EC6090 - ROBOTICS AND AUTOMATION

MINI PROJECT

LEGO-BASED MOBILE ROBORT NAVIGATION AND STRUCTURE IDENTIFICATION

Department of Electrical and Electronic Engineering

University of Jaffna

JUNE 2025

Group 31

2021/E/102 – SALMAN M.M

2021/E/117 – JAMES P.S.V

2021/E/136 – BANDARA S.M.C.U

2021/E/171 – DULANYA W.D.A

CONTENT

1.INTRODUCTION

This project focuses on the development of an autonomous LEGO EV3 robot capable of navigating a circular line track and identifying the shape of a central structure using ultrasonic sensor measurements. The robot follows the track using dual color sensors and collects distance data as it moves. After completing a circular loop, it calculates the average distance to the structure and classifies the shape as either a square, triangle, or circle using a simple mathematical model.

2.OBJECTIVE

- To implement autonomous line-following behavior using two color sensors.
- To record distance data from an ultrasonic sensor during motion.
- To calculate the average distance over time.
- To compare the average against a fixed threshold to determine the structure's shape.
- To display the result via screen and LED color on the EV3 brick.

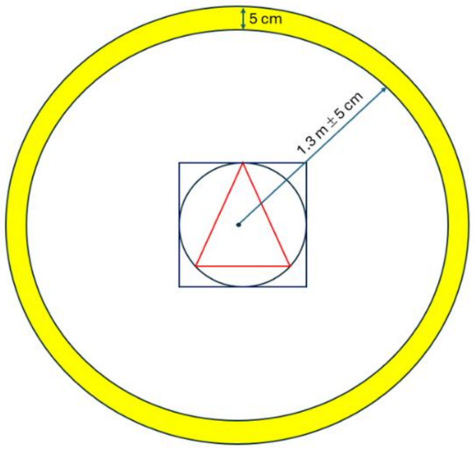


Figure 1: TESTING AREA

3. HARDWARE COMPONENTS

- LEGO EV3 Brick
- 2 Large Motors (connected to ports A and D)
- 2 Color Sensors (connected to ports 1 and 3)
- 1 Ultrasonic Sensor (connected to port 4)
- Circular yellow line on a gray surface
- Central structures: Square, Triangle, Circle

4.METHODOLOGY

4.1 Navigation and Line Following

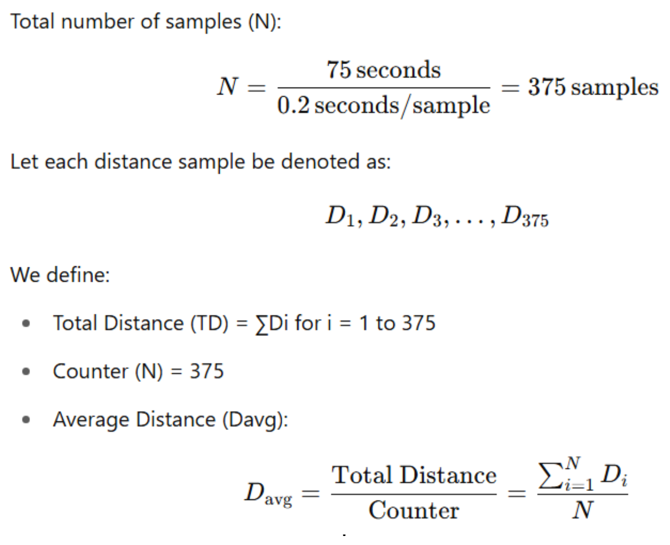
The robot follows a circular black line using dual color sensors:

- If both sensors detect reflected light intensity > 67 (white surface): Robot turns right (left motor slower)
- Else: Robot turns left (right motor slower)

This logic keeps the robot aligned with the black line throughout the circular path.

4.2 Distance Sampling

As the robot moves, it records the distance from the central object using the ultrasonic sensor at 0.2-second intervals for 75 seconds.



