Maze Solver Robot

Author	Berescu Silvia-Maria
Series and Group	332CD
GitHub	Maze Solver Robot

Introduction

This project involves building a small autonomous robot capable of navigating and escaping a maze with opaque walls, using ultrasonic sensors and a wall-follower algorithm.

- **What it does:** The robot detects walls in front and to the left using HC-SR04 ultrasonic sensors, makes real-time navigation decisions, and displays the measured distances and current decision on an LCD screen (FORWARD / TURN LEFT / TURN RIGHT).
- **Purpose:** Autonomous maze solving without human intervention, using an ATmega328P microcontroller.
- **Starting idea:** Demonstrating the use of microcontroller peripherals (PWM, I2C, UART, Interrupts) in a fully functional and autonomous physical product.

Hypothesis: I believe that using the Left-Hand Wall Follower algorithm with HC-SR04 ultrasonic sensors and real-time visual feedback through an LCD will allow the robot to solve a simple maze in under 60 seconds, as the algorithm guarantees finding the exit in any simply connected maze.

General Description

The block diagram below shows all project modules and how they interact:

Main modules and their roles:

- **ATmega328P-XMINI** — main microcontroller; coordinates all sensors and actuators via GPIO, PWM, I2C and external interrupts
- **HC-SR04 (front)** — measures distance to the front wall; ECHO pulse is captured precisely via the INT1 external interrupt
- **HC-SR04 (left)** — measures distance to the left wall via GPIO polling
- **L298N** — dual H-bridge motor driver; receives PWM + GPIO commands from the ATmega and drives the two DC motors
- **LCD 16×2 I2C** — displays measured distances and the current robot decision in real time
- **2WD chassis kit** — mechanical platform with 2 DC gear motors, wheels and a caster wheel
- **2x 18650 batteries (7.4V)** — autonomous power supply; L298N provides regulated 5V to the ATmega, sensors and LCD

Hardware Design

Bill of Materials

No.	Component	Qty	Role
1	2WD chassis kit (line follower)	1	Mechanical platform
2	ATmega328P-XMINI	1	Main microcontroller
3	L298N motor driver module	1	DC motor control via PWM
4	HC-SR04 ultrasonic sensor	2	Obstacle detection
5	LCD 16×2 with I2C module (PCF8574)	1	Robot status display
6	18650 Li-ion battery 2500mAh	2	Autonomous 7.4V power
7	Dual 18650 charger	1	Battery recharging
8	2×18650 battery holder + switch	1	Battery mount + ON/OFF
9	Male-female dupont wires	40	Component connections
10	400-point breadboard	1	Connection prototyping

Pin Mapping — Arduino Labels vs ATmega328P Physical Pins

The table below clarifies the mapping for all pins used in this project:

ATmega Port/Pin	Used For
PD2	HC-SR04 Front — TRIG
PD3 (INT1)	HC-SR04 Front — ECHO
PD4	HC-SR04 Left — TRIG
PD5 **(OC0B)**	L298N ENB — PWM Motor Right
PD6 **(OC0A)**	L298N ENA — PWM Motor Left
PB0	L298N IN1
PB1	L298N IN2
PB2	L298N IN3
PB3	L298N IN4
PB4	HC-SR04 Left — ECHO
PC4 (SDA)	LCD I2C — SDA
PC5 (SCL)	LCD I2C — SCL
VCC	5V power input
GND	Ground

PD3/INT1: connected to the INT1 external interrupt line, used for precise ECHO pulse measurement on the front sensor. Interrupt-driven measurement is more accurate than polling because the ATmega reacts instantly to the signal edge without busy-waiting.

PD5/OC0B and PD6/OC0A: both are hardware PWM outputs of Timer0 (Channel B and Channel A respectively). Using both channels allows independent speed control of each motor, which is essential for straight-line driving.

PB4: plain GPIO pin used for HC-SR04 Left ECHO via polling. It has no special hardware function requirements since the left sensor uses software polling.

Electrical Schematic

The schematic uses **net labels** instead of long wires for clarity. Two pins sharing the same net label name are electrically connected.

