Department of Electrical and Electronic Engineering

MEC104(S2) Experimental, Computer Skills and Sustainability

# Lab 3 :Park-Maintainence robot

**Group number：A26**

Group members： Tiankuo.Jiao 1929098

Yuahng.Jiang 1927888

Zhuoyuan.Liu 1927020 Submission date:06/04/2021

# Abstract

*In this task, we designed an automatic water sprinkler robot through Arduino Uno 3. The robot can be used in anywhere that need watering. We installed two motors on the robot to control the robot to move forward, backward and turn. In addition, we also installed a servo motor to control the water spraying.*

*In this mission, we used the following sensors to detect the environment condition and control the motion of motors and micro-servo motor:1.Force sensor 2.Ultrasonic range sensor 3.Gas sensor 4.Temperature sensor. We used the force sensor to detect the remaining water in the container. Temperature and gas sensors were applied to monitoring the environment condition.*

*The Arduino’s main code includes a default loop which functions repeatedly and a default setup which runs once. Before entering the main body ,the variables can be defined as global variables, whose values can be changed in self-defined function and automatically change in the main loop without the needs for parameters delivering.*

*After the rain water accumulates, the micro-servo motor is ready to run. At the same time, we also need to consider the conditions of temperature and gas in the environment. According to the magnitude and the changing rate of the temperature or the gas concentration, the motor runs at different speed to achieve the higher efficiency and sustainability.*

*In the future, we will continue to improve the functions and sustainability of the car and further reduce the cost of car production.*

***Catalogue***

[Abstract 2](#_bookmark0)

1. [Introduction 4](#_bookmark1)
   1. [Background of the robot 4](#_bookmark2)
   2. [Robot Introduction 4](#_bookmark3)
   3. [Overall function introduction 5](#_bookmark4)
2. [Sustainability 6](#_bookmark5)
3. [Design 8](#_bookmark6)
   1. [General description 8](#_bookmark7)
   2. [Detailed design steps 9](#_bookmark8)
4. [Theory 11](#_bookmark9)
   1. [micro-servo motor 11](#_bookmark10)
   2. [sensors 11](#_bookmark11)
5. [Experiment Method 12](#_bookmark12)
   1. [Mechanical Part 12](#_bookmark13)
   2. [Code Part 13](#_bookmark14)
      1. [State machine 15](#_bookmark15)
      2. [Speed control 16](#_bookmark16)
      3. [Problems Solving 16](#_bookmark17)
6. [Results 17](#_bookmark18)
7. [Discussion 19](#_bookmark19)
8. [Conclusion 20](#_bookmark20)
9. [Author list and contribution 22](#_bookmark21)
10. [References list 23](#_bookmark22)
11. [Appendix 23](#_bookmark23)

# Introduction

## Background of the robot

With the development of the city, the further improvement of urban greening is very important. Therefore, we can find that more and more cities begin to build green parks for citizens' recreation. However, the construction of such parks requires dedicated staff to water the trees and flowers in the parks, which is inconvenient and costly. Therefore, after a series of discussions, our group decided to design an **automatic watering robot**. Among them, the water of the robot comes from rain, which can also achieve the purpose of environmental protection and sustainability.

Based on the experience of designing a smart car last time, we finally decided to design a walking-watering robot which can move.

## Robot Introduction

The automatic watering robot designed by us is composed of water storage bucket, motor, servo motor and various sensors. The water storage bucket is mainly used to store the accumulated rainwater when it rains. The motor controls the robot's forward, backward, turning and other operations. Sensors control when the robot waters the trees and flowers. Moreover, the servo motor is used as a sprinkler. In this design, the devices we use are as follows:

1. Two motors: Control the car's forward, backward,turning.
2. One power supply: To power H bridge motor driver
3. One servo motor: it is responsible for the function of water spraying. When all conditions are met, the servo motor starts to work immediately.
4. A Force sensor: it is responsible for detecting the amount of rain water. When the collected rain water makes the value of the Force sensor reach 1N, the servo motor starts to work.
5. An ultrasonic distance sensor: it is responsible for detecting the distance between the car and the obstacle. When the ultrasonic sensor detects that the distance between the car and the obstacle is less than 50, the robot will first back and then turn left to avoid the obstacle.
6. A gas sensor: it is responsible for detecting the current gas concentration. When the gas concentration is less than the data we set, the servo motor will stop working. When the gas concentration is high, the forward motor will rotate slower, allowing the plants to be poured with more water.
7. A temperature sensor: Detect the current temperature state. When the temperature is below 25 *C* , the servo motor will stop working. When the

temperature is higher, the rotation speed of the forward motor will slow down to pour with more water.