Total number of samples (N):

$$N = \frac{75\,\text{seconds}}{0.2\,\text{seconds/sample}} = 375\,\text{samples}$$

Let each distance sample be denoted as:

$$D_1, D_2, D_3, \ldots, D_{375}$$

We define:

- Total Distance (TD) = ∑Di for i = 1 to 375
- Counter (N) = 375
- Average Distance (Davg):

$$D_{\text{avg}} = \frac{\text{Total Distance}}{\text{Counter}} = \frac{\sum_{i=1}^{N} D_i}{N}$$

4.3 Shape Classification Using Threshold

threshold value is used to normalize the average distance:

- Threshold (T) = 90 cm
- Final Value (F) = Davg − Threshold

Shape classification logic:

- If $F \geq 50 \rightarrow$ Shape = Square
- If $F \geq 20 \rightarrow$ Shape = Circle
- Else $\rightarrow$ Shape = Triangle

This logic is based on the intuition that:

- A square has large flat sides → higher average distance
- A circle has small width → lower average distance
- A triangle falls in between

4.4 Display Logic

Once the shape is detected:

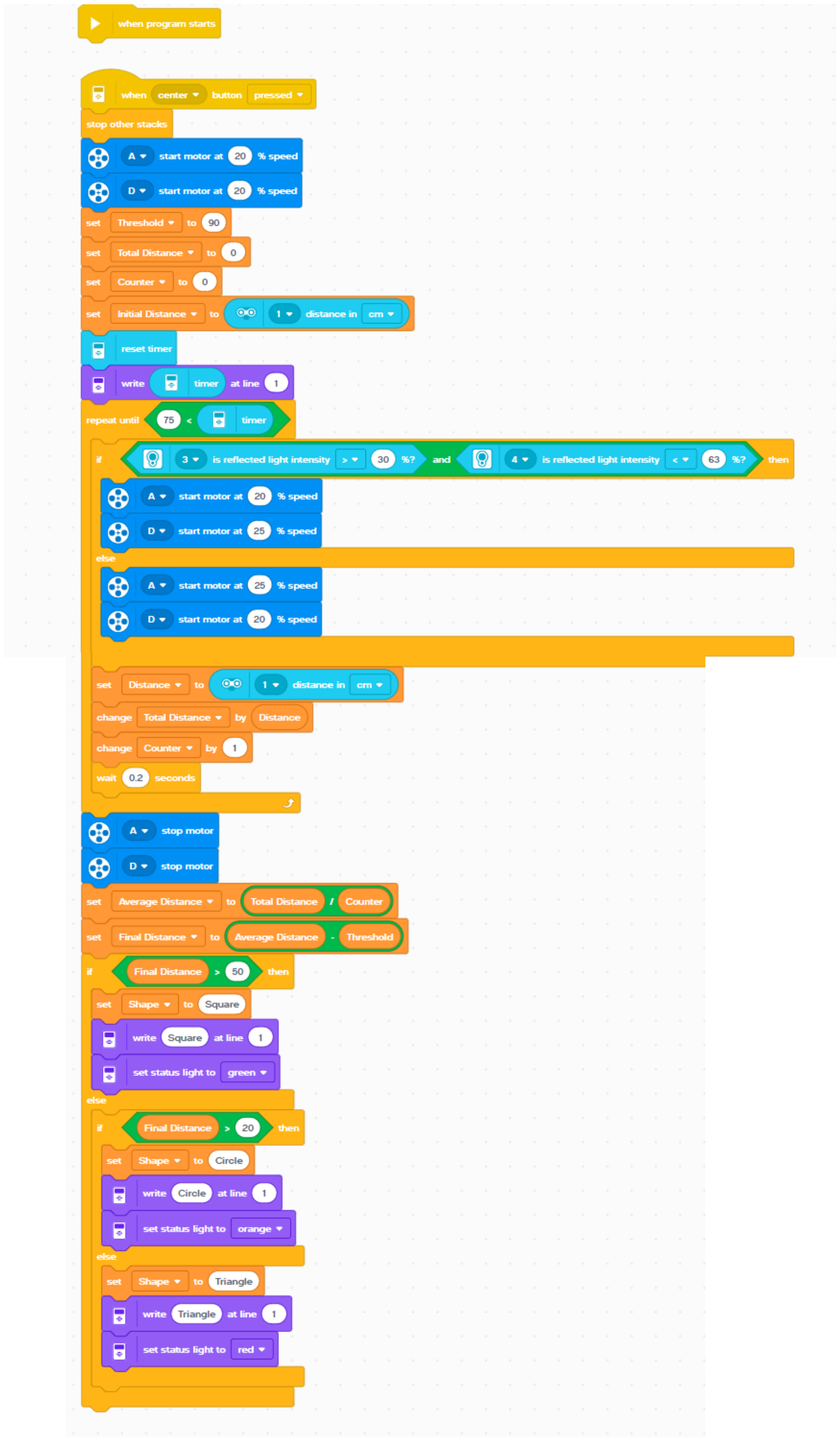
- It is shown on the EV3 display screen
- The EV3 status light changes color:
  - Green for Square
  - Orange for Circle
  - Red for Triangle