Schematic Explanation

Power Supply

1. The 2×18650 batteries (7.4V in series) connect through the ON/OFF switch on the battery holder to the **+12V** pin of the L298N (labelled VMS on the physical module)
2. The L298N drives the two DC motors directly at 7.4V through OUT1/OUT2 (Motor A) and OUT3/OUT4 (Motor B)
3. The L298N has an onboard 5V regulator. Its **+5V output** is connected to the VCC rail, powering the ATmega328P, both HC-SR04 sensors and the LCD
4. All GND pins across all components share a common ground

Motor Control

The L298N receives direction signals on IN1–IN4 from ATmega pins D8–D11:

IN1	IN2	Motor Left behavior
LOW	HIGH	Forward
HIGH	LOW	Reverse
LOW	LOW	Stop

The same logic applies to IN3/IN4 for Motor Right.

Speed is controlled via PWM on both **ENA** (D6/OC0A) and **ENB** (D5/OC0B) using Timer0 Fast PWM mode.

HC-SR04 Front Sensor

- **TRIG** - D2 (PD2): ATmega sends a 10µs HIGH pulse to start a measurement
- **ECHO** - D3 (PD3/INT1): the sensor returns a HIGH pulse proportional to distance

- INT1 triggers on both rising and falling edges — firmware records the Timer1 value at each edge to compute pulse duration
- Distance (cm) = pulse duration (µs) / 58

HC-SR04 Left Sensor

- **TRIG** - D4 (PD4)
- **ECHO** - D12 (PB4): GPIO polling — firmware busy-waits for the pulse to end
- D12 was chosen instead of D5 because D5 (OC0B) is needed for ENB PWM control
- Polling is acceptable here since the left wall changes less frequently than the front wall