1. One RGB : Indicator light

## Overall function introduction

1. First, the robot needs to store rain. When it rains, a bucket carried by the robot begins to accumulate water.
2. When the rain accumulates to a certain extent (making the Force sensor's force greater than 1N), the servo motor and the two motors start to work. Moreover, we default the two motors to start moving forward.

The principle of servo motor control water spraying is: servo motor control a pump downward to pressure water through the gear. Then, water will flow out through small holes on all sides. When all the rain water is spilled, the servo motor will stop working.

1. If the temperature drops below 25 *C* , the servo motor will stop working.

When the temperature is higher, the rotation speed of the forward motor will slow down to pour with more water.

1. Next, we judge the concentration of the gas. When the gas concentration is less than the data we set, the servo motor will stop working. Moreover, when the concentration of the gas is over the threshold value, the servo will start working. Also, when the gas concentration is high, the forward motor will rotate slower, allowing the plants to be poured with more water.
2. When all the rain water is sprayed out, the servo motor will stop working. Then, the robot will be in the next round of rainwater collection.

# Sustainability

The automatic sprinkler robot designed by us has the characteristics of automation and sustainability.

First of all, our robot uses the rain to water the plants, which can save water which can be used in other areas. As we all know, the rainwater is rarely recycled. Through the robot we designed, we can recycle the rain water to achieve the purpose of secondary utilization. When plants are short of water, we use rain to water them.

Moreover, we don't have to spend extra money to buy water to irrigate the flowers. The rain is free!!!

Secondly, we can save the workload of people by watering water in a robot way. In the past, we might water it by hand, which is time-consuming and inefficient. We can avoid this problem by designing robots that are fully autonomous which can operate on their own without having to be operated by human beings. The robot can also avoid obstacles automatically.

In addition, our robots are very intelligent. We designed the automatic robot to be able to decide whether to water or not , based on the environment at that time. Gas sensors and temperature sensors are installed on our robot, which can be sprayed according to the actual situation

To sum up, we believe that the robot we designed has the characteristics of sustainability and can meet the requirements of environmental protection and energy saving. In addition, the production cost of our robot is very low.

# Design

## General description

As is mentioned in the introduction part, we will build a smart robot used to perform specific reactions to the environment. Including:

When the temperature is higher than a specific value, split water to decrease the environment temperature.

1. When the quality of the air is bad, such as containing fog and haze or poisonous gases, split water to improve the quality of the air.
2. According the magnitude and change rate of the temperature and the air quality in the environment, the robot should be able to move at different speeds. If the magnitude of temperature or the density of dirty gases is high, the robot will move slower. If the decreasing rate of the temperature and the density of the gas is high, then the moving speed is high too. By doing this, the robot can have the highest efficiency. Thus, the robot would save electricity and achieve sustainability.
3. When the temperature is lower that 0 or equal to 0, the water is frozen so that the robot would shut down.

### Apart from that, the robot should also contain basic functions that can ensure the functions above can work properly:

1. The car should be able to move straight forward when there is enough water in the container. Simultaneously, the temperature is high or the density of gas is high. When come across an obstacle, the robot should be able to avoid it by firstly moving backward and then turn left.
2. The robot should be able to gather water in normal times. If the water stored in the robot is too little, the robot will not function at all.

## Detailed design steps

### Power source

Firstly, we need a extra 12V DC power source to power this Arduino UNO. Since in real life, if we simply connect the Arduino to a lower power source like personal computer, the power supply is not enough to achieve all the function.

### DC motors

Then in order to achieve the function of moving forward, moving backward and turn left, we need two motors to drive the robot.

When both speed is positive, the robot is moving forward. When both speed is negative, the robot is moving backward. When the left motor is negative and the right motor is positive, the robot is turning left.

### The force sensor

The robot need to have enough water to start functioning. So we need force sensor to detect the water pressure at the bottom surface of the container. If the robot contains too little water, which means the pressure is too small, the numerical value of the force sensor will be low. Then the other sensors on the robot should have no impact to the robot.

So the force sensor should control all the sensors and micro-servo motor and

motor.

### Gas sensor

In order to sense the quality of the environment, we need a gas sensor. In tinkerCAD, the function is achieved by moving the gas close by and far away. Then the density of the gas is changed. We can see this in serial window.

If the density is high, the micro-servo motor should spin fast on the condition of water supply is sufficient. If the density decreasing rate is high ,the motor’s spin speed should be high. So that the robot would move faster.

So the gas sensor should be able to control two parts: motor and micro-servo

motor.

### Temperature sensor

In order to detect the environmental temperature, the temperature sensor is needed. When the temperature is high, the the micro-servo motor should spin fast on the condition of water supply is sufficient. If the temperature’s decreasing rate is high, the motor’s spin speed should be high. So that the robot would move faster. Also, when the temperature is below 0℃, the whole system should not work.