5.FINAL LEGO SCRATCH PROGRAM BEHAVIOR

The code:

- Starts motors A and D at 20% speed
- Initializes variables: Total Distance = 0, Counter = 0, Threshold = 90
- Runs a loop for 75 seconds
    - Every 0.2s:
        - Reads ultrasonic distance (D)
        - Adds D to Total Distance
        - Increments Counter
- After 75s:
    - Computes Davg = Total Distance / Counter
    - Computes Final Value = Davg − Threshold
    - Classifies shape and displays it

6.SOURCE CODE



```
when program starts

when center button pressed
stop other stacks
A start motor at 20 % speed
D start motor at 20 % speed
set Threshold to 90
set Total Distance to 0
set Counter to 0
set Initial Distance to 1 distance in cm
reset timer
write timer at line 1
repeat until 75 < timer
    if 3 is reflected light intensity > 30 %? and 4 is reflected light intensity < 63 %? then
        A start motor at 20 % speed
        D start motor at 25 % speed
    else
        A start motor at 25 % speed
        D start motor at 20 % speed
    set Distance to 1 distance in cm
    change Total Distance by Distance
    change Counter by 1
    wait 0.2 seconds
A stop motor
D stop motor
set Average Distance to Total Distance / Counter
set Final Distance to Average Distance - Threshold
if Final Distance > 50 then
    set Shape to Square
    write Square at line 1
    set status light to green
else
    if Final Distance > 20 then
        set Shape to Circle
        write Circle at line 1
        set status light to orange
    else
        set Shape to Triangle
        write Triangle at line 1
        set status light to red
```

