113 學年度第一學期工程設計專題
Arduino PID 自動跟隨車

沈威宇

February 2, 2025

Contents

1 Summary

This project centers on the development of an Arduino-based auto-following kart using a PID (Proportional-Integral-Derivative) controller, blending control theory with practical implementation. The objective was to design an autonomous kart capable of maintaining a consistent distance from a leading object while navigating a dynamic environment.

At the core of the project lies control theory, which governs the behavior of dynamic systems. The PID controller adjusts outputs automatically based on feedback to minimize error and stabilize the system. It is composed of three main components: Proportional Control, which adjusts the output in response to the current error; Integral Control, which accounts for accumulated past errors; and Derivative Control, which predicts future errors by analyzing the rate of change.

As the most proficient coder on the team, I was responsible for implementing the PID controller and integrating it with Arduino. To maximize productivity, I utilized Debian Linux emulation on Android devices through tools such as proot and QEMU system emulation in Termux, allowing me to work efficiently during commutes. Key tools like Vim, G++, GDB, and Git ensured smooth development and debugging.

After gaining experience in designing and coding Arduino bots in Tinkercad, I created a basic PID controller in C++ that could function on both Arduino and non-Arduino systems. Advanced features were later introduced, including adaptive gain adjustment, which dynamically modifies proportional gain based on the error's rate of change, adaptive integral control to mitigate windup, and an extra derivative term, which handles rapid error changes and boosts damping effects. Inspired by the Ziegler–Nichols method, I invented a error history-based tuning system that utilizes a linked list structure for improved prediction accuracy. To optimize the PID controller, I developed a simulation framework featuring a "Car" class to emulate both the follower and leader vehicles in a dynamic environment. Multi-threading was employed to test various parameter combinations simultaneously, significantly accelerating the tuning process.

The physical kart was designed to balance cost-effectiveness with high functionality, drawing inspiration from the horseshoe crab's biological characteristics and fluid mechanics. The design of the kart began with detailed 3D modeling to visualize its structure and ensure proper alignment of all components. The kart's frame and other physical parts were fabricated using simple, low-cost, but sturdy materials such as wooden boards, TT motors, and a L293D motor driver. Two ultrasonic sensors were mounted strategically on either side of the kart to measure distances from obstacles. The difference in these measurements was used to adjust the kart's angular velocity, ensuring it stayed aligned with its intended path. The average of the distances controlled the kart's linear speed, allowing it to maintain a steady pace. A PID controller was employed to fine-tune both angular and linear velocities, ensuring the kart maintained a safe and consistent distance from the leader while avoiding sudden deviations. The L293D motor driver translated the velocity adjustments into precise motor control, enabling smooth and responsive movements.

Lastly, I created the project reports for our team using LaTeX, based on a template I designed. Through this process, I gained deeper insights into the practical use of XeTeX and LuaTeX while developing various useful commands and strengthened my C, C++, and JavaScript skills while converting between formats.

2 Tinkercad Practices

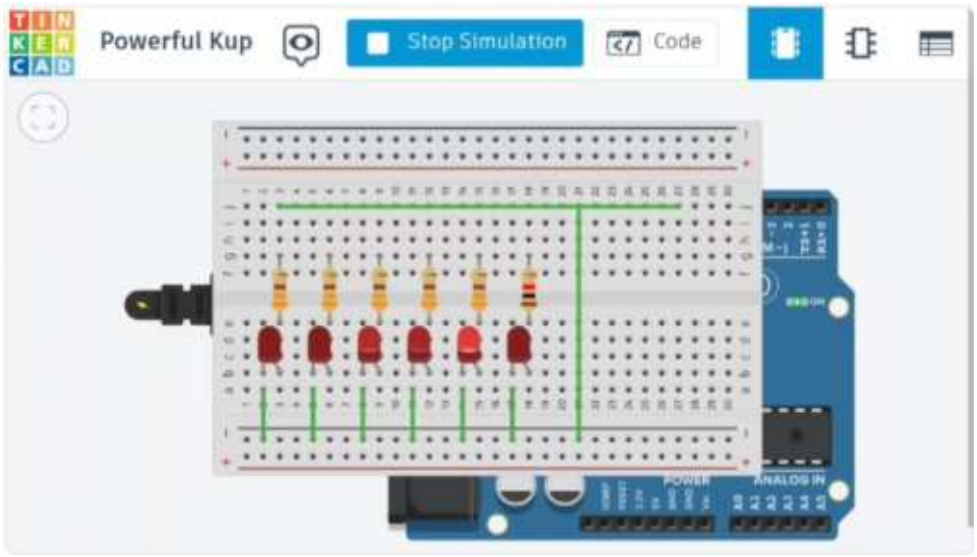
I Flashing LED Light Implementation

i Code

```
/**
 * @brief Controls a set of LEDs with a random blinking pattern.
 *
 * This Arduino sketch configures a set of LEDs connected to specific pins on the
 * microcontroller and controls them with a random blinking pattern.
 *
 * **Pattern Explanation:**
 * - The LED at the current index 'i' is turned on, then turned off after a delay.
 * - The delay alternates between 100 milliseconds (ms) and 200 ms based on the
 *   parity of the index 'i' (even indices have a 100 ms delay, odd indices have
 *   a 200 ms delay).
 * - After turning off the LED, a random number is generated to decide how much to
 *   increment the index 'i' for the next LED to be activated.
 * - The increment value is determined as follows:
 *   - If the random number is 1, the increment is a random number between 1 and 2.
 *   - If the random number is 0, the increment is a random number between 4 and 5.
 * - The index 'i' is updated by adding this increment value, and then wrapped around
 *   using modulo 6 to ensure it stays within the valid range of LED indices (0 to 5).
 *
 * As a result, the pattern of which LED is lit and the timing between LEDs is
 * randomized, creating a dynamic and unpredictable LED display.
 */

const uint8_t LEDs[] = {0b1000, 0b1001, 0b1010, 0b1011, 0b1100, 0b1101}; ///< Array of LED pin numbers.
const uint8_t total = sizeof(LEDs); ///< Total number of LEDs.
uint8_t i = 0; ///< Index of the currently active LED.

void setup() {
    /**
     * @brief Sets up the LED pins.
     *
     * Configures each LED pin as an OUTPUT and initializes each LED to LOW
     * (off) state.
     */
    for (uint8_t i = 0; i < total; i++) {
        pinMode(LEDs[i], OUTPUT); ///< Set the LED pin as an OUTPUT.
        digitalWrite(LEDs[i], LOW); ///< Initialize the LED pin to LOW (off).
    }
}

void loop() {
    /**
     * @brief Controls the LEDs with a blinking pattern.
     *
     * Turns on the LED at the current index, waits for a variable amount of time,
     * then turns it off. The delay time alternates between 100 ms and 200 ms based
     * on whether the current index is even or odd. After turning off the LED, a random
     * number is generated to decide how much to increment the index 'i' and which LED
     * to activate next.
```

```
     */
    digitalWrite(LEDs[i], HIGH); ///< Turn on the current LED.
    delay(100 + i % 2 * 100); ///< Delay for 100 ms or 200 ms based on index 'i'.
    digitalWrite(LEDs[i], LOW); ///< Turn off the current LED.

    // Generate a random number between 0 and 1.
    uint8_t temp = random(0, 2);

    // Generate a random increment based on the value of 'temp'.
    if (temp) {
        temp = random(1, 3); ///< Random increment between 1 and 2.
    } else {
        temp = random(4, 6); ///< Random increment between 4 and 5.
    }

    // Update the index 'i' and ensure it wraps around using modulo operation.
    i = (i + temp) % 6;
}
```

ii Screenshot



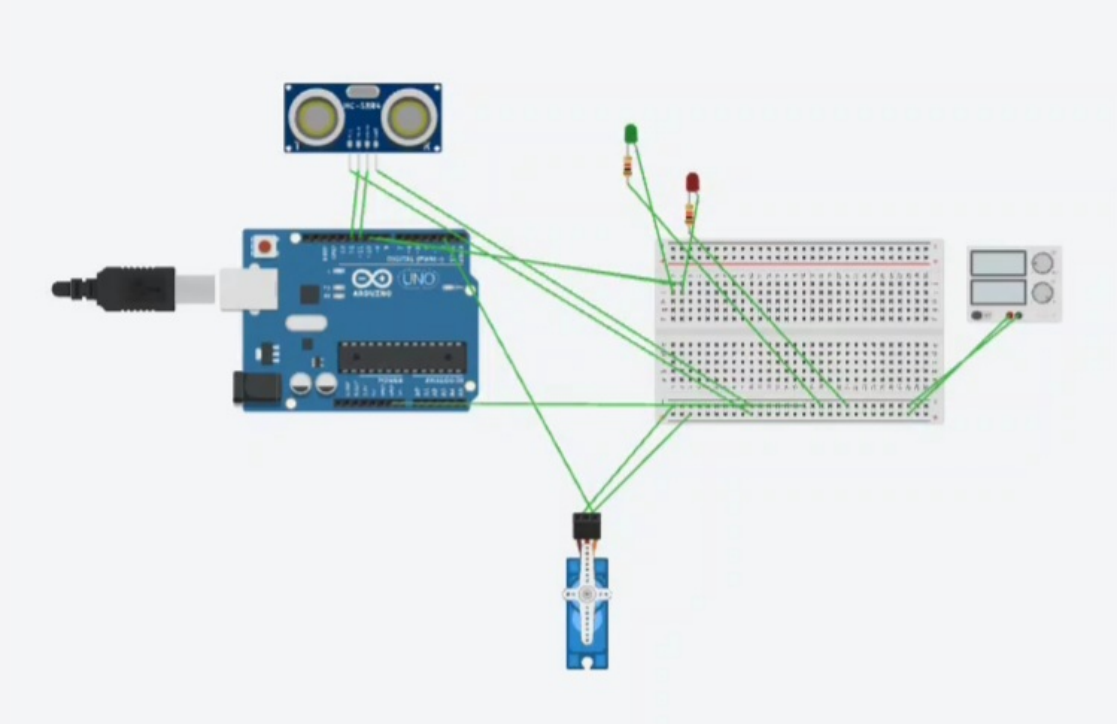
II Ultrasonic Distance-Based Servo Control with LED Indicators

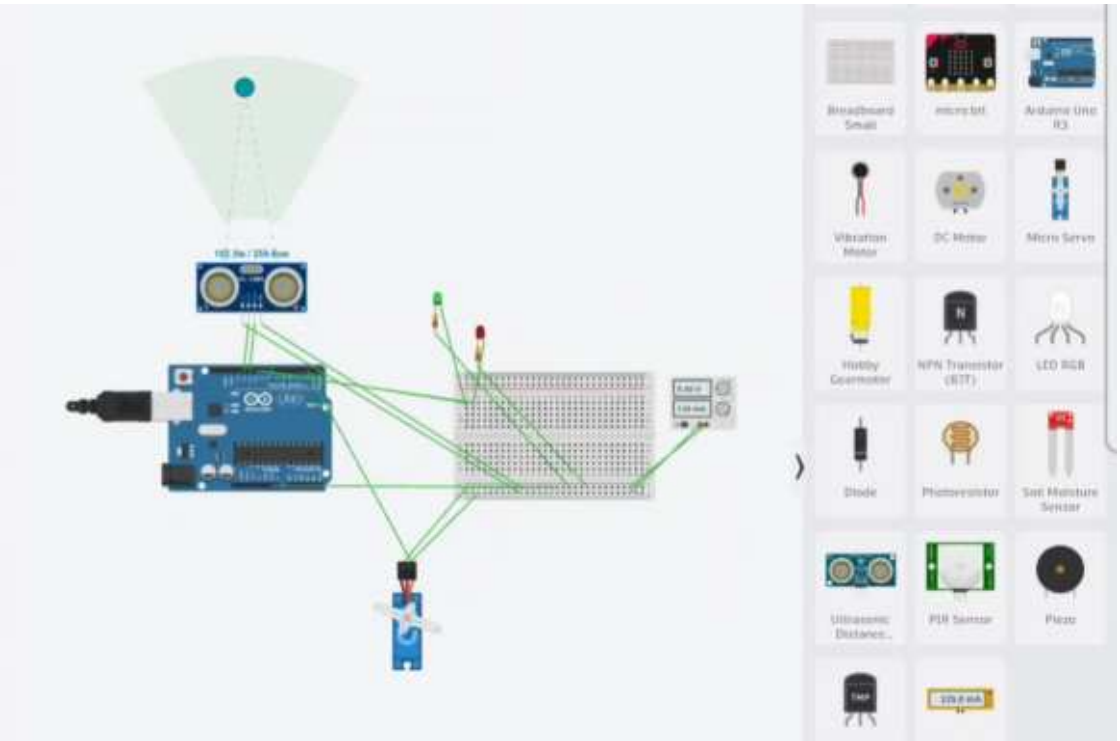
i Code

```
#include <Servo.h>

const int trigPin = 12; // Trig Pin
const int echoPin = 11; // Echo Pin
const int greenLedPin = 9; // Green LED pin
const int redLedPin = 10; // Red LED pin
long duration, cm; // Duration and distance in cm

Servo servo_2;

enum Direction { CLOCKWISE, COUNTERCLOCKWISE, NONE }; // Enum for servo direction
Direction lastDirection = NONE; // Variable to store the last direction

void setup() {
```

```
  Serial.begin(9600); // Set baud rate
  pinMode(trigPin, OUTPUT); // Set trigPin as OUTPUT
  pinMode(echoPin, INPUT); // Set echoPin as INPUT
  pinMode(greenLedPin, OUTPUT); // Set green LED pin as OUTPUT
  pinMode(redLedPin, OUTPUT); // Set red LED pin as OUTPUT
  servo_2.attach(2, 500, 2500); // Attach the servo to pin 2
}

void loop() {
  // Measure distance
  digitalWrite(trigPin, LOW);
  delayMicroseconds(5);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH); // Get the duration of the echo
  cm = (duration / 2) / 29.1; // Convert duration to distance in cm

  Serial.print("Distance: ");
  Serial.print(cm);
  Serial.println(" cm");

  // Control the servo and LEDs based on the distance
  if (cm > 160) {
    if (lastDirection != CLOCKWISE) {
      servo_2.write(servo_2.read() + 180); // Move clockwise
      lastDirection = CLOCKWISE;
    }
    digitalWrite(greenLedPin, HIGH); // Turn on green LED
    digitalWrite(redLedPin, LOW); // Turn off red LED
    Serial.println("Moving clockwise, green LED on");
  } else if (cm < 150) {
    if (lastDirection != COUNTERCLOCKWISE) {
      servo_2.write(servo_2.read() - 180); // Move counterclockwise
      lastDirection = COUNTERCLOCKWISE;
    }
    digitalWrite(greenLedPin, LOW); // Turn off green LED
    digitalWrite(redLedPin, HIGH); // Turn on red LED
    Serial.println("Moving counterclockwise, red LED on");
  }

  delay(100); // Wait before the next measurement
}
```

ii Screenshot



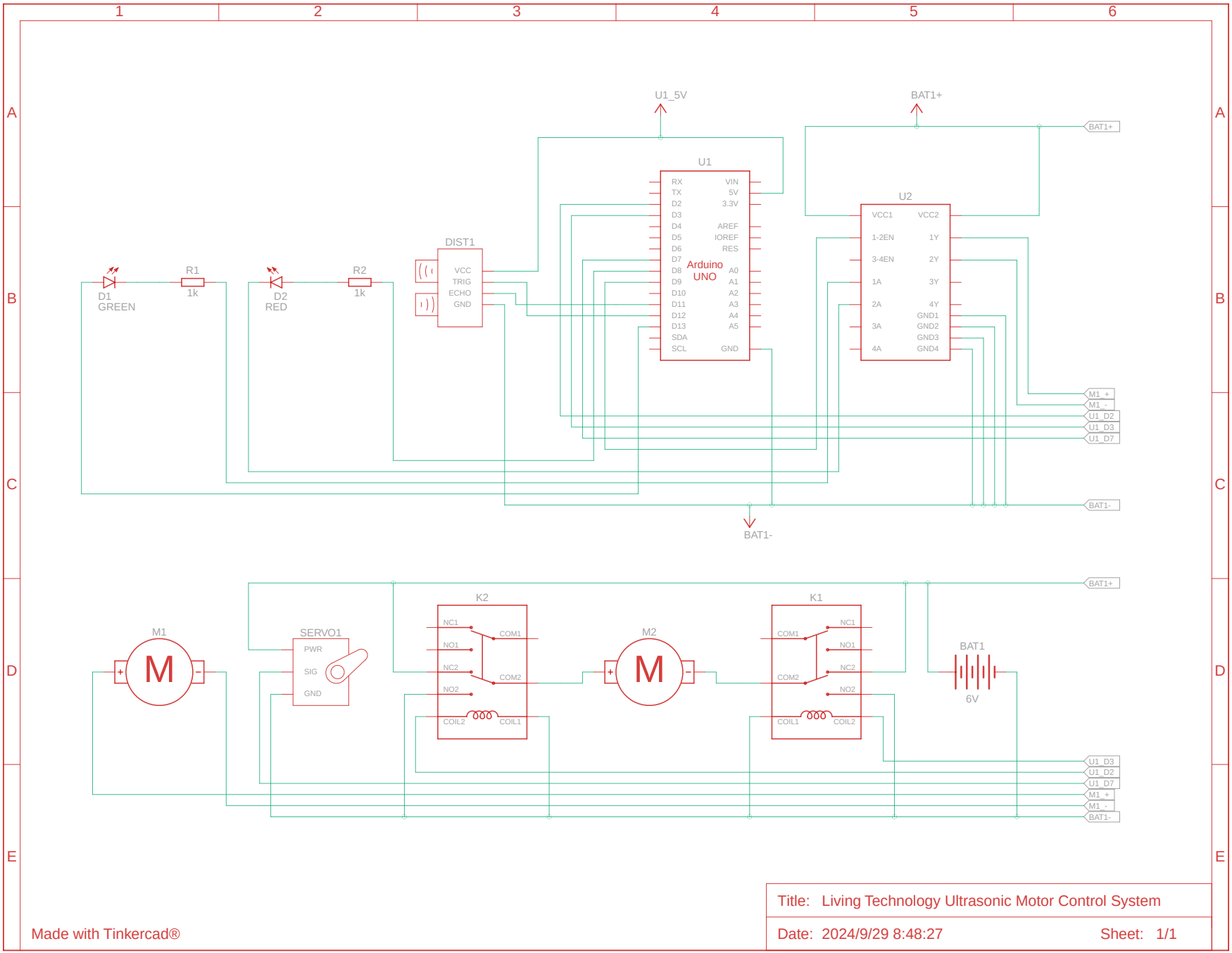


III Ultrasonic Distance-Based Servo and Relay Controlled Motor Control with LED Indicators

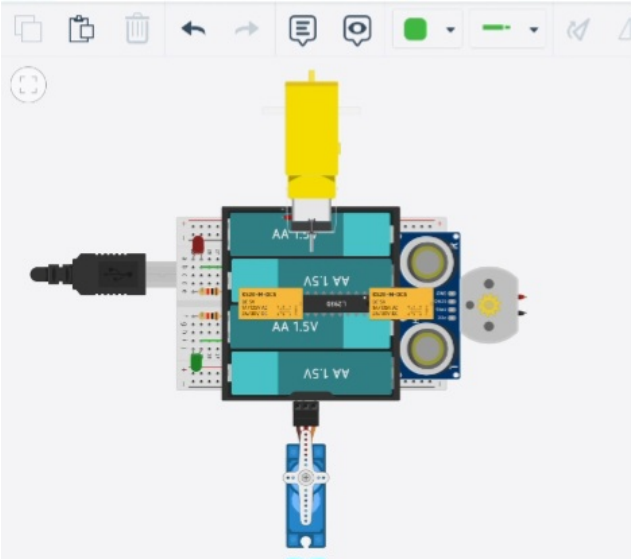
i Code

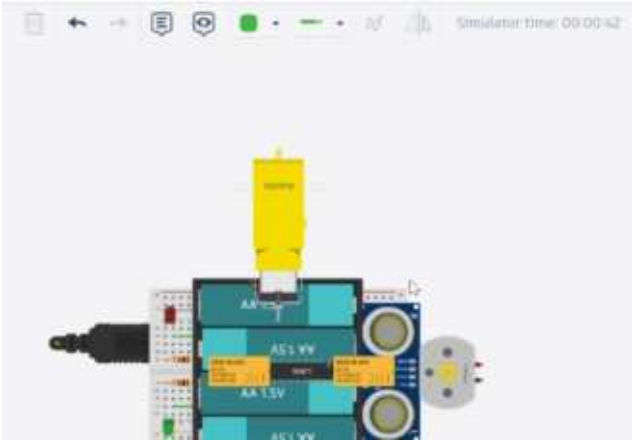
```
#include <Servo.h>

const int trigPin = 12; // Trig Pin
const int echoPin = 11; // Echo Pin
const int greenLedPin = 9; // Green LED pin
const int redLedPin = 10; // Red LED pin
long duration, cm; // Duration and distance in cm

Servo servo_2;

enum Direction { CLOCKWISE, COUNTERCLOCKWISE, NONE }; // Enum for servo direction
Direction lastDirection = NONE; // Variable to store the last direction

void setup() {
  Serial.begin(9600); // Set baud rate
  pinMode(trigPin, OUTPUT); // Set trigPin as OUTPUT
  pinMode(echoPin, INPUT); // Set echoPin as INPUT
  pinMode(greenLedPin, OUTPUT); // Set green LED pin as OUTPUT
  pinMode(redLedPin, OUTPUT); // Set red LED pin as OUTPUT
  servo_2.attach(2, 500, 2500); // Attach the servo to pin 2
}

void loop() {
  // Measure distance
  digitalWrite(trigPin, LOW);
  delayMicroseconds(5);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH); // Get the duration of the echo
  cm = (duration / 2) / 29.1; // Convert duration to distance in cm

  Serial.print("Distance: ");
  Serial.print(cm);
  Serial.println(" cm");

  // Control the servo and LEDs based on the distance
  if (cm > 160) {
    if (lastDirection != CLOCKWISE) {
      servo_2.write(servo_2.read() + 180); // Move clockwise
      lastDirection = CLOCKWISE;
    }
    digitalWrite(greenLedPin, HIGH); // Turn on green LED
    digitalWrite(redLedPin, LOW); // Turn off red LED
    Serial.println("Moving clockwise, green LED on");
  } else if (cm < 150) {
    if (lastDirection != COUNTERCLOCKWISE) {
      servo_2.write(servo_2.read() - 180); // Move counterclockwise
      lastDirection = COUNTERCLOCKWISE;
    }
    digitalWrite(greenLedPin, LOW); // Turn off green LED
    digitalWrite(redLedPin, HIGH); // Turn on red LED
    Serial.println("Moving counterclockwise, red LED on");
  }

  delay(100); // Wait before the next measurement
}
```

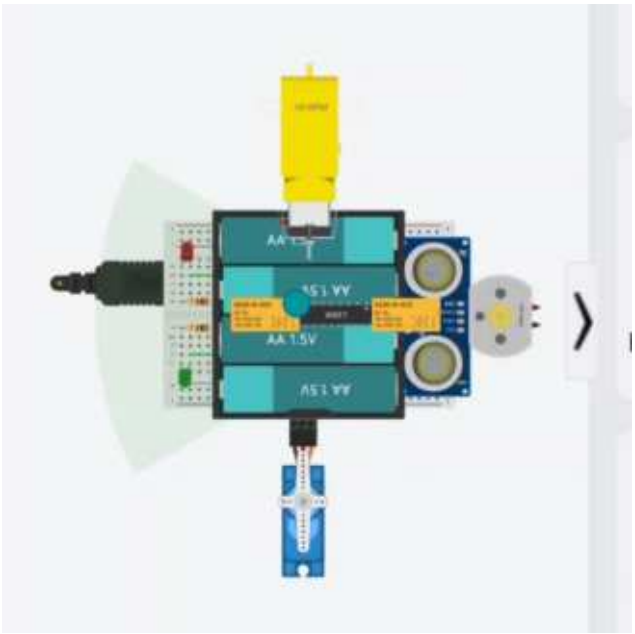
ii Sheet


1
2
3
4
5
6
A
B
C
D
E
U1_5V
BAT1+
U1
RX
TX
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D12
D13
SDA
SCL
VIN
5V
3.3V
AREF
IOREF
RES
A0
A1
A2
A3
A4
A5
GND
Arduino
UNO
U2
VCC1
1-2EN
3-4EN
1A
2A
3A
4A
VCC2
1Y
2Y
3Y
4Y
GND1
GND2
GND3
GND4
DIST1
VCC
TRIG
ECHO
GND
R1
1k
R2
1k
D1
GREEN
D2
RED
M1_+
M1_-
U1_D2
U1_D3
U1_D7
BAT1-
M1
M
SERVO1
PWR
SIG
GND
K2
K1
NC1
NO1
NC2
NO2
COM1
COM2
COIL1
COIL2
M2
BAT1
6V
Made with Tinkercad®
Title: Living Technology Ultrasonic Motor Control System
Date: 2024/9/29 8:48:27
Sheet: 1/1

iii Screenshot







3 Division of Labor

To maximize efficiency and leverage each member's expertise, we divided responsibilities as follows:

- 王品翔: An expert in 3D modeling, wiring, structural engineering, and the design of microprocessor-driven bots and karts. With extensive experience in CAD modeling, 3D printing, laser cutting, and microprocessor software development, as well as a track record of winning awards in microprocessor-based bot competitions, he leads the 3D modeling, hardware design, and component production aspects of the project.

- 沈威宇: Specializing in programming and markup languages such as C, C++, and LaTeX, with a strong foundation in advanced mathematics and physics. His portfolio includes developing machine learning-powered financial analysis tools, algorithmic trading systems, Linux virtual machines on Android, microprocessor-driven bots, and software projects in C++, Python, Node.js, and Java. He is responsible for the PID control system, Arduino programming, and the creation of the project report.
- 廖宏璿: Skilled in carpentry, wiring, structural engineering, and programmatic analysis in social sciences. He has contributed to projects like active noise-canceling headphones, microprocessor-driven bots, and social data algorithmic analysis. With expertise in designing and assembling physical structures, wiring microprocessor boards, and breadboards for active motion systems, he focuses on wiring design, wood structure assembly, and parameter tuning.

While roles are defined, teamwork remains at the heart of our approach, with all members actively supporting one another to achieve project goals.

4 PID Controller, Software Simulation, Parameters Optimization, and Arduino Code

I implemented a PID (Proportional-Integral-Derivative) controller designed to regulate the speed of the auto-following kart with a leader robot. The implementation includes the PID controller logic, a car simulation, a framework for testing various PID configurations, and the Arduino applications.

