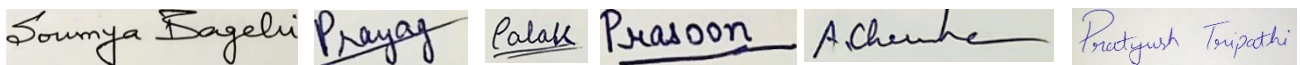


The Maze Solving robot – A Fun Robotics

Prof. Soumya Bagchi¹

Prayag Asrani², Palak Kumari³, Prasoon Krishna Yadav⁴, Abhirup Choudhury⁵, Pratyush Tripathi⁶

1. Assistant Professor, Department of Physics, 9734778872, sbagchi@iitism.ac.in
2. B-tech 2nd year Student, Department of Mechanical Engineering, 7419083487, 23je0741@iitism.ac.in
3. B-tech 2nd year Student, Department of Computer Science & Engineering, 6392760768, 23je0677@iitism.ac.in
4. B-tech 2nd year Student, Department of Electronics and Communication Engineering, 9555637152, 23je0737@iitism.ac.in
5. B-tech 2nd year Student, Department of Mathematics, 6289943945, 23je0023@iitism.ac.in
6. B-tech 2nd year Student, Department of Electrical Engineering, 8423297089, 23je0739@iitism.ac.in



Abstract:

We have developed a maze-solving robot based on the micromouse [1] competition concept, capable of solving mazes autonomously using the logic provided to the robot. The maze is solved by tracing the shortest path from the starting position to the target position in the minimum possible time. The logic is implemented as a program written in C++ programming language, based on simple algorithms and data structures compatible with the Arduino IDE [2]. The program is loaded into an Arduino Nano [3] microcontroller, which uses navigation algorithms to solve mazes in real-time. The robot is equipped with three infrared sensors for wall detection and marking the walls near the cell where the robot is currently located. A motor driver is used to control two encoder motors, enabling the robot to move. By optimizing the flood-fill algorithm, the robot explores multiple possible paths while navigating to the target. We have also tuned the infrared sensors for high accuracy in detecting maze boundaries, ensuring reliable performance under different lighting conditions.

Literature Survey (Prior Art):

The Micromouse competition, initiated in the late 1970s under the guidance of the Institute of Electrical and Electronics Engineers (IEEE)[4], challenges participants to design, develop, and program small robotic mice to navigate mazes efficiently. Over the years, this competition has inspired advancements in robotics and algorithm development, particularly in navigation strategies. Commonly employed algorithms include Depth-First Search (DFS), Breadth-First Search (BFS), and Flood-Fill [5], which optimize pathfinding and enable efficient maze-solving. Navigation strategies such as Wall-following and Dead-end-filling further enhance these capabilities, serving as the foundation for many maze-solving designs [6].

Robots utilizing microcontrollers like Arduino, Raspberry Pi, and BeagleBone are widely adopted in educational robotics, leveraging sensors such as infrared (IR), ultrasonic, and gyroscopes for navigation tasks [7]. Our design integrates an Arduino Nano for its compact form factor and compatibility. Additionally, behaviour-based robotic architectures, which combine simple behaviours like obstacle avoidance, wall-following, and goal-seeking, have proven effective for maze-solving robots [8]. Inspired by these prior works, our project builds upon these concepts while integrating advanced control techniques to enhance performance and reliability.

Detailed Description of the Invention including drawings:

Circuitry

We aimed to make our robot as small and cost-effective as possible. With a length of 45 mm and a width of 18 mm, the Arduino Nano is Arduino's smallest board [9] and weighs only 7 grams. We chose the Arduino Nano microcontroller, preloaded with Scratch programming, for ease of use. The Arduino Nano is a complete and breadboard-friendly board based on the ATmega328 (Arduino Nano 3.x).