So the temperature sensor should be able to control two parts: motor and

micro-servo motor. If the temperature is below 0, all system cease functioning.

### Ultrasonic distance sensor

In order to avoid the obstacle, the ultrasonic distance sensor is needed. This senso should be build in front of the robot. If the distance is too small, the motor will rotate counterclockwise to move backward. Then turn left to avoid the obstacle.

So the Ultrasonic distance sensor should be able to control the motor.

### Micro-servo motor

As we mentioned above, the speed of the micro-servo motor is related to three sensors: force sensor, temperature sensor and gas sensor. If the water pressure is low, the micro-servo motor should not move at all.

* 1. Circuit design

According to our discussion, the final circuit is follows:

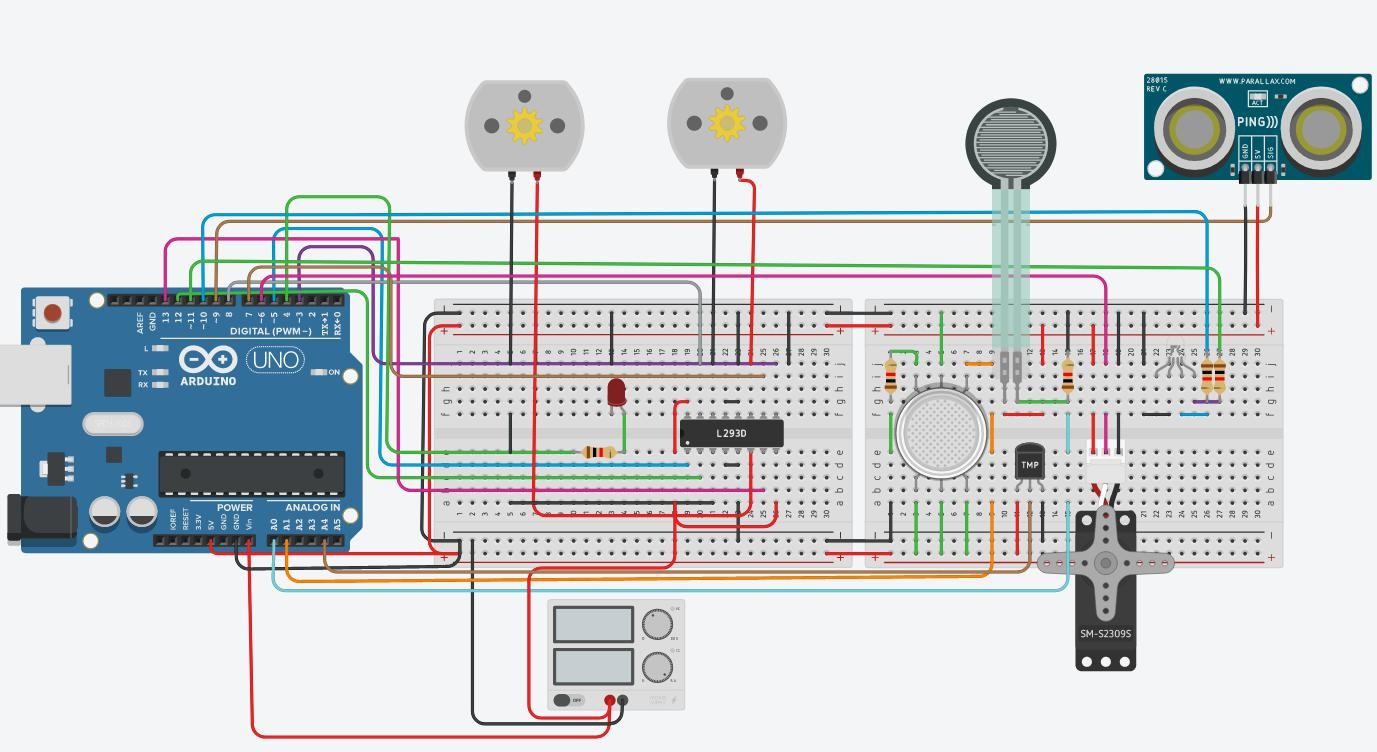


Fig.1. The simulated circuit

# Theory

## micro-servo motor

Servo motors are linear or rotary actuators that provide fast and accurate position control for closed loop position control applications. Unlike large industrial motors, servo motors are not used for continuous energy conversion. Servo motor has high speed response due to its small inertia, small diameter and long rotor length.[1]

The servo motor works in the servo mechanism and uses position feedback to control the speed and final position of the motor. Servo motor includes motor, feedback circuit, controller and other electronic circuits.