7.RESULTS

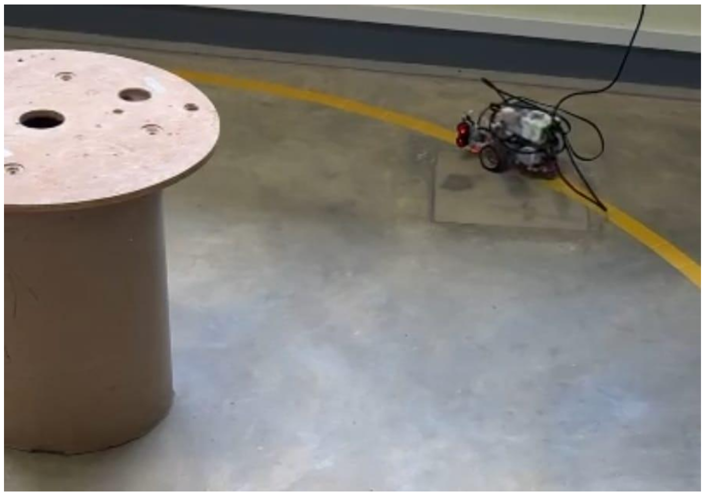


Figure 2: CIRCLE DETECTION

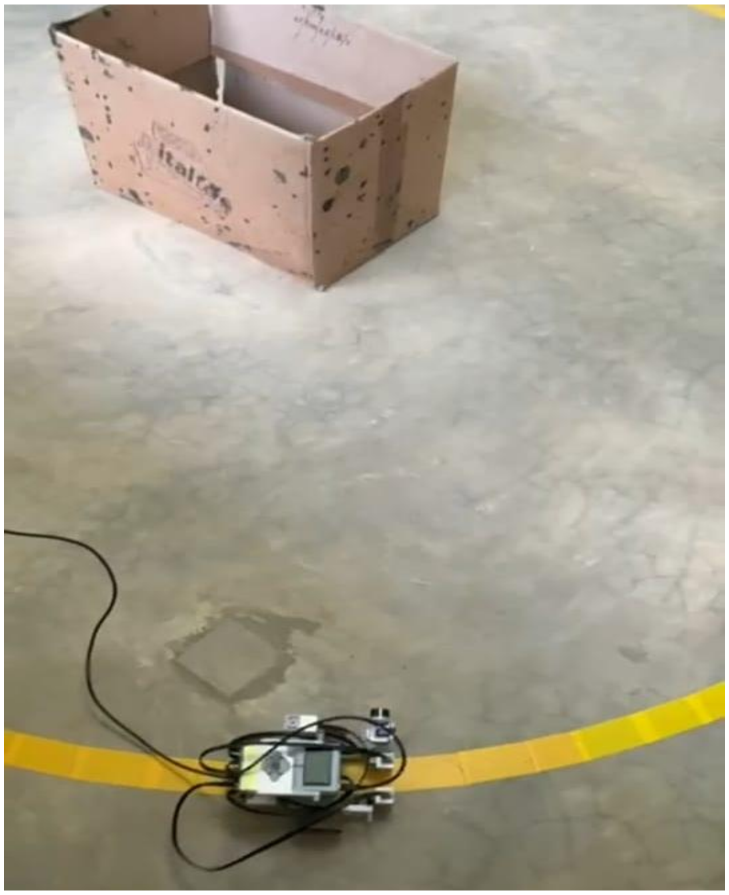


Figure 3: SQUARE DETECTION

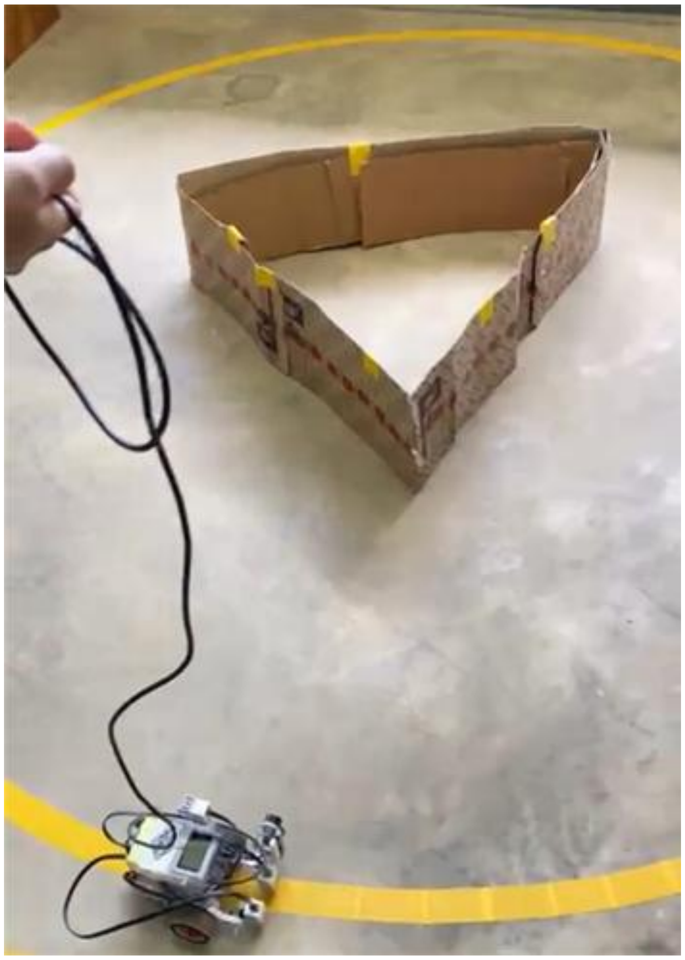


Figure 4: TRIANGLE DETECTION

When there is no object it show us a green light , because the average distance should be greater than 50.

8.DISCUSSION

This project successfully combined line-following navigation and shape classification using a LEGO EV3 robot equipped with color and ultrasonic sensors. The robot was designed to follow a circular black line while recording its distance to a central object placed inside the circle. After completing one loop, it calculated the average distance and used that information to determine whether the object was a square, triangle, or circle.

The use of dual color sensors allowed the robot to reliably follow the circular track with minimal deviation. The simple decision logic based on reflected light intensity (turning left or right when off-track) was sufficient for smooth navigation. While more advanced line-following methods like PID control could improve the precision and smoothness of motion, the current approach proved adequate for the goals of this project.

The ultrasonic sensor played a crucial role in shape detection. By collecting distance readings at 0.2-second intervals over 75 seconds, the robot generated a large set of data points. These measurements were added together to find the total distance, and then averaged to determine a representative value. A fixed threshold value of 90 cm was subtracted from the average to compute a “Final Value,” which simplified the shape classification logic.

