PROJECT REPORT

Maze Solving robot

Submitted to

LOVELY PROFESSIONAL UNIVERSITY

PHAGWARA, PUNJAB

Student Name: Ruhani Rawal

Registration Number: 12004649

Roll No.: 07

Section: E2006

TABLE OF CONTENTS

1.INTRODUCTION

- PROBLEM STATEMENT
- OBJECTIVES
- OVERVIEW

2. LITERATURE OVERVIEW

- PULSE WIDTH MODULATION

3. HARDWARE DESCRIPTION

- CONSTRUCTION
- ARDUINO
- ATMEGA 328P
- HC-SR04
- L298 MOTOR DRIVER
- VOLTAGE REGULATOR
- MOTORS
- BATTERIES

3. ALGORITHMS

- WALL FOLLOWING ALGORITHM
- FLOOD FILLL ALGORITHM

PROCESS OF CONSTRUCTION AND RESULT TO BE ADDED LATER

INTRODUCTION

ABSTRACT

A maze is a network of paths, typically from an entrance to exit. The concept of Maze is approximately thousand years old. Which was invented in Egypt. From then, many mathematicians made various algorithm to solve the maze.

It is based on one of the most important areas of robot, which is “Decision Making Algorithm”.

There are many types of maze solving robot using various type of algorithms.

In this project, the system design of Maze solving robot consist obstacle avoidance ultrasonic sensors and then sensors will detect the wall. To solve the maze, this robot will apply wall following algorithms such as left- or right-hand rule. It will also follow the Flood fill algorithm for finding the shortest path.

AUTONOMOUS MAZE SOLVING ROBOT

An autonomous robot, as the name suggests is a category of robot that can perform tasks intelligently depending on themselves, without any human assistance. Maze Solving Robot is one of the most popular autonomous robots. It is a small self-reliant robot that can solve a maze from a known starting position to the center area of the maze in the shortest possible time. A maze solving robot make multiple runs in a maze, first it creates a map of the maze layout and store it in its memory, then run through a shortest path.

PROBLEM STATEMENT

To design a hardware for maze solving robot, construct a software with the combination of wall following and flood fill algorithms then implement the software in the hardware of maze solving robot. At last, make a maze 6×6 maze to verify the robot.

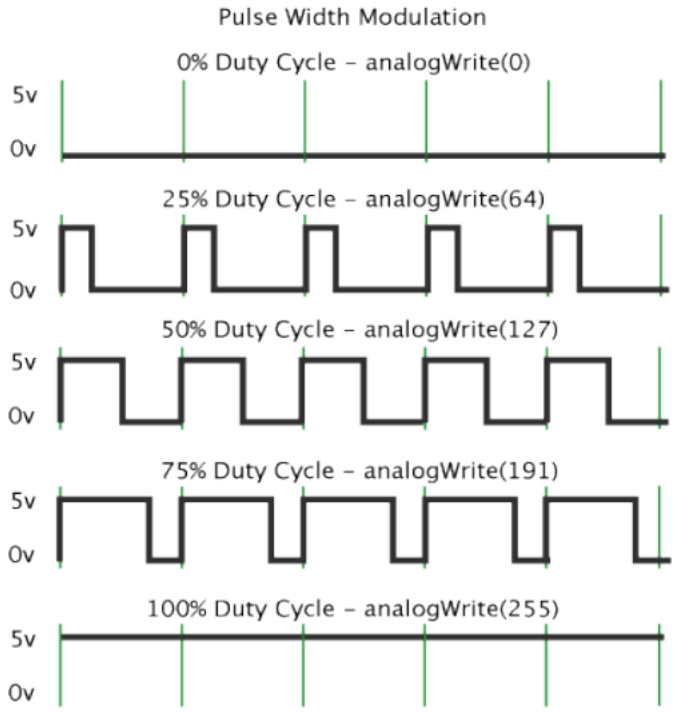
OBJECTIVES

- Understand and implement the wall follower and Flood fill algorithm.
- Design and build ultrasonic sensors array.
- Design and build Arduino based hardware.
- Programmed the robot to solve the maze.
- Make a small 6x6 maze.

LITERATURE OVERVIEW

PULSE WIDTH MODULATION

For controlling the motors speed, pulse width modulation (PWM) is used. Pulse width modulation is a simple method of controlling analogue devices via a digital signal through changing or modulating the pulse width. An analogue device is become digital by powering it with a pulse signal switches between on and off (5V and 0V). This digital control is used to create a square wave. The duty cycle is defined as the percentage of digital 'High' to digital 'Low' plus digital 'High' pulse-width during a PWM period. The average DC voltage value for 0% duty cycle is zero; with 25% duty cycle the average value is 1.25V (25% of 5V). With 50% duty cycle the average value is 2.5V, and if the duty cycle is 75%, the average voltage is 3.75V and so on. So, by varying the average voltage, the motor speed can be controlled.



The power loss in PWM switching devices is very low. In many cases it is near to 0. So, this is the main advantage of PWM. While being used, resistors will tend to loss more power because of its heat dissipation. So, PWM is efficient in controlling motors

HARDWARE

Arduino

Arduino is an open-source hardware development board. Arduino hardware consists of an open hardware design with an Atmel AVR processor. There are many types of Arduino board available in the market. But, in this project Arduino UNO has been used.

Arduino UNO is based on the ATmega328P microprocessor. It has 14 input/output pins. 6 digital pins can be used as PWM outputs. It has 6 analog input pins. It has a 16 MHz quartz crystal. It contains USB connection port, dc power port. The microcontroller is programmed via Arduino IDE.