## sensors

In our design, we used the following sensors to detect the environment condition and control the motion of motors and micro-servo motor:1.Force sensor 2.Ultrasonic range sensor 3.Gas sensor 4.Temperature sensor. We used the force

sensor to detect the remaining water in the container. Temperature and gas sensors were applied to monitoring the environment condition.

# Experiment Method

## Mechanical Part

### The mechanical structure of the system is described as follows:

The servo motor will have a **gear** fixed on its rotating part, and the pump will be fixed to a **gear rack**, therefore each time the servo rotates it can drive the pump to have a displacement which will squeeze the water in the tank and spill a portion, and each time of spilling is supposed to have a not-short period. The design of the tank is as below:

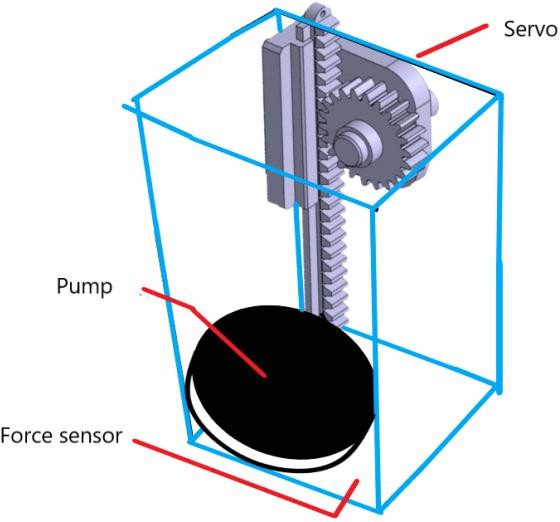


Fig. 2. The mechanical design of the tank

The tank is vertically positioned, and the water is squeezed downwards, which allows the force sensor to judge the amount of water by the **height** of water according to the **static pressure** equation. The water is spilled through a circular stripe around the bottom of the tank to the surrounding environment. The stripe should have tiny holes distributed all over the surface.

For the chassis, since only two motors are used, there should be an omni-directional wheel to stabilize. Like the picture below:



Fig. 3. The three-wheel structure

The driving power source for the robot must have sufficient power and capacity. It is suggested that a **high-power solar cell** with 12V output be provided. In this way, the robot can recharge when being static, which further improve its sustainability.

## Code Part

The coding and logic structure of the robot system is as follows:

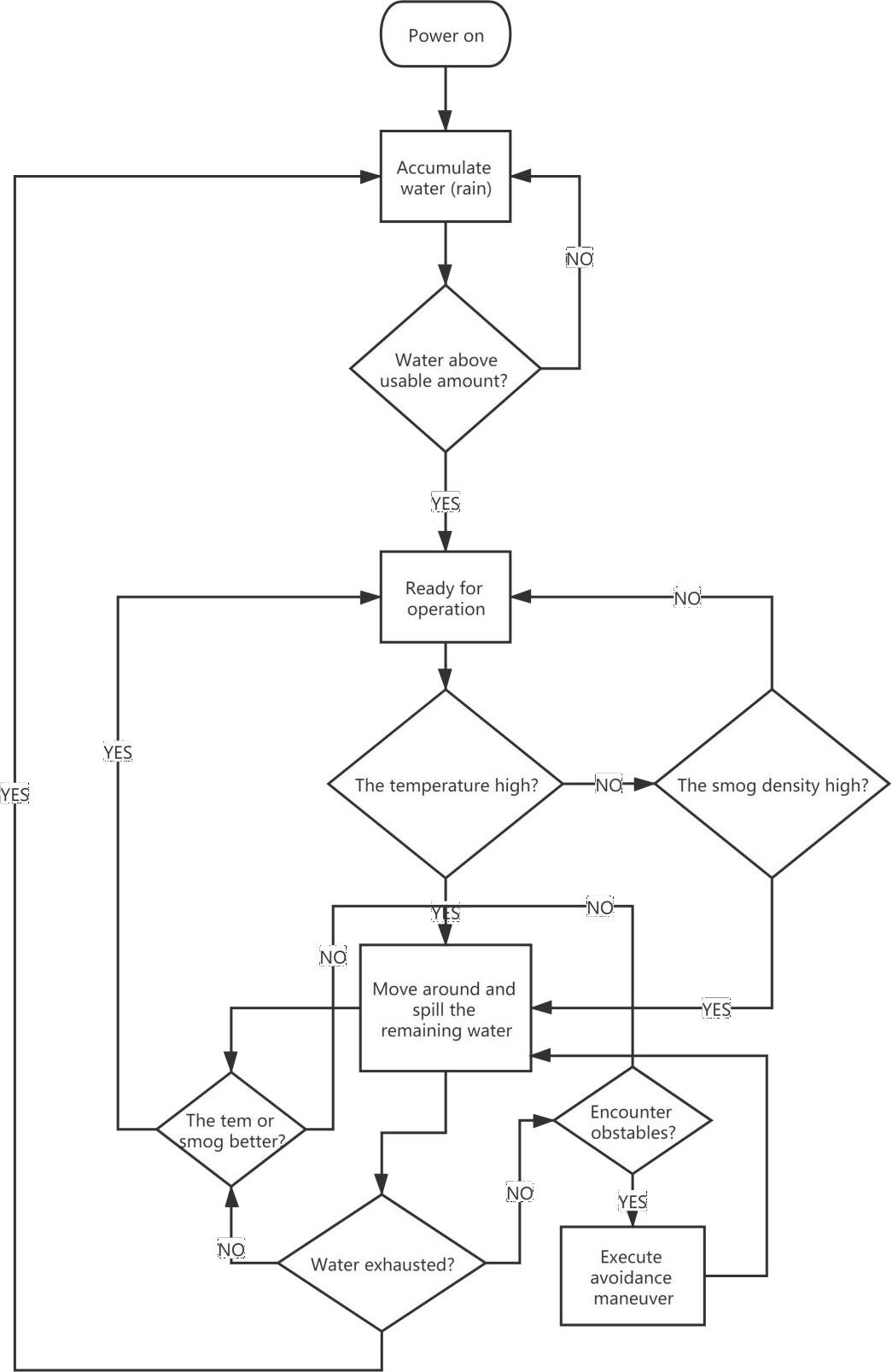


Fig. 4. The overall flow chart of the robot’s working logic

The Arduino’s main code body includes a default loop which is run repeatedly and a default setup which is run once. Before entering the main code body, the variables can be defined as global variables, whose values can be changed in

self-defined function and automatically change in the main loop without the needs for parameters delivering.

## State machine

The control of the machine’s state in the main loop and the subordinate ‘for’ loops was realized by state-machine method in which one flag was used, and no interruption method was used. The flag is used in these situations:

* + - 1. Judging if the water amount is over the usable value, which is represented by the force sensor reading over 1N. If yes, then it can proceed to the temperature and smog judging.
      2. Judging if the temperature is under 0℃. If yes then the freezing water is not suitable for working, then the machine will not function while sending warning to the upper monitor through possible IOT communication, in which case, the operator can choose to shut down the power.
      3. After entering the state of operating after smog or temperature is detected, judge if the bad situation is improved over the threshold value, in the ‘for’ loop of the major operations. If yes, like the temperature is below 35℃, then stop moving and stop spilling water.
      4. In the function of the major operations, judging if there is an obstacle ahead. If yes, then the pump will stop spilling water, while only the chassis will execute maneuver. This is to save water as the elementary avoidance could take long at any corner.
      5. For example, in the coding period below, the water amount judgement is set at the beginning of each main loop cycle. If there is no enough water, then flag=0 and the machine will not enter the operational mode.

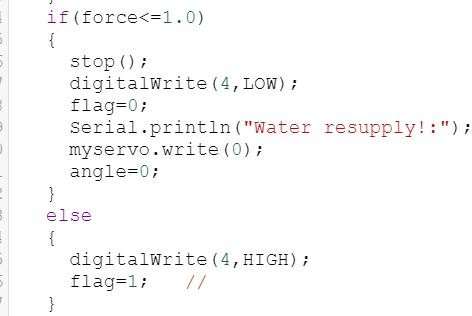


Fig.5. The water judgement

## Speed control

To reach maximum efficiency in each operating cycle, which means to lower the temperature and the smog density **by a value** as much as possible and **in an area** as large as possible in the park, the moving speed of the robot was set to be able to adapt according to the air sensors’ values and the rate in which the values decreasing. The ‘analogWrite()’ function can be input speed parameter which is calculated by: **The offset value-Air condition value+The decreasing rate**. Which means that if the air is bad around a point then the robot moves slower to make more improvement, if the condition improves fast then the robot moves faster to other points. However, this simple design cannot guarantee the highest efficiency. More complex algorithm and sensors are required.

## Problems Solving

When debugging, both logical and erroneous problems occurred. For example, when writing the major operation function ‘react()’, if it was implemented with the state parameter delivered in and become ‘react(flag)’, then the servo will rotate while operating avoidance maneuver, which was not expected. The reason was inferred to

be that after the delivery the global variable ‘flag’ became **local variable** and could only be changed **within the function** ‘react(flag)’. After deleting the parameter delivery, ‘flag’ could stay as global and the expectation was achieved with the servo ceasing rotating when avoiding obstacle.