Due to the limited number of pinouts [10] of Arduino Nano, we used the TA6586 motor driver [11]. It is effective because it requires fewer pin connections and combines pulse-width modulation pins with direction pins. For precise movements and position tracking, we used 6V N20 500 RPM Encoder Motors to analyze motor rotation. To detect walls at shorter distances, we chose three GP2Y0A41SK0F infrared sensors [12], as they have smaller minimum and maximum detection limits. The robot is powered by two 3.7V lithium polymer cells connected in series to supply the 6V required for the motors. To minimize inter-component connections through wires, we printed a double-layered custom circuit board using EasyEDA, a web-based electronic design automation tool. This circuit board connects all components to their respective ports. In the custom PCB, we integrated the Arduino Nano microcontroller schematic [13] by importing Eagle file[14] of it so that it is embedded directly into the board, eliminating the need to connect it externally. However, we chose not to embed the motor driver on the board because it can occasionally become faulty under undesirable conditions. Instead, the motor driver is connected explicitly by soldering it onto the board.

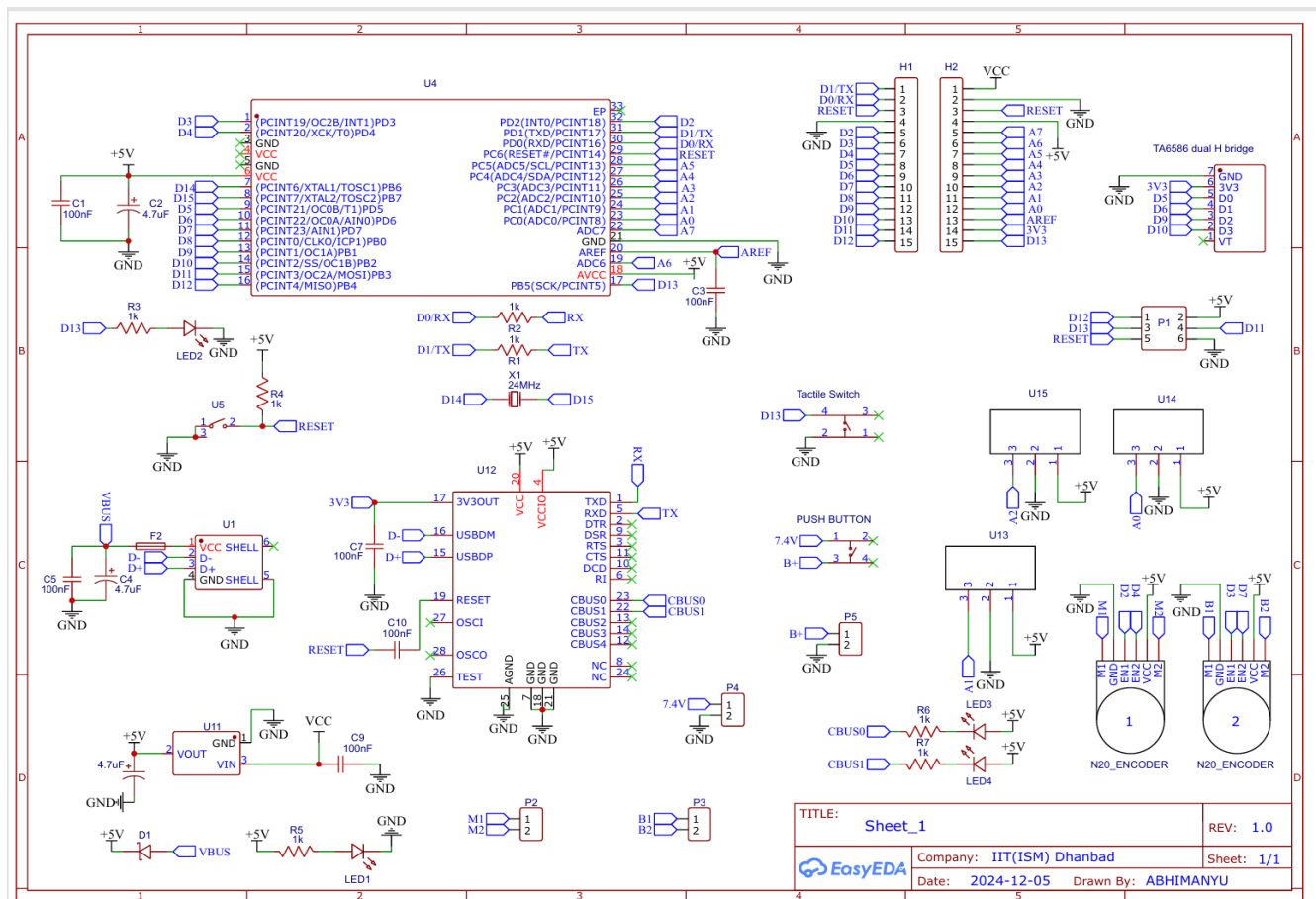


Figure 1: The schematics for the Custom Printed Circuit Board Designed

The main components for robot are listed below:

1. Arduino Nano
2. GP2Y0A41SK0F sharp infrared sensors
3. TA6586 Motor Driver
4. 3.7 V 1250 mAh lipo Battery - 2
5. N20 6V 500 RPM Encoder Motor - 2
6. N20 motor rubber small wheels - 2
7. N20 castor robot ball wheel – 1

| Item | Quantity | Reference | Part | Mfg P/N |