I PID.hpp

This header file implements a PID controller that includes traditional PID control logic along with enhancements such as adaptive gain adjustment, adaptive integral term, error history management, and additional derivative term. It is also designed to be compatible with both Arduino and non-Arduino environments.

i Code

```
#ifndef PID_HPP
#define PID_HPP

#ifdef ARDUINO
#include <Arduino.h>
#include <math.h>
#define to_string(a) String(a)
#else
#include <cmath>
#include <string>
#define String string
using namespace std;
#endif
#define MAX(a, b) (((a) > (b)) ? (a) : (b))
#define MIN(a, b) (((a) < (b)) ? (a) : (b))
#define ABS(a) ((a) < 0 ? -(a) : (a))
#define COPYSIGN(a, b) ((b) < 0 ? -ABS(a) : ABS(a))
#define CLAMP(a, b, c) (MAX(MIN((a), (b)), (c)))
```

```
19
20 struct Data {
21     unsigned long dt;
22     double e;
23 };
24
25 struct Node {
26     Data data;
27     Node* prev;
28 };
29
30 struct List {
31     Node* head;
32     Node* tail;
33     unsigned int size;
34 };
35
36 class PID {
37 public:
38     const double maxIntTm, maxAmp, minKp, maxKp, rTiM, TdM, TddM, eDPm, eDPa;
39     const unsigned long session;
40     double Kp, Ki, Kd, Kdd, preE, preDv, intg, preOut;
41     unsigned long preT;
42     List dtXs;
43
44     PID(double maxIntTm, double maxAmp, double minKp, double maxKp, double rTiM, double TdM, double TddM,
        double eDPm, double eDPa, unsigned long session, double Kp, double preE=0, unsigned long preT=0)
45         : maxIntTm(maxIntTm), maxAmp(maxAmp), minKp(minKp), maxKp(maxKp), rTiM(rTiM), TdM(TdM), TddM(TddM),
            eDPm(eDPm), eDPa(eDPa), session(session), Kp(Kp), Ki(0), Kd(0), Kdd(0), preE(preE), preDv(0),
            intg(0), preOut(0), preT(preT), dtXs({nullptr, nullptr, 0}) {}
46
47     double update(double e, unsigned long timestamp, String* debug = nullptr) {
48         unsigned long dt = timestamp - preT;
49         preT = timestamp;
50
51         if (dt == 0) {
52             if (debug != nullptr) {
53                 *debug += "dt=0\n";
54             }
55             return preOut;
56         }
57
58         double amp, ap;
59
60         if (session>0) {
61             unsigned long S = 0;
62             amp = 0;
63             if (dtXs.size > 1) {
64                 Node* pos = dtXs.tail;
65                 while (dtXs.size > 1 && pos != nullptr) {
66                     S += pos->data.dt;
67                     amp = MAX(ABS(pos->data.e), amp);
68                     if (S > session) {
69                         dtXs.head = pos;
70                         Node* cur = pos->prev;
71                         pos->prev = nullptr;
72                         while (cur != nullptr) {
73                             Node* tmp = cur->prev;
74                             delete cur;
75                             cur = tmp;
76                             dtXs.size--;
77                         }
78                         break;
79                     } else pos = pos->prev;
80                 }
```

```
            } else if (dtXs.size == 1) {
                amp = ABS(dtXs.tail->data.e);
            }
            ap = exp(-amp / maxAmp);
        } else {
            amp=0;
            ap=1;
        }

        Ki = Kp * rTiM * ap;
        Kd = Kp * TdM * ap;
        Kdd = Kd * TddM;

        double dv = (e - preE) / dt;
        double dd = (dv - preDv) / dt;

        intg += e * dt;
        double intTm = Ki * intg;
        if (ABS(intTm) > maxIntTm) {
            intg = COPYSIGN(maxIntTm / Ki, intTm);
            intTm = COPYSIGN(maxIntTm, intTm);
        }

        double propTm = Kp * e;
        double dvTm = Kd * dv;
        double ddTm = Kdd * dd;
        double out = propTm + intTm + dvTm + ddTm;

        double eDP = e * dv;
        if (ABS(eDP) > eDPm) {
            Kp *= (eDP < 0) ? (1+eDPa) : (1-eDPa);
            Kp = MIN(MAX(Kp, minKp), maxKp);
        }

        if (debug != nullptr) {
            *debug += "update() called\ndt: " + to_string(dt) + ", e: " + to_string(e) + ", dv: " + to_string
                (dv) + ", dd: " + to_string(dd) +", amp: " + to_string(amp) +", ap: " + to_string(ap) + ",\
                nKp: " + to_string(Kp) + ", Ki: " + to_string(Ki) + ", Kd: " + to_string(Kd) + ", Kdd: " +
                to_string(Kdd) + ",\npropTm: " + to_string(propTm) + ", intTm: " + to_string(intTm) + ", dvTm
                : " + to_string(dvTm) + ", ddTm: " + to_string(ddTm) + ",\neDP: " + to_string(eDP) + ", out:
                " + to_string(out) + "\n";
        }

        preE = e;
        preDv = dv;
        preOut = out;

        if (session>0) {
            Node* nn = new Node{{dt, e}, dtXs.tail};
            dtXs.tail = nn;
            if (dtXs.size == 0) dtXs.head = nn;
            dtXs.size++;
       }

        return out;
    }
};

#endif
```

ii Traditional PID Controller

The traditional PID controller is defined by the following equation:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)\,\mathrm{d}\tau + K_d \frac{\mathrm{d}e(t)}{\mathrm{d}t},$$

where:

- $u(t)$ is the control output.
- $e(t)$ is the error at time t, defined as the difference between the desired value and the measured value.
- K_p, K_i, and K_d are the proportional, integral, and derivative gains, respectively.

iii Adaptive Gain Adjustment

The proportional gain K_p is dynamically adjusted based on the interaction between the current error $e(t)$ and its rate of change (derivative). This adjustment is designed to improve the responsiveness of the controller to changing conditions. The adaptation can be mathematically expressed as follows:

$$K_p' = K_p \times (1 \pm eDP_a),$$

where:

- $eDP = e(t) \times \frac{de(t)}{dt}$
- eDP_m is a threshold that determines when to adjust the proportional gain.
- eDP_a is the adjustment factor that scales the change in K_p based on the error' s derivative.

The key idea behind this adjustment is to increase the proportional gain when the system is experiencing a large error combined with a significant rate of change, which could indicate an approaching overshoot or oscillation. Conversely, if the error is decreasing but remains large, it may reduce the proportional gain to prevent excessive correction, allowing the system to stabilize more smoothly.

iv Adaptive Integral Term

Integral control is essential for eliminating steady-state error; however, it can lead to issues like integral windup, where the integral term accumulates excessively during periods of sustained error. To mitigate this, the integral gain K_i is adaptively computed as follows:

$$K_i = K_p \times rTiM \times ap$$

Where:

- $ap = e^{-\frac{\text{amp}}{\text{maxAmp}}}$
- $rTiM$ is a constant that relates to the desired integral time constant.

The variable amp represents the maximum amplitude of error observed over a session. By using the exponential decay factor ap, the contribution of the integral term is diminished during periods of high error, effectively reducing the risk of windup. When the error is within an acceptable range, ap approaches 1, allowing the integral term to contribute effectively to the control output.

v Error History Management

Inspired by the **Ziegler–Nichols tuning method**, the implementation maintains a history of past errors and their timestamps to calculate the derivative and integral terms accurately. This history management is crucial for two main reasons:

- **Deriving the Rate of Change**: The derivative term requires knowledge of how fast the error is changing. By storing previous error values in a linked list, we can accurately compute the change in error over time ($\frac{de(t)}{dt}$):
$$dv = \frac{e(t) - preE}{dt}$$
- **Integral Calculation**: The integral of the error is accumulated over time, enabling the controller to respond to persistent deviations from the setpoint:
$$intg+ = e(t) \times dt$$

Using a linked list allows for efficient management of this error history. As new errors are added, older entries can be pruned to ensure that only relevant data is retained, which helps avoid excessive memory usage and keeps the calculations focused on the most recent behavior of the system.

vi Additional Derivative Term

In addition to the standard derivative term K_d, an additional derivative term K_{dd} is included in the PID controller. This term represents a second derivative component, which can be defined as follows:

$$K_{dd} = K_d \times TddM,$$

where $TddM$ is a multiplier that adjusts the impact of the second derivative term.
The inclusion of K_{dd} helps the controller anticipate changes in the error more effectively. By accounting for the acceleration of the error (i.e., how the rate of error change is itself changing), this additional term enhances the damping characteristics of the controller, allowing for more refined control responses, especially in systems that are subject to rapid changes or oscillations.

vii Data Structures

- **Data**: Holds the time difference (`dt`) and error value (`e`).
- **Node**: Represents an element in the linked list, containing `Data` and a pointer to the previous node.
- **List**: Manages the linked list, holding pointers to the head and tail nodes and the size of the list.

viii PID Class

The `PID` class encapsulates the PID control logic. It includes:

- **Member Variables:**

- Control gains: K_p, K_i, K_d, K_{dd} (additional derivative term).
- State variables: `preE`, `preDv`, `intg`, `preOut`, and `preT`, which store the previous error, derivative, integral term, last output, and last timestamp, respectively.
- `dtXs`: An instance of `List` to maintain historical error data.

- **Constructor:** Initializes the PID parameters and state variables.
- **Update Method:**
  - Takes the current error and timestamp, computes the control output, and updates the internal state.
  - Implements logic for calculating the adaptive gains, managing error history, and generating the control signal.

II testCar.hpp

This header file includes a `Car` class for simulating the motion of a car, a PID controller for adjusting the speed of an auto-following kart based on the distance to a leader car, and several utility functions for data processing and results management.

i Code

```cpp
#ifndef TESTCAR_HPP
#define TESTCAR_HPP

#include <iostream>
#include <cmath>
#include <string>
#include <vector>
#include <numeric>
#include <fstream>
#include <algorithm>
#include "PID.hpp"
using namespace std;

class Car
{
  public:
    Car(double initialSpeed, double initialPosition)
        : speed(initialSpeed), position(initialPosition) {}

    void updatePosition(double timeInterval)
    {
        position += speed * timeInterval;
    }

    void changeSpeed(double newSpeed)
    {
        speed = newSpeed;
    }

    double getPosition() const
    {
        return position;
    }

```

```
35      double getSpeed() const
36      {
37          return speed;
38      }
39
40      double speed;
41      double position;
42  };
43
44  void pidDebug(string message)
45  {
46      cout << message;
47  }
48
49  double sum_last_squared(const std::vector<double> &v, double prop)
50  {
51      if (v.empty())
52          return 0.0;
53
54      size_t n = v.size();
55      size_t count = static_cast<size_t>(std::ceil(n * prop));
56      size_t start_index = n - count;
57
58      double sum_of_squares = std::accumulate(v.begin() + start_index, v.end(), 0.0,
59                                              [](double sum, double value) {
60                                                  return sum + value * value;
61                                              });
62      return sum_of_squares;
63  }
64
65  vector<double> velocities(int steps)
66  {
67      vector<double> velocities(steps, 0.0);
68      double acceleration = 1;
69      double error = 0.2;
70
71      int phase_length = steps / 6;
72      double velocity = 0.0;
73
74      for (int i = 0; i < phase_length; ++i)
75      {
76          velocity += acceleration + ((i % 3 == 0) ? error : ((i % 3 == 1) ? (-error) : 0));
77          velocities[i] = velocity;
78      }
79
80      for (int i = phase_length; i < 2 * phase_length; ++i)
81      {
82          velocity += ((i % 3 == 0) ? error : ((i % 3 == 1) ? (-error) : 0));
83          velocities[i] = velocity;
84      }
85
86      for (int i = 2 * phase_length; i < 4 * phase_length; ++i)
87      {
88          velocity -= acceleration + ((i % 3 == 0) ? error : ((i % 3 == 1) ? (-error) : 0));
89          velocities[i] = velocity;
90      }
91
92      for (int i = 4 * phase_length; i < 5 * phase_length; ++i)
93      {
94          velocity += acceleration + ((i % 3 == 0) ? error : ((i % 3 == 1) ? (-error) : 0));
95          velocities[i] = velocity;
96      }
97
98      for (int i = 5 * phase_length; i < steps; ++i)
99      {
```

```
        if (ABS(velocity) >= (error * (steps - i)))
        {
            velocity += COPYSIGN(velocity, error);
        }
        else
        {
            velocity += ((i % 3 == 0) ? error : ((i % 3 == 1) ? (-error) : 0));
        }
        velocities[i] = velocity;
    }

    return velocities;
}

vector<double> test(double maxIntTm, double maxAmp, double minKp, double maxKp, double rTiM, double TdM,
    double TddM, double eDPm, double eDPa, unsigned long session, double Kp, double preE = 0, unsigned long
    preT = 0, double timeInterval = 10, int steps = 50, bool debug = false)
{
    double leaderInitialSpeed = 0;
    double followerInitialSpeed = 0;
    double initialDistance = 0;

    vector<double> vec;
    vector<double> vol = velocities(steps);

    Car leader(leaderInitialSpeed, initialDistance);
    Car follower(followerInitialSpeed, 0);

    PID pid = PID(maxIntTm, maxAmp, minKp, maxKp, rTiM, TdM, TddM, eDPm, eDPa, session, Kp, preE, preT);

    cout << fixed << setprecision(4);

    for (int i = 0; i < steps; i++)
    {
        leader.changeSpeed(leader.getSpeed() + vol[i]);
        leader.updatePosition(timeInterval);
        follower.updatePosition(timeInterval);

        double distanceToLeader = leader.getPosition() - follower.getPosition();
        vec.push_back(distanceToLeader);
        string* ptr = nullptr;
        if (debug) {
            cout << "TimeStep: " << i << ", distanceToLeader: " << distanceToLeader << ", FollowerSpeed: " <<
                follower.getSpeed() << ", LeaderSpeed: " << leader.getSpeed() << "\n";

            ptr = new string;
        }
        double fS = pid.update(distanceToLeader, i * timeInterval, ptr);
        follower.changeSpeed(follower.getSpeed() + MAX(-511, MIN(fS, 511)));
        if (debug) {
            cout << *ptr;
            delete ptr;
        }
    }

    return vec;
}

struct pidTest
{
    double maxIntTm, maxAmp, minKp, maxKp, rTiM, TdM, TddM, eDPm, eDPa;
    unsigned long session;
    double Kp, result;
};

```

```
bool write_results(const vector<pidTest> &data, const string &filename)
{
    auto sorted_data = data;
    sort(sorted_data.begin(), sorted_data.end(), [](const pidTest &a, const pidTest &b) {
        return a.result < b.result;
    });

    ofstream outfile(filename);
    if (!outfile)
    {
        cerr << "Error opening file for writing." << endl;
        return 0;
    }

    outfile << "maxIntTm,maxAmp,minKp,maxKp,rTiM,TdM,TddM,eDPm,eDPa,session,Kp,result\n";

    for (size_t i = 0; i < sorted_data.size(); ++i)
    {
        const pidTest &pt = sorted_data[i];
        outfile << pt.maxIntTm << "," << pt.maxAmp << "," << pt.minKp << "," << pt.maxKp << ","
                << pt.rTiM << "," << pt.TdM << "," << pt.TddM << "," << pt.eDPm << "," << pt.eDPa << ","
                << pt.session << "," << pt.Kp << "," << pt.result << "\n";
    }

    outfile.close();
    cout << "Data written to " << filename << endl;

    return 1;
}

struct ParameterRange
{
    double start;
    double end;
    double step;
};

#endif
```

ii Car Class

The `Car` class represents a simple vehicle model with basic motion dynamics. It includes attributes for speed and position, as well as methods to update these attributes.

iii PID Debugging Function

The `pidDebug` function is a utility for logging messages related to PID control operations.

iv Sum of Last Squared Values

```
double sum_last_squared(const std::vector<double> &v, double prop)
```

Calculates the sum of squares of the last $n \times$ prop elements of the provided vector `v`.

v Velocity Generation

```
vector<double> velocities(int steps)
```

Generates a vector of velocity values over a specified number of simulation steps, simulating different phases of motion (acceleration, deceleration, and adjustments).

vi PID Testing Function

```
vector<double> test(double maxIntTm, double maxAmp, double minKp, double maxKp, double rTiM, double TdM,
    double TddM, double eDPm, double eDPa, unsigned long session, double Kp, double preE = 0, unsigned long
    preT = 0, double timeInterval = 10, int steps = 1000)
```

This function simulates the interaction between a leader car and an auto-following kart using a PID controller to adjust the follower's speed based on the distance to the leader.

- **Parameters:**
  - `double maxIntTm`: Maximum integral time constant.
  - `double maxAmp`: Maximum amplitude of control output.
  - `double minKp`: Minimum proportional gain.
  - `double maxKp`: Maximum proportional gain.
  - `double rTiM`: Reference time interval multiplier.
  - `double TdM`: Derivative time multiplier.
  - `double TddM`: Second derivative time multiplier.
  - `double eDPm`: Maximum error derivative proportionality factor.
  - `double eDPa`: Error derivative adjustment factor.
  - `unsigned long session`: Session identifier for logging purposes.
  - `double Kp`: Initial proportional gain.
  - `double preE`: Previous error value (optional, defaults to 0).
  - `unsigned long preT`: Previous time value (optional, defaults to 0).
  - `double timeInterval`: Time interval for each step of simulation (default is 10).
  - `int steps`: Total number of simulation steps (default is 1000).
- **Returns:** A vector containing the distances between the leader and follower over time.

vii PID Test Data Structure

```
struct pidTest
{
double maxIntTm, maxAmp, minKp, maxKp, rTiM, TdM, TddM, eDPm, eDPa;
unsigned long session;
double Kp, result;
};
```

The `pidTest` structure is used to store parameters for PID testing, along with the result of the simulation.

viii Result Writing Function

```
bool write_results(const vector<pidTest> &data, const string &filename)
```

Writes the results of PID tests to a CSV file for analysis.

ix ParameterRange

A structure to define the range of parameters for optimization.

- **Attributes:**
  - `double start`: The starting value of the parameter range.
  - `double end`: The ending value of the parameter range.
  - `double step`: The increment step for the parameter.

III optimization.hpp

This header file defines the optimization functions and structures used to optimize the parameters of a PID controller in a car simulation without multi-threaded processing.

i Code

```
#ifndef OPTIMIZATION_HPP
#define OPTIMIZATION_HPP

#include "testCar.hpp"
using namespace std;

int optimize(vector<ParameterRange> ranges)
{
    vector<pidTest> results;

    for (double maxIntTm = ranges[0].start; maxIntTm <= ranges[0].end; maxIntTm += ranges[0].step)
    {
        for (double maxAmp = ranges[1].start; maxAmp <= ranges[1].end; maxAmp += ranges[1].step)
        {
            for (double minKp = ranges[2].start; minKp <= ranges[2].end; minKp += ranges[2].step)
            {
                for (double maxKp = max(minKp, ranges[3].start);
                     maxKp <= ranges[3].end; maxKp += ranges[3].step)
                {
                    for (double rTiM = ranges[4].start; rTiM <= ranges[4].end; rTiM += ranges[4].step)
                    {
                        for (double TdM = ranges[5].start; TdM <= ranges[5].end; TdM += ranges[5].step)
                        {
                            for (double TddM = ranges[6].start; TddM <= ranges[6].end; TddM += ranges[6].step
                                )
                            {
                                for (double eDPm = ranges[7].start; eDPm <= ranges[7].end; eDPm += ranges[7].
                                    step)
                                {
                                    for (double eDPa = ranges[8].start; eDPa <= ranges[8].end; eDPa += ranges
                                        [8].step)
                                    {
                                        for (unsigned long session = ranges[9].start; session <= ranges[9].
                                            end; session += ranges[9].step)
                                        {
```

```
                                        for (double Kp = MAX(minKp, ranges[10].start);
                                             Kp <= MIN(maxKp, ranges[10].end); Kp += ranges[10].step)
                                        {
                                            vector<double> tmp = test(maxIntTm, maxAmp, minKp, maxKp,
                                                rTiM, TdM, TddM, eDPm, eDPa, session, Kp);
                                            double result = sum_last_squared(tmp, 0.5);
                                            results.push_back(pidTest({maxIntTm, maxAmp, minKp, maxKp,
                                                rTiM, TdM, TddM, eDPm, eDPa, session, Kp, result}));
                                        }
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }

    if (!write_results(results, "results.csv"))
    {
        cerr << "Error writing results to file." << endl;
        return EXIT_FAILURE;
    }

    return EXIT_SUCCESS;
}

#endif
```

ii Optimize Function

Optimizes the parameters of the PID controller based on the provided ranges and writes results to a CSV file.

- **Parameters:**
  - `vector<ParameterRange> ranges`: A vector containing the ranges of parameters to optimize.
- **Returns:**
  - `int`: Returns `EXIT_SUCCESS` (0) if optimization completes successfully, otherwise returns `EXIT_FAILURE`.

IV optimization_multithread.hpp

This header file defines the optimization functions and structures used to optimize the parameters of a PID controller in a car simulation with multi-threaded processing.

```
#ifndef OPTIMIZATION_MULTITHREAD_HPP
#define OPTIMIZATION_MULTITHREAD_HPP

#include <vector>
#include <iostream>
#include <thread>
#include <mutex>
#include <queue>
```

```cpp
#include <atomic>
#include <chrono>
#include <iomanip>
#include <condition_variable>
#include <cmath>
#include "testCar.hpp"
using namespace std;

mutex resultsMutex;
mutex queueMutex;
atomic<int> totalCombinations(0);
atomic<int> processedCombinations(0);
queue<vector<double>> parameterQueue;
condition_variable cv;

void displayProgress()
{
    while (true)
    {
        {
            lock_guard<mutex> lock(queueMutex);
            if (processedCombinations >= totalCombinations)
            {
                break;
            }
            cout << "\rProgress: " << processedCombinations << "/" << totalCombinations
                 << " (" << fixed << setprecision(2)
                 << (static_cast<double>(processedCombinations) / totalCombinations * 100)
                 << "%)           " << flush;
        }
        this_thread::sleep_for(chrono::seconds(1));
    }
    cout << "\rProgress: 100% Complete!                    " << endl;
}

size_t calculateSteps(const ParameterRange &range)
{
    return static_cast<size_t>(ceil((range.end - range.start) / range.step)) + 1;
}

void workerFunction(vector<pidTest> &results, atomic<bool> &running)
{
    while (running)
    {
        vector<double> params;

        {
            unique_lock<mutex> lock(queueMutex);
            cv.wait(lock, [] { return !parameterQueue.empty(); });
            if (parameterQueue.empty())
            {
                continue;
            }
            params = parameterQueue.front();
            parameterQueue.pop();
        }

        vector<double> tmp = test(
            params[0], params[1], params[2], params[3],
            params[4], params[5], params[6], params[7],
            params[8], params[9], params[10]);

        double result = sum_last_squared(tmp, 0.5);

        {
```

```
            lock_guard<mutex> lock(resultsMutex);
            results.emplace_back(pidTest{
                params[0], params[1], params[2], params[3],
                params[4], params[5], params[6], params[7],
                params[8], static_cast<unsigned long>(params[9]),
                params[10], result});
        }

        processedCombinations++;
        cv.notify_one();
    }
}

int optimize(vector<ParameterRange> ranges)
{
    totalCombinations = 1;
    for (const auto &range : ranges)
    {
        size_t steps = calculateSteps(range);
        if (totalCombinations > 0 && steps > SIZE_MAX / totalCombinations)
        {
            cerr << "Error: Too many combinations, would cause overflow" << endl;
            return 1;
        }
        totalCombinations = totalCombinations * steps;
    }

    cout << "Total parameter combinations to test: " << totalCombinations << endl;

    for (double maxIntTm = ranges[0].start; maxIntTm <= ranges[0].end; maxIntTm += ranges[0].step)
    {
        for (double maxAmp = ranges[1].start; maxAmp <= ranges[1].end; maxAmp += ranges[1].step)
        {
            for (double minKp = ranges[2].start; minKp <= ranges[2].end; minKp += ranges[2].step)
            {
                for (double maxKp = max(minKp, ranges[3].start);
                     maxKp <= ranges[3].end; maxKp += ranges[3].step)
                {
                    for (double rTiM = ranges[4].start; rTiM <= ranges[4].end; rTiM += ranges[4].step)
                    {
                        for (double TdM = ranges[5].start; TdM <= ranges[5].end; TdM += ranges[5].step)
                        {
                            for (double TddM = ranges[6].start; TddM <= ranges[6].end; TddM += ranges[6].step
                                )
                            {
                                for (double eDPm = ranges[7].start; eDPm <= ranges[7].end; eDPm += ranges[7].
                                    step)
                                {
                                    for (double eDPa = ranges[8].start; eDPa <= ranges[8].end; eDPa += ranges
                                        [8].step)
                                    {
                                        for (unsigned long session = ranges[9].start; session <= ranges[9].
                                            end; session += ranges[9].step)
                                        {
                                            for (double Kp = MAX(minKp, ranges[10].start);
                                                 Kp <= MIN(maxKp, ranges[10].end); Kp += ranges[10].step)
                                            {
                                                parameterQueue.push(pidTest({maxIntTm, maxAmp, minKp, maxKp,
                                                    rTiM, TdM, TddM, eDPm, eDPa, session, Kp}));
                                            }
                                        }
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }
    }
```

```
                    }
                }
            }
        }
    }

    vector<pidTest> results;
    vector<thread> threads;
    atomic<bool> running(true);

    thread progressThread(displayProgress);
    unsigned int numThreads = thread::hardware_concurrency();
    cout << "Using " << numThreads << " threads" << endl;

    for (unsigned int i = 0; i < numThreads; ++i)
    {
        threads.emplace_back(workerFunction, ref(results), ref(running));
    }

    for (auto &thread : threads)
    {
        thread.join();
    }

    running = false;
    cv.notify_all();

    progressThread.join();

    if (!write_results(results, "results.csv"))
    {
        cerr << "Error writing results to file." << endl;
        return EXIT_FAILURE;
    }

    return EXIT_SUCCESS;
}

#endif
```

i Optimize Function

Optimizes the parameters of the PID controller based on the provided ranges and writes results to a CSV file.

- **Parameters:**
  - `vector<ParameterRange> ranges`: A vector containing the ranges of parameters to optimize.
- **Returns:**
  - `int`: Returns `EXIT_SUCCESS` (0) if optimization completes successfully, otherwise returns `EXIT_FAILURE`.

V main.cpp

The code file demonstrates the usage of the `optimization.hpp` (or `optimization_multithread.hpp`) header file to perform parameter optimization for a PID controller. It

defines a set of parameter ranges and calls the `optimize` function to evaluate different combinations of parameters.

i Code

```
#include "optimization.hpp" // or #include "optimization_multithread.hpp"

int main() {
    // Note that steps can't be zero
    vector<ParameterRange> ranges = {
        {461.9, 461.9, 1},     // maxIntTm
        {517.9, 517.9, 1},      // maxAmp
        {0.00265, 0.00265, 1},        // minKp
        {0.472, 0.472, 1},       // maxKp
        {1, 1, 1},        // rTiM
        {65.37, 65.37, 1},        // TdM
        {1.7, 1.7, 1},         // TddM
        {125, 125, 1}, // eDPm
        {0.15, 0.15, 1},     // eDPa
        {140, 140, 10},        // session
        {0.2436, 0.2436, 1}       // Kp
    };

    return optimize(ranges);
}
```

ii Notes

Note that steps can't be zero or the program won't stop.