# Results

According to the function we intended to make in the design part. We should check the outcome one by one.

### Moving forward at specific condition.

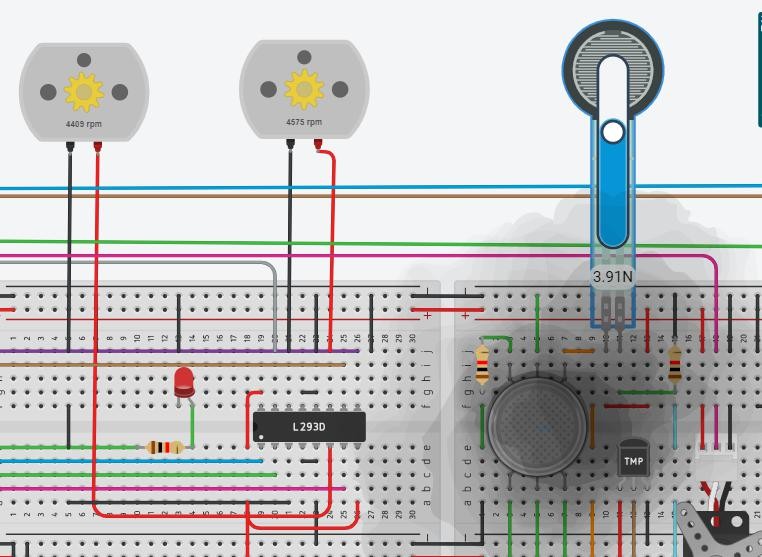
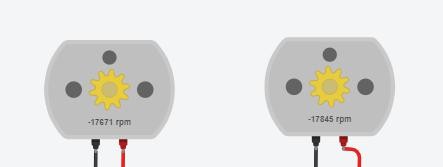


Fig.6. Result of function 1

As we can see, the motor starts to spin in positive angular speed when the force sensor is large enough and the gas density is high.

### Avoid obstacle



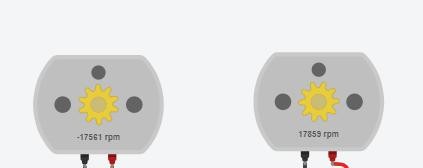


Fig.7. Result of function 2

As we can see, the motor firstly spin counterclockwise together to move backward then the left motor has negative speed while the right motor has positive speed. This means the robot is moving left.

### When temperature is high, the micro-servo motor start moving

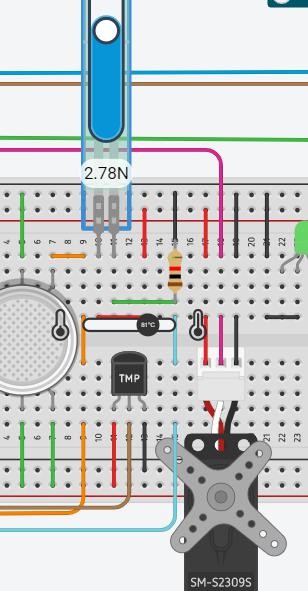


Fig.8. Result of function 3

As we can see, the water is sufficient, the temperature is high, the micro-servo motor is moving.

### When the gas density is high, the micro-servo motor start moving

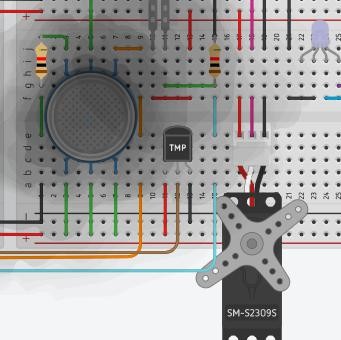


Fig.9. Result of function 4

As we can see from the serial window, the gas density is high. The micro-servo motor start moving.

### When water is insufficient, cease working 6.When temperature is below 0, cease working.

**7.The spinning rate of the micro-servo motor should be effected by temperature sensor or gas sensor**

# Discussion

The design of this intelligent robot presents an existence of a room for improvement.

For the chassis movement, the DC motors’ rotation speed have an instability, which might be caused by the external factors such as the wind drag, the variant ground friction, and the electromagnetic field, and the internal factors such as the imperfection of the bearings and the instability of the DC current. The inconsistency of the two motors would cause the robot to deviate from the path, or cause more severe events. Therefore, if a constant rotation speed of the motors is to be reached, the PID control method should be applied.

Further more, the obstacle-avoidance system is not enough, because there is only one supersonic sensor. If the robot is blocked from moving somehow but the motor is still rotating, even the motor with encoder cannot function in inspecting this error, which requires an acceleration sensor in addition that the Tinkercad lacks.