|------|----------|--------------------|--|------------------------|
| 1 | 5 | C1, C3, C4, C7, C9 | CAP 0.1uF +/- 5% 50V Ceramic | 08055C104KAT2A |
| 2 | 1 | C2 | CAP CERAMIC 10UF 16V -20% +80% Y5V | 08055C104KAT2A |
| 3 | 1 | D1 | Diode, Schottky 0.5A 20V | MBR0520LT1G |
| 4 | 1 | J1, J2 | Headers, 36PS 1 Row | |
| 5 | 1 | J4 | Connector, Mini-B Recept Rt. Angle | |
| 6 | 1 | J5 | Headers, 72PS 2 Rows | |
| 7 | 1 | LD1 | LED, RED 20mA 624nm 42mcd 140deg | EC04-0805QRC/E-AV |
| 8 | 1 | LD2 | LED, GREEN 2.1V 20mA 571nm 10mcd 140deg | 17-21SYGC/S530-E2/TR8 |
| 9 | 1 | LD3 | LED, YELLOW 2.0V 20mA 589nm 43mcd 140deg | 0805UYC/E-AV |
| 10 | 1 | LD4 | LED, Super Bright BLUE 80mcd 470nm 110degree | 17-21SUBHC/S530-A2/TR8 |
| 11 | 1 | R1 | Resistor Pack, 1K +/-5% 62.5mW 4RES SMD | YC164-JR-071KL |
| 12 | 1 | R2 | Resistor Pack, 680 +/-5% 62.5mW 4RES SMD | YC164-JR-07680RL |
| 13 | 1 | SW1 | Switch, Momentary Tact SPST 150gf 3.0x2.5mm | SKRKAE010 |
| 14 | 1 | U1 | IC, Microcontroller ATMEGA328 RISC 32kB Flash, 0.5kB EEPROM, 23 I/O Pins | ATMEGA328P-AUR |
| 15 | 1 | U2 | IC, USB to SERIAL UART | FT232RL-REEL |
| 16 | 1 | U3 | IC, Voltage regulator 5V, 500mA | UA78M05CDCYR |
| 17 | 1 | Y1 | Cystal, 16MHz 0.5% | CSTCE16M0V53-R0 |
| 18 | 1 | C5 | CAP CER 4.7UF 10V Y5V 0603 | GRM188F51A475ZE20D |
| 19 | 1 | F1 | FUSE PTC RESET 500MA SMD 0603 | MF-FSMF050X-2 |
| 20 | 1 | H1 | Dual H bridge Motor Driver | TA6586 |
| 23 | 3 | U9, U8, U7 | Sharp IR sensor | GP2Y0A41SK0F |
| 21 | 2 | U5, U6 | N20 6V 500 RPM Encoder Motor | |
| 24 | 1 | SW | 4 Pin Tactile push button Switch | |
| 25 | 1 | KEY | Slide Switch | SS-12D37-G030 |
| 22 | 2 | | 3.7V battery | |

Table 1: A complete list of material required during scratch fabrication of PCB.