VI main.ino

This Arduino code is for controlling a differential-drive robot using two ultrasonic sensors to measure distance, a PID (Proportional-Integral-Derivative) controller for speed and direction adjustments, and an L293D motor driver to control the motors based on sensor inputs.

i Code

```
// Arduino Pins
#define L293D_LEFT_EN 2
#define L293D_RIGHT_EN 3
#define L293D_LEFT_IN1 5
#define L293D_LEFT_IN2 6
#define L293D_RIGHT_IN1 7
#define L293D_RIGHT_IN2 8
#define TRIG_LEFT 4
#define TRIG_RIGHT 9
#define ECHO_LEFT 10
#define ECHO_RIGHT 11
// Constants
#define HALF_SOUND_SPEED 0.1715
#define TARGET_DISTANCE 15
#define DISTANCE_BETWEEN_ULTRASONIC_SENSORS 12
// Parameters to be tested
#define ANGULAR_SPEED_MULTIPLIER 0.8
#define LEFT_POSITIVE_SPEED_MULTIPLIER 1
#define LEFT_NEGATIVE_SPEED_MULTIPLIER 1.2
#define RIGHT_POSITIVE_SPEED_MULTIPLIER 1
```

```
#define RIGHT_NEGATIVE_SPEED_MULTIPLIER 1.2
// 1 for debug, 0 for not
#define DEBUG 1
// Adjust the above parameters
// The below code does not need to be changed

#include "PID.hpp"

PID avgPID(461.9, 517.9, 0.00265, 0.472, 1, 65.37, 1.7, 125, 0.15, 140, 0.2436);
PID angPID(461.9, 517.9, 0.00265, 0.472, 1, 65.37, 1.7, 125, 0.15, 140, 0.2436);
double avgV;
double angV;
#if DEBUG
    String* ptr = new String();
#endif

inline double leftIn() {
    digitalWrite(TRIG_LEFT, LOW);
    delayMicroseconds(2);
    digitalWrite(TRIG_LEFT, HIGH);
    delayMicroseconds(10);
    digitalWrite(TRIG_LEFT, LOW);
    return pulseIn(ECHO_LEFT, HIGH) * HALF_SOUND_SPEED;
}

inline double rightIn() {
    digitalWrite(TRIG_RIGHT, LOW);
    delayMicroseconds(2);
    digitalWrite(TRIG_RIGHT, HIGH);
    delayMicroseconds(10);
    digitalWrite(TRIG_RIGHT, LOW);
    return pulseIn(ECHO_RIGHT, HIGH) * HALF_SOUND_SPEED;
}

void setup()
{
    Serial.begin (9600);
    pinMode(TRIG_LEFT, OUTPUT);
    pinMode(TRIG_RIGHT, OUTPUT);
    pinMode(ECHO_LEFT, INPUT);
    pinMode(ECHO_RIGHT, INPUT);
    pinMode(L293D_LEFT_IN1, OUTPUT);
    pinMode(L293D_LEFT_IN2, OUTPUT);
    pinMode(L293D_RIGHT_IN1, OUTPUT);
    pinMode(L293D_RIGHT_IN2, OUTPUT);
    avgV = 0;
    angV = 0;
}

void loop() {
    double left = leftIn();
    double right = rightIn();
    double avg = left + right;
    double ang = atan2(left - right, DISTANCE_BETWEEN_ULTRASONIC_SENSORS);

    #if DEBUG
        *ptr = "";
        avgV += avgPID.update((avg - TARGET_DISTANCE), millis(), ptr);
        Serial.println("Average Speed PID Update:");
        Serial.println(*ptr);
        *ptr = "";
        angV += angPID.update((ang - TARGET_DISTANCE), millis(), ptr) * ANGULAR_SPEED_MULTIPLIER;
        Serial.println("Angular Speed Update:");
        Serial.println(*ptr);
        *ptr = "";
```

```
    #else
        avgV += avgPID.update((avg - TARGET_DISTANCE), millis());
        angV += angPID.update((ang - TARGET_DISTANCE), millis()) * ANGULAR_SPEED_MULTIPLIER;
    #endif

    double leftV = avgV + angV;
    double rightV = avgV - angV;

    leftOut(leftV);
    rightOut(rightV);

    delayMicroseconds(10000);
}

inline void leftOut(double leftV) {
    #if DEBUG
        Serial.println("Left Original Velocity: " + String(leftV));
    #endif
    leftV *= (leftV < 0)?LEFT_NEGATIVE_SPEED_MULTIPLIER:LEFT_POSITIVE_SPEED_MULTIPLIER;
    leftV = CLAMP(leftV, 255, -255);
    #if DEBUG
        Serial.println("Left Outputted Velocity: " + String(leftV));
    #endif
    if (leftV > 0) {
        analogWrite(L293D_LEFT_IN1, 0);
        analogWrite(L293D_LEFT_IN2, leftV);
    } else {
        analogWrite(L293D_LEFT_IN1, -leftV);
        analogWrite(L293D_LEFT_IN2, 0);
    }
}

inline void rightOut(double rightV) {
    #if DEBUG
        Serial.println("Right Original Velocity: " + String(rightV));
    #endif
    rightV *= (rightV < 0)?RIGHT_NEGATIVE_SPEED_MULTIPLIER:RIGHT_POSITIVE_SPEED_MULTIPLIER;
    rightV = CLAMP(rightV, 255, -255);
    #if DEBUG
        Serial.println("Right Outputted Velocity: " + String(rightV));
    #endif
    if (rightV > 0) {
        analogWrite(L293D_RIGHT_IN1, 0);
        analogWrite(L293D_RIGHT_IN2, rightV);
    } else {
        analogWrite(L293D_RIGHT_IN1, -rightV);
        analogWrite(L293D_RIGHT_IN2, 0);
    }
}
```

ii Pin Definitions

- **L293D motor control pins:**
  - `L293D_LEFT_EN` and `L293D_RIGHT_EN`: Enable pins for the left and right motors.
  - `L293D_LEFT_IN1`, `L293D_LEFT_IN2`, `L293D_RIGHT_IN1`, `L293D_RIGHT_IN2`: Input pins for controlling the direction of the motors.
- **Ultrasonic sensor pins:**
  - `TRIG_LEFT` and `TRIG_RIGHT`: Trigger pins for the left and right ultrasonic sensors.

- ECHO_LEFT and ECHO_RIGHT: Echo pins for the left and right ultrasonic sensors that return the duration of the reflected sound wave.

iii Constants

- **HALF_SOUND_SPEED**: Constant used to convert the echo time into a distance. The value 0.1715 is derived from the speed of sound.
- **TARGET_DISTANCE**: The target distance that the robot should maintain between the two sensors (in centimeters).
- **DISTANCE_BETWEEN_ULTRASONIC_SENSORS**: The distance between the two ultrasonic sensors mounted on the robot, which is used to calculate angular adjustments.

iv PID Tuning Parameters

- **PID Controllers (`avgPID` and `angPID`)**: Two PID controllers are used:
  - avgPID: Controls the average speed to keep the robot at the target distance.
  - angPID: Controls the angular speed to maintain proper alignment based on the sensor readings.

  The values provided for the PID controllers (like 461.9, 517.9, etc.) are the tuning parameters for each PID term (P, I, D).

v Functions and Code Flow

- **`leftIn()` and `rightIn()`**: These inline functions trigger the ultrasonic sensors by sending a pulse and measuring the time it takes for the sound to return. The return time is converted into a distance using the formula:

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

  The resulting distance is then returned.

- **`setup()`**: This function runs once when the program starts. It initializes serial communication, sets pin modes for the motor control and ultrasonic sensors, and initializes the velocity variables (avgV and angV).
- **`loop()`**: This function runs repeatedly. It does the following:
  1. Reads the distance from the left and right ultrasonic sensors using leftIn() and rightIn().
  2. Calculates the average distance (avg) and the angular difference (ang) between the two sensors.
  3. Applies the PID controllers to adjust the average speed and angular speed to maintain the target distance.

4. Sets the left and right velocities (`leftV` and `rightV`) based on the PID updates and applies any necessary adjustments to control the motors.
5. Sends the control signals to the motors using `leftOut()` and `rightOut()`.

vi Motor Control (`leftOut()` and `rightOut()`)

- `leftOut()` and `rightOut()`: These inline functions are responsible for sending control signals to the motor driver (L293D). The functions adjust the speed based on the calculated velocity and apply any speed multipliers to tune motor behavior. The `analogWrite()` function is used to set the motor speeds, where:
  - Positive velocities drive the motors forward.
  - Negative velocities drive the motors backward.

vii Debugging

If `DEBUG` is set to 1, additional information about the PID updates and motor velocities is printed to the serial monitor for troubleshooting and performance analysis.

VII all.ino

Combining `PID.hpp` and `main.ino` in one file.

i Code

```
// Arduino Pins
#define L293D_LEFT_EN 2
#define L293D_RIGHT_EN 3
#define L293D_LEFT_IN1 5
#define L293D_LEFT_IN2 6
#define L293D_RIGHT_IN1 7
#define L293D_RIGHT_IN2 8
#define TRIG_LEFT 4
#define TRIG_RIGHT 9
#define ECHO_LEFT 10
#define ECHO_RIGHT 11
// Constants
#define HALF_SOUND_SPEED 0.1715
#define TARGET_DISTANCE 15
#define DISTANCE_BETWEEN_ULTRASONIC_SENSORS 12
// Parameters to be tested
#define ANGULAR_SPEED_MULTIPLIER 0.8
#define LEFT_POSITIVE_SPEED_MULTIPLIER 1
#define LEFT_NEGATIVE_SPEED_MULTIPLIER 1.2
#define RIGHT_POSITIVE_SPEED_MULTIPLIER 1
#define RIGHT_NEGATIVE_SPEED_MULTIPLIER 1.2
// 1 for debug, 0 for not
#define DEBUG 1
// Adjust the above parameters
// The below code does not need to be changed

#ifdef ARDUINO
#include <Arduino.h>
#include <math.h>
#define to_string(a) String(a)
```

```
#else
#include <cmath>
#include <string>
#define String string
using namespace std;
#endif
#define MAX(a, b) (((a) > (b)) ? (a) : (b))
#define MIN(a, b) (((a) < (b)) ? (a) : (b))
#define ABS(a) ((a) < 0 ? -(a) : (a))
#define COPYSIGN(a, b) ((b) < 0 ? -ABS(a) : ABS(a))
#define CLAMP(a, b, c) (MAX(MIN((a), (b)), (c)))

struct Data {
    unsigned long dt;
    double e;
};

struct Node {
    Data data;
    Node* prev;
};

struct List {
    Node* head;
    Node* tail;
    unsigned int size;
};

class PID {
public:
    const double maxIntTm, maxAmp, minKp, maxKp, rTiM, TdM, TddM, eDPm, eDPa;
    const unsigned long session;
    double Kp, Ki, Kd, Kdd, preE, preDv, intg, preOut;
    unsigned long preT;
    List dtXs;

    PID(double maxIntTm, double maxAmp, double minKp, double maxKp, double rTiM, double TdM, double TddM,
        double eDPm, double eDPa, unsigned long session, double Kp, double preE=0, unsigned long preT=0)
        : maxIntTm(maxIntTm), maxAmp(maxAmp), minKp(minKp), maxKp(maxKp), rTiM(rTiM), TdM(TdM), TddM(TddM),
            eDPm(eDPm), eDPa(eDPa), session(session), Kp(Kp), Ki(0), Kd(0), Kdd(0), preE(preE), preDv(0),
            intg(0), preOut(0), preT(preT), dtXs({nullptr, nullptr, 0}) {}

    double update(double e, unsigned long timestamp, String* debug = nullptr) {
        unsigned long dt = timestamp - preT;
        preT = timestamp;

        if (dt == 0) {
            if (debug != nullptr) {
                *debug += "dt=0\n";
            }
            return preOut;
        }

        double amp, ap;

        if (session>0) {
            unsigned long S = 0;
            amp = 0;
            if (dtXs.size > 1) {
                Node* pos = dtXs.tail;
                while (dtXs.size > 1 && pos != nullptr) {
                    S += pos->data.dt;
                    amp = MAX(ABS(pos->data.e), amp);
                    if (S > session) {
                        dtXs.head = pos;
```

```
                    Node* cur = pos->prev;
                    pos->prev = nullptr;
                    while (cur != nullptr) {
                        Node* tmp = cur->prev;
                        delete cur;
                        cur = tmp;
                        dtXs.size--;
                    }
                    break;
                } else pos = pos->prev;
            }
        } else if (dtXs.size == 1) {
            amp = ABS(dtXs.tail->data.e);
        }
        ap = exp(-amp / maxAmp);
    } else {
        amp=0;
        ap=1;
    }

    Ki = Kp * rTiM * ap;
    Kd = Kp * TdM * ap;
    Kdd = Kd * TddM;

    double dv = (e - preE) / dt;
    double dd = (dv - preDv) / dt;

    intg += e * dt;
    double intTm = Ki * intg;
    if (ABS(intTm) > maxIntTm) {
        intg = COPYSIGN(maxIntTm / Ki, intTm);
        intTm = COPYSIGN(maxIntTm, intTm);
    }

    double propTm = Kp * e;
    double dvTm = Kd * dv;
    double ddTm = Kdd * dd;
    double out = propTm + intTm + dvTm + ddTm;

    double eDP = e * dv;
    if (ABS(eDP) > eDPm) {
        Kp *= (eDP < 0) ? (1+eDPa) : (1-eDPa);
        Kp = MIN(MAX(Kp, minKp), maxKp);
    }

    if (debug != nullptr) {
        *debug += "update() called\ndt: " + to_string(dt) + ", e: " + to_string(e) + ", dv: " + to_string
            (dv) + ", dd: " + to_string(dd) +", amp: " + to_string(amp) +", ap: " + to_string(ap) + ",\
            nKp: " + to_string(Kp) + ", Ki: " + to_string(Ki) + ", Kd: " + to_string(Kd) + ", Kdd: " +
            to_string(Kdd) + ",\npropTm: " + to_string(propTm) + ", intTm: " + to_string(intTm) + ", dvTm
            : " + to_string(dvTm) + ", ddTm: " + to_string(ddTm) + ",\neDP: " + to_string(eDP) + ", out:
            " + to_string(out) + "\n";
    }

    preE = e;
    preDv = dv;
    preOut = out;

    if (session>0) {
        Node* nn = new Node{{dt, e}, dtXs.tail};
        dtXs.tail = nn;
        if (dtXs.size == 0) dtXs.head = nn;
        dtXs.size++;
    }
```

```
        return out;
    }
};

PID avgPID(461.9, 517.9, 0.00265, 0.472, 1, 65.37, 1.7, 125, 0.15, 140, 0.2436);
PID angPID(461.9, 517.9, 0.00265, 0.472, 1, 65.37, 1.7, 125, 0.15, 140, 0.2436);
double avgV;
double angV;
#if DEBUG
    String* ptr = new String();
#endif

inline double leftIn() {
    digitalWrite(TRIG_LEFT, LOW);
    delayMicroseconds(2);
    digitalWrite(TRIG_LEFT, HIGH);
    delayMicroseconds(10);
    digitalWrite(TRIG_LEFT, LOW);
    return pulseIn(ECHO_LEFT, HIGH) * HALF_SOUND_SPEED;
}

inline double rightIn() {
    digitalWrite(TRIG_RIGHT, LOW);
    delayMicroseconds(2);
    digitalWrite(TRIG_RIGHT, HIGH);
    delayMicroseconds(10);
    digitalWrite(TRIG_RIGHT, LOW);
    return pulseIn(ECHO_RIGHT, HIGH) * HALF_SOUND_SPEED;
}

void setup()
{
    Serial.begin (9600);
    pinMode(TRIG_LEFT, OUTPUT);
    pinMode(TRIG_RIGHT, OUTPUT);
    pinMode(ECHO_LEFT, INPUT);
    pinMode(ECHO_RIGHT, INPUT);
    pinMode(L293D_LEFT_IN1, OUTPUT);
    pinMode(L293D_LEFT_IN2, OUTPUT);
    pinMode(L293D_RIGHT_IN1, OUTPUT);
    pinMode(L293D_RIGHT_IN2, OUTPUT);
    avgV = 0;
    angV = 0;
}

void loop() {
    double left = leftIn();
    double right = rightIn();
    double avg = left + right;
    double ang = atan2(left - right, DISTANCE_BETWEEN_ULTRASONIC_SENSORS);

    #if DEBUG
        *ptr = "";
        avgV += avgPID.update((avg - TARGET_DISTANCE), millis(), ptr);
        Serial.println("Average Speed PID Update:");
        Serial.println(*ptr);
        *ptr = "";
        angV += angPID.update((ang - TARGET_DISTANCE), millis(), ptr) * ANGULAR_SPEED_MULTIPLIER;
        Serial.println("Angular Speed Update:");
        Serial.println(*ptr);
        *ptr = "";
    #else
        avgV += avgPID.update((avg - TARGET_DISTANCE), millis());
        angV += angPID.update((ang - TARGET_DISTANCE), millis()) * ANGULAR_SPEED_MULTIPLIER;
    #endif
```

```
218
219     double leftV = avgV + angV;
220     double rightV = avgV - angV;
221
222     leftOut(leftV);
223     rightOut(rightV);
224
225     delayMicroseconds(10000);
226 }
227
228 inline void leftOut(double leftV) {
229     #if DEBUG
230         Serial.println("Left Original Velocity: " + String(leftV));
231     #endif
232     leftV *= (leftV < 0)?LEFT_NEGATIVE_SPEED_MULTIPLIER:LEFT_POSITIVE_SPEED_MULTIPLIER;
233     leftV = CLAMP(leftV, 255, -255);
234     #if DEBUG
235         Serial.println("Left Outputted Velocity: " + String(leftV));
236     #endif
237     if (leftV > 0) {
238         analogWrite(L293D_LEFT_IN1, 0);
239         analogWrite(L293D_LEFT_IN2, leftV);
240     } else {
241         analogWrite(L293D_LEFT_IN1, -leftV);
242         analogWrite(L293D_LEFT_IN2, 0);
243     }
244 }
245
246 inline void rightOut(double rightV) {
247     #if DEBUG
248         Serial.println("Right Original Velocity: " + String(rightV));
249     #endif
250     rightV *= (rightV < 0)?RIGHT_NEGATIVE_SPEED_MULTIPLIER:RIGHT_POSITIVE_SPEED_MULTIPLIER;
251     rightV = CLAMP(rightV, 255, -255);
252     #if DEBUG
253         Serial.println("Right Outputted Velocity: " + String(rightV));
254     #endif
255     if (rightV > 0) {
256         analogWrite(L293D_RIGHT_IN1, 0);
257         analogWrite(L293D_RIGHT_IN2, rightV);
258     } else {
259         analogWrite(L293D_RIGHT_IN1, -rightV);
260         analogWrite(L293D_RIGHT_IN2, 0);
261     }
262 }
```

VIII Development History

The development of this PID (Proportional-Integral-Derivative) controller project is rooted in control theory, a field of engineering that focuses on the behavior of dynamic systems. What makes us take the initiative is the auto-following kart lesson in our living technology class. We aim to make an auto-following kart from scratch with Arduino and wooden boards and make it as quality as possible on a constrained budget.

i Control Theory and PID Controllers

The development of this PID (Proportional-Integral-Derivative) controller project is rooted in control theory, a field of engineering that focuses on the behavior of dynamic systems. PID controllers are a cornerstone of modern control systems. They are designed to automatically adjust system outputs

based on feedback to minimize error, allowing for precise control of dynamic systems. The PID controller operates based on three fundamental components:

- **Proportional Control (P)**: Responds proportionally to the current error value, providing immediate correction based on how far the system is from the desired state.
- **Integral Control (I)**: Addresses accumulated past errors, ensuring that the system reaches the desired state even if there are persistent biases.
- **Derivative Control (D)**: Predicts future errors based on the rate of change of the error, providing a damping effect to reduce overshoot and oscillation.

The tuning of these parameters is crucial for achieving optimal performance. The traditional Ziegler–Nichols tuning method has served as a foundational approach for many engineers, providing empirical guidelines for setting PID parameters based on system behavior.

ii Development Tools

Balancing heavy academic responsibilities, including preparation for the GSAT, I primarily developed the advanced PID controller, simulation framework, and Arduino implementation using Linux emulation environments on my Android phone. This allowed me to utilize commuting time efficiently and maximize productivity during otherwise idle periods.
Leveraging my research on Android tools, such as **Android Non Root** (https://github.com/Willie169/Android-Non-Root) and **Termux Sh** (https://github.com/Willie169/termux-sh), I set up isolated Linux virtual environments on Termux to streamline my development process. A Debian Bookworm ARM64 proot environment served as the primary development platform due to its optimal performance and lightweight setup, a Debian Bookworm AMD64 QEMU system emulation installed for testing in a more realistic, computer-like environment, and an Alpine 3.21 Virt AMD64 QEMU system emulation configured for running Docker containers cloned from GitHub and GitLab repositories.
During development, I employed a suite of powerful and versatile tools. I primarily used `Vim` for editing to enhance coding speed and convenience, `G++` and `GCC` for compilation to run C and C++ programs, `GDB` for debugging to identify and resolve complex issues, and `git` for versioning to ensure my work remained organized, traceable, and collaborative-ready. These tools not only facilitated seamless development but also prepared me to adopt professional practices in future large-scale projects.

iii Initial Implementation

The journey began with a simple implementation of the traditional PID controller. The focus was on creating a basic class structure in C++ that encapsulated the PID logic, allowing for straightforward application in simulations.

- **Key Features**:
  - Basic PID calculations.
  - Initial support for both Arduino and non-Arduino environments through conditional compilation.

iv Enhancements and Adaptive Features

As the initial implementation was tested, several limitations became apparent, particularly regarding responsiveness and stability. This prompted the incorporation of advanced features such as:

- **Adaptive Gain Adjustment**: Inspired by the need for dynamic responsiveness, the proportional gain K_p was made adaptable based on the error's rate of change. This enhancement allowed for better handling of transient behaviors in the system.
- **Adaptive Integral Term**: To mitigate integral windup, the integral gain K_i was adjusted based on observed error amplitude. This feature improved system stability during sustained error conditions.

v Error History Management

To accurately compute the integral and derivative terms, a robust error history management system was implemented. This system utilized a linked list structure to store past error values and timestamps, enabling precise calculations for both the integral and derivative components. The integration of error history management was crucial for implementing more advanced tuning strategies, inspired by established control theory practices.

vi Additional Derivative Term

Recognizing the need for improved damping characteristics, an additional derivative term K_{dd} was introduced. This term accounted for the acceleration of error changes, allowing the controller to anticipate and react to rapid fluctuations more effectively.

vii Testing and Simulation Framework

With the PID logic in place, attention turned to building a comprehensive simulation framework to test various PID configurations. The introduction of the `Car` class for simulating follower and leader vehicles added a practical context for evaluating PID performance in a dynamic environment.

- **PID Testing Function**: A dedicated function was developed to simulate the interaction between the leader and follower karts, providing insights into how different PID parameters affected system behavior.

viii Optimization and Multi-threading

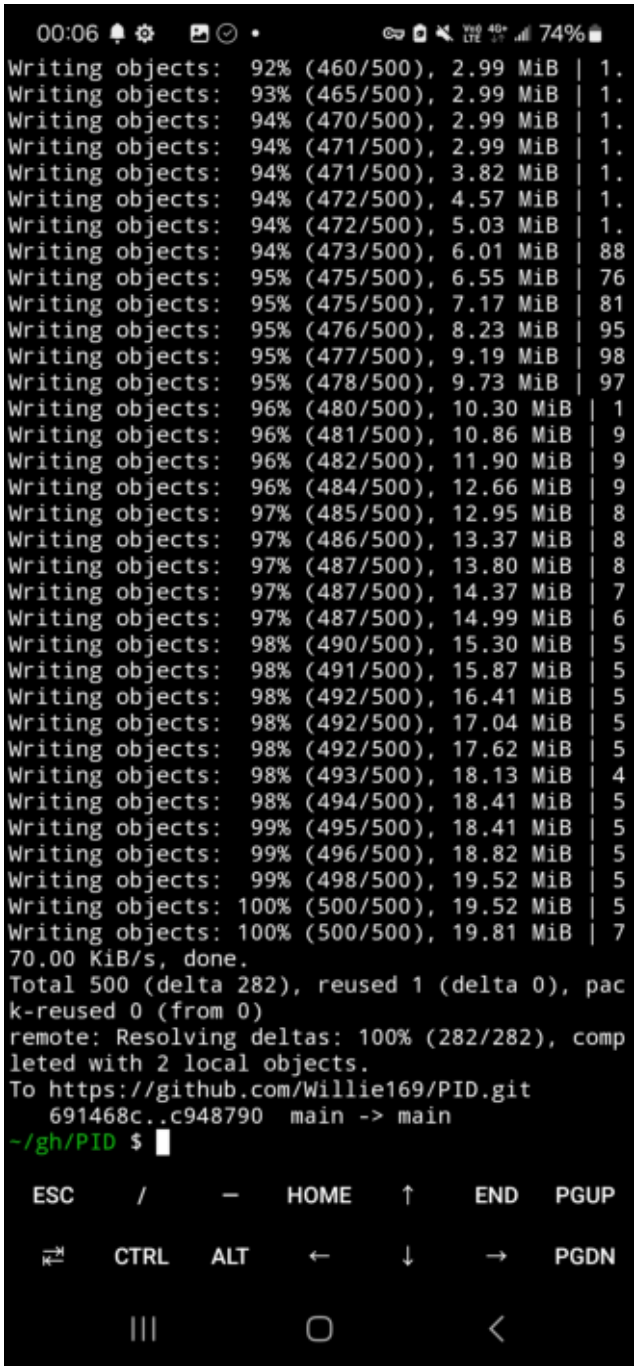
To facilitate further exploration of PID parameter tuning, optimization functions were integrated. Utilizing multi-threading allowed for the concurrent evaluation of multiple parameter combinations, greatly enhancing the testing efficiency. The optimization framework also included mechanisms for result logging and progress tracking, making it easier to analyze and visualize outcomes.

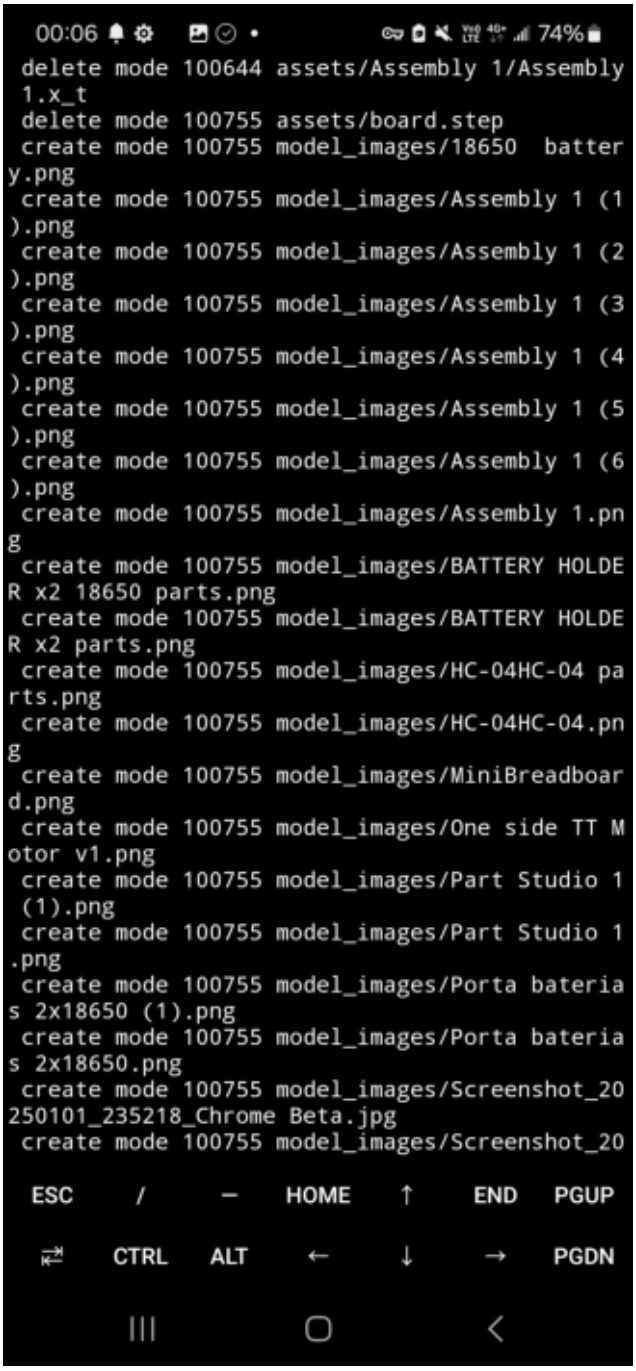
ix Arduino Auto-Following Kart

We use two ultrasonic sensors to measure the distance on both sides. The difference in distance helps adjust the robot's angular velocity, while the average of both distances adjusts its linear speed.

