MANTECH Publications

International Journal of Robotics and Autonomous Systems
Volume 6, Issue 2, July-December, 2021

A Bug Algorithm-Based Obstacle and Edge Detection System

Dr. Muhammad Masroor Ali[1], Mohammad Mahfuzul Islam[2]

Assistant Professor[1, 2]

Department of Mechanical Engineering

Western Ideal Institute

Corresponding Author's E-mail: *mahfuzulislammohmmad@gmail.com[1]*

Abstract

Because the purpose of an autonomous robot is to arrive at its destination without colliding with anything, obstacle avoidance is a significant difficulty in robotics. One type of autonomous robot is a real-time obstacle avoidance edge detection robot, which can identify obstacles and edges and take other routes that are free of them. This study shows a robotic robot with built-in intelligence that directs itself using a bug algorithm if it finds an impediment. This robotic robot is built with the AT Mega 8 series of microcontrollers (Arduino Uno R3). The ultrasonic sensor detects any barrier with edges and sends an instruction to the microcontroller. The microprocessor informs the robot to push in a different direction based on the received input signal by activating the motors attached to it via a motor driver. Depending on the situation, the robot may select the appropriate course of action. Here, an obstacle avoidance edge detection decision-making process occurs on its own. This robot was designed to think about what it could do on a daily basis.

Keywords: *Robotics, Arduino, Self-decisioning robot, Artificial Intelligence, Edge Detection, Ultrasonic Sensor, Obstacle Avoidance*

INTRODUCTION

The obstacle and edge detection robot is a self-decisioning robot that can make decisions based on the present situation. It can detect any impediment or edge and switch to a different path that is free of them. Some simple tasks that the robot can perform are as follows:

- If it can't discover an obstruction, it keeps moving forward at a constant

speed and can quickly detect a smooth surface if one is placed in front of it.

- It moves at a constant speed, comparable to the obstruction, until it finds an edge.
- It can detect every type of edge from a distance of only 5 cm;
- It can make logical decisions on its own to choose the most efficient route to take; and
- It can detect both edges and obstacles at the same time.

Prototype Design

The hardware needed to finish this project is as follows:

- Arduino Uno R3
- Motor Shield L293D
- 2x DC motors
- 3x Ultrasonic sonar sensor HC-SR04
- SG90 Micro Servo Motor

The robot chassis kit, which is propelled by two wheels, is the foundation of the system. The Arduino Uno R3 is mounted on top of the chassis, and the L293D motor shield is aligned with it. In the motor shield slots m1 and m2, two DC motors are connected. The SG90 servo motor is installed in the second slot. Pins A0-A5 connect three ultrasonic sensors to the motor shield. The entire system is powered by a lithium polymer ion battery.

All necessary connections are made via the jumper wires. Because the motor shield requires a lot of power to function properly, an 11V lithium polymer ion battery was used. The DC motors are connected using the m1 and m2 slots. However, the user can connect to any of the four slots.

One thing to note is that two ultrasonic sensors are used to detect the edge, and two basic extenders are used to expand the edge somewhat further away from the robot (user can use any kind of small stick or piece of metal or 3D printed arm). Above the servo motor is the third ultrasonic sensor (figure 1). Two ultrasonic sensors will be extended slightly so that the robot may stop while correctly detecting the edge. It takes some time to come to a halt due to inertia.

The prototype's pictures are depicted in Figure 1.

