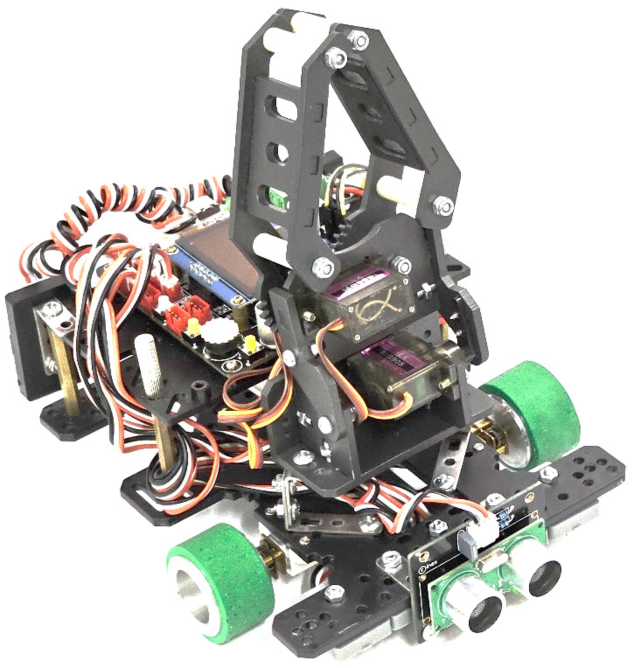
Intelligent Rescue Robot (Maze Mission)

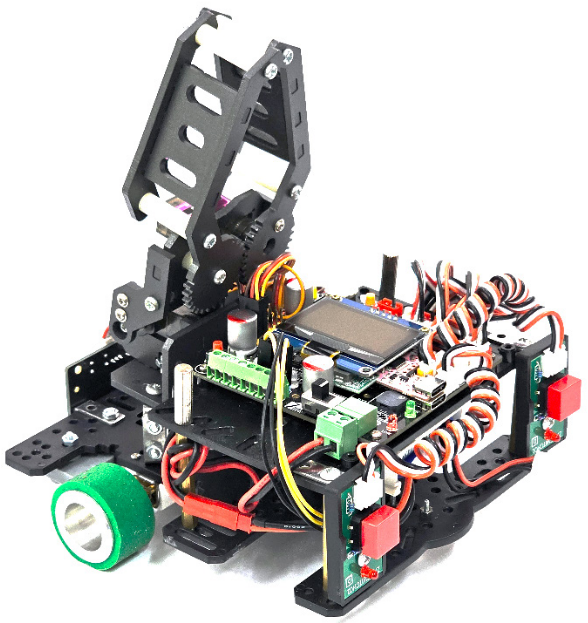
1 Introduction

We have designed the 2-wheel robot as an Intelligent Rescue Robot Game Tutorial for World Robot Games.

List of equipment

- POP32i Microcontroller Board × 1
- ZX-03R × 5
- ZX-SONAR × 1
- ZX-Switch01 × 2
- iRC1 Chassis × 1
- N20 Motor with wheel × 2
- Gripper-X ×1
- Li-po Battery 2 Cells 7.4V ×1