The final value served as a normalization tool to minimize the impact of sensor drift or environmental variability. Based on our testing, the resulting values clearly separated the three shapes:

- Final Value $\geq$ 50: Square
- Final Value $\geq$20: Circle
- Else: Triangle

This method worked well because the geometry of each shape affects the robot’s distance readings:

- Squares have broad, flat faces, resulting in higher average distances.
- Circles are evenly curved and generally allow the robot to stay closer.
- Triangles have sharp corners and sloped sides, producing medium or more varied distances.

Overall, the system’s logic was simple yet effective. The robot’s classification results were accurate and consistent across multiple test runs. In addition, the use of status lights and display messages provided clear visual feedback for the user, making it easy to verify the robot’s decision.

However, there are areas for improvement. Distance readings occasionally fluctuate due to ultrasonic sensor noise or vibration during movement. This could be addressed by averaging several readings per sample or using a median filter. Furthermore, adding more sophisticated navigation (e.g., PID-based control) would improve path stability, especially at higher speeds or on complex paths.

In conclusion, the project demonstrates that basic robotics principles line following, sensor data collection, averaging, and threshold comparison can be combined effectively to solve a classification problem. It provides a strong foundation for more advanced robotic systems that use sensors and decision logic to interpret their surroundings.

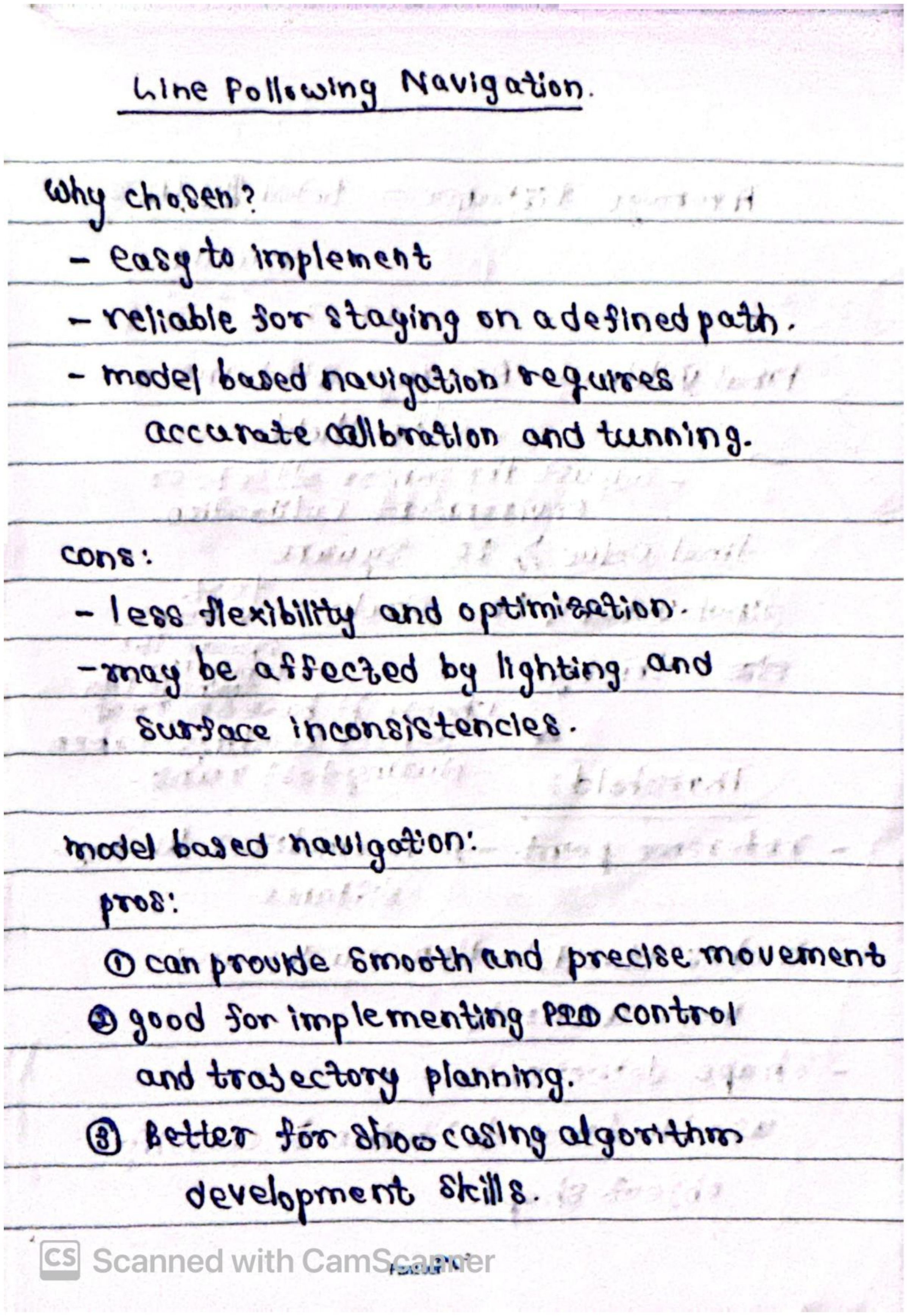
9.CONCLUSION

This project successfully integrates line-following navigation, sensor-based data logging, and conditional classification into a single LEGO EV3 system. By using a threshold-based method to normalize distance values, we overcome minor variations caused by sensor drift or environmental noise.

The robot is able to:

- Navigate autonomously using real-time sensor input
- Make logical decisions based on distance modeling
- Provide feedback through display and lights

10.ROUGH WORK



Line Following Navigation.

Why chosen?
- easy to implement
- reliable for staying on a defined path.
- model based navigation requires accurate calibration and tunning.

cons:
- less flexibility and optimization.
- may be affected by lighting and surface inconsistencies.

model based navigation:
pros:
① can provide smooth and precise movement
② good for implementing PID control and trajectory planning.
③ Better for showcasing algorithm development skills.

Average Distance = $\frac{\text{total Distance}}{\text{counter}}$

0.2 [illegible]

Final ~~Distance~~ = Average Distance − threshold −

- adjust for sensor offsets or environment Calibration.

final value $\geq$ 80 Square

final value $\leq$ 30 circle

else triangle

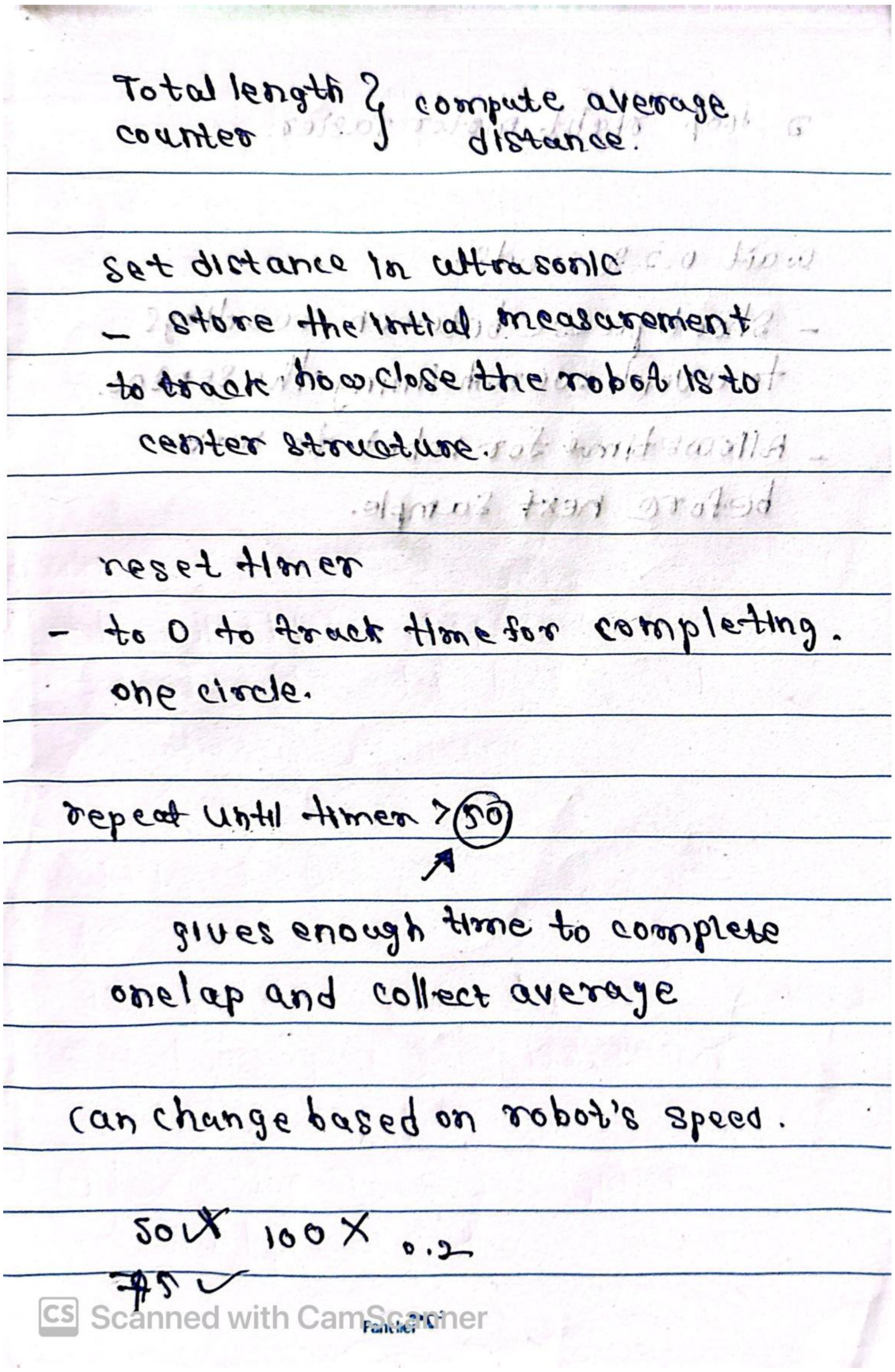
test values
square 160
circle 120
triangle 140