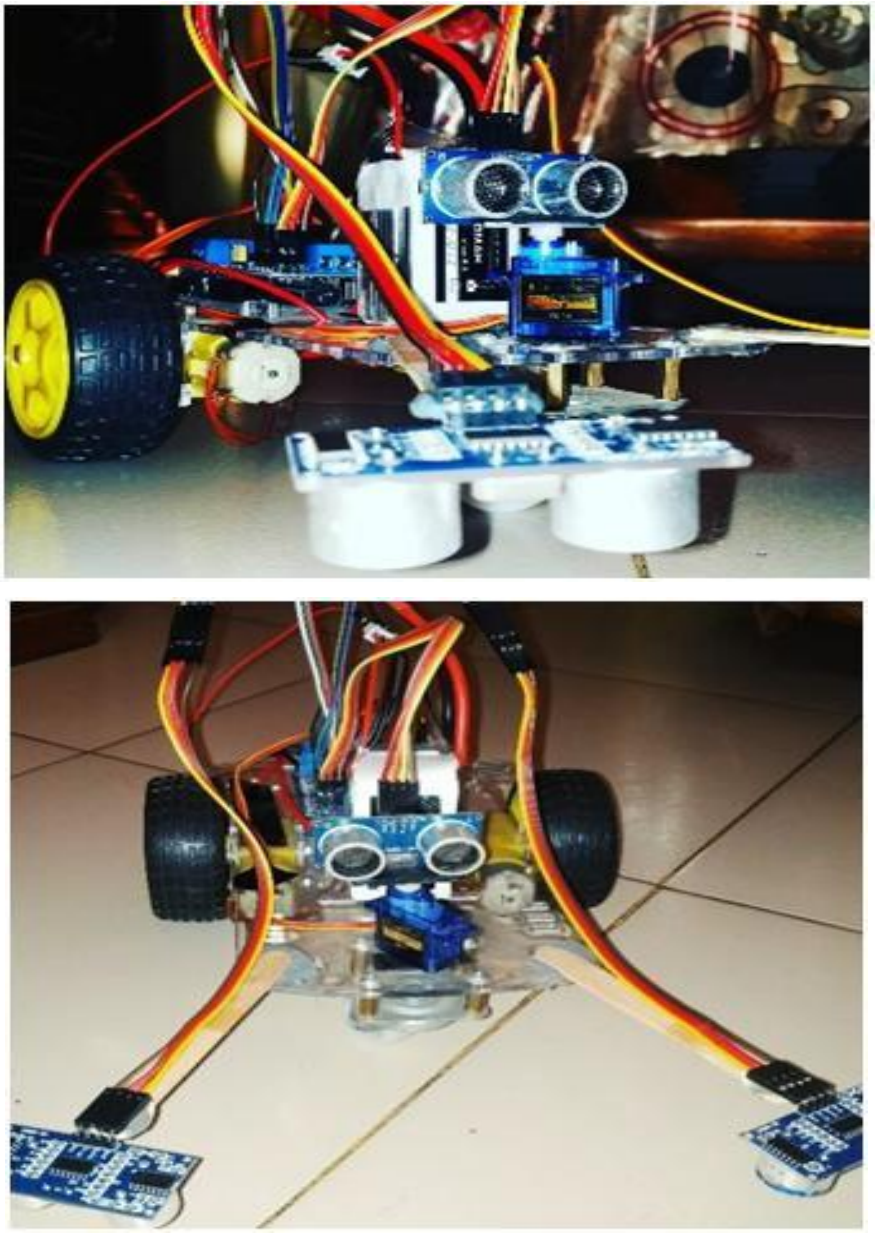


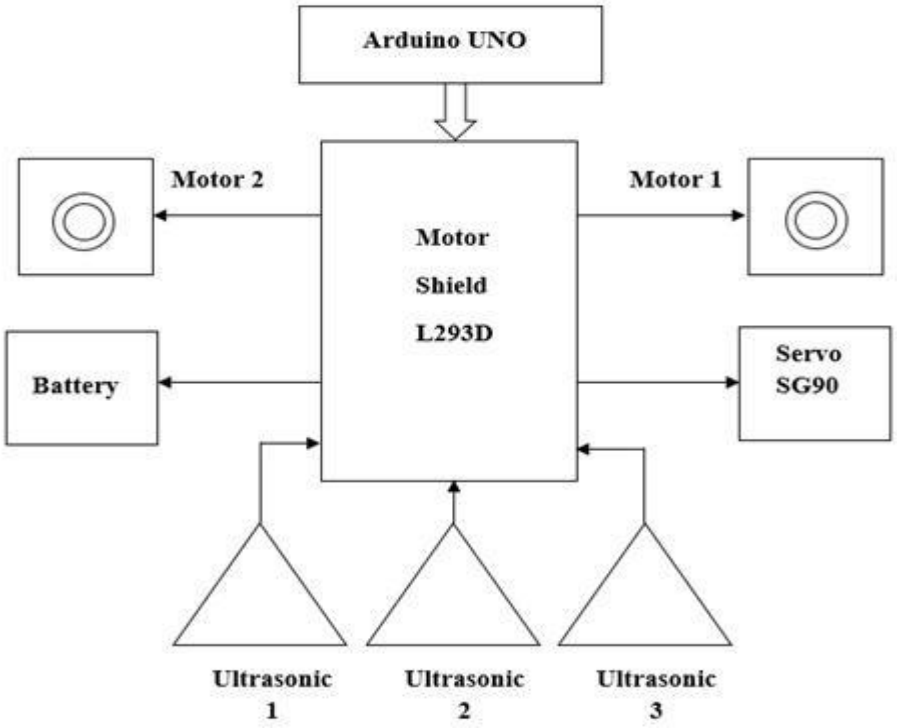
Figure 1: Snapshots of the Prototype

TECHNICAL PARTS

Block Diagram

The Arduino Uno R3 transmits a signal to the L293D motor shield, which can drive four dc motors at once, as shown in the system's block diagram. The motor shield can also be used to control two servo motors. However, only two dc motors and one servomotor were employed in this experiment, which received signals from the motor's shield [2].

As the measuring distance, the motor shield receives the signal from ultrasonic sensors. Figure 2 is a block diagram of the complete system.



System Architecture

For avoiding edge and hazard routes, the motor shield L293D was used. It's similar to an Arduino-to-dc-motors interface. It can control up to four dc motors at once. However, this Robot only requires two of these.

The motor 1 was linked to the m1 slot of the motor shield, while the motor 2 was connected to the m2 slot. The L293D engine shield may also run two servo motors simultaneously. One SG90 servo has been used, which is coupled to the servo motor's first terminal. In addition, three ultrasonic sensors were used. All ultrasonic sensors' ground and VCC pins have been connected to the motor shield's ground and 5V pins, respectively. A2 and A3 trig pins are connected to the ultrasonic 1 echo pin. 2 echo pins connected to A1 and trig pin attached to A0 for ultrasonic.

A4 and A5 are connected to the ultrasonic 3 echoes and the trigger pin, respectively. An 11V lithium polymer battery has been attached to the motor shield's external power supply port to power the complete gadget. Furthermore, the motor shield required external power, but the Arduino did not require any additional power. The system's architecture is depicted in Figure 3

