Problem 1

- Indoor Wall Painting Robot

    1. Cameras: A camera can be used to detect which part of the wall has been painted and which has not been painted and for obstacle detection and obstacle avoidance by the mobile base. The camera can also be used to make sure that the right color is being used for painting a wall.
    2. Proximity Sensors: Proximity sensors will be used to make sure that the robot does not crash into the wall, floor or ceiling. They can also be used to to detect if the robot is far from the wall so as to stop painting.
    3. PGV or Color sensors: An alternative to Cameras which can be used to detect colors and if a wall is painted or not
    4. 2D/3D LIDARs: These can be used by the mobile base of the robot to navigate in the room and detect and avoid obstacles along with the cameras. They are also be used to make map / 3D model of the room for the robot.

- 1. Cameras: Camera will be used to detect and segment areas of the pipe which are broken. It will also be used to make sure the robot is fixing the pipe properly. Lets take wielding a pipe for example. The Cameras will give the wielding fixtures the correct co-ordinates of the broken part of the pipe and how much of the pipe has been fixed.
    2. Ultrasonic Sensors / Sonar: These can be used to detect obstacles and the bed of the water body. They can also be used to detect cracks and fissures in a pipe.
    3. GPS: A GPS can be used to get the position of the robot and perceive its location in the world
    4. Gas Sensors: Gas sensors can be used to detect gas leaks from the pipelines.

Problem 2

- First we convert the frames of the video into gray scale and apply a Gaussian Blur over it to avoid noise.
- Then we pass this gray scale frame to the *HoughCircle* function in OpenCV. This function runs a Canny edge detector and then finds circles using the Hough Transformation technique.
- Once we get the circles form the *HoughCircle* function we then draw them over the frame and display it.[1]