Initially, it was expected that two Arduino UNO boards could be utilized to provide more IO Pins for more supersonic or infrared distance sensors, however this proposal was abandoned because if it is to be achieved a customized PCB board would be more effective.

For the coding structure, the interruption method could be more effective in multi-targets controlling than the flag method.

The robot can only solve one situation in each operation process. However, if the temperature and the smog density are both critical, it would be better if the robot could increase the efficiency of operating, like spilling water at a higher rate.

# Conclusion

In the open project, our group developed an automatic robot which can accumulate rain water and use the water to automatically improve the air quality in a

park, and the whole process was simulated using the TinkerCAD software. The mechanical structure of the robot is constructed by: a **water tank**, which is equipped with a servo with a gear which can drive the pump, and a force sensor at the bottom surface, and a holed stripe around the tank bottom; a **movable chassis**, where the tank is fixed and the Arduino board is fixed, and which is equipped with gas sensor, temperature sensor and distance sensor. The coding structure applied serial loops for the general operation and state machine for judgement in the loops.

The robot can realize the following functions if it is constructed in reality:

* 1. Accumulate rain water and judge if the amount is usable.
  2. Detect the temperature and the smog density in the environment.
  3. If either the temperature or the smog density is high and the water is enough and not freezing, move around and spill water to ameliorate the condition.
  4. If the conditions are better, stop working to save water. Then, if the conditions get bad again, spill the remaining water.
  5. Detect obstacles when moving forward and avoid obstacles.
  6. Repeat the process after water is used up.
  7. Send information to the upper monitor.

Generally, this robot is competent for achieving sustainability. Nevertheless, there still exists much room for improvement regarding controlling and stability.

# Author list and contribution

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Name** | **Contribution**  **Description** | **Percentage**  **（%）** | **S** |
| **1** | Zhuoyuan.Liu | Code part: design the motor, force sensor, integrate the circuit.  Report part: experiment method, conclusion, discussion. design the poster help  record the video. | 33.33 |  |
| **2** | Tiankuo.Jiao | Code part: design the temperature sensor, gas sensor Report part: abstract, introduction, sustainability design the poster help record the  video. | 33.33 |  |
| **3** | Yuhang.Jiang | Code part: design the ultrasonic distance sensor and micro-servo motor  Report part: Design, result, contribution table design the poster help record the  video. | 33.33 |  |

# References list

[1] Electronics Hub, “Servo-Types and Working Principle”, [Online].Available: https:[//www.electronicshub.org/servo-motors/](http://www.electronicshub.org/servo-motors/)

# Appendix

#include<Servo.h> Servo myservo;

//Macro definition## #define TMP A4 #define FORCE A0 #define gSensor A1

//3,4

const int PWM\_M1 = 3; const int IN1\_M1 = 7; const int IN2\_M1 = 8;

//1,2

const int PWM\_M2 = 5; const int IN1\_M2 = 13; const int IN2\_M2 = 12;

float temp; //temperature float force;//water pressure float state\_new;

float state\_old; float dstate;

int Fog; //Smog density int buzzer = 2;

int speed = 200; int speed1;

int distance; int servo = 6; int angle=0;

int sensorVal = 0; int optimum = 150; int flag=1;

int flag2=1;

int blue=10; int green=11;

void setup()

{

Serial.begin(9600); pinMode(TMP,INPUT);//Temperature

sensor

pinMode(FORCE, INPUT);//Force sensor

pinMode(4, OUTPUT);//LED pinMode(PWM\_M1,OUTPUT); pinMode(PWM\_M2,OUTPUT); pinMode(IN1\_M1,OUTPUT); pinMode(IN2\_M1,OUTPUT); pinMode(IN1\_M2,OUTPUT); pinMode(IN2\_M2,OUTPUT); pinMode(gSensor, INPUT); pinMode(servo, OUTPUT); myservo.attach(6);

}

void loop()

{

//\*\*\*\*\*\*Judge if the water is full\*\*\*\*\*\*\*\*\*//

force = (float) analogRead(FORCE); force=0.00005\*force\*force -

0.0033\*force + 0.2162; Serial.println("Water pressure:"); Serial.println(force);

if (angle>=170)

{

Serial.println("Water exhausted!");

stop();

}

for(angle; angle<180; angle+=flag\*10)

if(force<=1.0)

{

stop(); digitalWrite(4,LOW); flag=0;

Serial.println("Water resupply!:"); myservo.write(0);

angle=0;

}

else

{

{

critical.");

Serial.println("Fog:"); Serial.println(Fog); Serial.println("temp:"); Serial.println(temp); Serial.println("Temperature