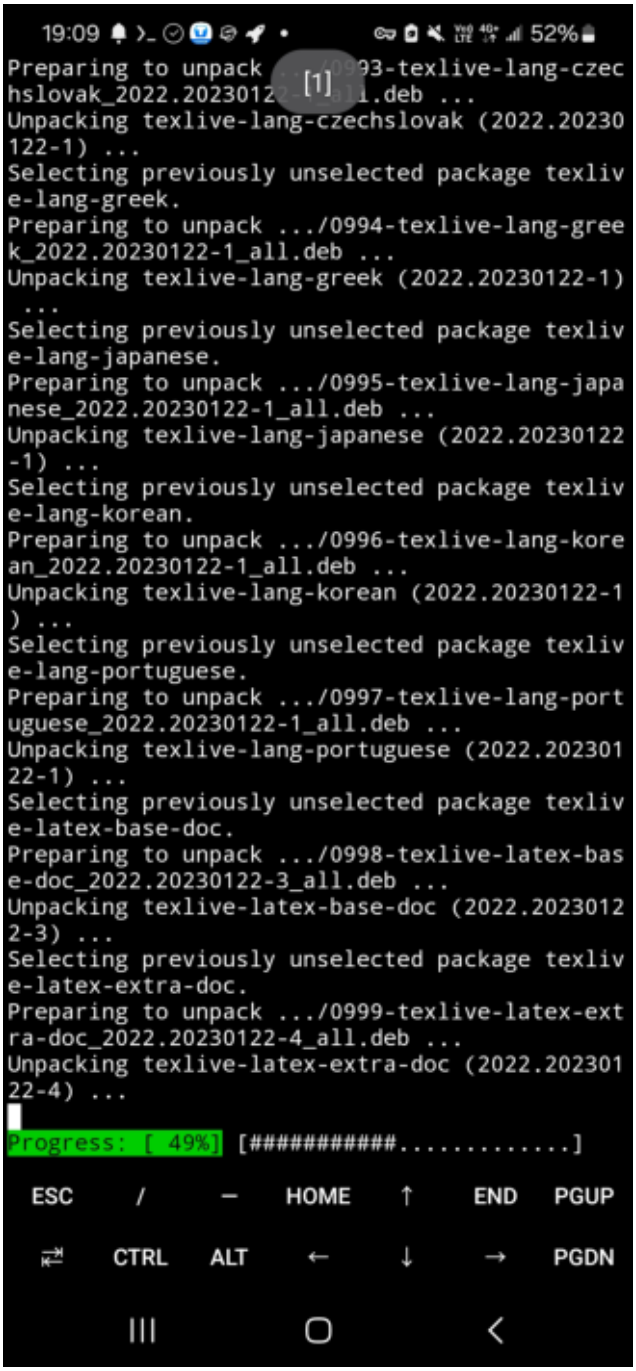
The PID controllers fine-tune these adjustments to ensure the robot stays at the target distance, while adjusting its orientation to keep the two sensors aligned. The L293D motor driver uses these velocity values to control the left and right motors, adjusting the robot's movement accordingly.

IX Working Screenshots






17:26
56%
#264416=PRODUCT_CONTEXT('',#264422,'mechanical
');
#264417=PRODUCT_CONTEXT('',#264422,'mechanical
');
#264418=PRODUCT_CONTEXT('',#264422,'mechanical
');
#264419=PRODUCT_CONTEXT('',#264422,'mechanical
');
#264420=PRODUCT_CONTEXT('',#264422,'mechanical
');
#264421=APPLICATION_PROTOCOL_DEFINITION('inter
national standard',
'ap242_managed_model_based_3d_engineering',201
1,#264422);
#264422=APPLICATION_CONTEXT('managed model bas
ed 3d engineering');
ENDSEC;
END-ISO-10303-21;
Assembly 1.step 278570,1 Bot
ESC / – HOME ↑ END PGUP
CTRL ALT ← ↓ → PGDN
1 2 3 4 5 6 7 8 9 0
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
?123 English


19:09
52%
[1]
Preparing to unpack .../0993-texlive-lang-czec
hslovak_2022.20230122-1.deb ...
Unpacking texlive-lang-czechslovak (2022.20230
122-1) ...
Selecting previously unselected package texliv
e-lang-greek.
Preparing to unpack .../0994-texlive-lang-gree
k_2022.20230122-1_all.deb ...
Unpacking texlive-lang-greek (2022.20230122-1)
...
Selecting previously unselected package texliv
e-lang-japanese.
Preparing to unpack .../0995-texlive-lang-japa
nese_2022.20230122-1_all.deb ...
Unpacking texlive-lang-japanese (2022.20230122
-1) ...
Selecting previously unselected package texliv
e-lang-korean.
Preparing to unpack .../0996-texlive-lang-kore
an_2022.20230122-1_all.deb ...
Unpacking texlive-lang-korean (2022.20230122-1
) ...
Selecting previously unselected package texliv
e-lang-portuguese.
Preparing to unpack .../0997-texlive-lang-port
uguese_2022.20230122-1_all.deb ...
Unpacking texlive-lang-portuguese (2022.202301
22-1) ...
Selecting previously unselected package texliv
e-latex-base-doc.
Preparing to unpack .../0998-texlive-latex-bas
e-doc_2022.20230122-3_all.deb ...
Unpacking texlive-latex-base-doc (2022.2023012
2-3) ...
Selecting previously unselected package texliv
e-latex-extra-doc.
Preparing to unpack .../0999-texlive-latex-ext
ra-doc_2022.20230122-4_all.deb ...
Unpacking texlive-latex-extra-doc (2022.202301
22-4) ...
Progress: [49%] [############.............]
ESC / – HOME ↑ END PGUP
CTRL ALT ← ↓ → PGDN

5 硬體設計與 3D 建模

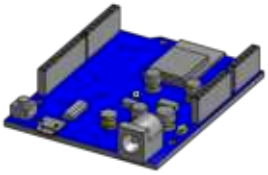
在這次的車輛設計中，我們以自然界的古老生物——鱟作為靈感來源，結合其生物學特性與流體力學特徵，利用 L293D 控制馬達，打造具有前瞻性與功能性的車輛外形與結構。鱟的流線型身體和穩定行走的構造，為我們提供了在設計中平衡美學與實用性的寶貴啟示。

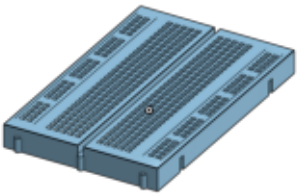
由於跟隨車需要做到即時與大動作地加速與轉彎反饋，我們採用了前驅作為動力系統的核心設計，並在後輪設計採用了萬向滾珠，提高了車輛加速與轉彎時的性能，使車輛在移動時能夠實現靈活的大角度轉彎，增加動力使用效率，並保持配重穩定性。

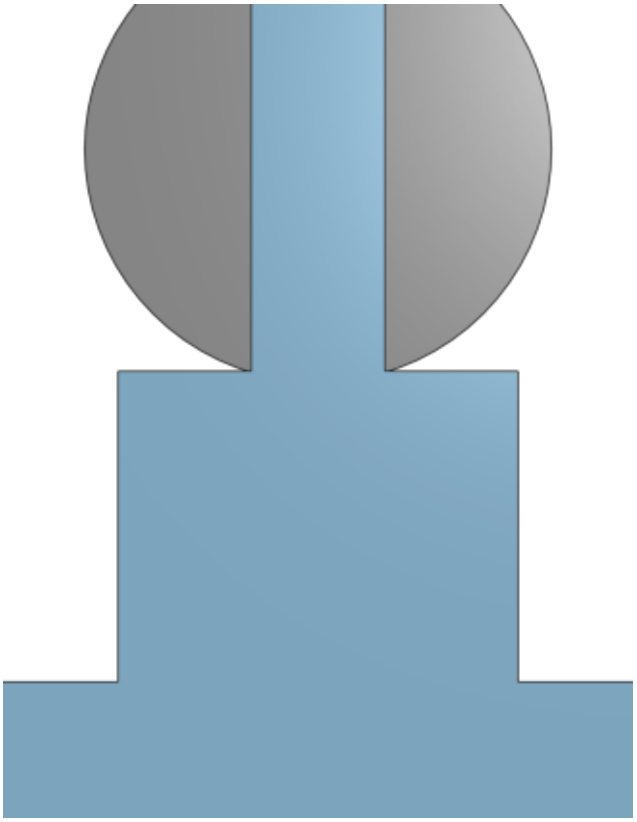
這輛車的設計展現了我們對自然生物演化的體悟，也彰顯了我們對未來交通工具的遠見。通

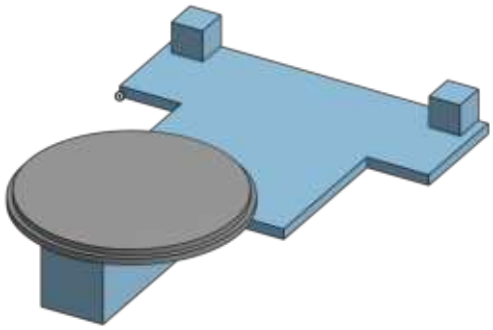
過模仿鱟的仿生設計結構，我們希望這樣的跟隨車設計可以具有提高機動性與增加適用場景的潛能，可以為載人、運貨、倉管、軍事、農業等潛在場景特化使用。

