



VISVESVARAYA TECHNOLOGICAL UNIVERSITY
“Jnana Sangama” Belagavi-590018, Karnataka

JSS MAHAVIDYAPEETHA



JSS Academy of Technical Education, Bengaluru
Department of Robotics & Automation

Presentation on

TITLE: SIMULATION OF MAZE SOLVING ROBOT

PROJECT GUIDE

K S MAHESH

ASSISTANT PROFESSOR

DEPARTMENT OF ROBOTICS AND AUTOMATION

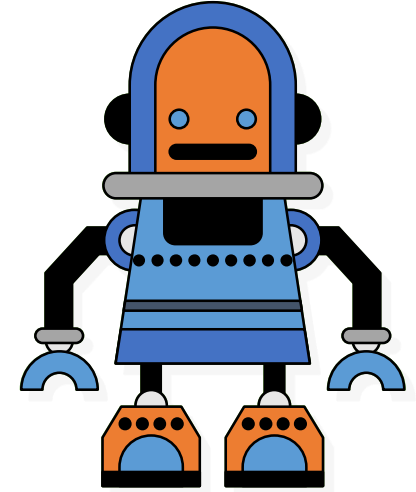
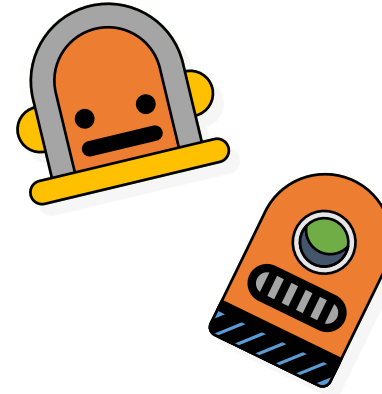
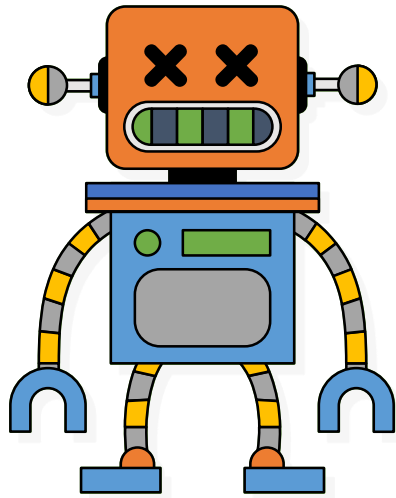
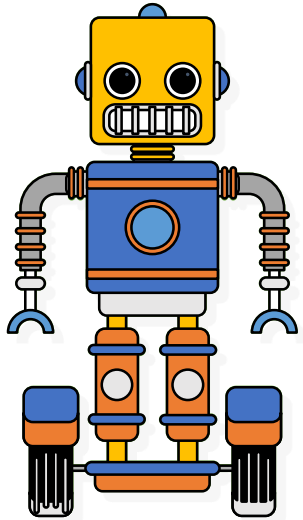
BATCH NO: 03

- | | |
|-------------------|-------------|
| 1. R B Kailash C | -1JS21RA013 |
| 2. Rohith | -1JS21RA015 |
| 3. Sudhanva V A B | -1JS21RA018 |
| 4. Vikas A Ram | -1JS21RA023 |

INTRODUCTION

OVERVIEW

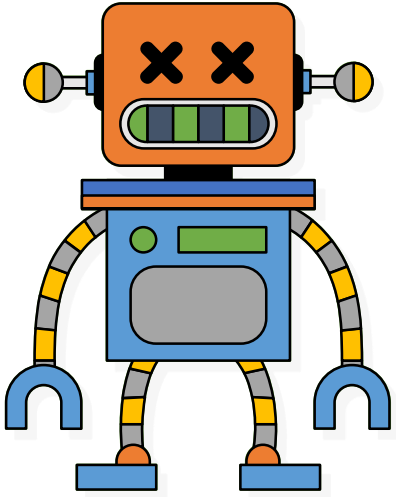
This project develops an autonomous robot that navigates mazes using sensors to detect obstacles and an algorithm to find the shortest path to the exit. Equipped with infrared and ultrasonic sensors, a microcontroller, and motors, the robot demonstrates efficient maze-solving capabilities. Potential applications include search and rescue missions, automated delivery systems, and robotic pathfinding.