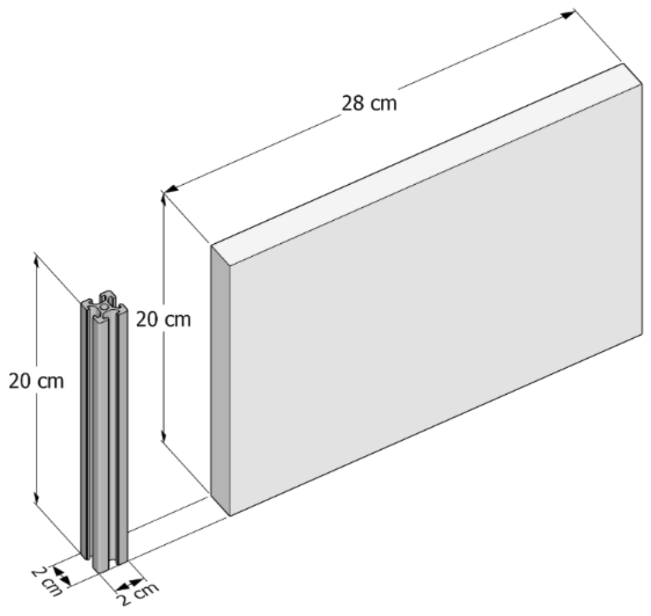


2 Maze Field Creation

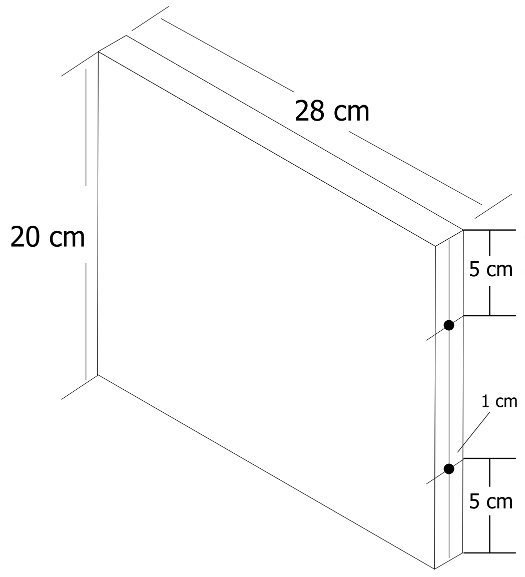
This example made the maze field look similar to an actual competition field.

List of equipment

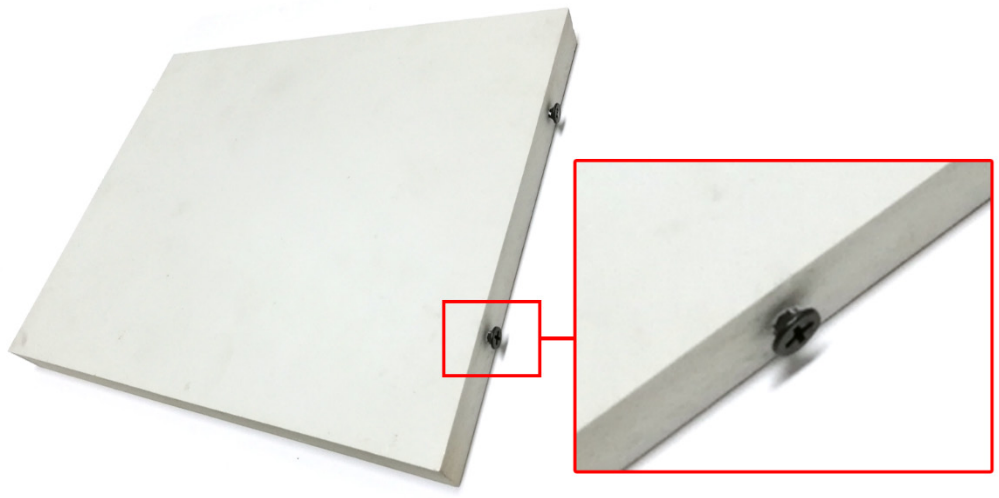
- Cut Aluminum Profile 2x2x20 cm
- Cut Plastic Wood Sheet 2x28x20 cm (create a wall).
- Flat-Head Tapping Screw
- Plastic Wood (create a large field or floor).
- Double-sided cloth tape