← choose it base on real sensor readings taken during test runs.

Threshold :

- reference point → normalizes average distance
- makes classification easier and more accurate.
- shape detection.

used in final detection to classify object shape.



Total length } compute average
counter } distance.

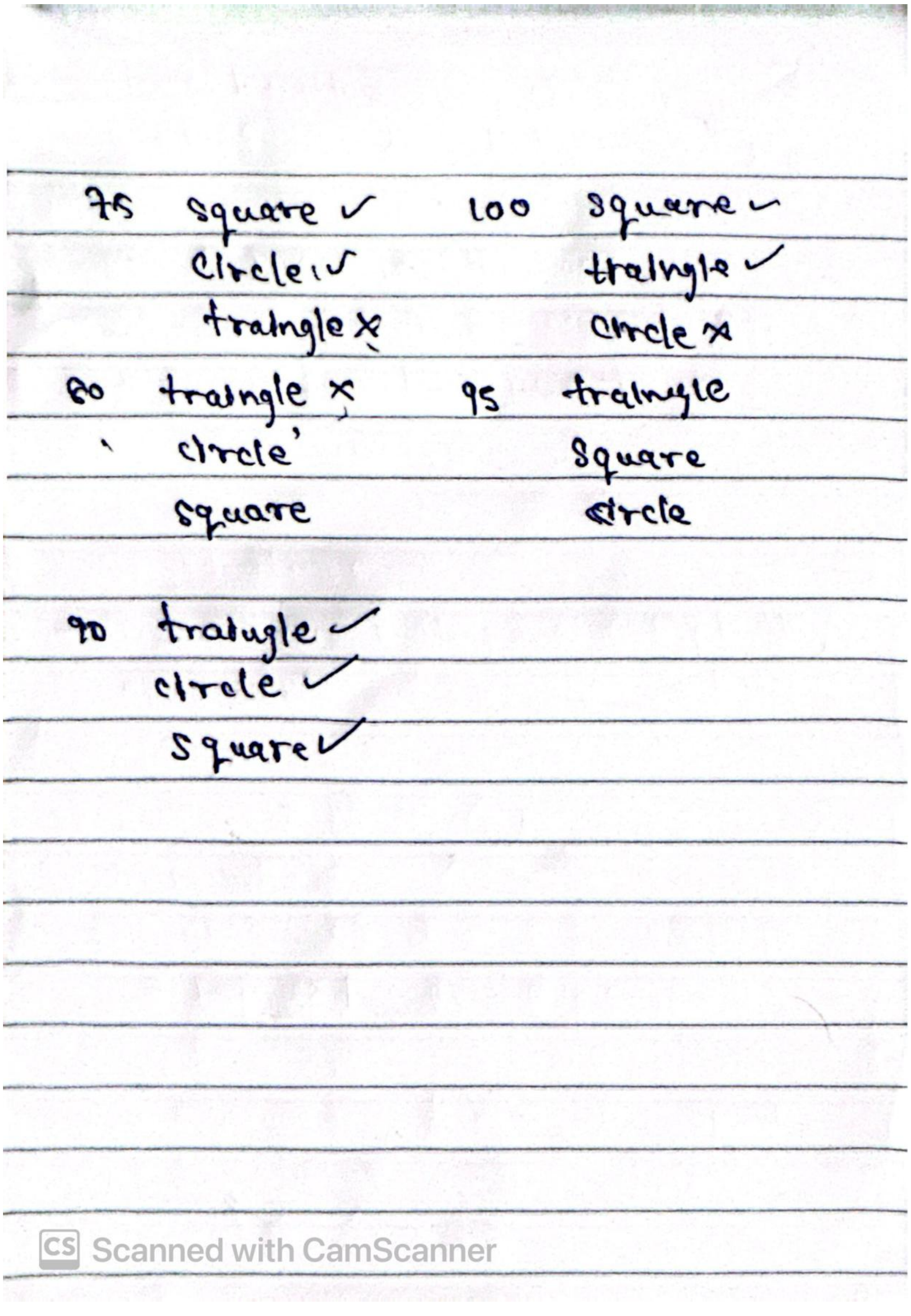
set distance in ultrasonic
- store the initial measurement
to track how close the robot is to
center structure.