OBJECTIVES AND GOALS

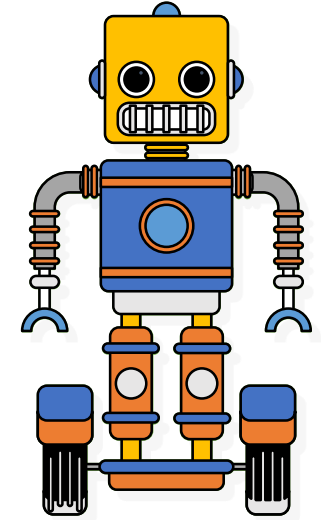
- ❖ Develop an autonomous robot capable of navigating through a maze and finding the shortest path to the exit
- ❖ Implement algorithms to efficiently explore the maze and determine the optimal route, such as the A* algorithm
- ❖ Explore potential real-world applications of maze-solving robots including search and rescue missions, automated delivery systems, and robotic pathfinding

PROBLEM STATEMENT



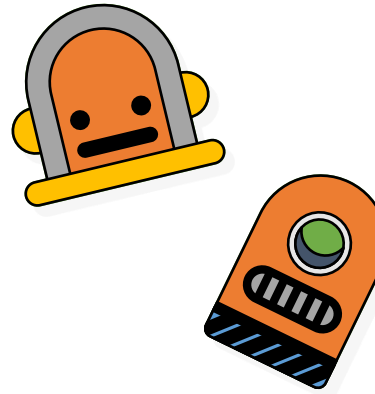
Exploration and Mapping

- Uncharted territories: Mapping unknown environments like caves, forests, or underwater terrains.
- Planetary exploration: Sending robots to explore other planets or moons.





Search and Rescue



- Urban environments: Navigating through collapsed buildings or disaster zones to locate survivors.
- Underground environments: Exploring caves or mines for trapped individuals.

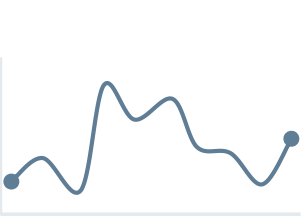


Logistics and Warehousing

- Inventory management: Optimizing warehouse layouts and locating items efficiently.
- Autonomous delivery: Navigating through complex indoor environments to deliver packages.

 TITLE	AUTHOR	WORKING FIELD	SUMMARY 
Design and Implementation of Autonomous Maze-Solving Robot based on an Enhanced Maze Solving Techniques	R Ramesh; D Selvalakshmi; M Naven Karthick; S L Nikileshkumar; S Cibivarshan; K Prasanth	IEEE [Institute of Electrical and Electronics Engineers]	This article discusses a self-governing robot that uses distance sensors and wheel pivot encoders to navigate and solve mazes by storing past information and following the shortest path.
Maze Solving Robot-B-Eng Design Project [2016]	Mohammed-Ahmed	Department of Electronic and Electrical Engineering	The details the development of two autonomous robots for maze navigation, using microcontrollers, sensors, wireless communication, and data loggers.
An Autonomous Maze-Solving Robotic System Based on an Enhanced Wall-Follower Approach	Shatha Alamri ,Hadeel Alamri ,Wejdan Alshehri ,Shuruq Alshehri ,Ahad Alaklabi and Tareq Alhmiedat	College of Computer Sciences and Information, King Saud University, Riyadh 11421, Saudi Arabia	Reviewing the recently developed wall-follower maze-solving robotic systems. Designing and implementing a maze-solving robotic system by employing a modified wall-follower method.



METHODOLOGY

Chassis Assembly:

- Assemble the robot chassis, attaching the motors and wheels securely.
- Mount the MSP430 development board on the chassis.

Motor and Motor Driver Connection:

- Connect the motors to the motor driver module.
- Connect the motor driver to the MSP430 microcontroller. For example, connect the motor driver’s input pins to the MSP430’s GPIO pins.

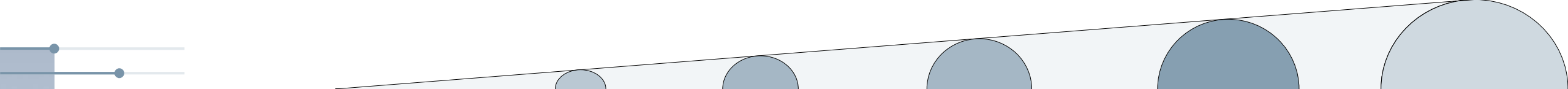
Sensor Placement and Connection:

- Place the sensors (e.g., IR sensors) on the front and sides of the robot for obstacle detection.
- Connect the sensor output pins to the MSP430’s ADC (Analog to Digital Converter) pins or digital input pins.