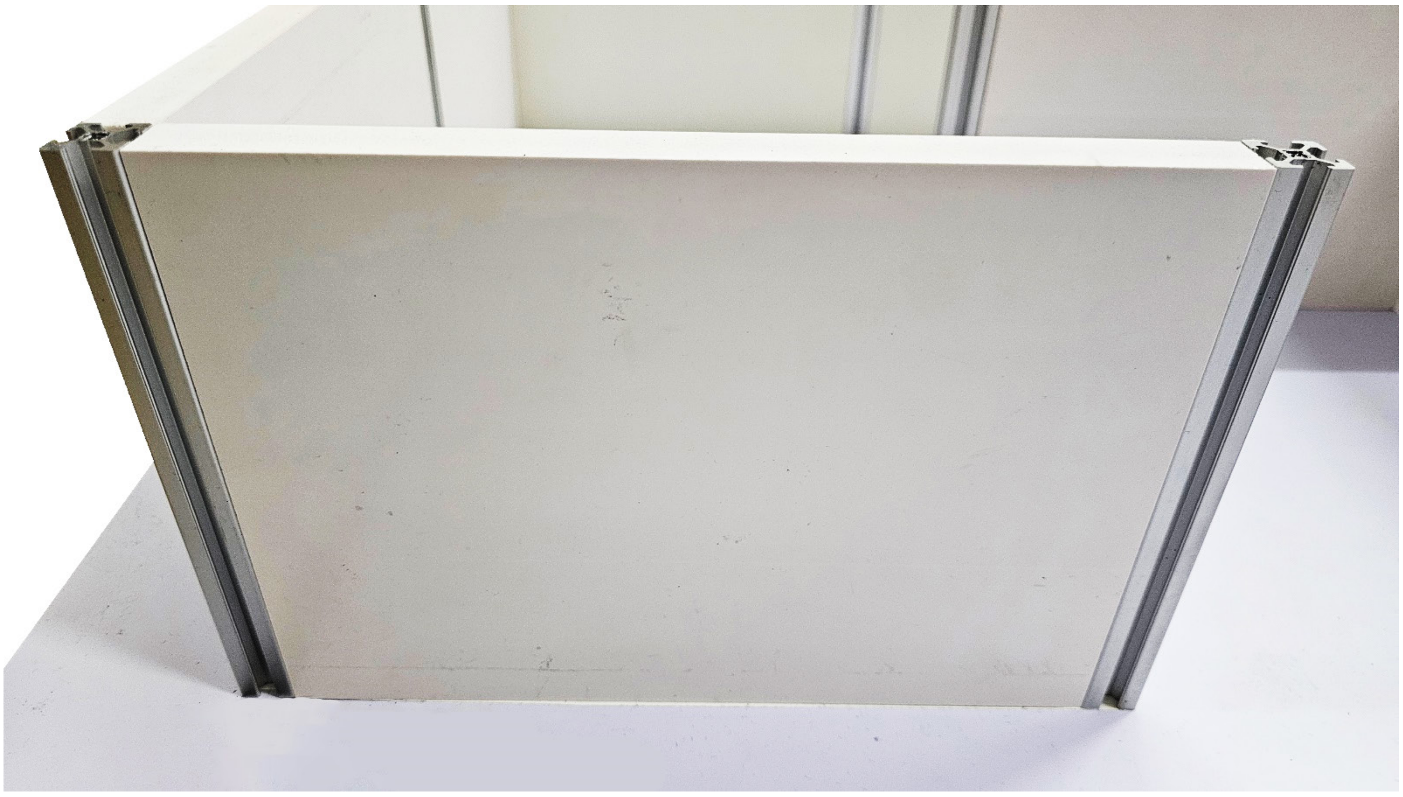
Drill 4 holes to thread screws into the plastic wood sheets as shown in the picture below.



For the field walls made of 28cm x 20cm, 2cm thick plastic wood sheets, the sides must be holed with self-tapping screws to insert into the aluminum profile rails. The screw positions should be measured 5cm down from the top rod, precisely in the middle of the plastic wood sheet, as shown in the picture. Do four holes and thread the self-tapping screws through, leaving a gap of about 3mm as shown in the picture below.

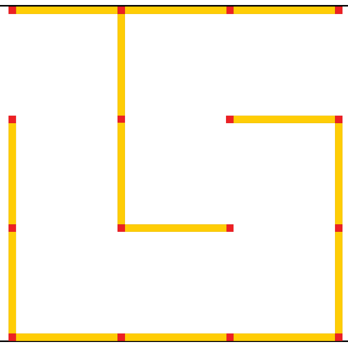
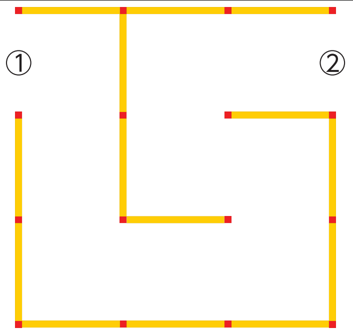
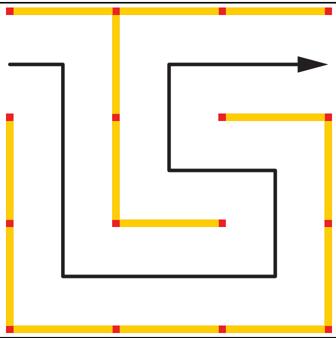


Use double-sided cloth tape to attach between the edge below the cut aluminum profile or the cut plastic wood and the plastic wood on the floor as shown in the picture below.