reset timer
- to 0 to track time for completing.
one circle.

repeat until timer > $\boxed{50}$
↗
gives enough time to complete
onelap and collect average

can change based on robot's speed.

50 ✗ 100 ✗ 0.2
75 ✓



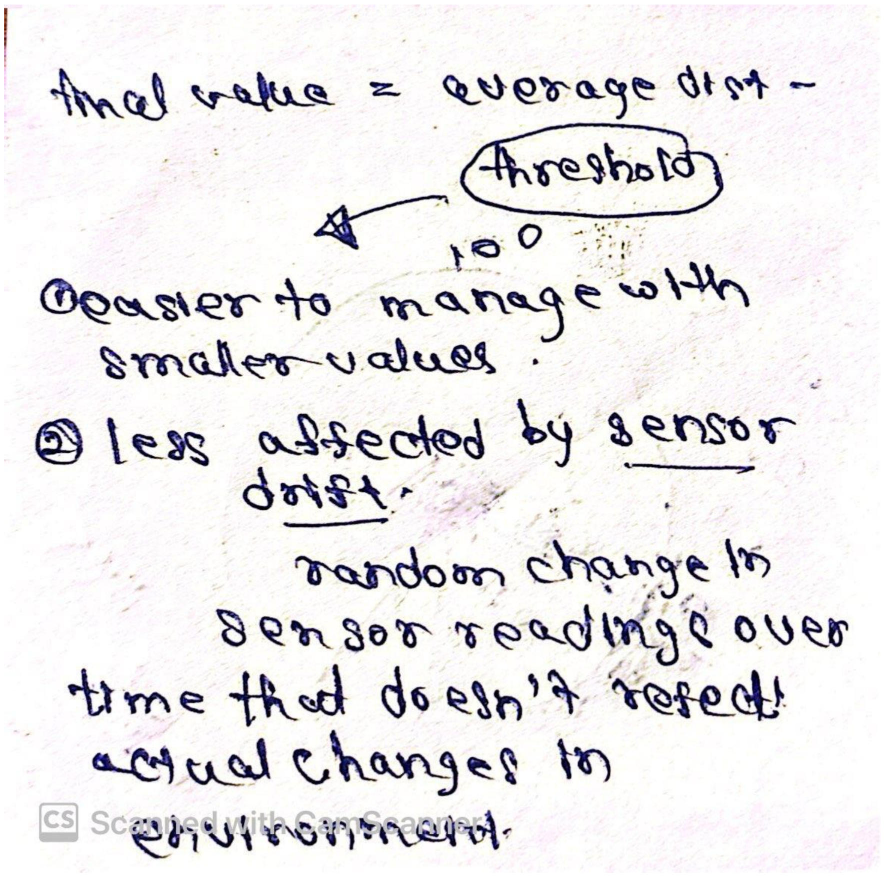
75 square ✓
circle ✓
traingle ✗

80 traingle ✗
circle'
square

90 traingle ✓
circle ✓
square ✓

100 square ✓
traingle ✓
circle ✗

95 traingle
square
circle



final value = average dist − $\boxed{\text{threshold}}$

100

① easier to manage with smaller values.

② less affected by sensor drift.

random change in sensor readings over time that doesn't reflect actual changes in environment.