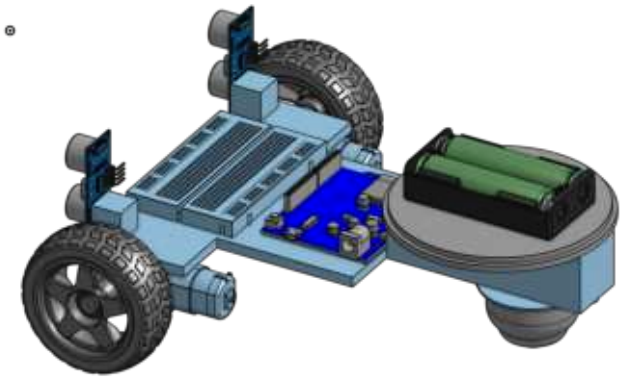
I 建模圖片

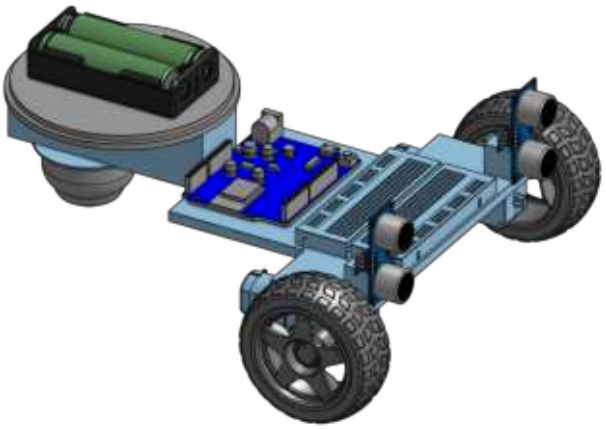


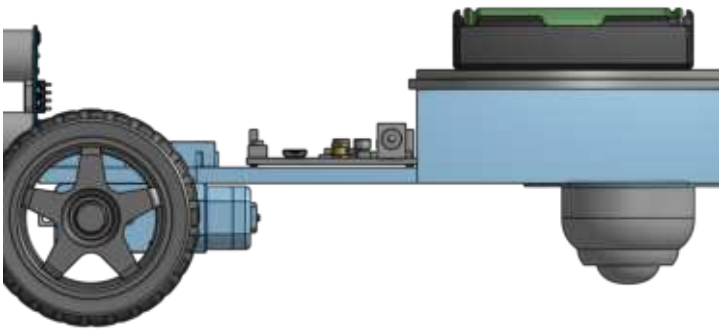


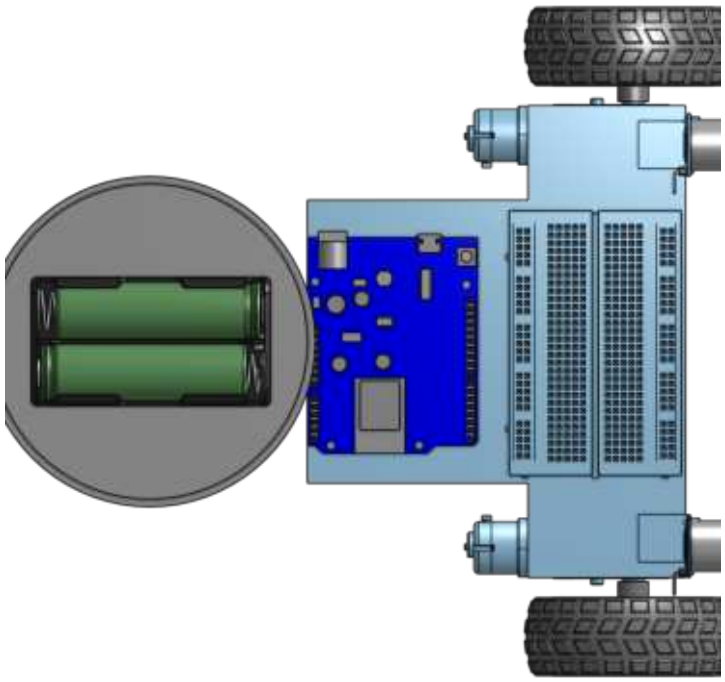


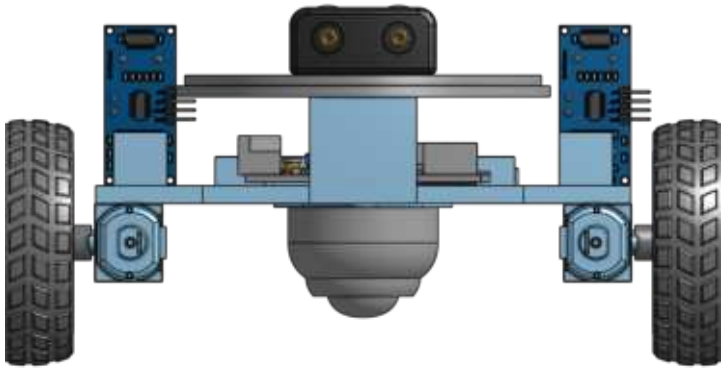


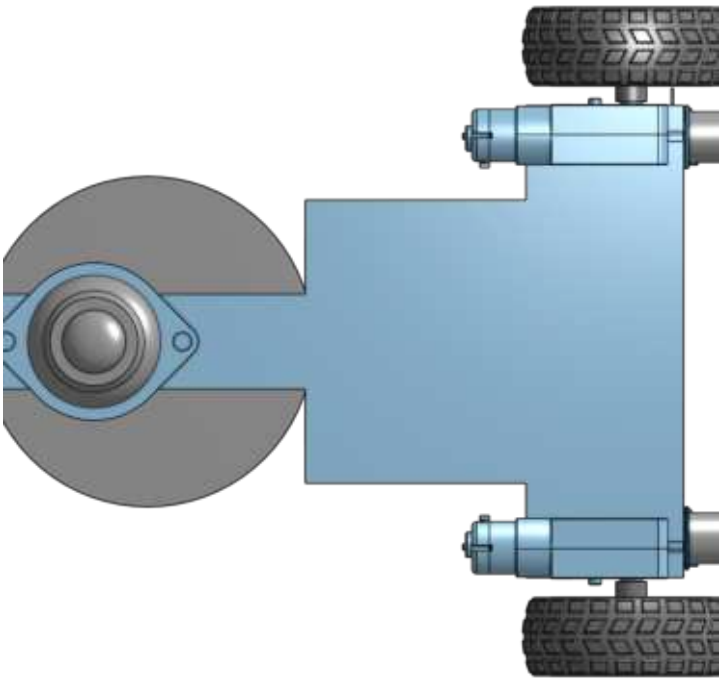


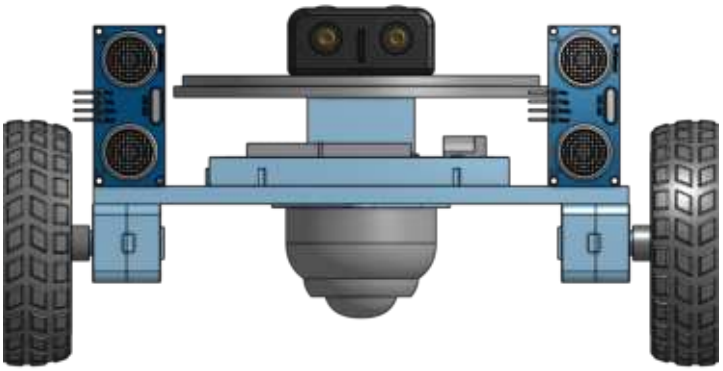




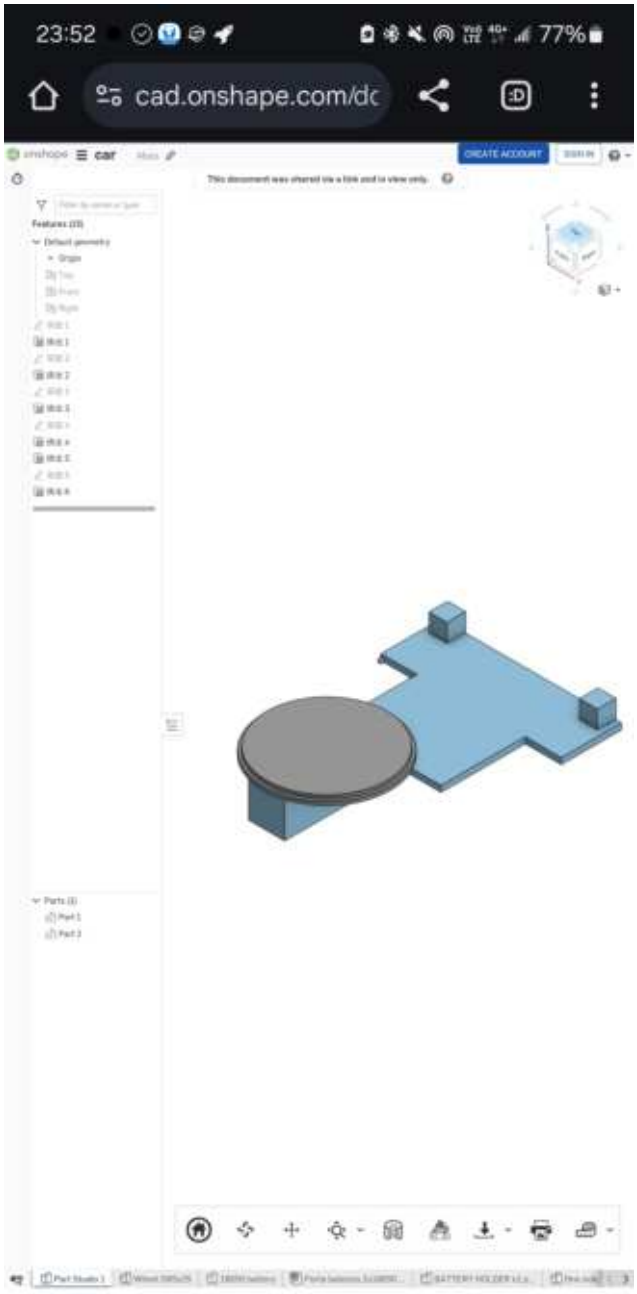




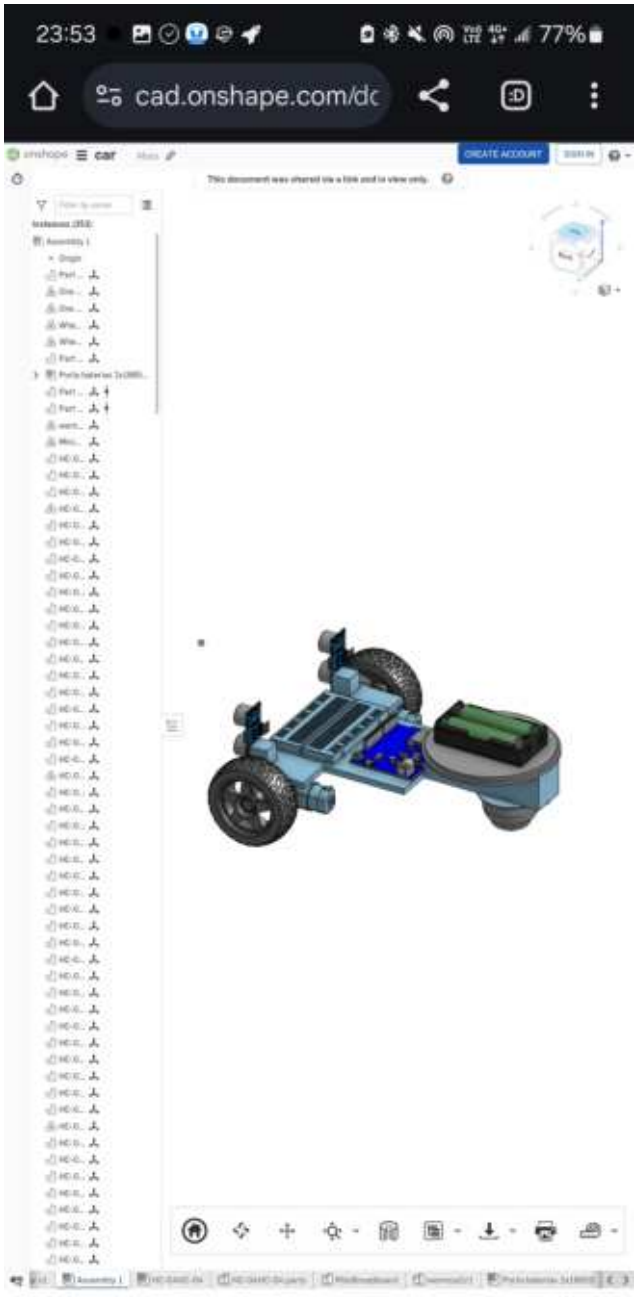


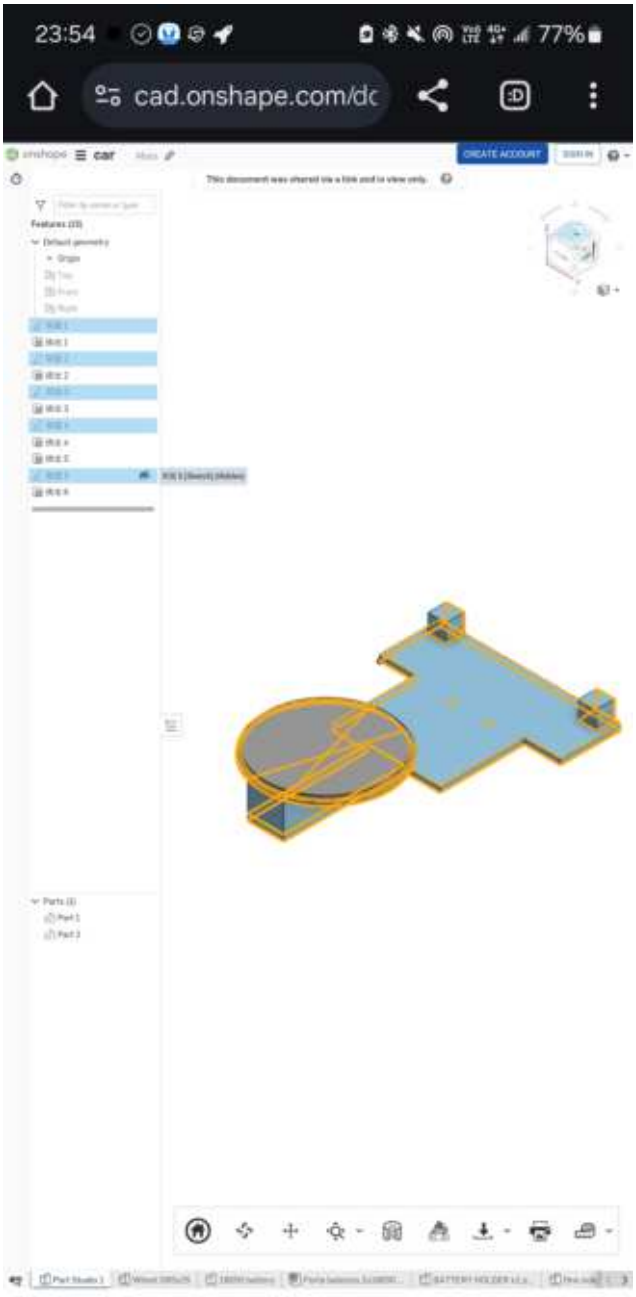


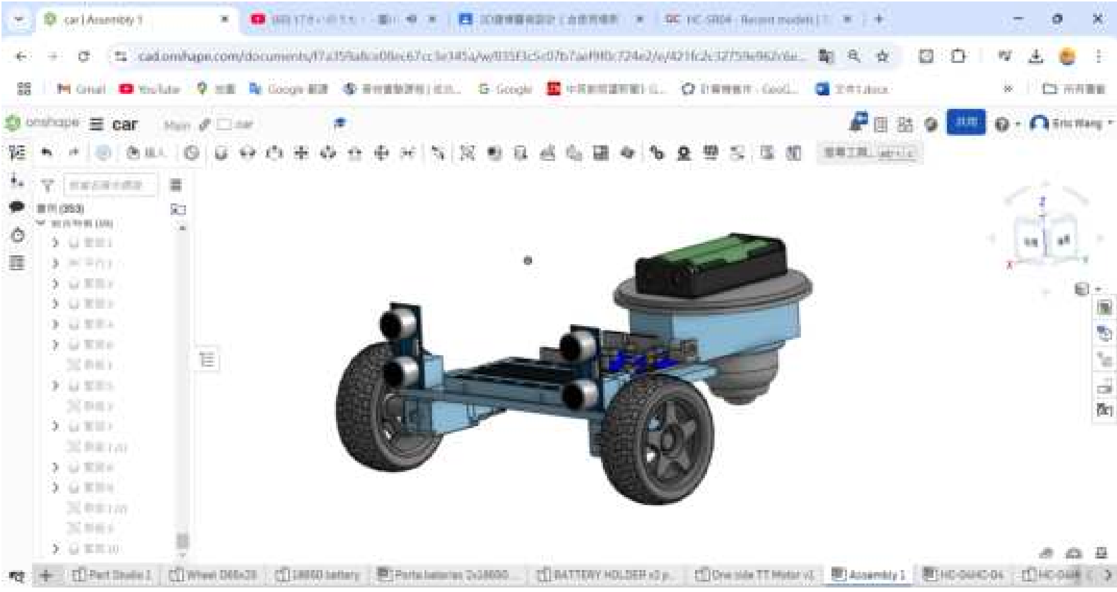
II 工作截圖

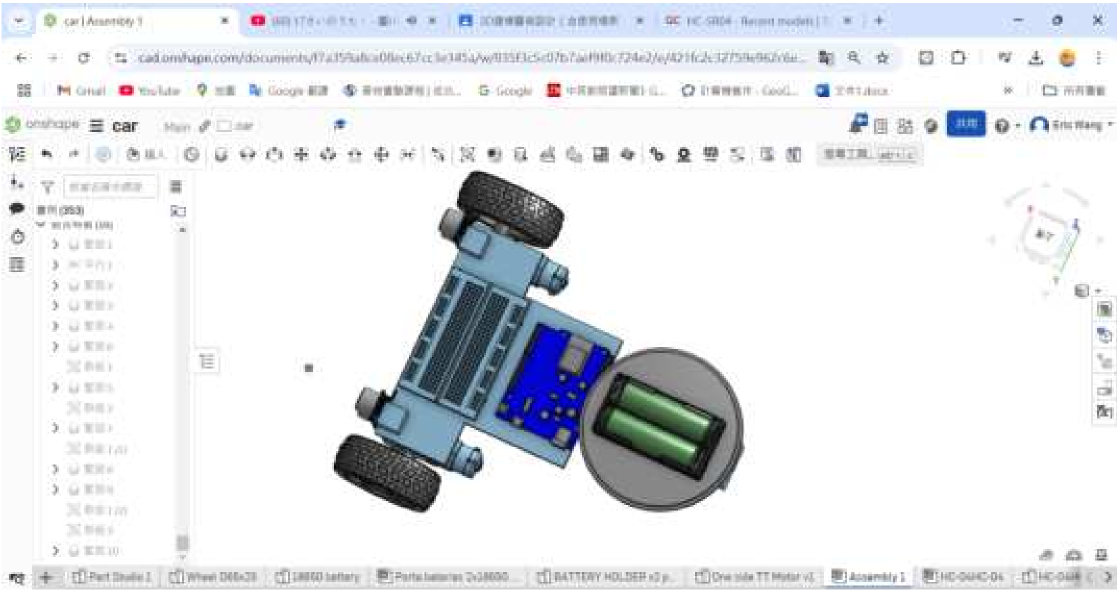

23:52
77%
cad.onshape.com/dc
onshape
car
CREATE ACCOUNT

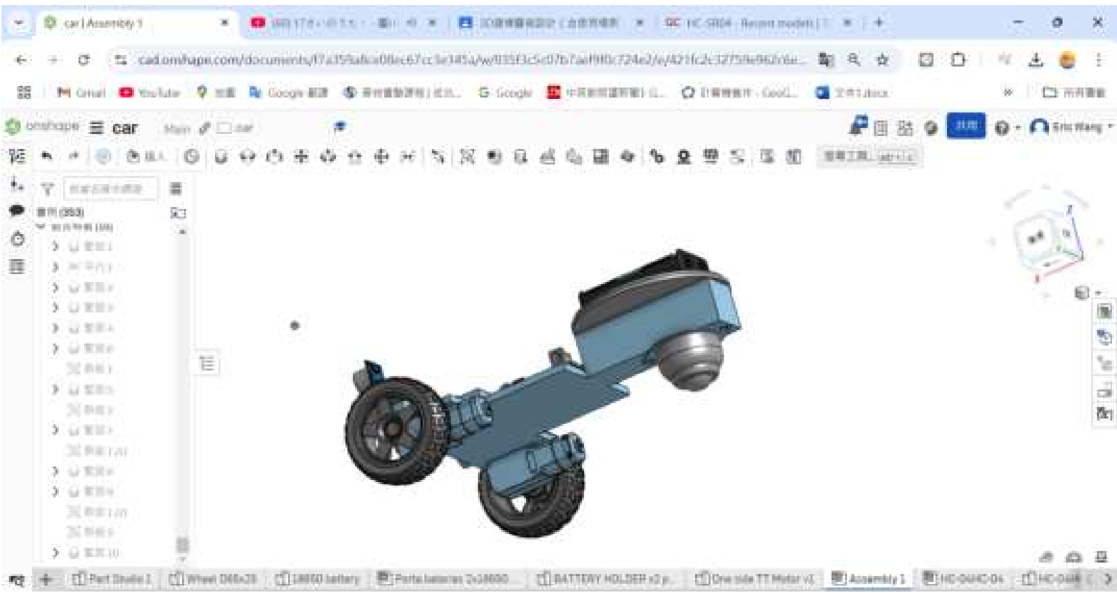

23:53
77%
cad.onshape.com/dc
onshape
car
CREATE ACCOUNT

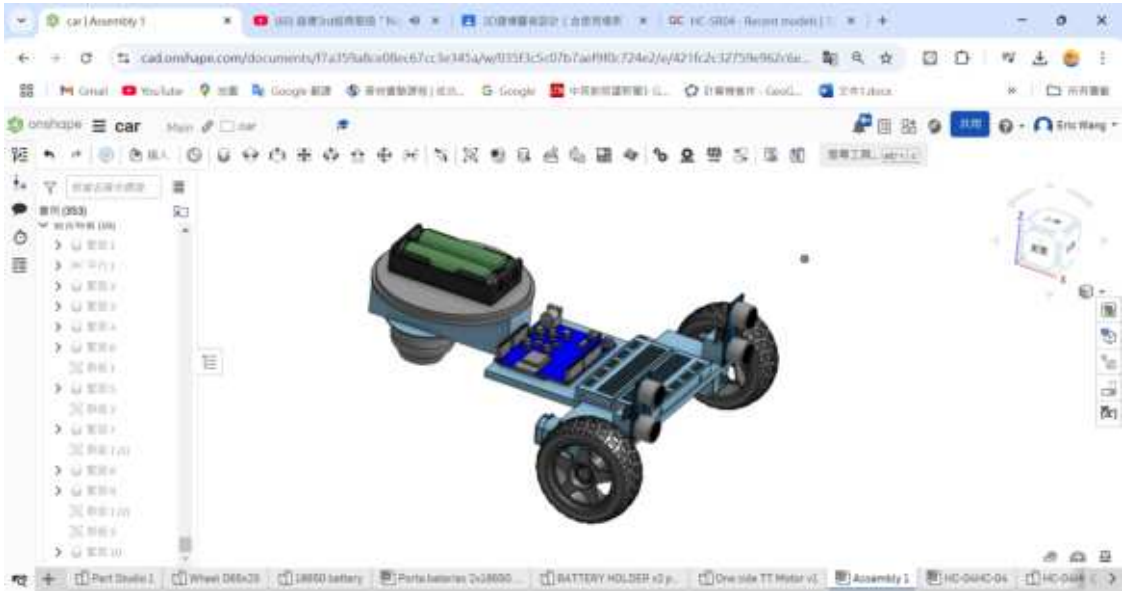

23:53
77%
cad.onshape.com/dc
car
CREATE ACCOUNT


23:54
77%
cad.onshape.com/dc
car
CREATE ACCOUNT


onshape
car
Part Studio 1
Assembly 1


onshape
car
Part Studio 1
Assembly 1


onshape
car
Part Studio 1
Assembly 1



III STEP Code

由於字數過多，共 2785701 行，故此報告僅錄部分，完整版請詳見以下連結

- https://github.com/Willie169/PID/blob/main/Assembly1.step
- https://github.com/Willie169/PID/tree/main/Assembly1Parts
- https://cad.onshape.com/documents/f7a359a8ce08ec67cc3e345a/w/035f3c5c07b7aef9f0c724e2/e/421fc2c32759e962c6ebb50d

```
ISO-10303-21;
HEADER;
/* Generated by software containing ST-Developer
 * from STEP Tools, Inc. (www.steptools.com)
 */
/* OPTION: using custom renumber hook */

FILE_DESCRIPTION(
/* description */ ('STEP AP242',
'CAx-IF Rec.Pracs.---Representation and Presentation of Product Manufa
cturing Information (PMI)---4.0---2014-10-13',
'CAx-IF Rec.Pracs.---3D Tessellated Geometry---0.4---2014-09-14','2;1'),

/* implementation_level */ '2;1');

FILE_NAME(
/* name */ 'Assembly 1',
/* time_stamp */ '2025-01-01T15:34:22Z',
/* author */ (''),
/* organization */ (''),
/* preprocessor_version */ 'ST-DEVELOPER v20',
/* originating_system */ 'ONSHAPE BY PTC INC, 1.191',
/* authorisation */ '  ');

FILE_SCHEMA (('AP242_MANAGED_MODEL_BASED_3D_ENGINEERING_MIM_LF { 1 0 10303 442 1 1 4 }'));
ENDSEC;

```

```
DATA;
#10=SHAPE_REPRESENTATION_RELATIONSHIP('','',#146,#11);
#11=MANIFOLD_SURFACE_SHAPE_REPRESENTATION('',(#144),#265);
#12=LINE('',#216,#27);
#13=LINE('',#218,#28);
#14=LINE('',#224,#29);
#15=LINE('',#226,#30);
#16=LINE('',#234,#31);
#17=LINE('',#236,#32);
#18=LINE('',#238,#33);
#19=LINE('',#240,#34);
#20=LINE('',#242,#35);
#21=LINE('',#244,#36);
#22=LINE('',#250,#37);
#23=LINE('',#252,#38);
#24=LINE('',#254,#39);
#25=LINE('',#260,#40);
#26=LINE('',#262,#41);
#27=VECTOR('',#169,1000.);
#28=VECTOR('',#170,1000.);
#29=VECTOR('',#175,1000.);
#30=VECTOR('',#176,1000.);
#31=VECTOR('',#183,1000.);
#32=VECTOR('',#184,1000.);
#33=VECTOR('',#185,1000.);
#34=VECTOR('',#186,1000.);
#35=VECTOR('',#187,1000.);
#36=VECTOR('',#188,1000.);
#37=VECTOR('',#193,1000.);
#38=VECTOR('',#194,1000.);
#39=VECTOR('',#195,1000.);
#40=VECTOR('',#200,1000.);
#41=VECTOR('',#201,1000.);
#42=CIRCLE('',#149,0.337597319506925);
#43=CIRCLE('',#150,0.285599344380192);
#44=CIRCLE('',#151,0.0207227374829273);
#45=CIRCLE('',#152,0.0258318878583709);
#46=CIRCLE('',#153,0.0947168202412612);
#47=CIRCLE('',#154,0.0622037543980458);
#48=CIRCLE('',#155,0.0878773095509511);
#49=CIRCLE('',#156,0.0447119737316936);
#50=CIRCLE('',#157,0.0390300396032908);
#51=CIRCLE('',#158,0.0368943935810302);
#52=CIRCLE('',#159,0.040691593617163);
#53=CIRCLE('',#160,0.233427795781408);
#54=ORIENTED_EDGE('',*,*,#82,.F.);
#55=ORIENTED_EDGE('',*,*,#83,.F.);
#56=ORIENTED_EDGE('',*,*,#84,.F.);
#57=ORIENTED_EDGE('',*,*,#85,.F.);
#58=ORIENTED_EDGE('',*,*,#86,.F.);
#59=ORIENTED_EDGE('',*,*,#87,.F.);
#60=ORIENTED_EDGE('',*,*,#88,.F.);
#61=ORIENTED_EDGE('',*,*,#89,.F.);
#62=ORIENTED_EDGE('',*,*,#90,.F.);
#63=ORIENTED_EDGE('',*,*,#91,.F.);
#64=ORIENTED_EDGE('',*,*,#92,.F.);
#65=ORIENTED_EDGE('',*,*,#93,.F.);
#66=ORIENTED_EDGE('',*,*,#94,.F.);
#67=ORIENTED_EDGE('',*,*,#95,.F.);
#68=ORIENTED_EDGE('',*,*,#96,.F.);
#69=ORIENTED_EDGE('',*,*,#97,.F.);
#70=ORIENTED_EDGE('',*,*,#98,.F.);
#71=ORIENTED_EDGE('',*,*,#99,.F.);
#72=ORIENTED_EDGE('',*,*,#100,.F.);
#73=ORIENTED_EDGE('',*,*,#101,.F.);
```

```
#74=ORIENTED_EDGE('',*,*,#102,.F.);
#75=ORIENTED_EDGE('',*,*,#103,.F.);
#76=ORIENTED_EDGE('',*,*,#104,.F.);
#77=ORIENTED_EDGE('',*,*,#105,.F.);
#78=ORIENTED_EDGE('',*,*,#106,.F.);
#79=ORIENTED_EDGE('',*,*,#107,.F.);
#80=ORIENTED_EDGE('',*,*,#108,.F.);
#81=ORIENTED_EDGE('',*,*,#109,.F.);
#82=EDGE_CURVE('',#110,#111,#138,.T.);
#83=EDGE_CURVE('',#112,#110,#42,.T.);
#84=EDGE_CURVE('',#113,#112,#43,.T.);
#85=EDGE_CURVE('',#114,#113,#12,.T.);
#86=EDGE_CURVE('',#115,#114,#13,.T.);
#87=EDGE_CURVE('',#116,#115,#44,.T.);
#88=EDGE_CURVE('',#117,#116,#45,.T.);
#89=EDGE_CURVE('',#118,#117,#14,.T.);
#90=EDGE_CURVE('',#119,#118,#15,.T.);
#91=EDGE_CURVE('',#120,#119,#46,.T.);
#92=EDGE_CURVE('',#121,#120,#47,.T.);
#93=EDGE_CURVE('',#122,#121,#48,.T.);
#94=EDGE_CURVE('',#123,#122,#16,.T.);
#95=EDGE_CURVE('',#124,#123,#17,.T.);
#96=EDGE_CURVE('',#125,#124,#18,.T.);
#97=EDGE_CURVE('',#126,#125,#19,.T.);
#98=EDGE_CURVE('',#127,#126,#20,.T.);
#99=EDGE_CURVE('',#128,#127,#21,.T.);
#100=EDGE_CURVE('',#129,#128,#49,.T.);
#101=EDGE_CURVE('',#130,#129,#50,.T.);
#102=EDGE_CURVE('',#131,#130,#22,.T.);
#103=EDGE_CURVE('',#132,#131,#23,.T.);
#104=EDGE_CURVE('',#133,#132,#24,.T.);
#105=EDGE_CURVE('',#134,#133,#51,.T.);
#106=EDGE_CURVE('',#135,#134,#52,.T.);
#107=EDGE_CURVE('',#136,#135,#25,.T.);
#108=EDGE_CURVE('',#137,#136,#26,.T.);
#109=EDGE_CURVE('',#111,#137,#53,.T.);
#110=VERTEX_POINT('',#210);
#111=VERTEX_POINT('',#211);
#112=VERTEX_POINT('',#213);
#113=VERTEX_POINT('',#215);
#114=VERTEX_POINT('',#217);
#115=VERTEX_POINT('',#219);
#116=VERTEX_POINT('',#221);
#117=VERTEX_POINT('',#223);
#118=VERTEX_POINT('',#225);
#119=VERTEX_POINT('',#227);
#120=VERTEX_POINT('',#229);
#121=VERTEX_POINT('',#231);
#122=VERTEX_POINT('',#233);
#123=VERTEX_POINT('',#235);
#124=VERTEX_POINT('',#237);
#125=VERTEX_POINT('',#239);
#126=VERTEX_POINT('',#241);
#127=VERTEX_POINT('',#243);
#128=VERTEX_POINT('',#245);
#129=VERTEX_POINT('',#247);
#130=VERTEX_POINT('',#249);
#131=VERTEX_POINT('',#251);
#132=VERTEX_POINT('',#253);
#133=VERTEX_POINT('',#255);
#134=VERTEX_POINT('',#257);
#135=VERTEX_POINT('',#259);
#136=VERTEX_POINT('',#261);
#137=VERTEX_POINT('',#263);
#138=B_SPLINE_CURVE_WITH_KNOTS('',3,(#206,#207,#208,#209),.UNSPECIFIED.,
```

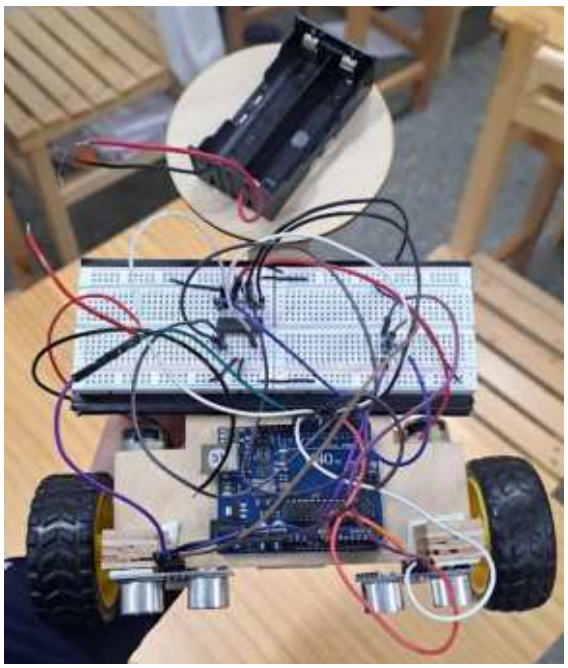
```
 .F.,.F.,(4,4),(0.,1.),.UNSPECIFIED.);
#139=EDGE_LOOP('',(#54,#55,#56,#57,#58,#59,#60,#61,#62,#63,#64,#65,#66,
#67,#68,#69,#70,#71,#72,#73,#74,#75,#76,#77,#78,#79,#80,#81));
#140=FACE_BOUND('',#139,.T.);
#141=PLANE('',#148);
#142=ADVANCED_FACE('',(#140),#141,.T.);
#143=OPEN_SHELL('',(#142));
#144=SHELL_BASED_SURFACE_MODEL('HC-04ShapeString011',(#143));
#145=SHAPE_DEFINITION_REPRESENTATION(#270,#146);
#146=SHAPE_REPRESENTATION('HC-04ShapeString011',(#147),#265);
#147=AXIS2_PLACEMENT_3D('',#204,#161,#162);
#148=AXIS2_PLACEMENT_3D('',#205,#163,#164);
#149=AXIS2_PLACEMENT_3D('',#212,#165,#166);
#150=AXIS2_PLACEMENT_3D('',#214,#167,#168);
#151=AXIS2_PLACEMENT_3D('',#220,#171,#172);
#152=AXIS2_PLACEMENT_3D('',#222,#173,#174);
#153=AXIS2_PLACEMENT_3D('',#228,#177,#178);
#154=AXIS2_PLACEMENT_3D('',#230,#179,#180);
#155=AXIS2_PLACEMENT_3D('',#232,#181,#182);
#156=AXIS2_PLACEMENT_3D('',#246,#189,#190);
#157=AXIS2_PLACEMENT_3D('',#248,#191,#192);
#158=AXIS2_PLACEMENT_3D('',#256,#196,#197);
#159=AXIS2_PLACEMENT_3D('',#258,#198,#199);
#160=AXIS2_PLACEMENT_3D('',#264,#202,#203);
#161=DIRECTION('',(0.,0.,1.));
#162=DIRECTION('',(1.,0.,0.));
#163=DIRECTION('',(0.,-1.,0.));
#164=DIRECTION('',(1.,0.,0.));
#165=DIRECTION('',(1.00665346793286E-17,1.,-6.03992080759717E-17));
#166=DIRECTION('',(0.326496682273424,-6.03759218496179E-17,-0.945198347683938));
#167=DIRECTION('',(3.03497576612031E-17,1.,-5.31816854430253E-17));
#168=DIRECTION('',(0.65357109707195,-6.00870875386104E-17,-0.756865127398646));
#169=DIRECTION('',(0.512510977513211,0.,-0.858680672851353));
#170=DIRECTION('',(0.662573882203037,0.,-0.748996562490374));
#171=DIRECTION('',(1.93633660727018E-17,1.,5.80900982181059E-17));
#172=DIRECTION('',(-0.0287255166194788,-5.75099039017191E-17,0.99958733720218));
#173=DIRECTION('',(-1.87376581651315E-17,1.,5.8294936513743E-17));
#174=DIRECTION('',(-0.564334095775335,-5.86994803495582E-17,0.825546502836415));
#175=DIRECTION('',(0.88311571945741,0.,0.469155225961751));
#176=DIRECTION('',(0.915644030191286,0.,0.401990062035195));
#177=DIRECTION('',(-8.2274021014388E-18,-1.,6.06770904981105E-17));
#178=DIRECTION('',(-0.0230332373689862,-6.04714891480939E-17,-0.999734699796053));
#179=DIRECTION('',(2.39142809929948E-17,-1.,5.6369376626345E-17));
#180=DIRECTION('',(-0.750072988249742,-5.52176285354746E-17,-0.661355057664264));
#181=DIRECTION('',(5.85245851861783E-17,-1.,1.80075646726705E-17));
#182=DIRECTION('',(-0.998364234827914,-5.73992900415726E-17,0.0571738980346342));
#183=DIRECTION('',(0.,0.,-1.));
#184=DIRECTION('',(-1.,0.,0.));
#185=DIRECTION('',(0.,0.,-1.));
#186=DIRECTION('',(1.,0.,0.));
#187=DIRECTION('',(0.,0.,-1.));
#188=DIRECTION('',(1.,0.,0.));
#189=DIRECTION('',(-1.77868683044478E-17,1.,5.85920367675922E-17));
#190=DIRECTION('',(-0.587604794038401,-5.78612828020793E-17,0.809148074225657));
#191=DIRECTION('',(-5.23045763484758E-17,1.,3.18375682121157E-17));
#192=DIRECTION('',(-0.9811218143935,-5.74742523009248E-17,0.193390758107018));
#193=DIRECTION('',(0.18841061712191,0.,0.982090341748528));
#194=DIRECTION('',(0.987825950711781,0.,0.155563141843966));
#195=DIRECTION('',(0.,0.,1.));
#196=DIRECTION('',(-5.74569447256013E-17,1.,-2.11683480568002E-17));
#197=DIRECTION('',(-0.71286765084448,-5.58045298567601E-17,-0.701298590031002));
#198=DIRECTION('',(-1.96912444550614E-17,1.,-5.79797753399033E-17));
#199=DIRECTION('',(0.0236506090873736,-5.74978476150228E-17,-0.99972028522472));
#200=DIRECTION('',(-1.,0.,0.));
#201=DIRECTION('',(0.,0.,1.));
```

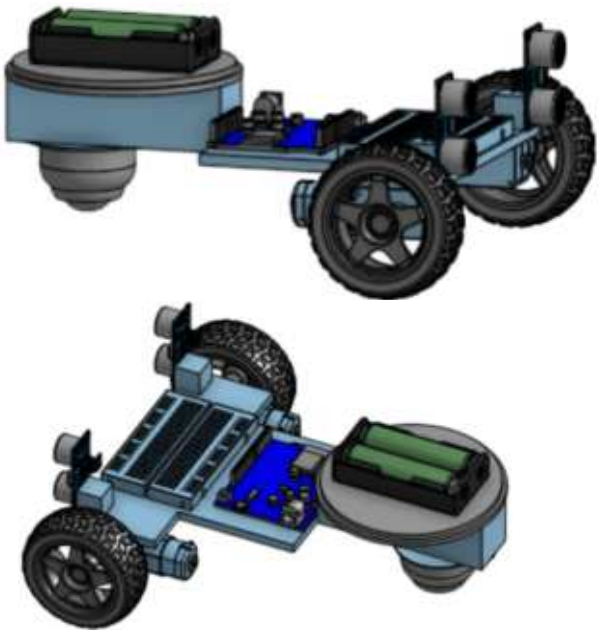
```
#202=DIRECTION('',(-5.80366339844372E-17,1.,-1.95230267225937E-17));
#203=DIRECTION('',(-0.762925074710322,-5.68989846224568E-17,-0.646486914313237));
#204=CARTESIAN_POINT('',(0.,0.,0.));
#205=CARTESIAN_POINT('',(2.2765,0.,0.));
#206=CARTESIAN_POINT('',(2.2765,0.,0.));
#207=CARTESIAN_POINT('',(2.20783333333333,0.,0.));
#208=CARTESIAN_POINT('',(2.15516666666667,0.,0.0196666666666667));
#209=CARTESIAN_POINT('',(2.1185,0.,0.059));
#210=CARTESIAN_POINT('',(2.2765,0.,0.));
#211=CARTESIAN_POINT('',(2.1185,0.,0.059));
#212=CARTESIAN_POINT('',(2.27727559523659,2.03827493791906E-17,0.337596428580471));
#213=CARTESIAN_POINT('',(2.3875,0.,0.0185));
#214=CARTESIAN_POINT('',(2.29634052317041,1.71608328067423E-17,0.289160184169284));
#215=CARTESIAN_POINT('',(2.483,0.,0.073));
#216=CARTESIAN_POINT('',(2.426,0.,0.1685));
#217=CARTESIAN_POINT('',(2.426,0.,0.1685));
#218=CARTESIAN_POINT('',(2.42025,0.,0.175));
#219=CARTESIAN_POINT('',(2.4145,0.,0.1815));
#220=CARTESIAN_POINT('',(2.40159527133997,1.1917626412237E-18,0.165285814019901));
#221=CARTESIAN_POINT('',(2.401,0.,0.186));
#222=CARTESIAN_POINT('',(2.40157781507673,1.51631839373443E-18,0.16017457531686));
#223=CARTESIAN_POINT('',(2.387,0.,0.1815));
#224=CARTESIAN_POINT('',(2.379,0.,0.17725));
#225=CARTESIAN_POINT('',(2.371,0.,0.173));
#226=CARTESIAN_POINT('',(2.36075,0.,0.1685));
#227=CARTESIAN_POINT('',(2.3505,0.,0.164));
#228=CARTESIAN_POINT('',(2.32318163500346,5.72766716736138E-18,0.254691691849534));
#229=CARTESIAN_POINT('',(2.321,0.,0.16));
#230=CARTESIAN_POINT('',(2.3181573559417,3.43474380386319E-18,0.222138767576854));
#231=CARTESIAN_POINT('',(2.2715,0.,0.181));
#232=CARTESIAN_POINT('',(2.34123356290857,5.04409517898811E-18,0.234475711664176));
#233=CARTESIAN_POINT('',(2.2535,0.,0.2395));
#234=CARTESIAN_POINT('',(2.2535,0.,0.672));
#235=CARTESIAN_POINT('',(2.2535,0.,0.672));
#236=CARTESIAN_POINT('',(2.4635,0.,0.672));
#237=CARTESIAN_POINT('',(2.4635,0.,0.672));
#238=CARTESIAN_POINT('',(2.4635,0.,0.8));
#239=CARTESIAN_POINT('',(2.4635,0.,0.8));
#240=CARTESIAN_POINT('',(2.2535,0.,0.8));
#241=CARTESIAN_POINT('',(2.2535,0.,0.8));
#242=CARTESIAN_POINT('',(2.2535,0.,1.04));
#243=CARTESIAN_POINT('',(2.2535,0.,1.04));
#244=CARTESIAN_POINT('',(2.155,0.,1.04));
#245=CARTESIAN_POINT('',(2.155,0.,1.04));
#246=CARTESIAN_POINT('',(2.15327297011566,2.58709215672868E-18,0.995321392560172));
#247=CARTESIAN_POINT('',(2.127,0.,1.0315));
#248=CARTESIAN_POINT('',(2.15129322327143,2.24322234347462E-18,1.00095195105217));
#249=CARTESIAN_POINT('',(2.113,0.,1.0085));
#250=CARTESIAN_POINT('',(2.073,0.,0.8));
#251=CARTESIAN_POINT('',(2.073,0.,0.8));
#252=CARTESIAN_POINT('',(1.946,0.,0.78));
#253=CARTESIAN_POINT('',(1.946,0.,0.78));
#254=CARTESIAN_POINT('',(1.946,0.,0.7095));
#255=CARTESIAN_POINT('',(1.946,0.,0.7095));
#256=CARTESIAN_POINT('',(1.98280081968144,2.05887428813966E-18,0.706873986198425));
#257=CARTESIAN_POINT('',(1.9565,0.,0.681));
#258=CARTESIAN_POINT('',(1.98203761902622,2.33967904901207E-18,0.712680211577199));
#259=CARTESIAN_POINT('',(1.983,0.,0.672));
#260=CARTESIAN_POINT('',(2.0635,0.,0.672));
#261=CARTESIAN_POINT('',(2.0635,0.,0.672));
#262=CARTESIAN_POINT('',(2.0635,0.,0.2225));
#263=CARTESIAN_POINT('',(2.0635,0.,0.2225));
#264=CARTESIAN_POINT('',(2.29658791853599,1.32818045626203E-17,0.209908015409663));
#265=(
GEOMETRIC_REPRESENTATION_CONTEXT(3)
```

```
GLOBAL_UNCERTAINTY_ASSIGNED_CONTEXT((#266))
GLOBAL_UNIT_ASSIGNED_CONTEXT((#269,#268,#267))
REPRESENTATION_CONTEXT('HC-04ShapeString011','TOP_LEVEL_ASSEMBLY_PART')
);
#266=UNCERTAINTY_MEASURE_WITH_UNIT(LENGTH_MEASURE(1.E-5),#269,
'DISTANCE_ACCURACY_VALUE','Maximum Tolerance applied to model');
#267=(
NAMED_UNIT(*)
SI_UNIT($,.STERADIAN.)
SOLID_ANGLE_UNIT()
);
#268=(
NAMED_UNIT(*)
PLANE_ANGLE_UNIT()
SI_UNIT($,.RADIAN.)
);
#269=(
LENGTH_UNIT()
NAMED_UNIT(*)
SI_UNIT(.MILLI.,.METRE.)
);
#270=PRODUCT_DEFINITION_SHAPE('','',#271);
#271=PRODUCT_DEFINITION('','',#273,#272);
#272=PRODUCT_DEFINITION_CONTEXT('',#279,'design');
#273=PRODUCT_DEFINITION_FORMATION_WITH_SPECIFIED_SOURCE('','',#275,
 .NOT_KNOWN.);
#274=PRODUCT_RELATED_PRODUCT_CATEGORY('','',(#275));
#275=PRODUCT('HC-04ShapeString011','HC-04ShapeString011',
'HC-04ShapeString011',(#277));
#276=PRODUCT_CATEGORY('','');
#277=PRODUCT_CONTEXT('',#279,'mechanical');
#278=APPLICATION_PROTOCOL_DEFINITION('international standard',
'ap242_managed_model_based_3d_engineering',2011,#279);
#279=APPLICATION_CONTEXT('managed model based 3d engineering');
ENDSEC;
END-ISO-10303-21;
```