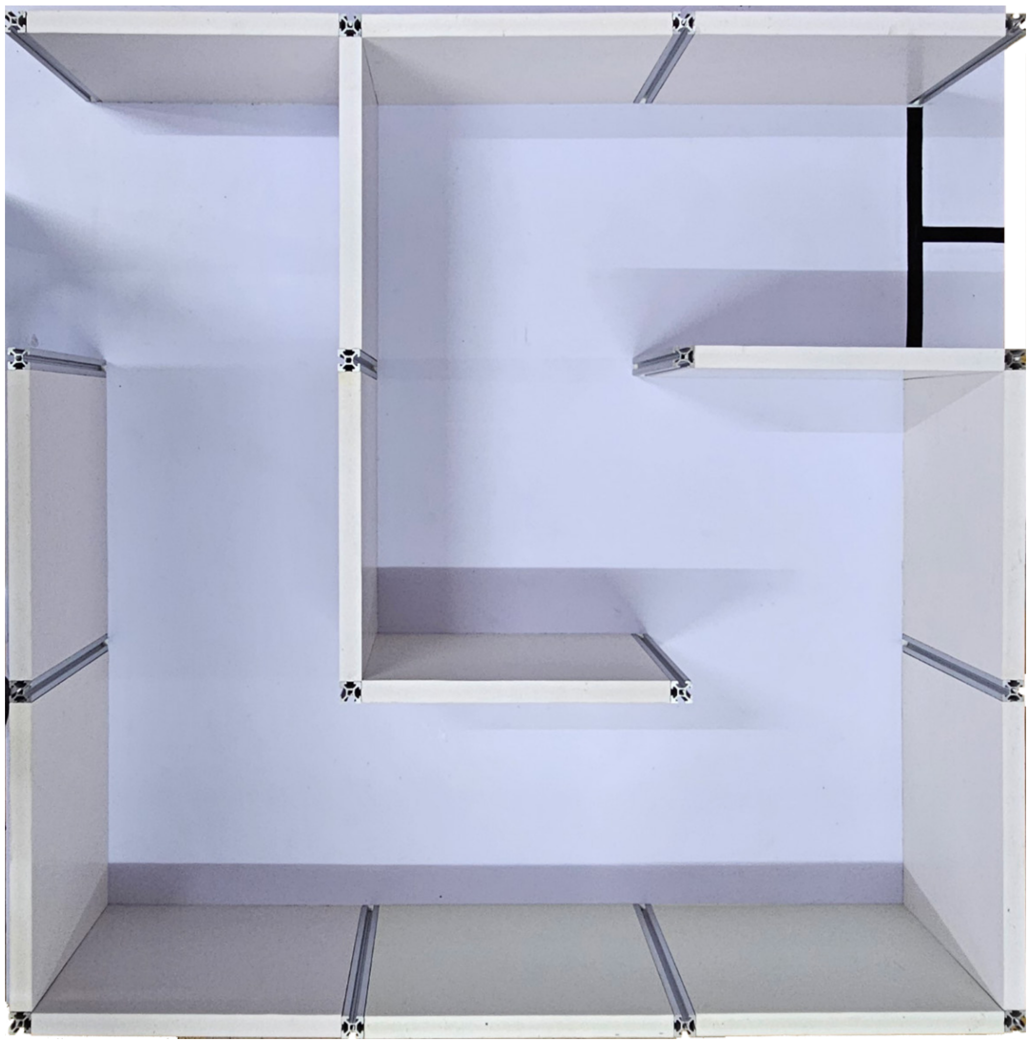


3 Maze Field Example

As shown in the steps below, we have created a maze field example for the maze mission.

		
Step 1: Create maze problem	Step 2: Determine **1** means start point **2** means finish point	Step 3: Let the robot move in the direction shown in the picture above.

See the actual field below like this.



4 Movement in The Maze

The designed maze has a blank to detect on the floor, but there is a wall, so we have to control the robot's movement to move as straight as we can. We have a 2-wheel robot, so we have to use a determined independent left-speed motor and right-speed motor function.

5 Wall Detection

As we said, the designed maze did not have a line, so we had to use other sensors to detect something in it. For example, we had to use an ultrasonic sensor to detect the wall by measuring the distance between its sensor and the wall.

To make sure that the robot is almost at the center of one grid during measuring to affect the next step. In this case, if the robot is far or near detecting the wall after it has completely spun left or right, while the robot is moving, it may crash, slip, slide, or strike the wall side.

6 Spin Left or Right Time

When the robot detects the wall, it should spin left or right to find an available path to the next path.

7 Wall Touch Backward

When the robot spined left or right by applying the time to approach exactly 90-degree rotation, it is also difficult to guess; one thing that can help you is the two switches assembled on the back of the robot. After that, let it move backward, and then they are touched by the wall. Meanwhile, the robot tries to set to be straight as it can.