Power Supply:

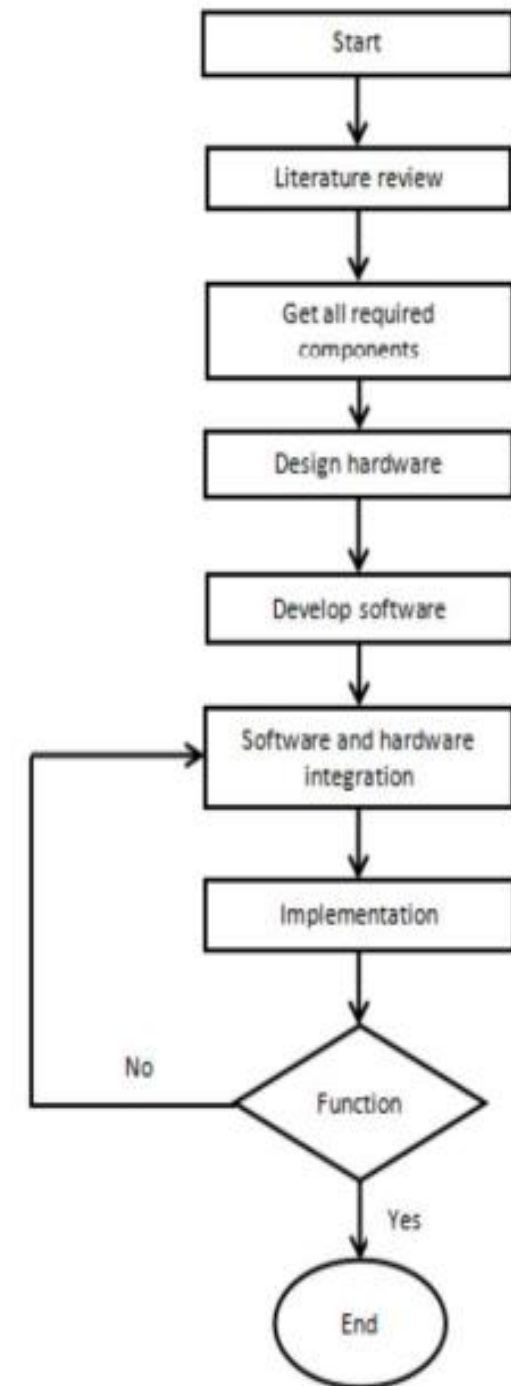
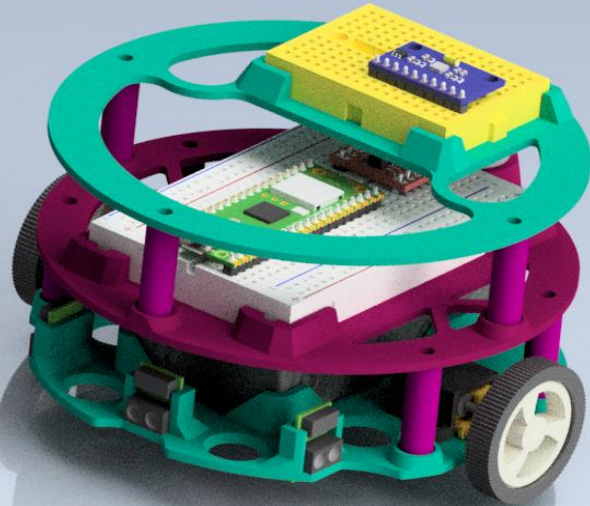
- Connect a battery pack to power the motors and the MSP430 board.
- Ensure proper voltage regulation for the MSP430 (typically 3.3V).

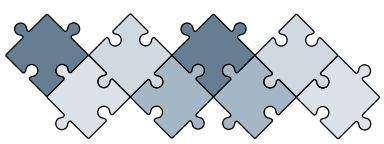
ITEM	QUANTITY	APPROXIMATE COST (INR)
MSP430(GR553)	1	4100
L298N Motor Driver	1	150
N20 Mini Motor	2	200 (100 each)
Ultrasonic Sensor	2{if needed}	200 (100 each)
IR Sensors	3	150 (50 each)
Li-Po Battery (12V)	1	1000
Jumper Cables	1 set	100
Chassis	1	300
ESP32 WIFI Module	1	500
Breadboard	1	100
BUCK Convertor	1	100