6 車體設計、電路配置與 Arduino 板實測

車體設計





本車的車體設計為三輪車，由兩個前輪驅動，而後輪以萬向滾輪使其可以跟著前輪二馬達的轉速差而轉向，據說這樣前輪驅動的三輪車是比較省力(節能)的形式。車體底板由薄木板切割，形如鱟，並且額外切割一片木板以加固；而後輪萬向滾輪部分以螺絲固定在厚木塊上，其上則有承載電池盒與行動電源(預想)的圓板。

零件部分，從前到後為超音波感測器、Arduino板、麵包板、電池盒與行動電源(實裝，建模時原本麵包板和Arduino板位置相反)。除了接線距離最短與空間安排剛好可以塞下以外，後面的萬向滾輪需要一定配重才能自由滾動(否則只能滑動)，因此電池盒和行動電源置於後方。

電路配置與接線

Component	L293D Pin	Arduino Pin	Other
Motor A Terminal1	Pin 3	-	-
Motor A Terminal2	Pin 6	-	-
Motor B Terminal1	Pin 11	-	-
Motor B Terminal2	Pin 14	-	-
EN1 (Motor A)	Pin 1	D2 (PWM)	-
EN2 (Motor B)	Pin 9	D3 (PWM)	-
IN1	Pin 2	D5	-
IN2	Pin 7	D6	-
IN3	Pin 10	D7	-
IN4	Pin 15	D8	-
Logic Power	Pin 16	Arduino 5V	-
Motor Power	Pin 8	6-12V Battery	-
Ground	Pin 4,5,12,13	Arduino GND	Battery GND

Final Updated Wiring Table

Component	Pin on Ultrasonic Sensor	Pin on Arduino D1
Ultrasonic Sensor 1	VCC	5V
	GND	GND
	TRIG	D0
	ECHO	D9
Ultrasonic Sensor 2	VCC	5V
	GND	GND
	TRIG	D1
	ECHO	D10

我們的電路配置與接線並沒有於tinkercad接出來，而是以gpt生成接線表格並檢查無誤後直接於麵包板與Arduino板接線。比較重要的部分在於L293D的各腳位接線，而在參考其他電路後也確認無誤(但有兩個腳位其實沒有使用到)。而程式碼的部分也依各腳位做修改。

```
// Arduino Pins
#define L293D_LEFT_EN 2
#define L293D_RIGHT_EN 3
#define L293D_LEFT_IN1 5
#define L293D_LEFT_IN2 6
#define L293D_RIGHT_IN1 7
#define L293D_RIGHT_IN2 8
#define TRIG_LEFT 0
#define TRIG_RIGHT 1
#define ECHO_LEFT 9
#define ECHO_RIGHT 10
```

Arduino D1 wifi板測試

D1板是ESP8266系列的板子之一，而我們其實並沒有使用其wifi的功能，而是做為普通Arduino板使用。然而，經過整整兩天的測試，程式依然無法上傳，因而放棄此方案。程式編譯成功，IDE裡板子的選擇也正確，然而一直遇到此問題：a fatal esptool.py error occurred: failed to connect to esp8266: timed out waiting for packet header

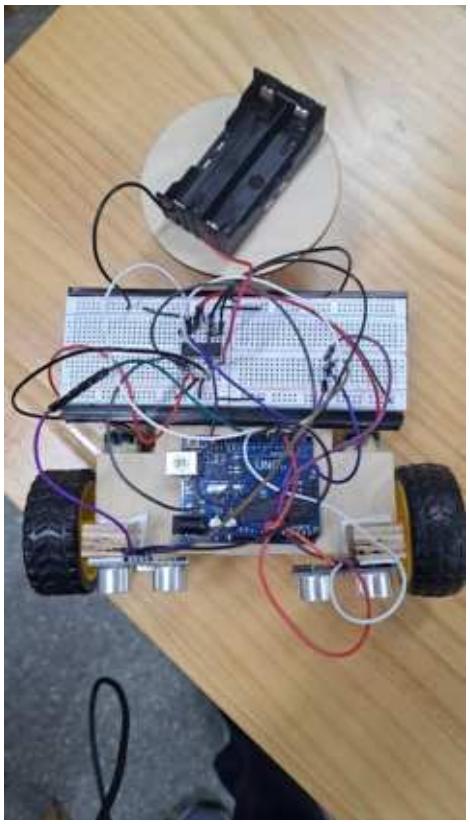
在經過查找相關網站、詢問GPT、寫簡單版本的程式嘗試燒錄並檢查所有可能的問題除錯之後，依然無法成功，可能是板子本身的問題也可能有其他問題。而以下為所有除錯，主要參考以下網站(當時沒有截圖)

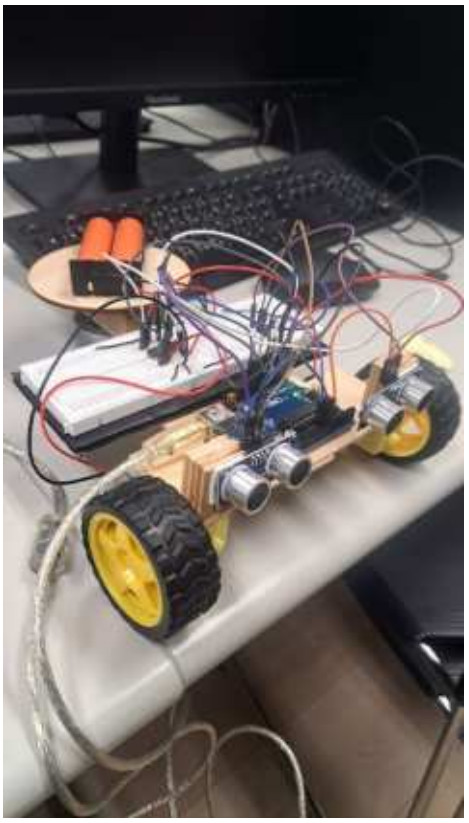
https://blog.csdn.net/weixin_65858849/article/details/131061031

1. 數據線問題：由於Arduino板上的燈有亮起，而在更早之前的測試也能燒錄，排除數據線問題的可能性
2. COM接口問題：於device manager確認Arduino裝置有在port被辨認到，而為了再次確認有卸載後再次安裝driver，確認此部分沒有問題
3. Baud率問題：於燒錄時已降低
4. 供壓不穩：以電腦接上應該無此問題
5. 串口被占用：後來將超音波偵測器的腳位都拔掉了
6. FLASH mode：這是最奇怪的部分，gpt與網站資料都說要將板子上的FLASH按鈕按下，於FLASH mode才能燒錄程式，然而再三確認後並沒有發現此按鈕(僅有RESET鍵)，可能是型號問題；而gpt給的建議是將GPIO0接至GND，而板子上也沒有標示為GPIO0的腳位，後來gpt說D1板的GPIO0為D3腳位，然而將其接至GND後也無法燒錄，最後在所有除錯皆確認並失敗之後改用UNO板。

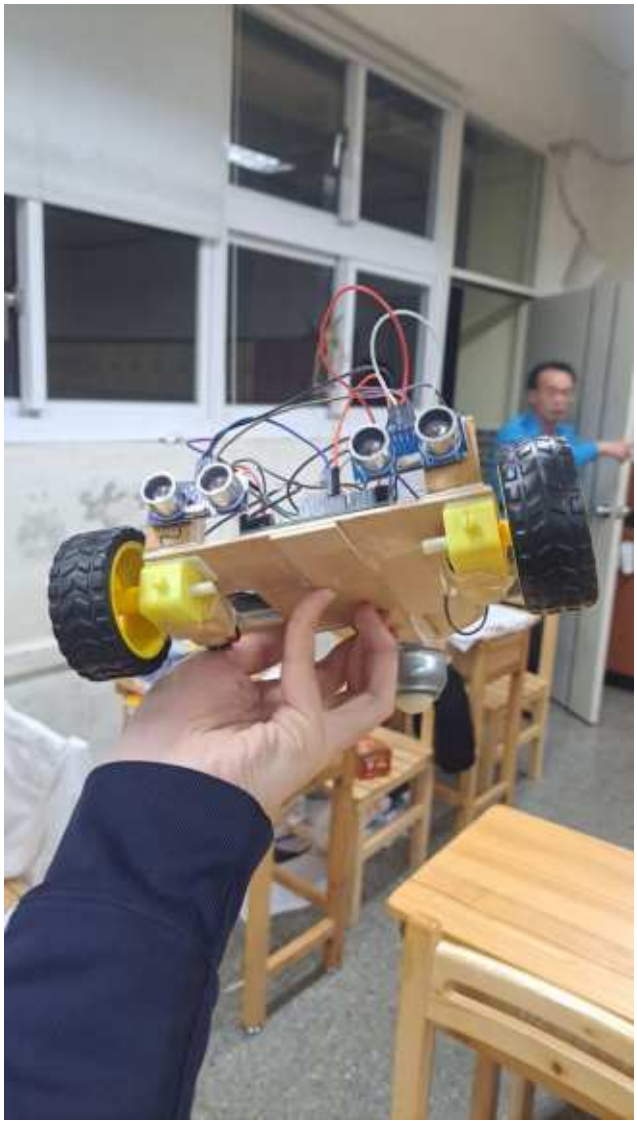
Arduino UNO R3板測試

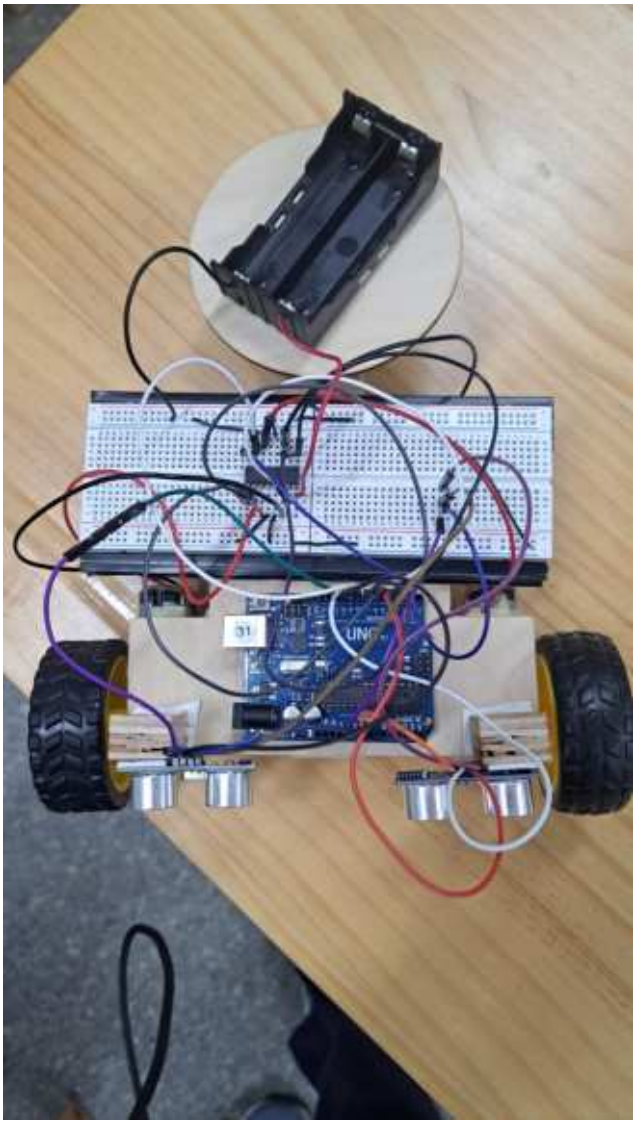
後來借到UNO板進行測試，電路以相同方式再次接線，唯一的不同是程式碼的腳位從D1等改成1。一開始燒錄依然出現問題，而查找資料後發現UNO板需要將0和1腳位的線拔掉才能燒錄，因為這是其通訊腳位。然而在燒錄成功後，超音波感測器有讀數了，在序列阜能看到超音波讀數與馬達輸出，然而馬達並沒有轉動，而以LED閃爍的測試程式上傳卻能成功運作。經過檢查接線都沒有問題，而懷疑可能是電池電壓不足或是馬達太鬆，然而充電後馬達依然沒有轉動。至此由於時間問題，已經無法再一步步除錯與測試，而就此打住了。

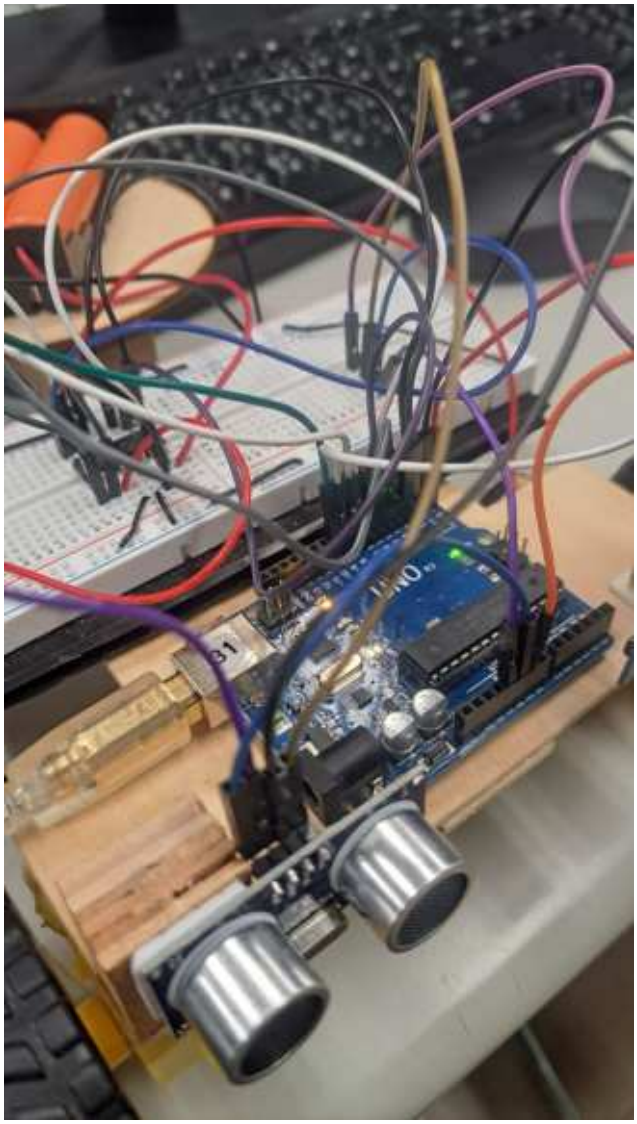


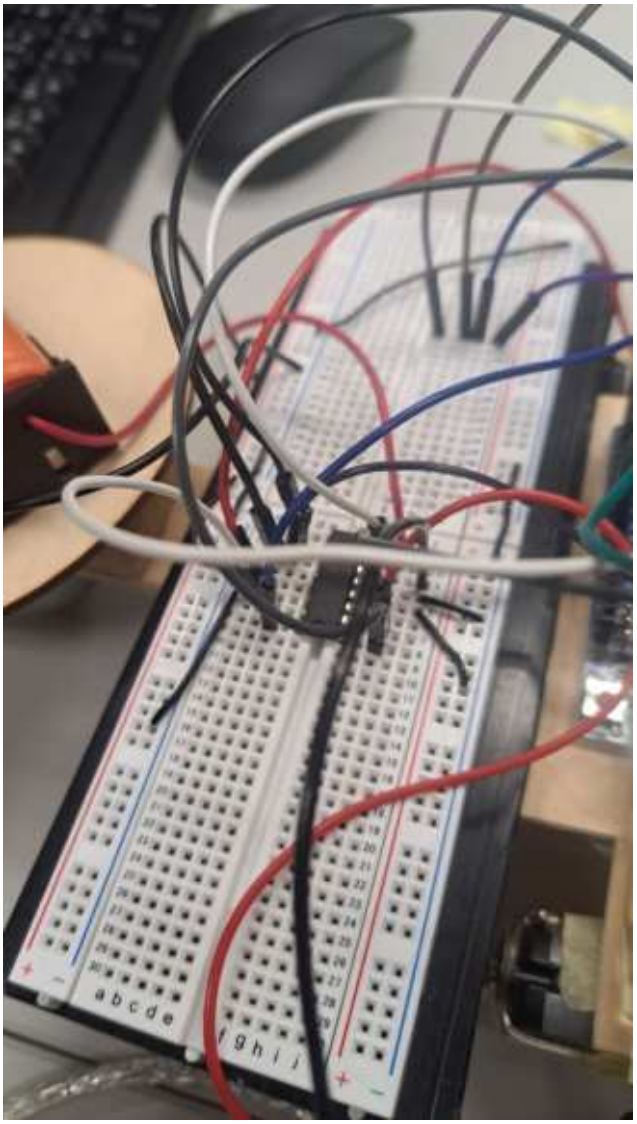


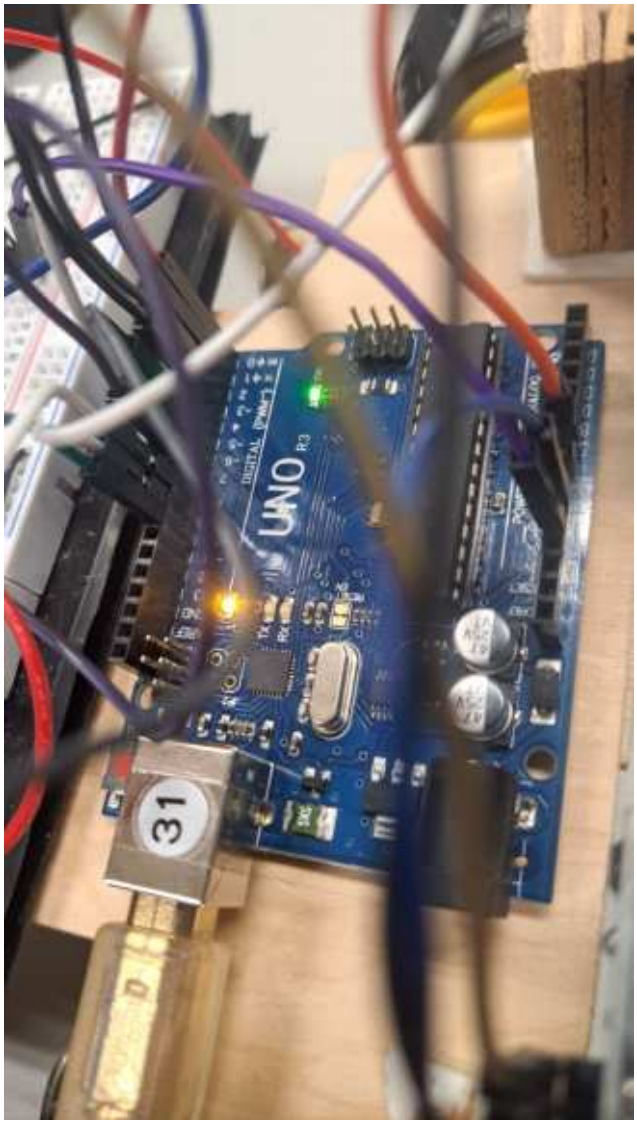
VI　零件與車體照片

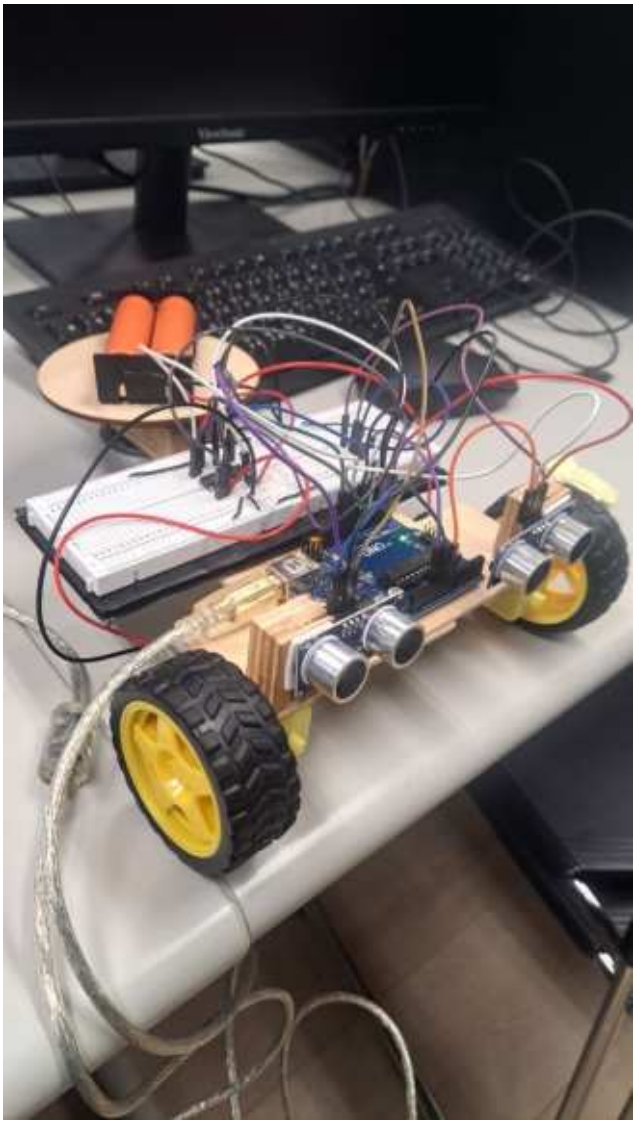












7 LaTeX Report Writing

I My LaTeX Learning Journey

My journey with LaTeX began when I realized the limitations of traditional word processors for creating professional-looking documents, especially those involving mathematical equations and technical formatting. These features are critical for my studies and research in advanced mathematics and physics.

With prior experience in programming languages such as HTML, C++, and JavaScript, I quickly grasped the basic syntax of LaTeX. I created numerous LaTeX documents on math and science, including my **tw-gifted-k12-notes** (https://github.com/Willie169/tw-gifted-k12-notes) project and many entries in my Learning Portfolio.

I expanded my knowledge of LaTeX packages and commands by consulting resources such as **Overleaf** (https://www.overleaf.com), **TeX - LaTeX Stack Exchange** (https://tex.stackexchange.com), and package documentation from **CTAN - Comprehensive TEX Archive Network** (https://ctan.org).

As I became more proficient in LaTeX, I ventured into designing templates specifically for XeTeX, with a focus on enhancing usability for CJK users.

II Template Design

When designing a template, I carefully select packages, fonts, and define custom commands to ensure compatibility with diverse document types. Below are the key aspects of my template:

- **Document Aesthetics:** Packages like `nowidow`, `titlesec`, and `titling` enable fine control over spacing, section styles, and title layouts.
- **Mathematical and Scientific Typesetting:** For advanced notations and formulae, I utilize packages such as `amsmath`, `mathtools`, `nicematrix`, `physics`, and `mhchem`.
- **Graphics and Tables:** The `graphicx`, `tikz`, `subcaption`, and `longtable` packages allow the creation of professional-quality tables and illustrations.
- **Fonts and Localization:** I integrate multilingual support with `fontspec`, `xeCJK`, and `zhnumber`, using fonts like TeX Gyre, STIX 2 Math, and Noto CJK to seamlessly handle Chinese and other non-Latin characters.
- **Customization:** Using `etoolbox`, `xparse`, and `tcolorbox`, I create reusable components and custom behaviors, such as styled boxes and command shortcuts.

The culmination of my efforts is a template available on GitHub (https://github.com/Willie169/LaTeX-ToolKit), which I have also used for this report.