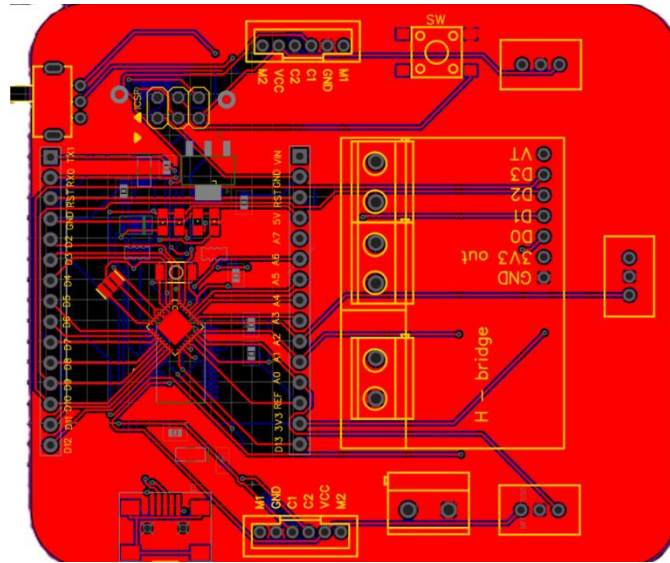


Figure 2a: Upper side of circuit board

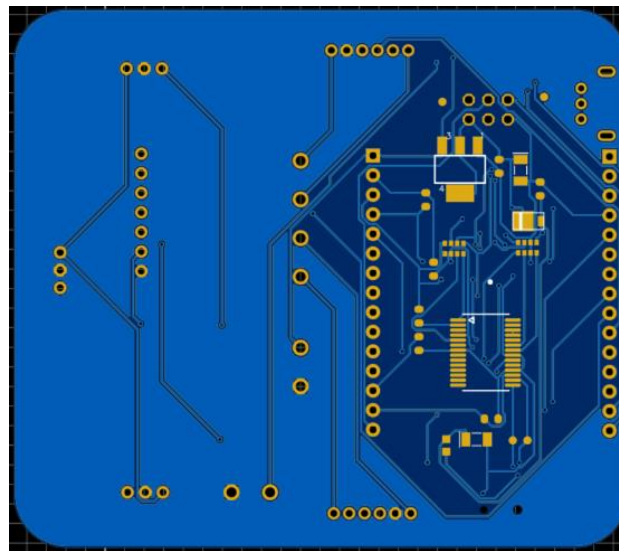


Figure 2b: Lower Side of circuit board

Figure 2: The pictorial representation of the designed Custom Printed Circuit Board.

Chassis

The chassis structure for the prototype of our robot was 3D printed using Poly Lactic Acid (PLA), chosen for its lightweight and durable properties. We designed the chassis using Solidworks, a solid modeling CAD software. The chassis has a square shape with dimensions of 7.5 cm x 7.5 cm on the top plane and consists of two layers. The height of the robot is 5.5 cm after attaching the N20 motor rubber small wheels, which have a diameter of 34 mm, and an N20 castor robot ball wheel with a height of 1.5 cm. Both layers are fastened with screws using four corner poles. The motors and the battery are sandwiched between the two layers. The lower chassis has motor cases precisely shaped on its upper side to fit the motors. Brackets are used to secure the motors with screws in the motor cases from the upper side at the back. A castor wheel is attached to the front side with screws below the lower chassis layer to provide front support. The battery is placed on the lower layer to

achieve proper mass balance and an accurate center of mass. The upper chassis layer holds a printed circuit board (PCB) with dimensions of 7 cm x 7 cm. The PCB is either pasted or screwed onto the upper side to ensure it is well-protected. The infrared sensors are screwed vertically onto 3D-printed vertical supports attached to both layers at the front, left, and right sides of the robot. The orientation of the sensors was determined based on their datasheet to ensure proper usage.