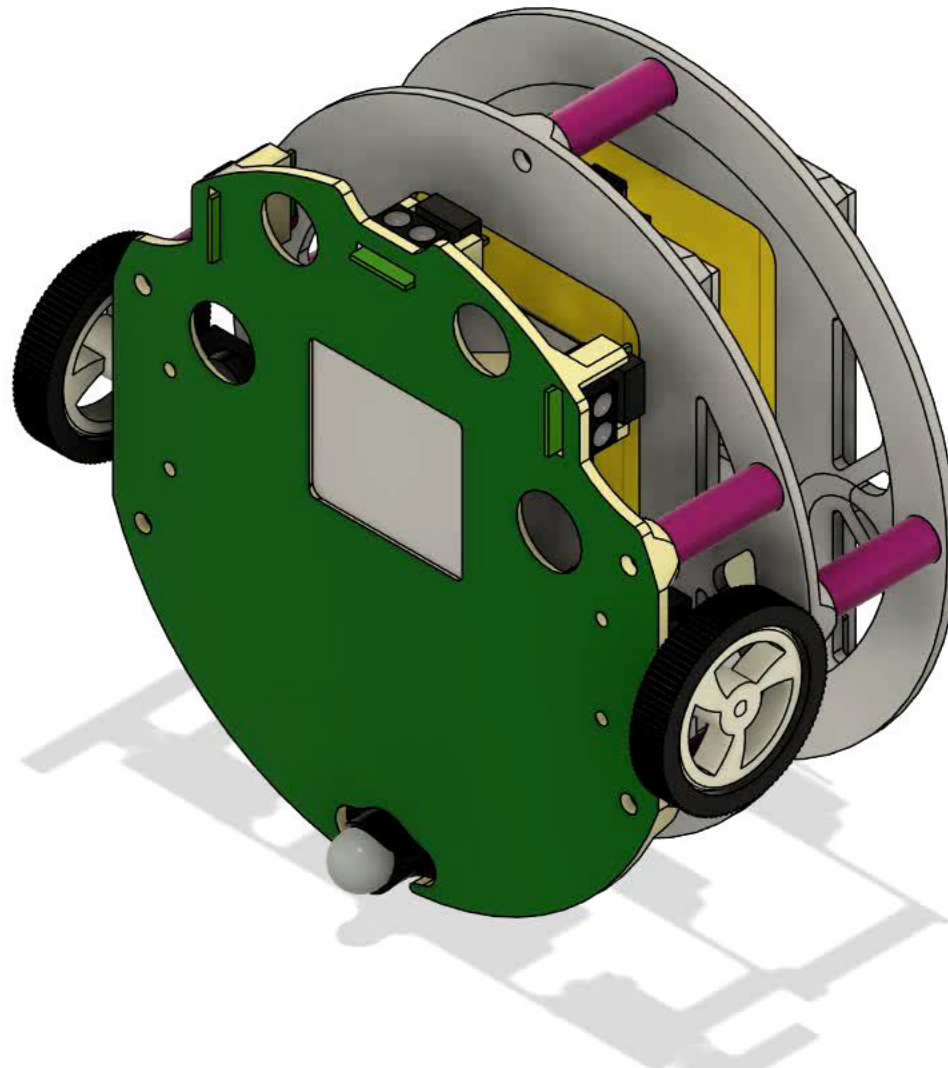
DESIGN

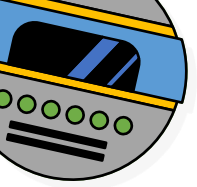
- Understand how maze solving robots work by referring to previous research papers.
- Understand different algorithm and figure out a way to use it to solve maze.
- Develop a code for finding the shortest path to solve the maze using suitable algorithm in with the help of **Code Composer Studio** .
- Making a 6' * 6' maze to test the robot.





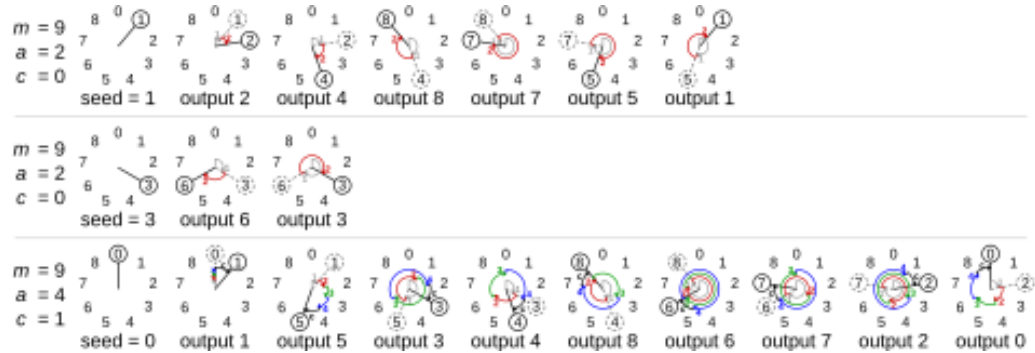
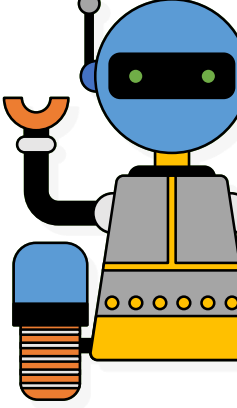
CAD MODELS