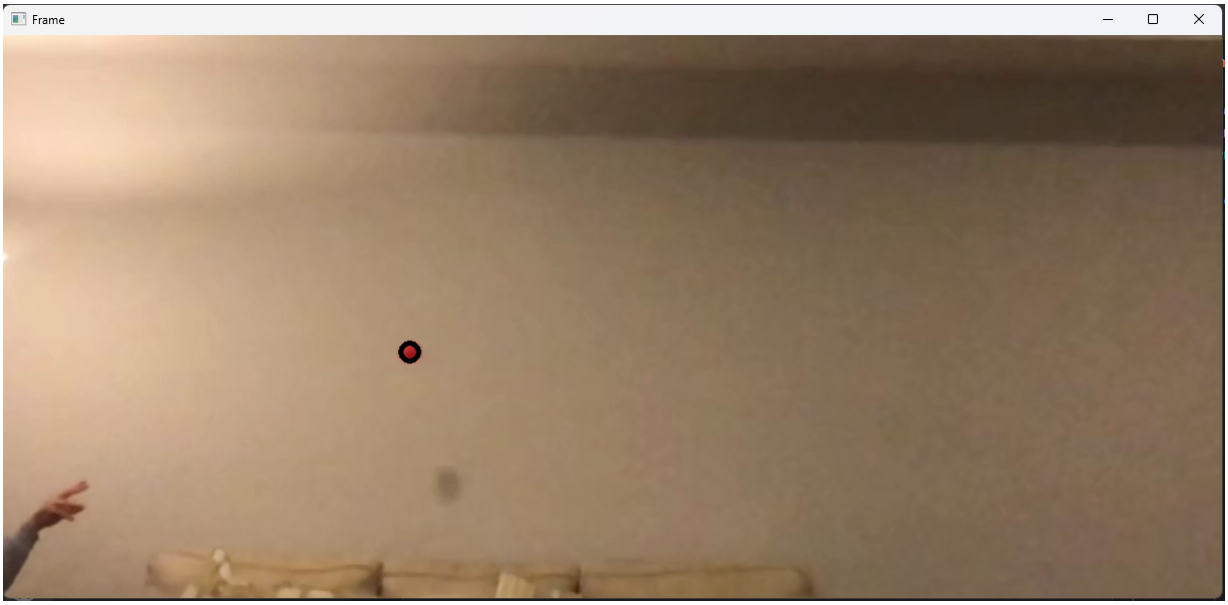

Figure 1: Black line detecting ball in video

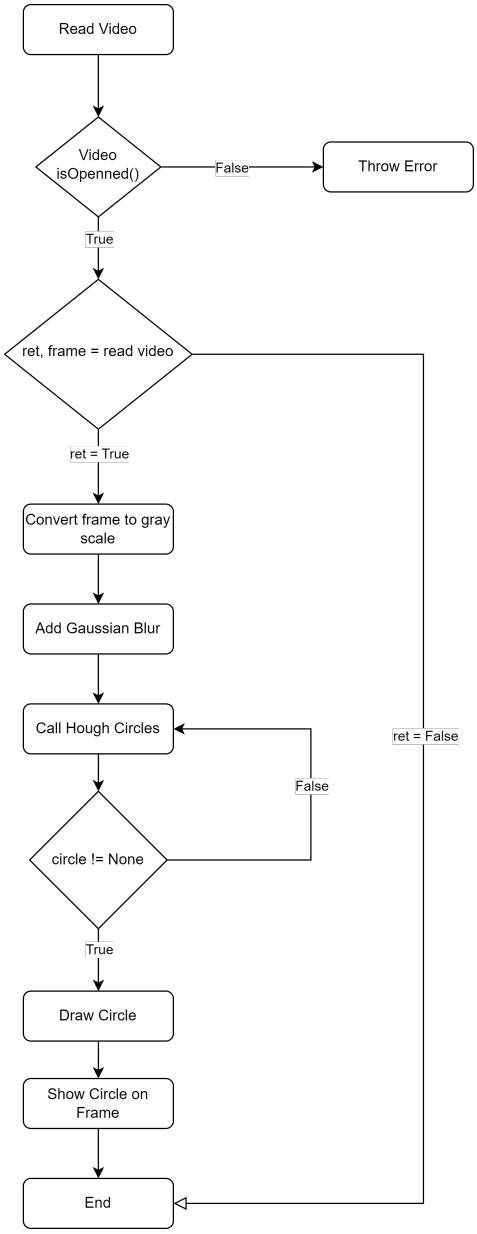

Figure 2: Question 2 Flowchart

Problem 3

- First we convert the image into gray scale and blur to pass it into the canny edge detector.
- We then pass the output of the canny detector to the *HoughLines* function by Open CV.
- We then draw the major Hough Lines that belong to the tain tracks
- We then find the vanishing point of the tracks
- We then crop the image such that only the tracks are visible
- We then find the homography for the cropped image to get the top view.
- the top view image is then used to calculate new Hough Lines which are used to calculate the distance between each row.
- The Average distance between each comes out to be 90.71 pixels

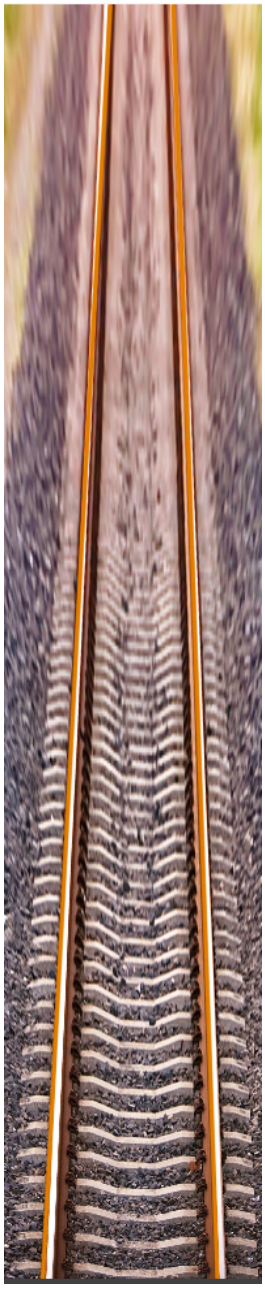

Figure 3: Top View of the Train Tracks

Problem 4

- First we convert the image into gray scale and blur to get the edges using canny edge detector.
- Then we erode and dilate the output of the canny detector to fill in gaps and remove unwanted lines.
- We then find the contours of the image using the $findContours$ function in Open CV
- We then draw bounding boxes along the contours. We also remove the contours which do not bound balloons by making use of the $approxPolyDP$ function provided by Open CV [2]
- The total number of contours detected gives us the number of balloons