III Template Preamble First Part

```
% This is template.tex of https://github.com/Willie169/LaTeX-ToolKit, licensed under either GPL-3.0-or-later, CC BY-SA 4.0-or-later, or LPPL-1.3c-or-later.
\ifnum\value{CJK}=1
\ifx\luatexversion\undefined
\usepackage{fontspec}
\usepackage[CJKspace]{xeCJK}
\else
\usepackage{luaotfload,luatexja-fontspec}
\fi
\usepackage{zhnumber}
\fi
\ifnum\value{markdown}=1
\usepackage{markdown}
\markdownSetup{autoIdentifiers=true,blankBeforeBlockquote=true,blankBeforeCodeFence=true,blankBeforeDivFence=true,blankBeforeHeading=true,blankBeforeList=true, bracketedSpans=true,citations=true,definitionLists=true,fancyLists=true,fencedCodeAttributes=true,fencedDivs=true,gfmAutoIdentifiers=true,hashEnumerators=true,headerAttributes=true,inlineCodeAttributes=true,inlineNotes=true,jekyllData=true,linkAttributes=true,lineBlocks=true,mark=true,notes=true,pipeTables=true,preserveTabs=false,rawAttribute=true,smartEllipses=true,strikeThrough=true,stripIndent=true,subscripts=true,superscripts=true,tableAttributes=true,tableCaptions=true,taskLists=true,texMathDollars=true,texMathSingleBackslash=true}
\else\ifnum\value{markdown}=2
\usepackage{markdown}
\markdownSetup{blankBeforeBlockquote=true,blankBeforeCodeFence=true,blankBeforeDivFence=true,blankBeforeHeading=true,blankBeforeList=true, pipeTables=true,preserveTabs=false,smartEllipses=true,strikeThrough=true,stripIndent=true,subscripts=true,superscripts=true,taskLists=true,texMathDollars=true,texMathSingleBackslash=true}
```

```
\else\ifnum\value{markdown}=3
\usepackage{markdown}
\markdownSetup{blankBeforeBlockquote=true,blankBeforeCodeFence=true,blankBeforeDivFence=true,
    blankBeforeHeading=true,blankBeforeList=true, preserveTabs=false,smartEllipses=true,stripIndent=true}
\else\ifnum\value{markdown}=4
\usepackage{markdown}
\markdownSetup{autoIdentifiers=true,bracketedSpans=true,citations=true,definitionLists=true,fancyLists=true,
    fencedCodeAttributes=true,fencedDivs=true,gfmAutoIdentifiers=true,hashEnumerators=true,headerAttributes=
    true,inlineCodeAttributes=true,inlineNotes=true,jekyllData=true,linkAttributes=true,lineBlocks=true,mark=
    true,notes=true,pipeTables=true,preserveTabs=false,rawAttribute=true,smartEllipses=true,strikeThrough=
    true,stripIndent=true,subscripts=true,superscripts=true,tableAttributes=true,tableCaptions=true,taskLists
    =true,texMathDollars=true,texMathSingleBackslash=true}
\else\ifnum\value{markdown}=5
\usepackage{markdown}
\markdownSetup{pipeTables=true,preserveTabs=false,smartEllipses=true,strikeThrough=true,stripIndent=true,
    subscripts=true,superscripts=true,taskLists=true,texMathDollars=true,texMathSingleBackslash=true}
\else\ifnum\value{markdown}=6
\usepackage{markdown}
\markdownSetup{preserveTabs=false,smartEllipses=true,stripIndent=true}
\ifnum\value{markdown}=7
\usepackage{markdown}
\markdownSetup{autoIdentifiers=true,blankBeforeBlockquote=true,blankBeforeCodeFence=true,blankBeforeDivFence=
    true,blankBeforeHeading=true,blankBeforeList=true, bracketedSpans=true,citations=true,definitionLists=
    true,fancyLists=true,fencedCodeAttributes=true,fencedDivs=true,gfmAutoIdentifiers=true,hashEnumerators=
    true,headerAttributes=true,inlineCodeAttributes=true,inlineNotes=true,jekyllData=true,linkAttributes=true
    ,lineBlocks=true,mark=true,notes=true,pipeTables=true,rawAttribute=true,strikeThrough=true,subscripts=
    true,superscripts=true,tableAttributes=true,tableCaptions=true,taskLists=true,texMathDollars=true,
    texMathSingleBackslash=true}
\else\ifnum\value{markdown}=8
\usepackage{markdown}
\markdownSetup{blankBeforeBlockquote=true,blankBeforeCodeFence=true,blankBeforeDivFence=true,
    blankBeforeHeading=true,blankBeforeList=true, pipeTables=true,subscripts=true,superscripts=true,taskLists
    =true,texMathDollars=true,texMathSingleBackslash=true}
\else\ifnum\value{markdown}=9
\usepackage{markdown}
\markdownSetup{blankBeforeBlockquote=true,blankBeforeCodeFence=true,blankBeforeDivFence=true,
    blankBeforeHeading=true,blankBeforeList=true}
\else\ifnum\value{markdown}=10
\usepackage{markdown}
\markdownSetup{autoIdentifiers=true,bracketedSpans=true,citations=true,definitionLists=true,fancyLists=true,
    fencedCodeAttributes=true,fencedDivs=true,gfmAutoIdentifiers=true,hashEnumerators=true,headerAttributes=
    true,inlineCodeAttributes=true,inlineNotes=true,jekyllData=true,linkAttributes=true,lineBlocks=true,mark=
    true,notes=true,pipeTables=true,rawAttribute=true,strikeThrough=true,subscripts=true,superscripts=true,
    tableAttributes=true,tableCaptions=true,taskLists=true,texMathDollars=true,texMathSingleBackslash=true}
\else\ifnum\value{markdown}=11
\usepackage{markdown}
\markdownSetup{pipeTables=true,strikeThrough=true,subscripts=true,superscripts=true,taskLists=true,
    texMathDollars=true,texMathSingleBackslash=true}
\else\ifnum\value{markdown}=12
\usepackage{markdown}
\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi
\usepackage{etoolbox,xparse,xpatch,ifthen,titlesec,titling,setspace,needspace,titletoc}
\usepackage[defaultlines=8]{nowidow}
\usepackage{amsmath,amssymb,amsthm,dsfont,esint,mathrsfs,mathtools,metalogo,microtype,stmaryrd,textcomp,
    textgreek,upgreek,wasysym,yfonts}
\usepackage{array,booktabs,caption,cellspace,diagbox,enumitem,float,graphicx,hanging,longtable,nicematrix,
    placeins,ragged2e,subcaption}
\usepackage{braket,chemfig,csvsimple,listings,listingsutf8,pdfpages,physics,tikz,xcolor}
\usepackage[version=4]{mhchem}
\usepackage[hidelinks,bookmarksnumbered=true,pdfstartview=FitH]{hyperref}
\usepackage{tcolorbox}
\tcbuselibrary{breakable,documentation,fitting,hooks,listings,listingsutf8,poster,raster,skins,theorems,
    xparse}
\usepackage{unicode-math}
\setmathfont{XITS Math}
```

```
\setmathfont[range={cal,bfcal},StylisticSet=1]{XITS Math}
\ifnum\value{Fonts}=1
\setmainfont{TeX Gyre Heros}[Ligatures=TeX]
\setsansfont{TeX Gyre Termes}[Ligatures=TeX]
\setmonofont{TeX Gyre Cursor}[Ligatures=TeX]
\else\ifnum\value{Fonts}=2
\setmainfont{TeX Gyre Termes}[Ligatures=TeX]
\setsansfont{TeX Gyre Heros}[Ligatures=TeX]
\setmonofont{TeX Gyre Cursor}[Ligatures=TeX]
\else\ifnum\value{Fonts}=3
\setmainfont{TeX Gyre Heros}[Ligatures=TeX]
\setmonofont{TeX Gyre Cursor}[Ligatures=TeX]
\else\ifnum\value{Fonts}=4
\setmainfont{TeX Gyre Termes}[Ligatures=TeX]
\setmonofont{TeX Gyre Cursor}[Ligatures=TeX]
\else\ifnum\value{Fonts}=5
\setmainfont{TeX Gyre Heros}[Ligatures=TeX]
\setsansfont{TeX Gyre Termes}[Ligatures=TeX]
\else\ifnum\value{Fonts}=6
\setmainfont{TeX Gyre Termes}[Ligatures=TeX]
\setsansfont{TeX Gyre Heros}[Ligatures=TeX]
\else\ifnum\value{Fonts}=7
\setmainfont{TeX Gyre Heros}[Ligatures=TeX]
\else\ifnum\value{Fonts}=8
\setmainfont{TeX Gyre Termes}[Ligatures=TeX]
\fi\fi\fi\fi\fi\fi\fi\fi
```

IV Template Preamble Second Part Generator C++

```
// This is luatexja.cpp of https://github.com/Willie169/LaTeX-ToolKit, licensed under either GPL-3.0-or-later, CC BY-SA 4.0-or-later, or LPPL-1.3c-or-later.
#include<iostream>
#include<string>
#include <algorithm>
#include <cctype>
using namespace std;
class Latex {
public:
Latex() {}
const string cmain = "\\setmainjfont";
const string csans = "\\setsansjfont";
const string cmono = "\\setmonojfont";
const string nff = "\n\\newfontfamily\\";
const string path = "Path=/usr/share/fonts/noto-cjk/,";
const string sansf = "NotoSans";
const string seriff = "NotoSerif";
const string monof = "NotoSansMonoCJK";
const string otf = ".otf";
const string cf = "\\ifnum\\value{CJKFonts}=";
const string cl  ="\\ifnum\\value{CJKLanguage}=";
const string cn = "\\ifnum\\value{NotoCJKFamily}=";
const string el = "\\else";
const string cp = "}[";
const string end = "Ligatures=TeX]\n";
const string endd = "Ligatures=TeX]";
const string pre = "\\ifnum\\value{CJK}=1\n\\ifx\\luatexversion\\undefined\n\\ifnum\\value{CJKFonts}=0\\else\n";
string ulang[5]={"TC","SC","HK","JP","KR"};
string llang[5]={"tc","sc","hk","jp","kr"};
string fw[5]={"UprightFont","ItalicFont","BoldFont","BoldItalicFont","SmallCapsFont"};
string sansw[5]={"Regular","Medium","Bold","Black","DemiLight"};
string serifw[5]={"Regular","Medium","Bold","Black","Light"};
string monofwn[2]={"UprightFont","BoldFont"};
```

```
string monow[2]={"Regular","Bold"};
string sans(string lang){
  string ret="{"+sansf+lang+cp+path;
  for (int i=0; i<5; i++) ret+=fw[i]+"="+sansf+lang+"-"+sansw[i]+otf+",";
  ret+=end;
  return ret;
}
string serif(string lang){
  string ret="{"+seriff+lang+cp+path;
  for (int i=0; i<5; i++) ret+=fw[i]+"="+seriff+lang+"-"+serifw[i]+otf+",";
  ret+=end;
  return ret;
}
string mono(string lang){
  string ret="{"+monof+lang+cp+path;
  for (int i=0; i<2; i++) ret+=monofwn[i]+"="+monof+lang+"-"+monow[i]+otf+",";
  ret+=end;
  return ret;
}
string ssans(string lang){
  string ret="Sans"+lang+"{"+sansf+lang+cp+path;
  for (int i=0; i<5; i++) ret+=fw[i]+"="+sansf+lang+"-"+sansw[i]+otf+",";
  ret+=endd;
  return ret;
}
string sserif(string lang){
  string ret="Serif"+lang+"{"+seriff+lang+cp+path;
  for (int i=0; i<5; i++) ret+=fw[i]+"="+seriff+lang+"-"+serifw[i]+otf+",";
  ret+=endd;
  return ret;
}
string lsans(string lang){
  string ret=lang+"{"+sansf+lang+cp+path;
  for (int i=0; i<5; i++) ret+=fw[i]+"="+sansf+lang+"-"+sansw[i]+otf+",";
  ret+=endd;
  return ret;
}
string lserif(string lang){
  string ret=lang+"{"+seriff+lang+cp+path;
  for (int i=0; i<5; i++) ret+=fw[i]+"="+seriff+lang+"-"+serifw[i]+otf+",";
  ret+=endd;
  return ret;
}
string smono(string ulang){
  string lang = ulang;
  transform(lang.begin(), lang.end(), lang.begin(), [](unsigned char c) { return tolower(c); });
  string ret="Mono"+ulang+"{"+monof+lang+cp+path;
  for (int i=0; i<2; i++) ret+=monofwn[i]+"="+monof+lang+"-"+monow[i]+otf+",";
  ret+=endd;
  return ret;
}
void yes() {
  cout << pre;
  for (int i=0; i<5; i++) {
    cout << cl+to_string(i)+"\n";
    cout << cf+"1"+"\n";
    cout << cmain << sans(ulang[i]);
    cout << csans << serif(ulang[i]);
    cout << cmono << mono(llang[i]);
    cout << "\\else";
    cout << cf+"2"+"\n";
    cout << cmain << serif(ulang[i]);
    cout << csans << sans(ulang[i]);
    cout << cmono << mono(llang[i]);
    cout << "\\else";
```

```
    cout << cf+"3"+"\n";
    cout << cmain << sans(ulang[i]);
    cout << cmono << mono(llang[i]);
    cout << "\\else";
    cout << cf+"4"+"\n";
    cout << cmain << serif(ulang[i]);
    cout << cmono << mono(llang[i]);
    cout << "\\else";
    cout << cf+"5"+"\n";
    cout << cmain << sans(ulang[i]);
    cout << csans << serif(ulang[i]);
    cout << "\\else";
    cout << cf+"6"+"\n";
    cout << cmain << serif(ulang[i]);
    cout << csans << sans(ulang[i]);
    cout << "\\else";
    cout << cf+"7"+"\n";
    cout << cmain << sans(ulang[i]);
    cout << "\\else";
    cout << cf+"8"+"\n";
    cout << cmain << serif(ulang[i]);
    cout << "\\fi\\fi\\fi\\fi\\fi\\fi\\fi\\fi";
    if (i!=4) cout << "\\else\n";
  }
  cout << "\\fi\\fi\\fi\\fi\\fi\\fi\n";
  cout << cn << "0\n";
  cout << "\\else" << cn << "1";
  for (int i=0; i<5; i++) {
    cout << nff << ssans(ulang[i]);
    cout << nff << lsans(ulang[i]);
    cout << nff << sserif(ulang[i]);
    cout << nff << smono(ulang[i]);
  }
  cout << "\\else" << cn << "2";
  for (int i=0; i<5; i++) {
    cout << nff << ssans(ulang[i]);
    cout << nff << sserif(ulang[i]);
    cout << nff << lserif(ulang[i]);
    cout << nff << smono(ulang[i]);
  }
  cout << "\\else" << cn << "3";
  for (int i=0; i<5; i++) {
    cout << nff << ssans(ulang[i]);
    cout << nff << sserif(ulang[i]);
    cout << nff << smono(ulang[i]);
  }
  cout << "\\else" << cn << "4";
  for (int i=0; i<5; i++) {
    cout << nff << ssans(ulang[i]);
    cout << nff << lsans(ulang[i]);
    cout << nff << sserif(ulang[i]);
  }
  cout << "\\else" << cn << "5";
  for (int i=0; i<5; i++) {
    cout << nff << ssans(ulang[i]);
    cout << nff << sserif(ulang[i]);
    cout << nff << lserif(ulang[i]);
  }
  cout << "\\else" << cn << "6";
  for (int i=0; i<5; i++) {
    cout << nff << ssans(ulang[i]);
    cout << nff << sserif(ulang[i]);
  }
  cout << "\n\\fi\\fi\\fi\\fi\\fi\\fi\n";
}
```

```
};
int main() {
  Latex latex=Latex();
  latex.yes();
  return 0;
}
```

V Template Preamble Third Part Generator C++

```
// This is xeCJK.cpp of https://github.com/Willie169/LaTeX-ToolKit, licensed under either GPL-3.0-or-later or CC BY-SA 4.0-or-later.
#include<iostream>
#include<string>
#include <algorithm>
#include <cctype>
using namespace std;
class Latex {
public:
Latex() {}
const string cmain = "\\setCJKmainfont";
const string csans = "\\setCJKsansfont";
const string cmono = "\\setCJKmonofont";
const string nff = "\n\\newfontfamily\\";
const string path = "Path=/usr/share/fonts/noto-cjk/,";
const string sansf = "NotoSans";
const string seriff = "NotoSerif";
const string monof = "NotoSansMonoCJK";
const string otf = ".otf";
const string cf = "\\ifnum\\value{CJKFonts}=";
const string cl  ="\\ifnum\\value{CJKLanguage}=";
const string cn = "\\ifnum\\value{NotoCJKFamily}=";
const string el = "\\else";
const string cp = "}[";
const string end = "Ligatures=TeX]\n";
const string endd = "Ligatures=TeX]";
const string pre = "\\else\n\\ifnum\\value{CJKFonts}=0\\else\n";
string ulang[5]={"TC","SC","HK","JP","KR"};
string llang[5]={"tc","sc","hk","jp","kr"};
string fw[5]={"UprightFont","ItalicFont","BoldFont","BoldItalicFont","SmallCapsFont"};
string sansw[5]={"Regular","Medium","Bold","Black","DemiLight"};
string serifw[5]={"Regular","Medium","Bold","Black","Light"};
string monofwn[2]={"UprightFont","BoldFont"};
string monow[2]={"Regular","Bold"};
string sans(string lang){
  string ret="{"+sansf+lang+cp+path;
  for (int i=0; i<5; i++) ret+=fw[i]+"="+sansf+lang+"-"+sansw[i]+otf+",";
  ret+=end;
  return ret;
}
string serif(string lang){
  string ret="{"+seriff+lang+cp+path;
  for (int i=0; i<5; i++) ret+=fw[i]+"="+seriff+lang+"-"+serifw[i]+otf+",";
  ret+=end;
  return ret;
}
string mono(string lang){
  string ret="{"+monof+lang+cp+path;
  for (int i=0; i<2; i++) ret+=monofwn[i]+"="+monof+lang+"-"+monow[i]+otf+",";
  ret+=end;
  return ret;
}
string ssans(string lang){
  string ret="Sans"+lang+"{"+sansf+lang+cp+path;
```

```
  for (int i=0; i<5; i++) ret+=fw[i]+"="+sansf+lang+"-"+sansw[i]+otf+",";
  ret+=endd;
  return ret;
}
string sserif(string lang){
  string ret="Serif"+lang+"{"+seriff+lang+cp+path;
  for (int i=0; i<5; i++) ret+=fw[i]+"="+seriff+lang+"-"+serifw[i]+otf+",";
  ret+=endd;
  return ret;
}
string lsans(string lang){
  string ret=lang+"{"+sansf+lang+cp+path;
  for (int i=0; i<5; i++) ret+=fw[i]+"="+sansf+lang+"-"+sansw[i]+otf+",";
  ret+=endd;
  return ret;
}
string lserif(string lang){
  string ret=lang+"{"+seriff+lang+cp+path;
  for (int i=0; i<5; i++) ret+=fw[i]+"="+seriff+lang+"-"+serifw[i]+otf+",";
  ret+=endd;
  return ret;
}
string smono(string ulang){
  string lang = ulang;
  transform(lang.begin(), lang.end(), lang.begin(), [](unsigned char c) { return tolower(c); });
  string ret="Mono"+ulang+"{"+monof+lang+cp+path;
  for (int i=0; i<2; i++) ret+=monofwn[i]+"="+monof+lang+"-"+monow[i]+otf+",";
  ret+=endd;
  return ret;
}
void yes() {
  cout << pre;
  for (int i=0; i<5; i++) {
    cout << cl+to_string(i)+"\n";
    cout << cf+"1"+"\n";
    cout << cmain << sans(ulang[i]);
    cout << csans << serif(ulang[i]);
    cout << cmono << mono(llang[i]);
    cout << "\\else";
    cout << cf+"2"+"\n";
    cout << cmain << serif(ulang[i]);
    cout << csans << sans(ulang[i]);
    cout << cmono << mono(llang[i]);
    cout << "\\else";
    cout << cf+"3"+"\n";
    cout << cmain << sans(ulang[i]);
    cout << cmono << mono(llang[i]);
    cout << "\\else";
    cout << cf+"4"+"\n";
    cout << cmain << serif(ulang[i]);
    cout << cmono << mono(llang[i]);
    cout << "\\else";
    cout << cf+"5"+"\n";
    cout << cmain << sans(ulang[i]);
    cout << csans << serif(ulang[i]);
    cout << "\\else";
    cout << cf+"6"+"\n";
    cout << cmain << serif(ulang[i]);
    cout << csans << sans(ulang[i]);
    cout << "\\else";
    cout << cf+"7"+"\n";
    cout << cmain << sans(ulang[i]);
    cout << "\\else";
    cout << cf+"8"+"\n";
    cout << cmain << serif(ulang[i]);
```

```
    cout << "\\fi\\fi\\fi\\fi\\fi\\fi\\fi\\fi";
    if (i!=4) cout << "\\else\n";
  }
  cout << "\\fi\\fi\\fi\\fi\\fi\\fi\n";
  cout << cn << "0\n";
  cout << "\\else" << cn << "1";
  for (int i=0; i<5; i++) {
    cout << nff << ssans(ulang[i]);
    cout << nff << lsans(ulang[i]);
    cout << nff << sserif(ulang[i]);
    cout << nff << smono(ulang[i]);
  }
  cout << "\\else" << cn << "2";
  for (int i=0; i<5; i++) {
    cout << nff << ssans(ulang[i]);
    cout << nff << sserif(ulang[i]);
    cout << nff << lserif(ulang[i]);
    cout << nff << smono(ulang[i]);
  }
  cout << "\\else" << cn << "3";
  for (int i=0; i<5; i++) {
    cout << nff << ssans(ulang[i]);
    cout << nff << sserif(ulang[i]);
    cout << nff << smono(ulang[i]);
  }
  cout << "\\else" << cn << "4";
  for (int i=0; i<5; i++) {
    cout << nff << ssans(ulang[i]);
    cout << nff << lsans(ulang[i]);
    cout << nff << sserif(ulang[i]);
  }
  cout << "\\else" << cn << "5";
  for (int i=0; i<5; i++) {
    cout << nff << ssans(ulang[i]);
    cout << nff << sserif(ulang[i]);
    cout << nff << lserif(ulang[i]);
  }
  cout << "\\else" << cn << "6";
  for (int i=0; i<5; i++) {
    cout << nff << ssans(ulang[i]);
    cout << nff << sserif(ulang[i]);
  }
  cout << "\n\\fi\\fi\\fi\\fi\\fi\\fi\\fi\n\\fi\\fi\n";
}
};
int main() {
  Latex latex=Latex();
  latex.yes();
  return 0;
}
```