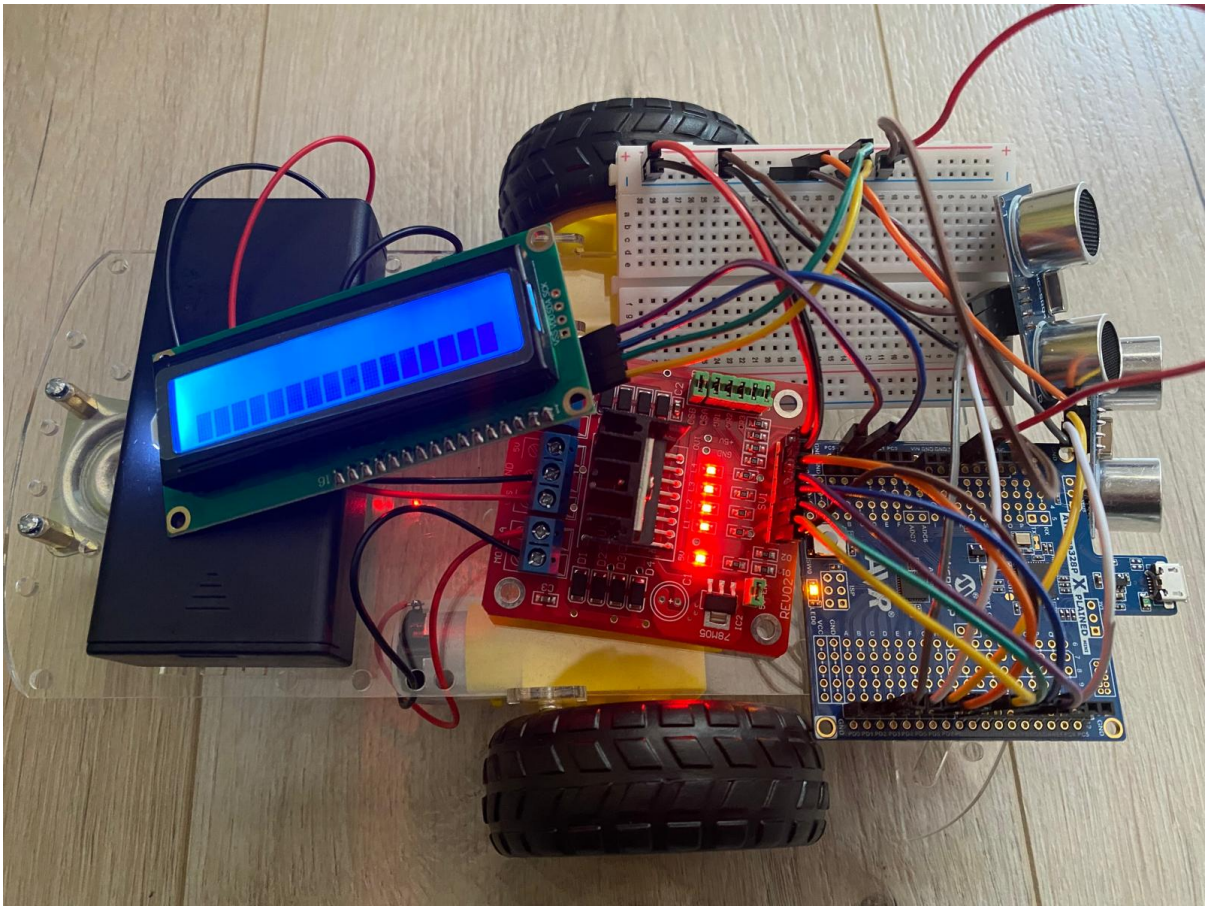
LCD 16x2 I2C

- **SDA** - A4 (PC4): I2C serial data
- **SCL** - A5 (PC5): I2C serial clock
- The onboard PCF8574 expander converts I2C to the parallel HD44780 interface internally
- I2C address: 0x27 (PCF8574 default)
- Only 2 signal wires needed instead of 16 for parallel mode

Power Supply Summary

1. 2×18650 batteries (7.4V) → ON/OFF switch → L298N +12V (VMS)
2. L298N → motors at 7.4V
3. L298N 5V OUT → ATmega328P + HC-SR04 x2 + LCD I2C

Hardware Progress



The image shows the fully wired robot with all components mounted and powered via the 2×18650 battery pack (7.4V).

Component Identification

- **ATmega328P-XMINI** — powered via the 5V regulated output of the L298N module. The onboard amber LED is lit, confirming the board is receiving stable 5V power from the battery pack through the L298N regulator. GND from the ATmega is connected to the breadboard ground rail to establish a common ground reference for all components.
- **L298N motor driver** — all 4 red indicator LEDs are on, confirming the module is powered and IN1–IN4 lines are in a defined logic state. The 5V regulated output feeds the breadboard power rail which distributes power to all logic components.
- **LCD 16×2 I2C** — the blue backlight is on, confirming the LCD module is receiving stable 5V power from the battery pack via the L298N regulated output. No characters are displayed yet as the firmware has not been uploaded at this stage. I2C communication will be verified once the ATmega is programmed.
- **HC-SR04 ultrasonic sensors** — both front and left sensors are mounted and connected to ATmega via the breadboard.
- **Breadboard** — distributes the 5V and GND rails from the L298N to all logic components. The red (+) rail carries 5V and the blue (-) rail carries GND, with the ATmega GND also connected to the same rail for common ground.
- **Battery holder 2×18650** — mounted on the chassis.

What the Indicators Confirm

- **L298N red LEDs on** — motor driver is powered and input pins IN1–IN4 are in a defined state, confirming correct power wiring.
- **LCD backlight on** — the 5V regulated output of the L298N is confirmed working, supplying stable power to the LCD module via the breadboard power rail.
- **ATmega amber LED on** — microcontroller is powered and operational.
- **Common GND established** — ATmega GND pin is connected to the breadboard ground rail shared with L298N, sensors and LCD, ensuring all components share the same voltage reference.