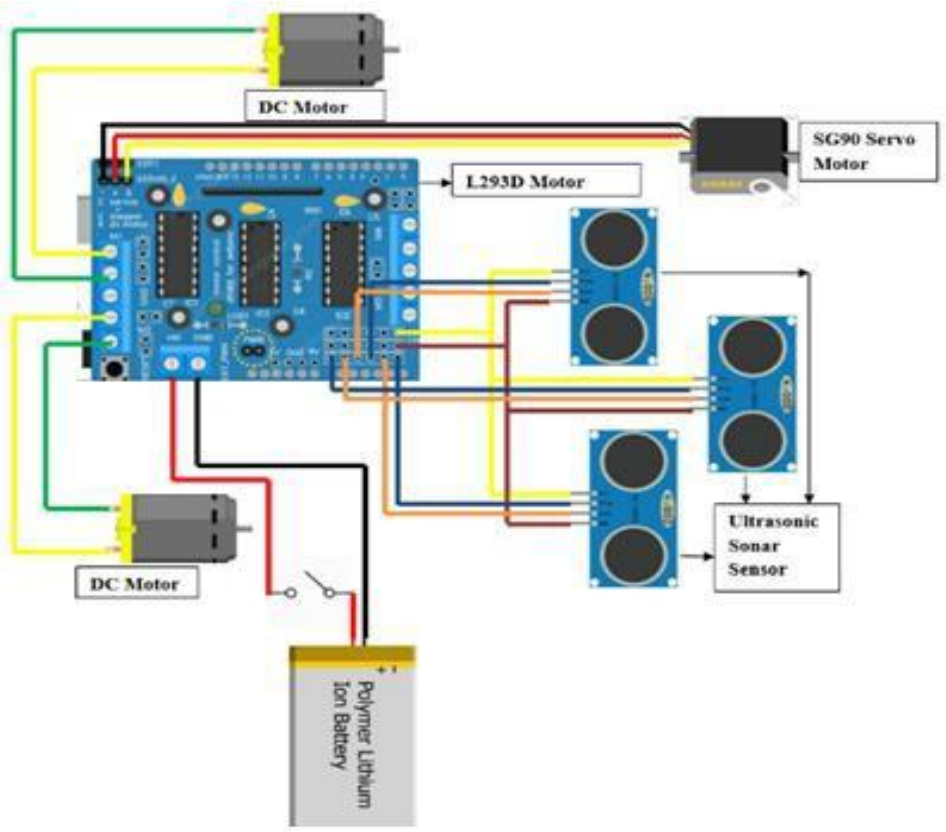
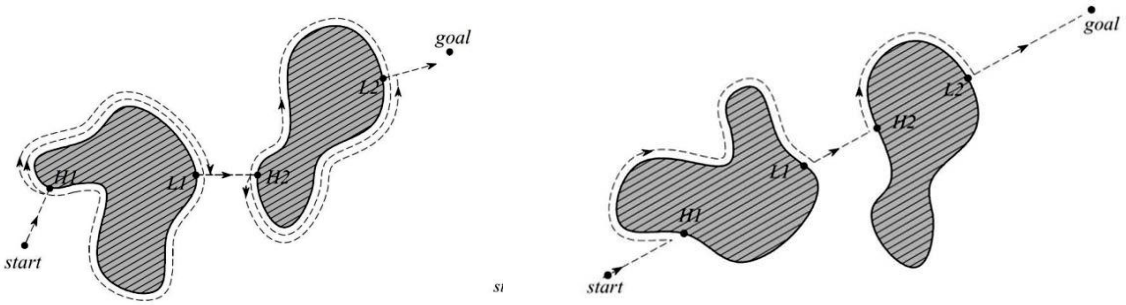


Figure 3: Architecture of the System

Working Principle of the system using Bug Algorithm

Modern techniques allow the robot to skip the obstruction utilising some type of quantitative calculation of obstacle dimensions, whereas primitive algorithms used to stop the robot to prevent the collision. Modern techniques allow the robot to bypass obstacles using some form of quantitative measurement of the dimensions of obstacles.



(a) Bug 1 ***(b) Bug 2***

Figure 4: Path Planning with Bug Algorithm [5]

In the Bug 1 algorithm, the robot performs the following steps:

- The robot is bound towards the destination.
- If the robot encounters an impediment, it navigates around it and records / recalls its distance from the objective at each point.
- After navigating around the obstacle, the robot returns to the position closest to the obstacle using wall-following and then continues on the route to the target.

Figure 4(a) shows how a point robot navigates an area with two obstacles to reach the target using the Bug 1 algorithm. It's possible that the Bug 1 algorithm will be pricey. The distance d travelled by the robot is bounded by D= dD + 12 Pi if the cost of the straight line path from the starting to the destination is D and the diameter of the obstruction is Pi. To avoid circumnavigating the obstruction, the Bug 2 algorithm was proposed[6].

Within the Bug 2 algorithm, the robot performs the following steps:

- The robot moves towards the m-line destination, which is defined as the line that connects the starting place and the destination.
- If an impediment is spotted, the robot will follow it until it finds an m-line that is closer to the destination.
- The robot then moves away from the impediment and returns to the m-line in order to reach the goal.

Figure 4(b) shows how a point robot navigates an area with two obstacles using the Bug 2 algorithm to achieve its destination.

It would be more cost-effective if the robot could move directly to the destination once a straight line path from its current location to the destination is possible. As a result, Bug 1 is an exhaustive search algorithm because it considers all options, but Bug 2 is a greedy algorithm. In many cases, Bug 2 outperforms Bug 1, while Bug 1 has a more predictable efficiency. In the actual world, however, we have sensors like range sensors that are more powerful than touch sensors and can anticipate a limited range[4]. The Tangent Error algorithm is then implemented.

The following is the working principle:

Step 1: Start

Step 2: Initialization

Step 3: Is there any obstacle or edge?

