# Assignment 1

Gohil Dwijesh 2017CS50407
Rushil Gupta 2017CS10487

August 2019

## 1 Medial axis detection of Moving Objects

### 1.1 Algorithm:

1. Extraction of frames from the video under processing
2. Background subtraction: We used the in-built KNN background subtractor in OpenCV.
3. Preprocessing for Canny: erode function using variable sized kernels, blurring.
4. Canny edge detection : The processed image was given as input to the Canny Edge Detector.
5. Now, we calculate the number of pixels having the intensity greater than a particular threshold, let's say numpix. Based on the difference of the number with the number for the previous frames, we classify the current frame as:
   - Dynamic: In this state we calculate threshold(for votes) of hough line transform depending upon the size of the rod seen in the frame. If it seems that our rod size is almost same for x consecutive frames, we declare static state(to overcome time complexity issue), where we predict threshold for next y(here y = 2x) consecutive frames.
   - static: We'll use the predicted threshold for Hough transform.
6. We apply the Hough transform to detect lines in the frame. If we don't get enough number of lines then we reduce the threshold by some amount and apply Hough transform. Repeat the same until we get enough number of lines.
7. Now we get enough number of lines. We will store each lines in a min heap, ordered with respect to theta of each line, so that we can get parallel lines as we pop lines from the heap.
8. Now we get a set of tuple lines which are parallel. While calculating the average for finding the medial line, we consider only those tuple lines whose theta is within certain limit of the learned theta. We take average of tuple lines that follow the mentioned criteria to get middle line, finally take average of each of the middle lines to get a medial axis of the rod.
9. Now we get the medial axis. While doing this we are also maintaining past few theta values of medial axis. We used these theta values to ignore unwanted parallel lines that does not correspond to orientation of the rod.
10. Now, we have a default initial value for length of rod. Based on difference in numpix for two consecutive frames, we compute sigmoid if difference $<0$ and multiply it with previous length value or compute 1/sigmoid and multiply it with previous length if difference $>0$. In case of difference = 0, we reduce the length by a very small value.
11. We also multiplied the length by a factor dependent on current theta and previous frame's theta to count for the change in the orientation of the rod and suitably modify the length.
12. Finally we draw the segment on the background image.

## 1.2 Classes:

- Line: stores line as rho and theta
- Node, Linked_list : store linked list of lines.
- Myheap: Store lines, ordered with respect to theta.
- stochastic_node, stochastic_list: maintain past history of good theta values. It helps in learning good theta value from the past history.

## 1.3 Functions:

- number_of_ones: calculates number pixels having intensity value greater than 127 for a given frame.
- video_to_frames: splits the video into frames.
- video_to_knnFrames: First splits video into frames and then it uses KNN background subtractor and stores the result in a directory.
- pre_canny: It uses erode and blurring function to remove the noise.
- check_two_sides: It checks, for a given two or three lines, whether these lines belongs to the extreme parallel edges of the rod or not.
- draw_hough_lines: Given line parameters, it draws a line on the given background image.
- get_avg_lined: It returns the average of the given lines.
- hough_lines: It finds the medial line of the rod and draws the corresponding segment on the given background image.
- knn_to_hough_frame: It takes the background subtracted image as input, performs canny edge detection followed by a call to hough_lines function that performs hough line detection and returns the background image with the medial line drawn. This function outputs the above specified image which is then added to the video.

## 1.4 Link to results:

https://drive.google.com/drive/folders/1ffQIG23lgH5SvHj2hcGdQwWdGNhw8qMl?usp=sharing