Software Design

Development Environment

- **Visual Studio Code + PlatformIO Core 6.1.19**
- **Framework:** Arduino (avr-minicore) used only as build system, all peripheral code written directly on AVR registers
- **Compiler:** avr-gcc (toolchain-atmelavr 7.3.0)
- **Upload:** custom avrdude with `xplainedmini` programmer over USB mEDBG

Project Structure

File	Role
`main.cpp`	Motor functions, navigation algorithm, setup/loop
`sensors.cpp/.h`	HC-SR04 front (INT1 interrupt) and left (polling)
`lcd.cpp/.h`	TWI/I2C driver + HD44780 LCD functions

Lab Concepts Used and Justification

Lab concept	Justification
GPIO	TRIG/ECHO sensor pins and motor direction pins controlled directly via DDRx/PORTx registers
PWM	Timer0 Fast PWM on OC0A and OC0B for independent speed control of each motor; allows straight-line correction by setting different OCR values
Interrupts	INT1 on PD3 for front HC-SR04 ECHO, interrupt-driven measurement is more accurate than polling because the MCU reacts instantly to signal edges without busy-waiting
I2C	LCD 16×2 via PCF8574 on PC4/PC5; TWI implemented on registers following the same pattern as the lab
Timers	Timer0 for PWM generation; Timer1 as free-running counter inside INT1 ISR to timestamp ECHO pulse edges and compute distance

Navigation Algorithm

The robot implements the Left-Hand Wall Follower algorithm. The maze is constructed so the robot always has a left wall present. The decision logic runs continuously in the main loop:

1. If **left is free** (distance > 20cm): turn left 90° then move forward — this is the core left-hand rule, taking every available left turn
2. Else if **front is free** (distance > 20cm): move forward, follow the left wall
3. Else: turn right 90°, blocked on both sides, only option is right

This guarantees finding the exit in any simply connected maze.

Sensor Implementation

The **front sensor** uses INT1 hardware interrupt for precise ECHO pulse measurement. Timer1 is started on the rising edge of ECHO and stopped on the falling edge. Distance is computed from the elapsed ticks. This approach is more accurate than polling and frees the CPU between measurements.

The **left sensor** uses GPIO polling. PB4 (D12) was chosen instead of D5 because D5 (OC0B) was needed for ENB PWM. Polling is acceptable here since the left wall changes slowly, precision requirements are lower than for the front sensor.

Motor Control

Timer0 Fast PWM at 7.8kHz drives both motors independently via OC0A (left) and OC0B (right).

Important finding: The motor direction logic is inverted, determined experimentally during hardware testing.

Motor B runs at a slightly higher PWM value than Motor A to compensate for mechanical differences between the two motors, ensuring straight-line driving.

Calibration

- **Turn delay:** calibrated by placing the robot on a flat surface, calling the turn function, and measuring the resulting angle physically. Adjusted until 90° was achieved consistently.
- **Motor speeds:** OCR0A=180-200, OCR0B=190-200 — calibrated by observing straight-line driving. Motor B speed increased until drift was eliminated.
- **Distance thresholds:** DIST_FRONT_STOP and DIST_LEFT_FAR both set to 20cm — chosen to match the corridor width of the test maze and provide sufficient stopping margin.

Validation

Each component was validated independently before integration: motors tested individually for direction and speed, each sensor tested by placing hand at known distances and verifying LCD

readings, turn angles verified physically on grid paper. Full algorithm validated in test maze sections observing behavior for each decision case.

Demo Video

To be added after final maze test

Results

Conclusions

Download

Journal

Week	Activity	Status
Week 1	Component selection, ordering parts	Done
Week 2	Chassis assembly, mounting motors and wheels	Done
Week 3	L298N wiring + electrical schematic	Done
Week 4	HC-SR04 integration + LCD I2C integration	Done
Week 5	Navigation algorithm implementation + maze test	Planned
Week 6	Code optimization, final demo	Planned

Bibliography / Resources

Hardware Resources

- HC-SR04 Datasheet: https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf
- L298N Datasheet: http://www.st.com/web/en/resource/technical/document/datasheet/CD00000240.pdf
- ATmega328P Datasheet: https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf
- PCF8574 Datasheet: https://www.ti.com/lit/ds/symlink/pcf8574.pdf

Export to PDF