(a) If yes, go to step 5

(b) No, run the robot in forward direction

Step 4: If there is any obstacle or edge found

(a) Yes, move backward

(b) No, go to step 3

Step 5: Distance measurement to turn left or right

If right distance > left distance

(a) Yes, turn right

(b) No, turn left

Step 6: Continue Step 3

Step 7: If power failed go to step 8

Step 8: Stop

System Flowchart

The flowchart of the system is given in figure no 5.

In the flow chart, it shows the route taken while avoiding obstacles with an edge detecting robot. The robot will not start moving after turning on the electricity unless we start from the beginning. After startup, the first thing it will do is search for an impediment or edge in front of it. When there is no barrier or edge in the way, the robot can freely step forward. In order to move forward, it will seek out challenge and edge in every situation.

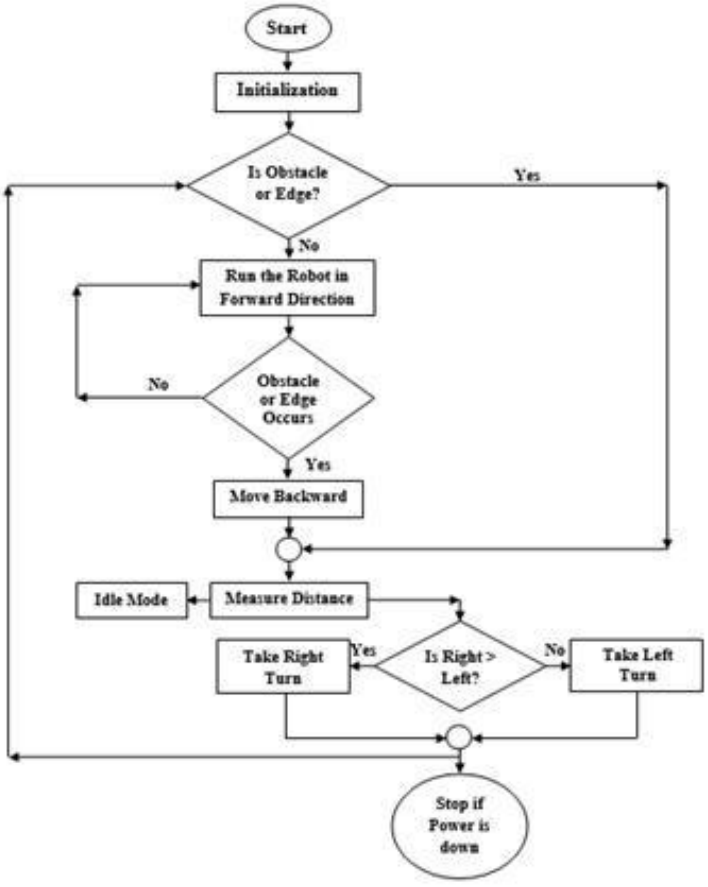


Figure 5: Flowchart of the system

As long as there are no holes or edges, it will effortlessly move along. Regardless of whether it is confronted with a challenge or at the bottom, it will do the same action. It might be right at the start or right in the middle of its journey. After identifying an impediment or edge in some space, the first thing it will do is move backward (3 cm). The detection range for obstacles is 15 cm, although it can be adjusted based on the robot's speed. After travelling backward, it will come to a halt and glance to the right and left, respectively. Looking right and left involves using an ultrasonic sensor to measure the distance between right and left. The robot will travel to the side that has more space than the other, depending on whether the room is on the right or left. The procedure will then be repeated from the very beginning. A question emerges in this situation. What if the two distances are exactly the same? This is unlikely to happen with ultrasonic distance measurements of one-tenth of a metre. Let's move on to the next section: what happens if an edge is detected? When it comes to impediments, the robot will follow the same methods. When it detects an edge, it comes to a halt, turns to the right or left depending on the greater distance, and then continues forward until it encounters another obstacle or edge. For searching the edges and obstructions, there is no chance of slipping or falling from the top. We never have to apply pressure to discover the side where there is no edge because the robot will stop automatically if it detects one.

Decision Making Process through Distance Measurement

Based on the value of the distance measurement obtained from three ultrasonic sensors, table 1 illustrates when the robot will halt and when it will go forward.