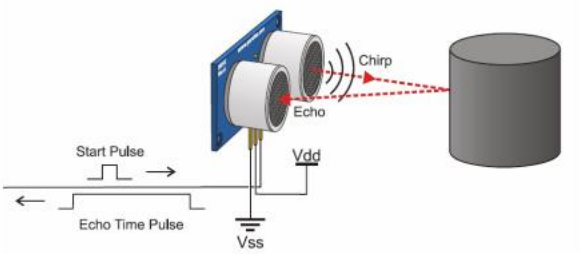
ATmega328P

ATmega328P is an 8-bit AVR RISC-based microcontroller. It has 32KB ISP flash memory, 1024B EEPROM, 2KB SRAM, 23 general purpose I/O lines, 6-channel 10-bit A/D converter, 32 general purpose working registers, 3 flexible timers, serial programmable USART, a byte-oriented 2-wire serial interface and SPI serial port.

Ultrasonic ranging sensor (HC-SR04)

An Ultrasonic sensor is a device that can measure the distance to an object by using sound waves. HC-SR04 is one of the common ultrasonic sensors. It has two parts; one is transmitter another is receiver. A pulse is triggered in the trigger pin, then a sound wave sends to the object, then the reflected wave from the object is received by the receiver. The received signal go to the circuit for measured the distance. A working diagram of sonar sensor is given. The formula of measuring distance using sonar sensor is given below:

$$distance = \frac{speed\ of\ sound\ \times time\ taken}{2}$$



Features:

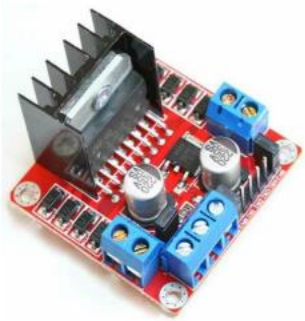
- Operating Voltage: 5V DC
- Operating Current: 15mA
- Measure Angle: 15°
- Ranging Distance: 2cm - 4m

L298 Motor driver

This dual bidirectional motor driver. Most popular L298 Dual H-Bridge Motor Driver Integrated Circuit is used here.

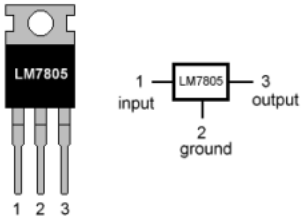
Features:

- Logic voltage- 5V
- Drive voltage- 5V to 35V
- Logic Current- 0mA-36mA
- Drive current- 2A (MAX each bridge)
- Maximum Power – 25W



Voltage Regulator

The voltage regulator is used to supply constant 5V to the ultrasonic sensors. A very common voltage regulator IC is 7805 which has been used in this project.



Motors

Two DC gear motors have been used to drive this robot. This gear motor is ideal for robotic car or line-tracing robot

Features:

- Operating voltage: 3V ~ 6V DC
- Speed without load: 800g.cm
- Maximum torque: 90±10rpm
- Reduction ratio: 1:48
- Load current: 190mA (max. 250mA)

Batteries

4 AA batteries have been used to power the motor driver. Each battery relates to other in series connection. One 9V battery is used to power the Arduino board, another is used to power the voltage regulator.

ALGORITHMS

WALL FOLLOWING ALGORITHM

The most common algorithm for maze solving robot is Wall following algorithm. The robot will take its direction by following either left or right wall. This algorithm also called, left hand-right hand rules. Whenever the robot reaches a junction, it will sense for the opening walls and select it direction giving the priority to the selected wall. By taking the walls as guide, this strategy is capable to make the robot reaches the finish point of the maze without solving it. But this algorithm is not efficient method to solve a maze. Cause, the wall follower algorithm will fail to solve some maze construction, such as a maze with a closed loop region. The instructions used in the algorithm for both left and right wall is given in a table below:

Right wall following routine	Left wall following routine
if there is no wall at right,	if there is no wall at left
turn right	turn left
else	else
if there is no wall at straight	if there is no wall at straight
keep straight	keep straight
else	else
if there is no wall at left	if there is no wall at right
turn left	turn right
else	else
turn around	turn around

FLOOD FILL ALGORITHM

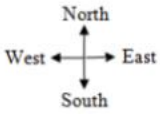
The most efficient Maze solving algorithm is flood fill algorithm. It is derived from “Bellman Ford Algorithm”. The algorithm works by assigning value for all cells in the maze, where these values indicate the steps from any cell to the destination cell. The first array is holding the walls map values, while the other one is storing the distance values.

Array Positions

20	21	22	23	24
15	16	17	18	19
10	11	12	13	14
5	6	7	8	9
0	1	2	3	4

Flood Fill

4	3	2	3	4
3	2	1	2	3
2	1	**0**	1	2
3	2	1	2	3
4	3	2	3	4



Bit No.	7	6	5	4	3	2	1	0
Wall					W	S	E	N

Map Array

array	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
value	7	3	6	3	6	1	4	9	12	5	5	5	11	2	4	5	5	11	12	5	13	9	10	14	13

Distance Array

array	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
value	4	3	2	3	4	3	2	1	2	3	2	1	0	1	2	3	2	1	2	3	4	3	2	3	4

In every cell, robot will follow the following steps:

1. Update the wall map.
2. Flood the maze with the new distance values.
3. Decide which neighboring cell has the lowest distance value.
4. Move to the neighboring cell with the lowest distance value.