TESTING AND VALIDATION

RANDOM MOUSE

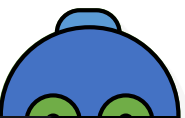
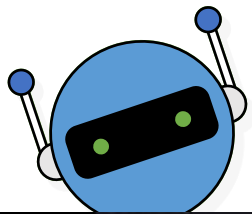
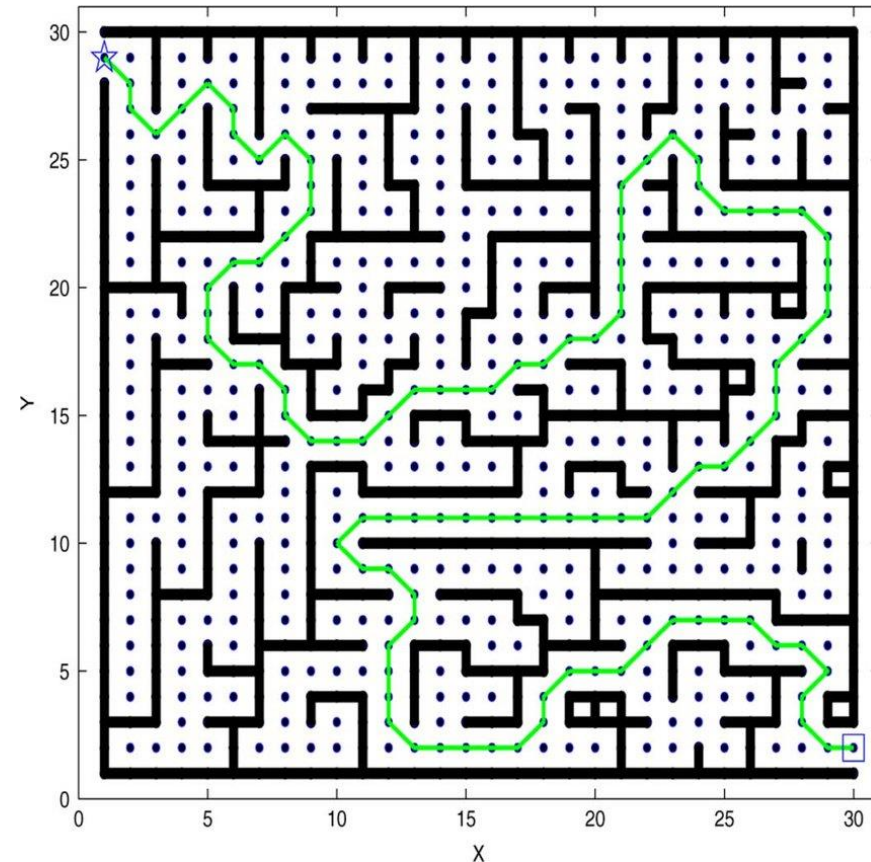


Sequence;

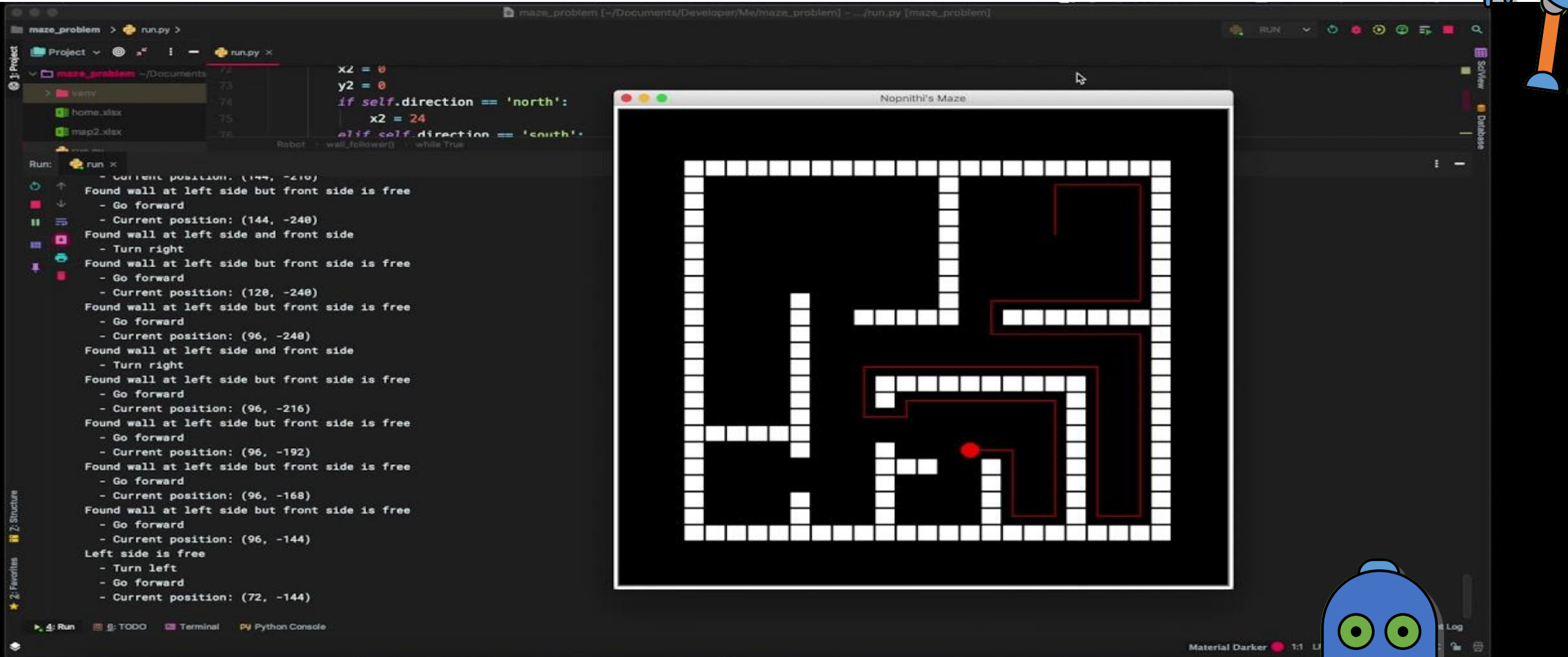
$$x_i \equiv a x_{i-1} + b \pmod{m}$$
$$x_1 \equiv a x_0 + b \pmod{m}$$
$$x_2 \equiv a x_1 + b \pmod{m}$$