Table 1: Decision Making Process through Distance Measurement

Right Ultrasonic Value	Left Ultrasonic Value	Middle Ultrasonic Value	Decision
<5 cm	≥5 cm	>15 cm	Stop Robot
<5 cm	≥5 cm	≤15 cm	Stop Robot
<5cm	<5 cm	>15 cm	Move Forward
<5 cm	<5 cm	≤15 cm	Stop Robot
≥5 cm	≥5 cm	>15 cm	Stop Robot
≥5cm	≥5 cm	≤15 cm	Stop Robot
≥5cm	<5 cm	>15 cm	Stop Robot
≥5cm	<5 cm	≤15 cm	Stop Robot

One thing to keep in mind is that the ultrasonic sensors on the right and left are both facing down to detect the edge, while the middle ultrasonic sensor is facing forward to identify obstacles. When the value of the right and left ultrasonic sensors is greater than or equal to 5 cm, the robot considers it an edge and begins looking for an alternative path. When the ultrasonic sensor in the front of the middle face registers a reading of less than 15 cm, the system recognises it as an impediment and stops the Robot instantly. We can see that there is just one sweet place where the Robot can push ahead when we look at the entire table. When the mean ultrasonic value is larger than 15 cm and the ultrasonic right and left sensor values are less than 5 cm, you've found the sweet spot.

Path Planning to Detect Obstacle and Edge

Figure 6 demonstrates the robot's route planning with obstacles and boundaries.

On this board, there are four obstacles and four edges. There are two stages: 'Start,' which is where the robot starts, and 'Target,' which is where the robot will arrive. The robot will begin by pressing the 'Start' button, then locate the first obstacle, obstacle 1, then turn left using its algorithm before continuing forward.

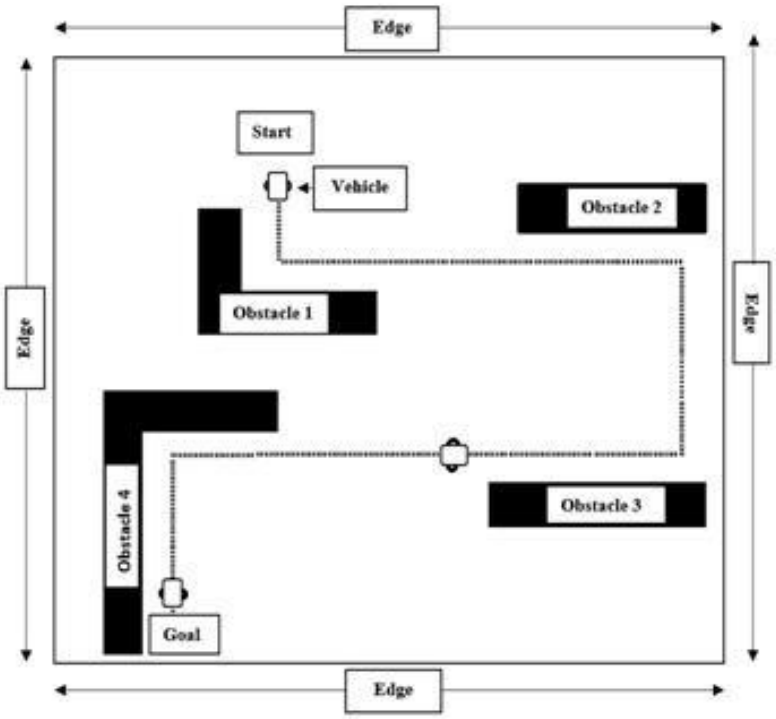


Figure 6: Path Planning of the Robot with Obstacles and Edges

When it comes to the edge, it will look for another path, and it will notice that obstacle 2 is blocking the left side, so it will turn right. Then it will restart moving forward until it hits obstacle number three. When detecting obstacle 3, you'll see that both the right and left sides have the same gap (this is because there is nothing in the ultrasonic sensor range). We have explained in the algorithm that if the distances are equal, our comparable robot will choose the right side to travel on. After turning right, it will begin going forward once more, finally detecting obstacle 4. And the algorithm will once again assist it in turning left and achieving its objective.

