definitely implement the probabilistic hough transform after I learn the mathematics behind it properly.