8 Code Sequence Description

1) Header

```
#define SONAR_READ_CM (analog(5) * 100) / 4095
#define SPD 45
#define kpsonar 7
#define kpspin_t 350
#define kpifr 1500
#define kpfreq 500
#define kpstop_t 300
#include <POP32.h>
```

Description: We have defined exampled some value as macros value;

- SONAR_READ_CM: Read analog raw data and convert it to centimeter.
- SPD: Set default motor speed.
- kpsonar: Set the distance between the ultrasonic sensor and the wall threshold in centimeter.
- kpspin_t: Set spinning time in milliseconds.
- kpifr: Set infrared sensor value threshold.
- kpfreq: Set sound frequency to alarm.
- kpstop_t: Set stoping time in milliseconds.

and have included POP-32 Library.

2) Function "Move forward, then stop temporarily when it detects the wall."

```
void fdSONAR(int cm, int freq, int ao_t) {
  fd2(SPD - 2, SPD);
  while (SONAR_READ_CM > cm);
  ao();
  sound(freq, ao_t);
}
```

Description: Use a determined independent left-speed motor and right-speed motor function to move precisely straightforward. (In this example, we had tested the robot moving tilted right, so we had to decrease the left-speed motor to make the robot straightforward.)

The robot continues moving when it is far from the wall until it is near the wall. Then, it stops temporarily.

3) Function "Spin left, then stop temporarily."

```
void slMAZE(int sl_t, int ao_t) {
  sl(SPD);
  delay(sl_t);
  ao();
  delay(ao_t);
}
```

Description: Use a spinning left robot time and stop temporarily time.

4) Function "Spin right, then stop temporarily."

```
void srMAZE(int sr_t, int ao_t) {
  sr(SPD);
  delay(sr_t);
  ao();
  delay(ao_t);
}
```

Description: Use a spinning right robot time and stop temporarily time.

5) Function "Move backward, touch the wall, and set it to be straight by 2-Switch."

```
void bkSW(int freq, int ao_t) {
  while (1) {
    if (!in(22) && !in(23)) {
      bk(SPD);
      sound(freq, ao_t);
      ao();
      delay(ao_t);
      break;
    } else if (in(22) && !in(23)) {
      sl(SPD);
    } else if (!in(22) && in(23)) {
      sr(SPD);
    } else {
      bk(SPD);
    }
  }
}
```

Description: Unless any switch is touched, it will move straight backward.

If the right switch is touched, it tries to spin left.

If the left switch is touched, it tries to spin right.

If both switches are touched, the motor will move straight backward while touching the wall temporarily to ensure it sets to be straight, then stop and exit the loop by applying "break;".

6) Function “Track and stop on the line.”

```
void stopline(int ifr, int freq, int ao_t) {
  fd2(SPD - 2, SPD);
  while (analog(2) > ifr);
  ao();
  sound(freq, ao_t);
}
```

Description: It is the same as No.2) but uses an infrared sensor to detect black lines instead of an ultrasonic sensor.

7) Main Movement Sequence Code

```
void setup() {
  // put your setup code here, to run once:
}
void loop() {
  // put your main code here, to run repeatedly:
  waitSW_OK_bmp();
  fdSONAR(kpsonar, kpfreq, kpstop_t);
  srMAZE(kpspin_t, kpstop_t);
  bkSW(kpfreq, kpstop_t);
  fdSONAR(kpsonar, kpfreq, kpstop_t);
  slMAZE(kpspin_t, kpstop_t);
  bkSW(kpfreq, kpstop_t);
  fdSONAR(kpsonar, kpfreq, kpstop_t);
  slMAZE(kpspin_t, kpstop_t);
  bkSW(kpfreq, kpstop_t);
  fdSONAR(kpsonar, kpfreq, kpstop_t);
  slMAZE(kpspin_t, kpstop_t);
  bkSW(kpfreq, kpstop_t);
  fdSONAR(kpsonar, kpfreq, kpstop_t);
  srMAZE(kpspin_t, kpstop_t);
  bkSW(kpfreq, kpstop_t);
  fdSONAR(kpsonar, kpfreq, kpstop_t);
  srMAZE(kpspin_t, kpstop_t);
  bkSW(kpfreq, kpstop_t);
  stopline(kpifr, kpfreq, kpstop_t);
}
```

Description: Wait to press the OK button, then let the robot do the mission sequence. After it has completed the mission, it will start waiting to press the OK button again.

9 Code Download

https://pastebin.com/DwQiF8US

10 Video Demo

https://youtu.be/uzIP8nUF6zk

ENG Rev1.0 20240919