APPLICATIONS

In several industries, there is a tremendous amount of impediment and edge potential that robots must avoid. The following are a few notable examples:

- From Bangladesh's standpoint, the majority of rural roads lack a guardrail. As a result, there's always a chance it'll fall through the crack. This module, which may be added into the Robot, will assist in the handling of undesired situations.

- It's common for different building materials to fall from unfinished levels in multi-story building construction. Such items may be simply transferred across the ledge.

- The majority of the routes across the highlands in Bangladesh are devoid of guardrails. As a result, there is always the possibility of plunging down the chasm. This module may be installed in the Robot and will assist in the event of an unwelcome mishap.

- Falling various construction materials from completed levels is rather prevalent in multistory building development. These materials can be efficiently transported using the robot's edge and obstacle avoidance capabilities.

- A large number of disabled persons rely on wheelchairs. This concept may be used in wheelchairs since it can detect obstacles and edges on its own.

LIMITATIONS

Because an ultrasonic sensor calculates distance using ultrasonic sound, sharp edges might make it difficult for the robot to estimate distances properly. It has the potential to lead to poor judgments. As a result, impediments and edges must be as smooth as possible when employing

ultrasonic sensors. If we utilise infrared sensors instead of ultrasonic sensors, this problem may not arise. However, because IR sensors are light-sensitive, we favour ultrasonic sensors. If an impediment is detected within the range of the front ultrasonic sensor, the vehicle will most likely avoid it. If the ultrasonic sensor is placed very close to the surface, the robot can circumvent this constraint. Last but not least, this prototype was not built to travel through slop. To get around this constraint, it should be constructed differently.

FUTURE RECOMMENDATIONS

- If the present project is connected to a camera robot, it is feasible to drive beyond line of sight and range due to the vast range of networks.

- Use as a firefighting robot: By adding a temperature sensor, a water tank, and other programming adjustments, we can turn this robot into a firefighting robot.

- We will use wireless technologies to enhance this project utilising IR (or) RF (or) ZIGBEE.

- We may use the DTMF receiver with the mobile phone.

- This robot may be used to choose and place the proper object by giving it commands, however depending on the application, the ultrasonic sensor should be removed.

FUTURE WORKS

The ability for robots to adapt to their surroundings is a crucial topic of robotics study. It doesn't matter if you're in an aquatic environment, on land, underground, in the air, or in space.

- A completely autonomous robot can:
- Work for a lengthy period of time without human interaction or the need for a power supply;
- Avoid dangerous situations.

The most efficient way to improve my robot's accuracy is to add better sensors; this will raise the project's cost, but it will also extend the precision and the problem space where the robot may be employed. Robots with better actuators will be quicker and more efficient.

CONCLUSION

The goal of this project is to create an autonomous robot that can identify obstacles in its path and navigate according to the rules we establish for it. The aforementioned Arduino controller and ultrasonic sensor have been

investigated, with the ultrasonic sensor being chosen as the control result to suit its application in the automobile prototype system. The key challenge was to detect and avoid the barrier. The obstacle avoidance algorithm was successfully implemented with low mistakes by writing the programme in C. Obstacle avoidance is a valuable method that may be applied in automobiles to avert several accidents and fatalities.

REFERENCES

1. J.C. Baker, A Simon, "Sensor and navigation data fusion for an autonomous guided vehicle", Proceedings of Intelligent vehicle symposium, pp. 156-161, 2000.

2. Z. Yi, Y. Khing, C. Chin Seng, and Z. Xiao Wei, "Multiultrasonic sensor fusion for mobile robots", Proceedings of intelligent vehicle symposium, pp. 387-391, 2000.

3. Buniyamin N., Wan Ngah W.A.J., Sariff N., Mohamad Z., "A Simple Local Path Planning Algorithm for Autonomous Mobile Robots", INTERNATIONAL JOURNAL OF SYSTEMS APPLICATIONS, ENGINEERING & DEVELOPMENT, Issue 2, Volume 5, 2011

4. G.D. Hager, Z. Dodds, "Robotic Motion Planning: Bug Algorithms", unpublished.

5. M. ohaib, S.M Pasha, N. Javaid, J Iqbal, "Intelligent Bug Algorithm (IBA): A Novel Strategy to Navigate Mobile Robots Autonomously"