Linear congruential generator (LCG)

An algorithm that yields a sequence of pseudo-randomized numbers calculated with a discontinuous piecewise linear equation.



TESTING AND VALIDATION

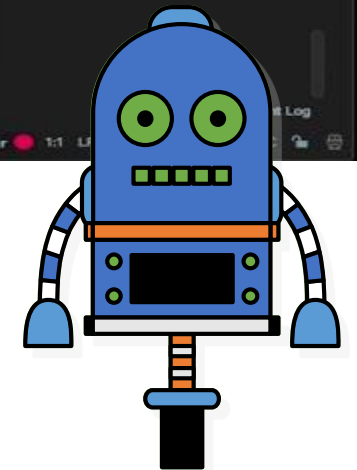


```
maze_problem > run.py >
Project > maze_problem ~/Documents/Developer/Me/maze_problem - .../run.py {maze_problem}
venv
home.xlsx
map2.xlsx
Run: run x
- Current position: (144, -210)
Found wall at left side but front side is free
- Go forward
- Current position: (144, -240)
Found wall at left side and front side
- Turn right
Found wall at left side but front side is free
- Go forward
- Current position: (120, -240)
Found wall at left side but front side is free
- Go forward
- Current position: (96, -240)
Found wall at left side and front side
- Turn right
Found wall at left side but front side is free
- Go forward
- Current position: (96, -216)
Found wall at left side but front side is free
- Go forward
- Current position: (96, -192)
Found wall at left side but front side is free
- Go forward
- Current position: (96, -168)
Found wall at left side but front side is free
- Go forward
- Current position: (96, -144)
Left side is free
- Turn left
- Go forward
- Current position: (72, -144)
x2 = 0
y2 = 0
if self.direction == 'north':
    x2 = 24
elif self.direction == 'south':
    x2 = 24
Robot: wall_follower() while True
```

Nopnith's Maze

WALL FOLLOWING

The "Wall Following" algorithm is a popular maze-solving strategy for robots. One common variant is the "Right-Hand Rule," where the robot maintains contact with the right wall throughout the maze. This strategy is suitable for a maze-solving robot with limited sensors and simple logic.



TESTING PHASE

The image shows a software interface for testing a maze-solving algorithm. On the left is a 20x20 grid representing a maze. The grid contains numbers from 1 to 89, with some cells being empty (0). A red line traces a path through the maze, starting from the top-left corner (cell 59) and ending at the bottom-right corner (cell 51). The path is composed of red lines connecting the cells. The right side of the interface contains a 'Controls' panel with buttons for 'Build' and 'Run'. Below these are tabs for 'Build Output', 'Run Output', and 'Stats'. The 'Stats' tab is currently selected, displaying various metrics:

- Total Distance: 0
- Current Run Distance: 0
- Total Effective Distance: 0
- Current Run Effective Distance: 0
- Total Turns: 0
- Current Run Turns: 0
- Best Run Distance: (empty field)
- Best Run Effective Distance: (empty field)
- Best Run Turns: (empty field)
- Score: 2000

The 'Config' section at the top right shows the 'Maze' set to 'resources/mazes/example3.r' and the 'Mouse' set to 'wall following c++'.

FABRICATION AND MANUFACTURING



**Li-ON 3D PLA Printing
Filament 1.75mm for 3D
Printers- Smooth
Printing material with
Wide Color Option (1 kg
Spool) (Red)**

MANUFACTURING A MAZE SOLVING ROBOT

Design and Planning: Determine maze type, robot dimensions, sensor/actuator selection, microcontroller, and power source.