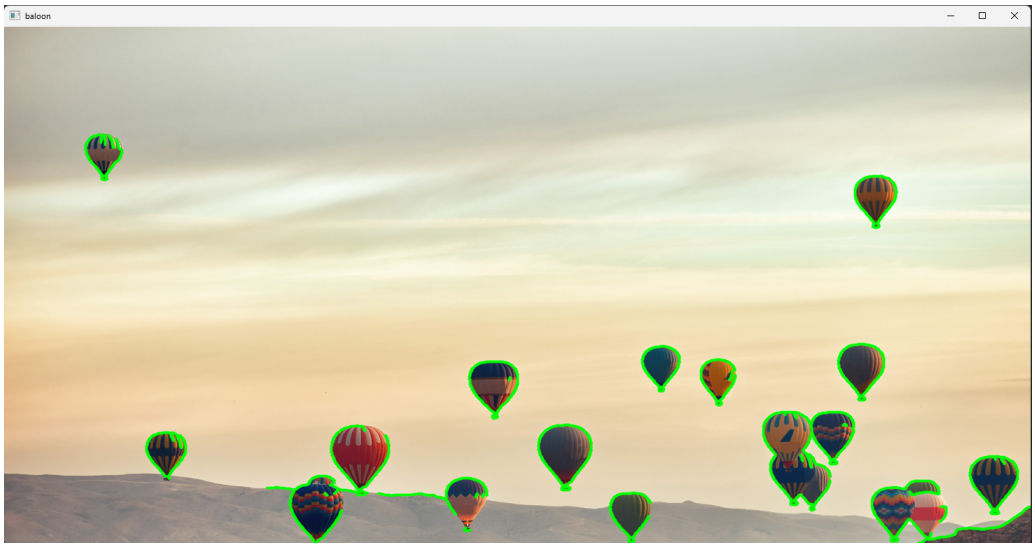

Figure 4: Balloon's Contours

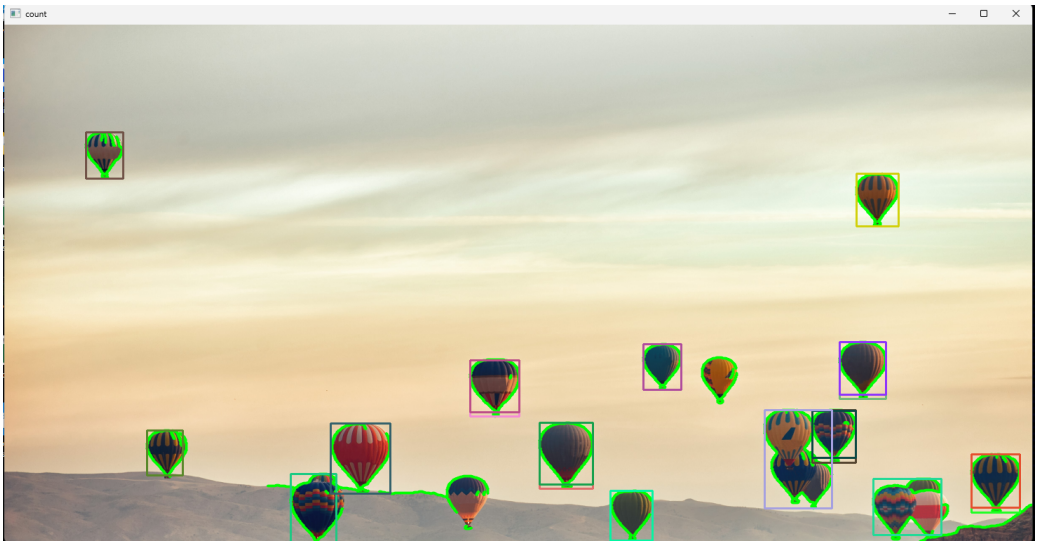

Figure 5: Detected Balloons

Problem 5

K-Means Clustering: We have the numbers $25, 13, 2, 11, 4, 6, 15, 22$ which are to be made into 3 clusters.

- 1^{st} Iteration: Assume the centre as follows:

$$c_1 = 5$$
$$c_2 = 10$$
$$c_3 = 20$$

  We assign 2, 4, 6 to c_1, 11, 13 to c_2 and 15, 22, 25 to c_3.

- 2^{nd} Iteration

  We update the centres as:

$$c_1 = \frac{2+4+6}{3} = 4$$
$$c_2 = \frac{11+13}{2} = 12$$
$$c_3 = \frac{15+22+25}{3} = 20\frac{2}{3}$$

  We assign 2, 4, 6 to c_1, 11, 13, 15 to c_2 and 22, 25 to c_3

- $3^{rd} Iteration$

  We update the centres as:

$$c_1 = \frac{2+4+6}{3} = 4$$
$$c_2 = \frac{11+13+15}{3} = 13$$
$$c_3 = \frac{22+25}{2} = 23\frac{1}{2}$$

  We assign 2, 4, 6 to c_1, 11, 13, 15 to c_2 and 22, 25 to c_3

- 4^{th} Iteration

  This is the same as the 3^{rd} Iteration hence the centres do not move and the clusters have been formed.

References

1. OpenCV Hough Circle Transform. OpenCV Documentation. Retrieved March 16, 2023, from https://docs.opencv.org/3.4/d4/d70/tutorial_hough_circle.html

2. OpenCV (n.d.). Creating Bounding boxes and circles for contours. OpenCV Documentation. Retrieved March 16, 2023, from https://docs.opencv.org/3.4/da/d0c/ tutorial_bounding_rects_circles.html

3. Parts of the Code for Question 3 were taken from previous assignments