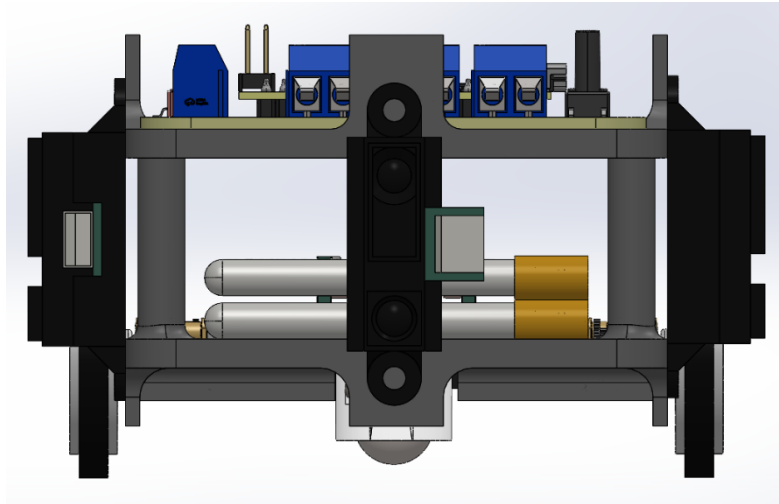


Figure 3a: Front view of Chassis

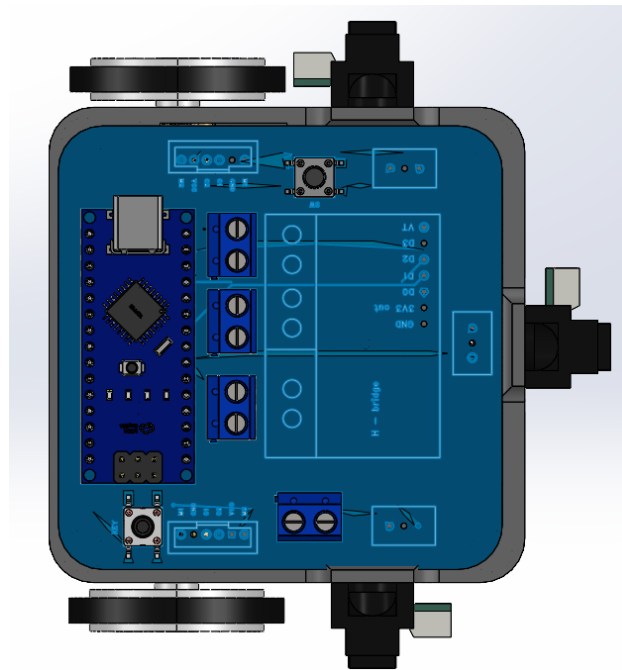


Figure 3b: Top view of Chassis

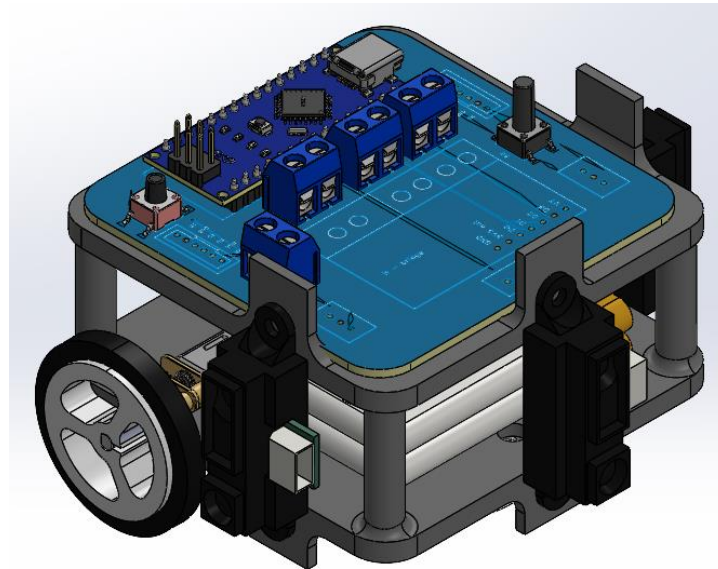
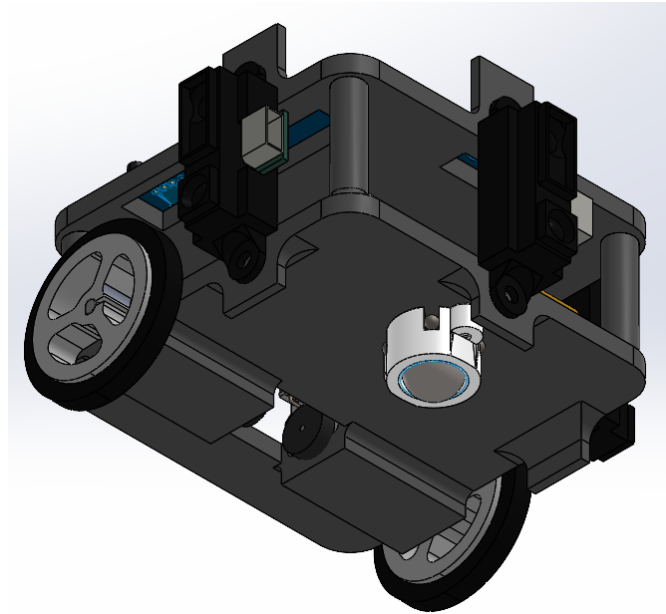