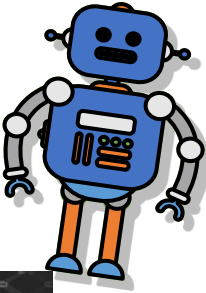
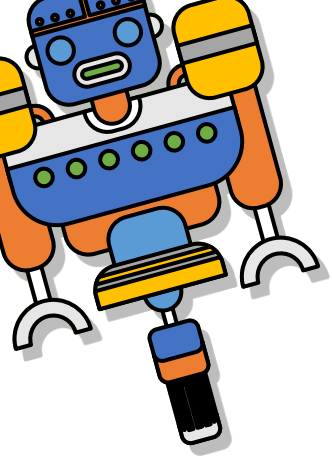
Component Procurement: Gather necessary materials for the robot's structure, electronics, and power supply.

Assembly: Construct the robot's chassis, mount sensors and actuators.

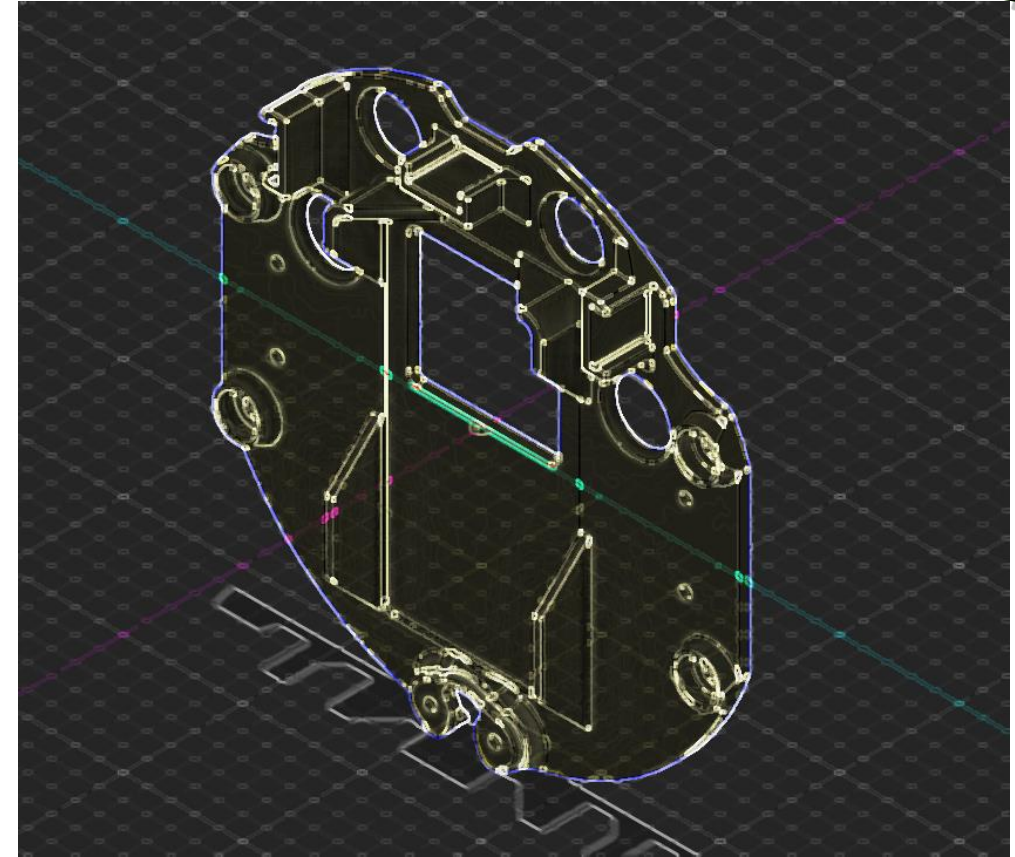
Software Development: Create algorithms for maze solving, sensor data processing, and motor control.

Testing and Refinement: Test the robot's performance in different mazes, calibrate sensors, and optimize performance.

RESULTS AND ANALYSIS



- **Wheel Imperfection:**
The robot's wheels experienced resistance issues, causing it to deviate from a straight path, necessitating modifications to the algorithm.
- **Bump Sensor Solution:**
The integration of bump sensors effectively detected walls and obstacles, playing a crucial role in the robot's maze-solving capabilities.
- **Algorithm Adaptation:**
Due to wheel inconsistencies, modifications will be made to the algorithm to compensate for the robot's imperfect movement, ensuring successful maze navigation.



CONCLUSION

AI[Artificial intelligence] Application:

The project successfully demonstrated the application of artificial intelligence for maze-solving, highlighting its potential for future advancements.

Project Challenges:

Implementing the project required adaptations to the original plan due to unfamiliarity with the learning kit.