Serial.println("angle:"); Serial.println(angle); state\_old=temp;

temp =

digitalWrite(4,HIGH); flag=1;

}

temp = map(((analogRead(TMP) - 20)

\* 3.04), 0, 1023, -40, 125);

Fog = analogRead(gSensor); if (temp<=0)

{

Serial.println("Below frozen point.");

flag=0;

}

map(((analogRead(TMP) - 20) \* 3.04), 0,

1023, -40, 125);

state\_new=temp;

dstate=5\*(state\_old-state\_new);

react(flag); if(temp<35)

{

flag=0; stop(); break;

}

Serial.println("Fog:"); }

Serial.println(Fog); }

Serial.println("temp:"); Serial.println(temp);

else if (Fog>=optimum&&flag==1)

{

if(temp>=35&&flag==1)

{

analogWrite(green,temp\*5);

//\*\*\*\*\*\*\*\*\*\*\*\*\*// angle=170-17\*force; myservo.write(angle); Serial.println("angle:"); Serial.println(angle);

//\*\*\*\*This part is for real-situation design!

//Can be deleted for simulation.

analogWrite(blue,Fog);

//\*\*\*\*\*\*\*\*\*\*\*\*\*Can be deleted for simulation.//

angle=170-17\*force; myservo.write(angle); Serial.println("angle:"); Serial.println(angle);

//\*\*\*\*Because This part is for real-situation design!

for(angle; angle<180; angle+=flag\*10)

{

Serial.println("Fog:"); Serial.println(Fog); Serial.println("temp;"); Serial.println(temp); Serial.println("Smog critical."); Serial.println("Angle:"); Serial.println(angle); react(flag);

Serial.println("Flag after react;"); Serial.println(flag); state\_old=Fog;

Fog = analogRead(gSensor); state\_new=Fog; dstate=state\_old-state\_new; if(Fog<optimum)

{

flag=0; stop(); break;

}

}

}

else

{

noTone(buzzer); myservo.write(0); delay(500);

}

delay(10);

}

void react(int flag)

{

myservo.write(angle); delay(1000); forward();

distance= 0.01723\*readUltrasonicDistance(9,9);

Serial.println("distance"); Serial.println(distance); if(distance>0&&distance<50)

{

flag=0; backward();

delay(1000); turnLeft(); delay(1000);

}

else

flag=1; if(force==0)

{

stop(); flag=0;

}

}

void forward()

{

Serial.println("forward");

speed1= speed-state\_new+dstate; if(speed1>255)

speed1=255; if(speed1<50)

speed1=50; analogWrite(PWM\_M1, speed1); analogWrite(PWM\_M2, speed1); digitalWrite(IN1\_M1, HIGH); digitalWrite(IN2\_M1, LOW); digitalWrite(IN1\_M2, HIGH); digitalWrite(IN2\_M2, LOW);

}

void backward(){ Serial.println("backward"); analogWrite(PWM\_M1, speed); analogWrite(PWM\_M2, speed);

digitalWrite(IN1\_M1, LOW); digitalWrite(IN2\_M1, HIGH);

digitalWrite(IN1\_M2, LOW); digitalWrite(IN2\_M2, HIGH);

}

void turnLeft(){ Serial.println("turnLeft"); analogWrite(PWM\_M1, speed);

analogWrite(PWM\_M2, speed);

digitalWrite(IN1\_M1, HIGH); digitalWrite(IN2\_M1, LOW);

digitalWrite(IN1\_M2, LOW); digitalWrite(IN2\_M2, HIGH);

}

// Reads the echo pin, and returns the sound wave travel time in microseconds

return pulseIn(echoPin, HIGH);

}

void turnRight(){ Serial.println("turnRight"); analogWrite(PWM\_M1, speed); analogWrite(PWM\_M2, speed);

digitalWrite(IN1\_M1, LOW); digitalWrite(IN2\_M1, HIGH);

digitalWrite(IN1\_M2, HIGH); digitalWrite(IN2\_M2, LOW);

}

void stop(){ Serial.println("stop"); digitalWrite(IN1\_M1, LOW); digitalWrite(IN2\_M1, LOW);

digitalWrite(IN1\_M2, LOW); digitalWrite(IN2\_M2, LOW);

}

long readUltrasonicDistance(int triggerPin, int echoPin)

{

pinMode(triggerPin, OUTPUT); // Clear the trigger

digitalWrite(triggerPin, LOW); delayMicroseconds(2);

// Sets the trigger pin to HIGH state for 10 microseconds

digitalWrite(triggerPin, HIGH); delayMicroseconds(10); digitalWrite(triggerPin, LOW); pinMode(echoPin, INPUT);