Figure 3c: Isotropic view of Chassis

Figure 3: Computer Aided Design of Chassis

Algorithm

We initially tried a basic wall-following algorithm to solve the maze, but it took more time. To address this, we implemented the flood-fill algorithm along with maze mapping. Additionally, we compared the modified flood-fill algorithm [15] with the standard flood-fill algorithm to improve the robot's overall performance. To ensure precise movement, we implemented PID control using the motor's encoder readings, which allows the robot to accurately reach its target positions. Positional errors were minimized by calculating wall distances through IR sensors and properly aligning the robot to correct deviations. At last, we tried every possible optimization step to improve the robot. After tracing any possible path, the map data is stored in the EEPROM (electrically Erasable Programmable Read-only Memory) of the Arduino Nano, which has a capacity of 1 KB. This data can be updated in the EEPROM whenever needed. Additionally, the EEPROM data can be reset to its default state when the maze-solving process restarts.

Reference:

1. Howell, T. *Micromouse History*. Retrieved from <https://micromouseonline.com/micromouse-book/history/>
2. Monk, S. (2012). *Programming Arduino: Getting Started with Sketches*. McGraw-Hill Education.
3. Arduino. *Arduino Nano Datasheet*. Retrieved from <https://docs.arduino.cc/resources/datasheets/A000005-datasheet.pdf>
4. IEEE. *IEEE Micromouse History*. Retrieved from <https://www.ieee.org/>
5. IEEE. (2010). An Algorithm of Micromouse Maze Solving. *Proceedings of the IEEE Conference*. Retrieved from <https://ieeexplore.ieee.org/document/5578409>
6. Nishi, T., & Yasuda, T. (1990). A study on algorithms for Micromouse competitions. *Robotics and Automation Journal*.
7. Elger, D. (1987). Wall-Following Algorithms for Autonomous Robots. *Control Systems Magazine*.
8. Brooks, R. A. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*.
9. Arduino. *Arduino Nano Board Information*. Retrieved from https://content.arduino.cc/assets/arduino_nano_size.pdf
10. Arduino. *Pinout Diagram of Arduino Nano*. Retrieved from <https://docs.arduino.cc/resources/pinouts/A000005-full-pinout.pdf>
11. Toshiba. *Motor Driver Circuit TA6586*. Retrieved from https://www.micros.com.pl/mediaserver/UITA6586_0001.pdf
12. Sharp. *GP2Y0A41SK0F Infrared Sensor Datasheet*. Retrieved from https://global.sharp/products/device/lineup/data/pdf/datasheet/gp2y0a41sk_e.pdf
13. Arduino. *Arduino Nano Schematics*. Retrieved from https://content.arduino.cc/assets/NanoV3.3_sch.pdf
14. Arduino. *Arduino Nano Eagle File*. Retrieved from <https://content.arduino.cc/assets/Nano-reference.zip>
15. Mars University. Quantitative Comparison of Flood Fill and Modified Flood Fill Algorithms. Retrieved from <https://marsuniversity.github.io/ece387/FloodFill.pdf>