Performance and Future Scope:

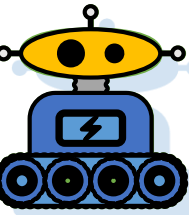
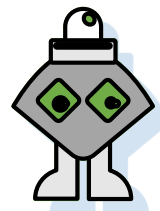
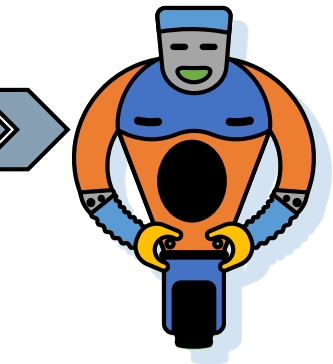
While achieving success, the robot's navigation and obstacle avoidance capabilities can be further enhanced.

Potential Impact:

Maze-solving robots represent a promising field with anticipated growth and development in the coming years.

AI [Artificial intelligence] and Future:

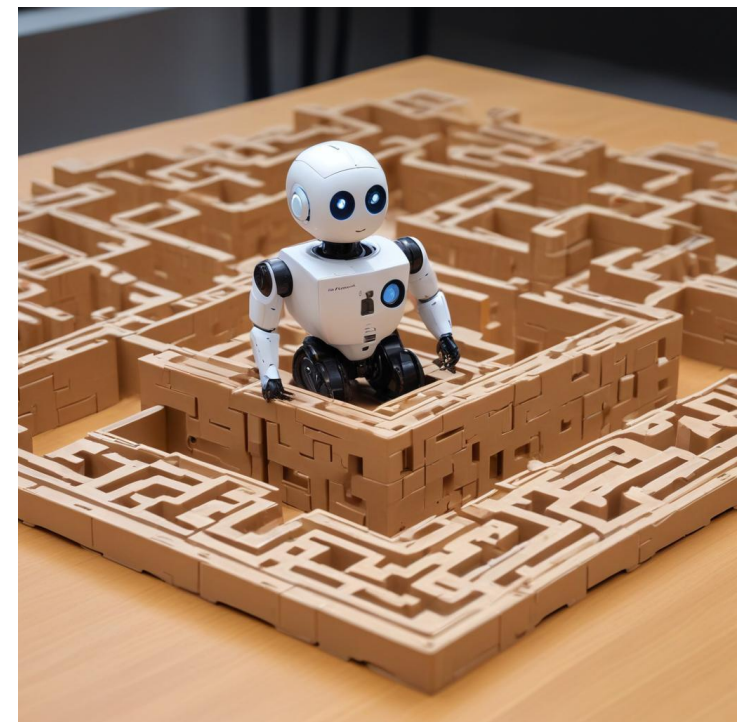
The project's outcome suggests that AI could play a significant role in future technological advancements.



FUTURE WORK

Future Work and Possibilities for Maze Solving Robots:

- **Advanced Algorithms:**
Explore more sophisticated algorithms like Deep Q-Learning for complex maze environments and real-time decision making.
- **Sensor Fusion:**
Combine multiple sensors (lidar, camera, ultrasonic, etc.) for enhanced perception and mapping capabilities.
- **Dynamic Maze Environments:**
Develop robots capable of adapting to changing maze conditions, such as moving obstacles or walls.
- **Real-World Applications:**
Apply maze-solving technology to practical applications like search and rescue, warehouse automation, and exploration.
- **Autonomous Learning:**
Implement machine learning techniques for robots to learn and improve their maze-solving abilities through experience.



REFERENCES

1. An Autonomous Maze-Solving Robotic System Based on an Enhanced Wall-Follower Approach
<https://www.mdpi.com/2075-1702/9/7/81>
2. Design and Implementation of Autonomous Maze-Solving Robot based on an Optimized Flood-Fill Algorithm
<https://www.ijert.org/design-and-implementation-of-autonomous-maze-solving-robot-based-on-an-optimized-flood-fill-algorithm>
3. Maze Solving Robot with Automated Obstacle Avoidance
<https://www.ijrte.org/wp-content/uploads/papers/v7i6/F4721047619.pdf>
4. An Efficient Autonomous Maze Solving Robot based on Microcontroller
<https://www.ijeat.org/wp-content/uploads/papers/v8i6/F8735088619.pdf>
5. An Efficient Algorithm for Robot Maze-Solving
https://www.researchgate.net/publication/330361795_An_Efficient_Algorithm_for_Robot_Maze-Solving
6. Autonomous Maze Solving Robotics: Algorithms and Systems
<https://www.sciencedirect.com/science/article/pii/S1877050916315338>
7. Design and Implementation of a Robot for Maze-Solving using Flood-Fill Algorithm
<https://www.sciencedirect.com/science/article/abs/pii/S221201731300400X>
8. A Comprehensive and Comparative Study Of Maze-Solving Techniques
<https://ieeexplore.ieee.org/document/8424331>

THANK YOU