VI Template Preamble Forth Part

```
\righthyphenmin=4
\lefthyphenmin=4
\pretolerance=5000
\tolerance=9000
\emergencystretch=1em
\setlength{\parindent}{0pt}
\everymath{\displaystyle}
\everydisplay{\displaystyle}
\captionsetup{labelformat=empty}
\renewcommand{\maketitle}{
```

```
\begin{titlepage}
\begin{center}
\vspace*{\fill}
{\huge \bfseries \thetitle\par}
\vskip 1.5em
{\Large \theauthor\par}
\vskip 1em
{\large \thedate\par}
\vspace*{\fill}
\end{center}
\end{titlepage}
}
\lstset{
basicstyle=\ttfamily\scriptsize,
keywordstyle=\color{blue},
stringstyle=\color{red},
commentstyle=\color{green!50!black},
numberstyle=\small\color{gray},
numbers=left,
stepnumber=1,
numbersep=5pt,
backgroundcolor=\color{white},
showspaces=false,
showstringspaces=false,
showtabs=false,
frame=single,
rulecolor=\color{black},
tabsize=4,
captionpos=b,
breaklines=true,
breakatwhitespace=false
}
\makeatletter\csvset{autotabularcenter/.style={file=#1,after head=\csv@pretable\begin{tabular}{|*{\
    csv@columncount}{c|}}\csv@tablehead,table head=\hline\csvlinetotablerow\\\hline,late after line=\\,lable
    foot=\\\hline,late after last line=\csv@tablefoot\end{tabular}\csv@posttable,command=\csvlinetotablerow},
    autobooktabularcenter/.style={file=#1,after head=\csv@pretable\begin{tabular}{*{\csv@columncount}{c}}\
    csv@tablehead,table head=\toprule\csvlinetotablerow\\\midrule,late after line=\\,table foot=\\\bottomrule
    ,late after last line=\csv@tablefoot\end{tabular}\csv@posttable,command=\csvlinetotablerow}}\makeatother
\newcommand{\csvautotabularcenter}[2][]{\csvloop{autotabularcenter={#2},#1}}
\newcommand{\csvautobooktabularcenter}[2][]{\csvloop{autobooktabularcenter={#2},#1}}
\newtheorem{theorem}{Theorem}
\newtheorem{lemma}{Lemma}
\makeatletter
\newcommand{\overleftrightarrowsmall}{\mathpalette{\overarrowsmall@\leftrightarrowfill@}}
\newcommand{\overrightarrowsmall}{\mathpalette{\overarrowsmall@\rightarrowfill@}}
\newcommand{\overleftarrowsmall}{\mathpalette{\overarrowsmall@\leftarrowfill@}}
\newcommand{\overarrowsmall@}[3]{\vbox{\ialign{##\crcr#1{\smaller@style{#2}}\crcr\noalign{\nointerlineskip}$\
    m@th\hfil#2#3\hfil$\crcr}}}\def\smaller@style#1{\ifx#1\displaystyle\scriptstyle\else\ifx#1\textstyle\
    scriptstyle\else\scriptscriptstyle\fi\fi}
\makeatother
\newcommand{\overlinesmall}[1]{\mskip.5\thinmuskip\overline{\mskip-.5\thinmuskip{#1}\mskip-.5\thinmuskip}\
    mskip.5\thinmuskip}
\newcommand{\underlinesmall}[1]{\mskip.5\thinmuskip\underline{\mskip-.5\thinmuskip{#1}\mskip-.5\thinmuskip}\
    mskip.5\thinmuskip}
\let\olra\overleftrightarrowsmall
\let\ora\overrightarrowsmall
\let\ola\overleftarrowsmall
\let\ol\overlinesmall
\let\ul\underlinesmall
\let\olras\overleftrightarrowsmall
\let\oras\overrightarrowsmall
\let\olas\overleftarrowsmall
\let\ols\overlinesmall
\let\uls\underlinesmall
\ProvideDocumentCommand{\parallelsum}{}{\mathbin{\!/\mkern-5mu/\!}}
```

```
\ProvideDocumentCommand{\TextCircled}{m}{\text{\raisebox{.6pt}{\textcircled{\raisebox{-.9pt}{#1}}}}}
\makeatletter\tikzset{dot diameter/.store in=\dot@diameter,dot diameter=1pt,dot spacing/.store in=\
    dot@spacing,dot spacing=5.5pt,dots/.style={line width=\dot@diameter,line cap=round,dash pattern=on 0pt
    off \dot@spacing}}\makeatother
\let\FB\FloatBarrier
\let\La\Leftarrow
\let\Lar\Leftarrow
\let\Lra\Leftrightarrow
\let\Lrar\Leftrightarrow
\let\Ns\Needspace
\let\Ra\Rightarrow
\let\Rar\Rightarrow
\let\amp\&
\let\apx\approx
\let\cd\cdot
\let\cds\cdots
\let\chm\chemfig
\let\clp\clearpage
\let\cpt\caption
\let\ctr\centering
\let\da\downarrow
\let\dar\downarrow
\let\dds\ddots
\let\dgb\diagbox
\let\f\frac
\let\fr\frac
\let\hf\hfill
\let\hs\hspace
\let\icg\includegraphics
\let\icp\includepdf
\let\ift\infty
\let\ip\input
\let\la\leftarrow
\let\lar\leftarrow
\let\lds\ldots
\let\lh\lineheight
\let\lra\leftrightarrow
\let\lrar\leftrightarrow
\let\lrh\leftrightharpoons
\let\lw\linewidth
\let\mb\mathbf
\let\mbb\mathbb
\let\mbbi\mathbbit
\let\mbbit\mathbbit
\let\mbc\mathbfcal
\let\mbf\mathbf
\let\mbfcal\mathbfcal
\let\mbffrak\mathbffrak
\let\mbfit\mathbfit
\let\mbfk\mathbffrak
\let\mbfscr\mathbfscr
\let\mbi\mathbfit
\let\mbs\mathbfscr
\let\mc\mathcal
\let\mcal\mathcal
\let\mds\mathds
\let\mfk\mathfrak
\let\mfrak\mathfrak
\let\mi\mathit
\let\mit\mathit
\let\mnm\mathnormal
\let\mr\mathrm
\let\ms\mathscr
\let\mscr\mathscr
\let\msf\mathsf
```

```
\let\mtt\mathtt
\let\mup\mathup
\let\npgb\nopagebreak
\let\opn\operatorname
\let\pc\%
\let\pgb\pagebreak
\let\pk\textperthousand
\let\prp\propto
\let\ra\rightarrow
\let\rar\rightarrow
\let\rl\relax
\let\rlh\rightleftharpoons
\let\rsb\raisebox
\let\rtb\rotatebox
\let\sq\sqrt
\let\tb\textbf
\let\tc\TextCircled
\let\tcb\tcolorbox
\let\tfr\tfrac
\let\th\textheight
\let\ti\textit
\let\tnm\textnormal
\let\tr\textrm
\let\tsc\textsc
\let\tsf\textsf
\let\ttt\texttt
\let\tup\textup
\let\tw\textwidth
\let\tx\text
\let\ua\uparrow
\let\uar\uparrow
\let\vds\vdots
\let\vf\vfill
\let\vs\vspace
\ProvideDocumentCommand{\Arg}{}{\operatorname{Arg}}
\ProvideDocumentCommand{\E}{m}{\relax\ifmmode 10^{#1}\else\(10^{#1}\)\fi}
\ProvideDocumentCommand{\RNum}{m}{\uppercase\expandafter{\romannumeral #1\relax}}
\ProvideDocumentCommand{\XE}{m}{\relax\ifmmode\times 10^{#1}\else\(\times 10^{#1}\)\fi}
\ProvideDocumentCommand{\X}{}{\relax\ifmmode\times\else\(\times\)\fi}
\ProvideDocumentCommand{\acosh}{}{\operatorname{acosh}}
\ProvideDocumentCommand{\acos}{}{\operatorname{acos}}
\ProvideDocumentCommand{\acsch}{}{\operatorname{acsch}}
\ProvideDocumentCommand{\acsc}{}{\operatorname{acsc}}
\ProvideDocumentCommand{\arccosh}{}{\operatorname{arccosh}}
\ProvideDocumentCommand{\arccsch}{}{\operatorname{arccsch}}
\ProvideDocumentCommand{\arccsc}{}{\operatorname{arccsc}}
\ProvideDocumentCommand{\arcsech}{}{\operatorname{arcsech}}
\ProvideDocumentCommand{\arcsec}{}{\operatorname{arcsec}}
\ProvideDocumentCommand{\arcsinh}{}{\operatorname{arcsinh}}
\ProvideDocumentCommand{\arctanh}{}{\operatorname{arctanh}}
\ProvideDocumentCommand{\arctantwo}{}{\operatorname{arctan2}}
\ProvideDocumentCommand{\asech}{}{\operatorname{asech}}
\ProvideDocumentCommand{\asec}{}{\operatorname{asec}}
\ProvideDocumentCommand{\asinh}{}{\operatorname{asinh}}
\ProvideDocumentCommand{\asin}{}{\operatorname{asin}}
\ProvideDocumentCommand{\atanh}{}{\operatorname{atanh}}
\ProvideDocumentCommand{\atantwo}{}{\operatorname{atan2}}
\ProvideDocumentCommand{\atan}{}{\operatorname{atan}}
\ProvideDocumentCommand{\bBNM}{}{\begin{BNiceMatrix}}
\ProvideDocumentCommand{\bBmt}{}{\begin{Bmatrix}}
\ProvideDocumentCommand{\bNM}{}{\begin{NiceMatrix}}
\ProvideDocumentCommand{\bVNM}{}{\begin{VNiceMatrix}}
\ProvideDocumentCommand{\bVmt}{}{\begin{Vmatrix}}
\ProvideDocumentCommand{\ba}{}{\begin{aligned}}
\ProvideDocumentCommand{\bbNM}{}{\begin{bNiceMatrix}}
```

```
\ProvideDocumentCommand{\bbmt}{}{\begin{bmatrix}}
\ProvideDocumentCommand{\bcs}{}{\begin{cases}}
\ProvideDocumentCommand{\bcttbrn}{}{\begin{center}\begin{figure}[h]\centering\begin{tabular}}
\ProvideDocumentCommand{\bcttbr}{}{\begin{center}\begin{figure}[H]\centering\begin{tabular}}
\ProvideDocumentCommand{\bctfn}{}{\begin{center}\begin{figure}[h]\centering}
\ProvideDocumentCommand{\bctf}{}{\begin{center}\begin{figure}[H]\centering}
\ProvideDocumentCommand{\bcttn}{}{\begin{center}\begin{table}[h]\centering}
\ProvideDocumentCommand{\bctt}{}{\begin{center}\begin{table}[H]\centering}
\ProvideDocumentCommand{\bct}{}{\begin{center}}
\ProvideDocumentCommand{\ben}{}{\begin{enumerate}}
\ProvideDocumentCommand{\beq}{}{\begin{equation}}
\ProvideDocumentCommand{\bfH}{}{\begin{figure}[H]}
\ProvideDocumentCommand{\bf}{}{\begin{figure}}
\ProvideDocumentCommand{\bg}{m}{\begin{#1}}
\ProvideDocumentCommand{\bit}{}{\begin{itemize}}
\ProvideDocumentCommand{\blt}{}{\begin{longtable}}
\ProvideDocumentCommand{\bmac}{}{\[\begin{aligned}\begin{cases}}
\ProvideDocumentCommand{\bma}{}{\[\begin{aligned}}
\ProvideDocumentCommand{\bmc}{}{\[\begin{cases}}
\ProvideDocumentCommand{\bmt}{}{\begin{matrix}}
\ProvideDocumentCommand{\bm}{}{\[}
\ProvideDocumentCommand{\bpNM}{}{\begin{pNiceMatrix}}
\ProvideDocumentCommand{\bpmt}{}{\begin{pmatrix}}
\ProvideDocumentCommand{\bsh}{}{\relax\ifmmode\backslash\else\textbackslash\fi}
\ProvideDocumentCommand{\btH}{}{\begin{table}[H]}
\ProvideDocumentCommand{\btbr}{}{\begin{tabular}}
\ProvideDocumentCommand{\btk}{}{\begin{tikzpicture}}
\ProvideDocumentCommand{\bt}{}{\begin{table}}
\ProvideDocumentCommand{\bvNM}{}{\begin{vNiceMatrix}}
\ProvideDocumentCommand{\bvmt}{}{\begin{vmatrix}}
\ProvideDocumentCommand{\closure}{}{\operatorname{closure}}
\ProvideDocumentCommand{\col}{}{\operatorname{col}}
\ProvideDocumentCommand{\ctgn}{o m o}{\begin{center}\begin{figure}[h]\centering\IfNoValueTF{#3}{\IfNoValueTF
    {#2}{\includegraphics{#1}}{\includegraphics[#1]{#2}}}{\includegraphics[#1]{#2}\caption{#3}}\end{figure}\
    end{center}}
\ProvideDocumentCommand{\ctg}{o m o}{\begin{center}\begin{figure}[H]\centering\IfNoValueTF{#3}{\IfNoValueTF
    {#2}{\includegraphics{#1}}{\includegraphics[#1]{#2}}}{\includegraphics[#1]{#2}\caption{#3}}\end{figure}\
    FloatBarrier\end{center}}
\ProvideDocumentCommand{\df}{m m o}{\IfNoValueTF{#3}{\frac{\mathrm{d}#1}{\mathrm{d}#2}}{\frac{\mathrm{d
    }^{#1}#2}{\mathrm{d}#3^{#1}}}}
\ProvideDocumentCommand{\dpre}{m m o}{\IfNoValueTF{#3}{\frac{\mathrm{d}}{\mathrm{d}#2}\left(#1\right)}{\frac
    {\mathrm{d}^{#1}}{\mathrm{d}#3^{#1}}\left(#2\right)}}
\ProvideDocumentCommand{\eBNM}{}{\end{BNiceMatrix}}
\ProvideDocumentCommand{\eBmt}{}{\end{Bmatrix}}
\ProvideDocumentCommand{\eNM}{}{\end{NiceMatrix}}
\ProvideDocumentCommand{\eVNM}{}{\end{VNiceMatrix}}
\ProvideDocumentCommand{\eVmt}{}{\end{Vmatrix}}
\ProvideDocumentCommand{\eam}{}{\end{aligned}\]}
\ProvideDocumentCommand{\ea}{}{\end{aligned}}
\ProvideDocumentCommand{\ebNM}{}{\end{bNiceMatrix}}
\ProvideDocumentCommand{\ebmt}{}{\end{bmatrix}}
\ProvideDocumentCommand{\ecam}{}{\end{cases}\end{aligned}\]}
\ProvideDocumentCommand{\ecm}{}{\end{cases}\]}
\ProvideDocumentCommand{\ecs}{}{\end{cases}}
\ProvideDocumentCommand{\ect}{}{\end{center}}
\ProvideDocumentCommand{\ed}{m}{\end{#1}}
\ProvideDocumentCommand{\eit}{}{\end{itemize}}
\ProvideDocumentCommand{\een}{}{\end{enumerate}}
\ProvideDocumentCommand{\eeq}{}{\end{equation}}
\ProvideDocumentCommand{\efB}{}{\end{figure}\FloatBarrier}
\ProvideDocumentCommand{\efctn}{}{\end{figure}\end{center}}
\ProvideDocumentCommand{\efct}{}{\end{figure}\FloatBarrier\end{center}}
\ProvideDocumentCommand{\ef}{}{\end{figure}}
\ProvideDocumentCommand{\elt}{}{\end{longtable}}
\ProvideDocumentCommand{\emt}{}{\end{matirx}}
```

```
\ProvideDocumentCommand{\em}{}{\]}
\ProvideDocumentCommand{\epNM}{}{\end{pNiceMatrix}}
\ProvideDocumentCommand{\epmt}{}{\end{pmatrix}}
\ProvideDocumentCommand{\eq}{}{=}
\ProvideDocumentCommand{\etB}{}{\end{table}\FloatBarrier}
\ProvideDocumentCommand{\etbr}{}{\end{tabular}}
\ProvideDocumentCommand{\etbrctn}{}{\end{tabular}\end{figure}\end{center}}
\ProvideDocumentCommand{\etbrct}{}{\end{tabular}\end{figure}\FloatBarrier\end{center}}
\ProvideDocumentCommand{\etctn}{}{\end{table}\end{center}}
\ProvideDocumentCommand{\etct}{}{\end{table}\FloatBarrier\end{center}}
\ProvideDocumentCommand{\etk}{}{\end{tikzpicture}}
\ProvideDocumentCommand{\et}{}{\end{table}}
\ProvideDocumentCommand{\evNM}{}{\end{vNiceMatrix}}
\ProvideDocumentCommand{\evmt}{}{\end{vmatrix}}
\ProvideDocumentCommand{\gt}{}{>}
\ProvideDocumentCommand{\itg}{m m o o}{\IfNoValueTF{#4}{\IfNoValueTF{#3}{\int #1\,\mathrm{d}{#2}}{\int
    _0^{#1}#2\,\mathrm{d}{#3}}}{\int_{#1}^{#2} #3\,\mathrm{d}{#4}}}
\ProvideDocumentCommand{\lt}{}{<}
\ProvideDocumentCommand{\mdp}{O{}}{\[#1\]}
\ProvideDocumentCommand{\mil}{O{}}{\(#1\)}
\ProvideDocumentCommand{\mn}{}{-}
\ProvideDocumentCommand{\nthm}{}{\Needspace{1\textheight-1em}}
\ProvideDocumentCommand{\nth}{o}{\IfNoValueTF{#1}{\Needspace{1\textheight}}{\Needspace{#1\textheight}}}
\ProvideDocumentCommand{\null}{}{\operatorname{null}}
\ProvideDocumentCommand{\n}{}{\\}
\ProvideDocumentCommand{\pf}{m m o}{\IfNoValueTF{#3}{\frac{\partial #1}{\partial #2}}{\frac{\partial^{#1}
    #2}{\partial #3^{#1}}}}
\ProvideDocumentCommand{\pht}{m}{\phantom{#1}}
\ProvideDocumentCommand{\pl}{}{+}
\ProvideDocumentCommand{\ppre}{m m o}{\IfNoValueTF{#3}{\partial #2 \left( #1 \right)}{\frac{\partial^{#1}}{\
    partial #3^{#1}} \left( #2 \right)}}
\ProvideDocumentCommand{\rank}{}{\operatorname{rank}}
\ProvideDocumentCommand{\sh}{}{/}
\ProvideDocumentCommand{\ts}{o}{\IfNoValueTF{#1}{\text{\protect\ }}{\text{\protect\foreach\i in{1,2,...,#1}{\
    protect\ }}}}
\ProvideDocumentCommand{\ttimes}{}{\(\times\)}
\ProvideDocumentCommand{\vsf}{}{\vspace*{\fill}}
\ProvideDocumentCommand{\tAlpha}{O{}}{\text{\textAlpha}}
\ProvideDocumentCommand{\tBeta}{O{}}{\text{\textBeta}}
\ProvideDocumentCommand{\tGamma}{O{}}{\text{\textGamma}}
\ProvideDocumentCommand{\tDelta}{O{}}{\text{\textDelta}}
\ProvideDocumentCommand{\tEpsilon}{O{}}{\text{\textEpsilon}}
\ProvideDocumentCommand{\tZeta}{O{}}{\text{\textZeta}}
\ProvideDocumentCommand{\tEta}{O{}}{\text{\textEta}}
\ProvideDocumentCommand{\tTheta}{O{}}{\text{\textTheta}}
\ProvideDocumentCommand{\tIota}{O{}}{\text{\textIota}}
\ProvideDocumentCommand{\tKappa}{O{}}{\text{\textKappa}}
\ProvideDocumentCommand{\tLambda}{O{}}{\text{\textLambda}}
\ProvideDocumentCommand{\tMu}{O{}}{\text{\textMu}}
\ProvideDocumentCommand{\tNu}{O{}}{\text{\textNu}}
\ProvideDocumentCommand{\tXi}{O{}}{\text{\textXi}}
\ProvideDocumentCommand{\tOmicron}{O{}}{\text{\textOmicron}}
\ProvideDocumentCommand{\tPi}{O{}}{\text{\textPi}}
\ProvideDocumentCommand{\tRho}{O{}}{\text{\textRho}}
\ProvideDocumentCommand{\tSigma}{O{}}{\text{\textSigma}}
\ProvideDocumentCommand{\tTau}{O{}}{\text{\textTau}}
\ProvideDocumentCommand{\tUpsilon}{O{}}{\text{\textUpsilon}}
\ProvideDocumentCommand{\tPhi}{O{}}{\text{\textPhi}}
\ProvideDocumentCommand{\tChi}{O{}}{\text{\textChi}}
\ProvideDocumentCommand{\tPsi}{O{}}{\text{\textPsi}}
\ProvideDocumentCommand{\tOmega}{O{}}{\text{\textOmega}}
\ProvideDocumentCommand{\talpha}{O{}}{\text{\textalpha}}
\ProvideDocumentCommand{\tbeta}{O{}}{\text{\textbeta}}
\ProvideDocumentCommand{\tgamma}{O{}}{\text{\textgamma}}
\ProvideDocumentCommand{\tdelta}{O{}}{\text{\textdelta}}
```

```
\ProvideDocumentCommand{\tepsilon}{O{}}{\text{\textepsilon}}
\ProvideDocumentCommand{\tzeta}{O{}}{\text{\textzeta}}
\ProvideDocumentCommand{\teta}{O{}}{\text{\texteta}}
\ProvideDocumentCommand{\ttheta}{O{}}{\text{\texttheta}}
\ProvideDocumentCommand{\tiota}{O{}}{\text{\textiota}}
\ProvideDocumentCommand{\tkappa}{O{}}{\text{\textkappa}}
\ProvideDocumentCommand{\tlambda}{O{}}{\text{\textlambda}}
\ProvideDocumentCommand{\tmu}{O{}}{\text{\textmu}}
\ProvideDocumentCommand{\tnu}{O{}}{\text{\textnu}}
\ProvideDocumentCommand{\txi}{O{}}{\text{\textxi}}
\ProvideDocumentCommand{\tomicron}{O{}}{\text{\textomicron}}
\ProvideDocumentCommand{\tpi}{O{}}{\text{\textpi}}
\ProvideDocumentCommand{\trho}{O{}}{\text{\textrho}}
\ProvideDocumentCommand{\tsigma}{O{}}{\text{\textsigma}}
\ProvideDocumentCommand{\ttau}{O{}}{\text{\texttau}}
\ProvideDocumentCommand{\tupsilon}{O{}}{\text{\textupsilon}}
\ProvideDocumentCommand{\tphi}{O{}}{\text{\textphi}}
\ProvideDocumentCommand{\tchi}{O{}}{\text{\textchi}}
\ProvideDocumentCommand{\tpsi}{O{}}{\text{\textpsi}}
\ProvideDocumentCommand{\tomega}{O{}}{\text{\textomega}}
\ProvideDocumentCommand{\vAlpha}{}{\varAlpha}
\ProvideDocumentCommand{\vBeta}{}{\varBeta}
\ProvideDocumentCommand{\vGamma}{}{\varGamma}
\ProvideDocumentCommand{\vDelta}{}{\varDelta}
\ProvideDocumentCommand{\vEpsilon}{}{\varEpsilon}
\ProvideDocumentCommand{\vZeta}{}{\varZeta}
\ProvideDocumentCommand{\vEta}{}{\varEta}
\ProvideDocumentCommand{\vTheta}{}{\varTheta}
\ProvideDocumentCommand{\vIota}{}{\varIota}
\ProvideDocumentCommand{\vKappa}{}{\varKappa}
\ProvideDocumentCommand{\vLambda}{}{\varLambda}
\ProvideDocumentCommand{\vMu}{}{\varMu}
\ProvideDocumentCommand{\vNu}{}{\varNu}
\ProvideDocumentCommand{\vXi}{}{\varXi}
\ProvideDocumentCommand{\vOmicron}{}{\varOmicron}
\ProvideDocumentCommand{\vPi}{}{\varPi}
\ProvideDocumentCommand{\vRho}{}{\varRho}
\ProvideDocumentCommand{\vSigma}{}{\varSigma}
\ProvideDocumentCommand{\vTau}{}{\varTau}
\ProvideDocumentCommand{\vUpsilon}{}{\varUpsilon}
\ProvideDocumentCommand{\vPhi}{}{\varPhi}
\ProvideDocumentCommand{\vChi}{}{\varChi}
\ProvideDocumentCommand{\vPsi}{}{\varPsi}
\ProvideDocumentCommand{\vOmega}{}{\varOmega}
\ProvideDocumentCommand{\valpha}{}{\varalpha}
\ProvideDocumentCommand{\vbeta}{}{\varbeta}
\ProvideDocumentCommand{\vgamma}{}{\vargamma}
\ProvideDocumentCommand{\vdelta}{}{\vardelta}
\ProvideDocumentCommand{\vepsilon}{}{\varepsilon}
\ProvideDocumentCommand{\vzeta}{}{\varzeta}
\ProvideDocumentCommand{\veta}{}{\vareta}
\ProvideDocumentCommand{\vtheta}{}{\vartheta}
\ProvideDocumentCommand{\viota}{}{\variota}
\ProvideDocumentCommand{\vkappa}{}{\varkappa}
\ProvideDocumentCommand{\vlambda}{}{\varlambda}
\ProvideDocumentCommand{\vmu}{}{\varmu}
\ProvideDocumentCommand{\vnu}{}{\varnu}
\ProvideDocumentCommand{\vxi}{}{\varxi}
\ProvideDocumentCommand{\vomicron}{}{\varomicron}
\ProvideDocumentCommand{\vpi}{}{\varpi}
\ProvideDocumentCommand{\vrho}{}{\varrho}
\ProvideDocumentCommand{\vsigma}{}{\varsigma}
\ProvideDocumentCommand{\vtau}{}{\vartau}
\ProvideDocumentCommand{\vupsilon}{}{\varupsilon}
\ProvideDocumentCommand{\vphi}{}{\varphi}
```

```
\ProvideDocumentCommand{\vchi}{}{\varchi}
\ProvideDocumentCommand{\vpsi}{}{\varpsi}
\ProvideDocumentCommand{\vomega}{}{\varomega}
\ProvideDocumentCommand{\uAlpha}{}{\upAlpha}
\ProvideDocumentCommand{\uBeta}{}{\upBeta}
\ProvideDocumentCommand{\uGamma}{}{\upGamma}
\ProvideDocumentCommand{\uDelta}{}{\upDelta}
\ProvideDocumentCommand{\uEpsilon}{}{\upEpsilon}
\ProvideDocumentCommand{\uZeta}{}{\upZeta}
\ProvideDocumentCommand{\uEta}{}{\upEta}
\ProvideDocumentCommand{\uTheta}{}{\upTheta}
\ProvideDocumentCommand{\uIota}{}{\upIota}
\ProvideDocumentCommand{\uKappa}{}{\upKappa}
\ProvideDocumentCommand{\uLambda}{}{\upLambda}
\ProvideDocumentCommand{\uMu}{}{\upMu}
\ProvideDocumentCommand{\uNu}{}{\upNu}
\ProvideDocumentCommand{\uXi}{}{\upXi}
\ProvideDocumentCommand{\uOmicron}{}{\upOmicron}
\ProvideDocumentCommand{\uPi}{}{\upPi}
\ProvideDocumentCommand{\uRho}{}{\upRho}
\ProvideDocumentCommand{\uSigma}{}{\upSigma}
\ProvideDocumentCommand{\uTau}{}{\upTau}
\ProvideDocumentCommand{\uUpsilon}{}{\upUpsilon}
\ProvideDocumentCommand{\uPhi}{}{\upPhi}
\ProvideDocumentCommand{\uChi}{}{\upChi}
\ProvideDocumentCommand{\uPsi}{}{\upPsi}
\ProvideDocumentCommand{\uOmega}{}{\upOmega}
\ProvideDocumentCommand{\ualpha}{}{\upalpha}
\ProvideDocumentCommand{\ubeta}{}{\upbeta}
\ProvideDocumentCommand{\ugamma}{}{\upgamma}
\ProvideDocumentCommand{\udelta}{}{\updelta}
\ProvideDocumentCommand{\uepsilon}{}{\upepsilon}
\ProvideDocumentCommand{\uzeta}{}{\upzeta}
\ProvideDocumentCommand{\ueta}{}{\upeta}
\ProvideDocumentCommand{\utheta}{}{\uptheta}
\ProvideDocumentCommand{\uiota}{}{\upiota}
\ProvideDocumentCommand{\ukappa}{}{\upkappa}
\ProvideDocumentCommand{\ulambda}{}{\uplambda}
\ProvideDocumentCommand{\umu}{}{\upmu}
\ProvideDocumentCommand{\unu}{}{\upnu}
\ProvideDocumentCommand{\uxi}{}{\upxi}
\ProvideDocumentCommand{\uomicron}{}{\upomicron}
\ProvideDocumentCommand{\upi}{}{\uppi}
\ProvideDocumentCommand{\urho}{}{\uprho}
\ProvideDocumentCommand{\usigma}{}{\upsigma}
\ProvideDocumentCommand{\utau}{}{\uptau}
\ProvideDocumentCommand{\uupsilon}{}{\upupsilon}
\ProvideDocumentCommand{\uphi}{}{\upphi}
\ProvideDocumentCommand{\uchi}{}{\upchi}
\ProvideDocumentCommand{\upsi}{}{\uppsi}
\ProvideDocumentCommand{\uomega}{}{\upomega}
\titleclass{\subsubparagraph}{straight}[\subparagraph]
\newcounter{subsubparagraph}[subparagraph]
\titlespacing*{\subsubparagraph}{\parindent}{3.25ex plus 1ex minus .2ex}{1em}
\titleformat{\section}[block]{\Large\bfseries}{\thesection}{1em}{}
\titleformat{\subsection}{\large\bfseries}{\thesubsection}{1em}{}
\titleformat{\subsubsection}{\normalsize\bfseries}{\thesubsubsection}{1em}{}
\titleformat{\paragraph}{\normalsize\bfseries}{\theparagraph}{1em}{}
\titleformat{\subparagraph}{\normalsize\bfseries}{\thesubparagraph}{1em}{}
\titleformat{\subsubparagraph}{\normalsize\bfseries}{\thesubsubparagraph}{1em}{}
\ProvideDocumentCommand{\sct}{s o}{\IfBooleanTF{#1}{\IfNoValueTF{#2}{\section*}{\section*{#2}}}{\IfNoValueTF
    {#2}{\section}{\section{#2}}}}
\ProvideDocumentCommand{\ssc}{s o}{\IfBooleanTF{#1}{\IfNoValueTF{#2}{\subsection*}{\subsection*{#2}}}{\
    IfNoValueTF{#2}{\subsection}{\subsection{#2}}}}
\ProvideDocumentCommand{\sssc}{s o}{\IfBooleanTF{#1}{\IfNoValueTF{#2}{\subsubsection*}{\subsubsection
```

```
    *{#2}}}{\IfNoValueTF{#2}{\subsubsection}{\subsubsection{#2}}}}
\ProvideDocumentCommand{\pr}{s o}{\IfBooleanTF{#1}{\IfNoValueTF{#2}{\paragraph*}{\paragraph*{#2}}}{\
    IfNoValueTF{#2}{\paragraph}{\paragraph{#2}}}}
\ProvideDocumentCommand{\spr}{s o}{\IfBooleanTF{#1}{\IfNoValueTF{#2}{\subparagraph*}{\subparagraph*{#2}}}{\
    IfNoValueTF{#2}{\subparagraph}{\subparagraph{#2}}}}
\ProvideDocumentCommand{\sspr}{s o}{\IfBooleanTF{#1}{\IfNoValueTF{#2}{\subsubparagraph*}{\subsubparagraph
    *{#2}}}{\IfNoValueTF{#2}{\subsubparagraph}{\subsubparagraph{#2}}}}
\renewcommand\theparagraph{\arabic{paragraph}.}
\renewcommand\thesubparagraph{(\arabic{subparagraph})}
\renewcommand\thesubsubparagraph{\boxed{\arabic{subsubparagraph}}}
\titlecontents{section}[2em]{\addvspace{1em}}{\thecontentslabel\quad}{}{\titlerule*[1pc]{.}\contentspage}
\titlecontents{subsection}[4em]{\addvspace{0.5em}}{\thecontentslabel\quad}{}{\titlerule*[1pc]{.}\contentspage
    }
\titlecontents{subsubsection}[6em]{\addvspace{0.5em}}{\thecontentslabel\ }{}{\titlerule*[1pc]{.}\contentspage
    }
\ifnum\value{CJK}=1\ifnum\value{SectionLanguage}=1\ifnum\value{CJKLanguage}=0
\renewcommand\thesection{第\zhnumber[style={Traditional,Normal}]{\arabic{section}}節}
\renewcommand\thesubsection{\zhnumber[style={Traditional,Normal}]{\arabic{subsection}}、}
\renewcommand\thesubsubsection{(\zhnumber[style={Traditional,Normal}]{\arabic{subsubsection}})}
\else\ifnum\value{CJKLanguage}=1
\renewcommand\thesection{第\zhnumber[style={Simplified,Normal}]{\arabic{section}}节}
\renewcommand\thesubsection{\zhnumber[style={Simplified,Normal}]{\arabic{subsection}}、}
\renewcommand\thesubsubsection{(\zhnumber[style={Simplified,Normal}]{\arabic{subsubsection}})}
\else
\renewcommand\thesection{\arabic{section}}
\renewcommand\thesubsection{\Roman{subsection}}
\renewcommand\thesubsubsection{\roman{subsubsection}}
\fi\fi
\else
\renewcommand\thesection{\arabic{section}}
\renewcommand\thesubsection{\Roman{subsection}}
\renewcommand\thesubsubsection{\roman{subsubsection}}
\fi\else
\renewcommand\thesection{\arabic{section}}
\renewcommand\thesubsection{\Roman{subsection}}
\renewcommand\thesubsubsection{\roman{subsubsection}}
\fi
\makeatletter
\@ifclassloaded{article}{}{
\titleformat{\chapter}[block]{\LARGE\bfseries}{\thechapter}{1em}{}
\ProvideDocumentCommand{\ch}{s o}{\IfBooleanTF{#1}{\IfNoValueTF{#2}{\chapter*}{\chapter*{#2}}}{\IfNoValueTF
    {#2}{\chapter}{\chapter{#2}}}}
\titlecontents{chapter}[0em]{\addvspace{1em}}{\thecontentslabel\quad}{}{\titlerule*[1pc]{.}\contentspage}
\ifnum\value{CJK}=1\ifnum\value{SectionLanguage}=1\ifnum\value{CJKLanguage}=0
\renewcommand\thechapter{第\zhnumber[style={Traditional,Normal}]{\arabic{chapter}}章}
\else\ifnum\value{CJKLanguage}=1
\renewcommand\thechapter{第\zhnumber[style={Simplified,Normal}]{\arabic{chapter}}章}
\else
\renewcommand\thechapter{Chapter \arabic{chapter}}
\fi\fi
\else
\renewcommand\thechapter{Chapter \arabic{chapter}}
\fi\else
\renewcommand\thechapter{Chapter \arabic{chapter}}
\fi
}
\makeatother
\ifnum\value{CJK}=1\ifnum\value{ZhRenew}=1\ifnum\value{CJKLanguage}=0
\newcommand{\temtoday}{\zhtoday}
\renewcommand{\contentsname}{\hfill\text{目錄}\hfill}
\else\ifnum\value{CJKLanguage}=1
\newcommand{\temtoday}{\zhtoday}
\renewcommand{\contentsname}{\hfill\text{目录}\hfill}
\else
\newcommand{\temtoday}{\today}
```

```
\fi\fi
\else
\newcommand{\temtoday}{\today}
\fi\else
\newcommand{\temtoday}{\today}
\fi
\newcommand{\temtitle}{\thispagestyle{empty}\Needspace{1\textheight}\maketitle\setcounter{page}{1}}
\newcommand{\temtoc}{\Needspace{1\textheight}\tableofcontents}
\newcommand{\temdoc}{\Needspace{1\textheight}\setcounter{page}{1}}
\newcommand{\titletocdoc}{\onehalfspacing\temtitle\temtoc\temdoc}
\newcommand{\titledoc}{\onehalfspacing\temtitle\temdoc}
\newcommand{\tocdoc}{\onehalfspacing\temtoc\temdoc}
```

VII Template Preamble Fifth Part Generator C

```c
// This is alphabet.c of https://github.com/Willie169/LaTeX-ToolKit, licensed under either GPL-3.0-or-later, CC BY-SA 4.0-or-later, or LPPL-1.3c-or-later.
#include <stdio.h>

int main() {
    char array[52];
    char i = 0;
    for (char c = 'A'; c <= 'Z'; c++) array[i++] = c;
    for (char c = 'a'; c <= 'z'; c++) array[i++] = c;
    for (int j = 0; j < 52; j++) {
        printf("\\ProvideDocumentCommand{\\tx%c}{O{}}{\\text{%c}}\n\\ProvideDocumentCommand{\\tb%c}{O{}}{\\textbf{%c}}\n\\ProvideDocumentCommand{\\bf%c}{O{}}{\\relax\\ifmmode\\mathbf{%c}\\else\\text{\\(\\mathbf{%c}\\)}\\fi}\n\\ProvideDocumentCommand{\\rm%c}{O{}}{\\relax\\ifmmode\\mathrm{%c}\\else\\text{\\(\\mathrm{%c}\\)}\\fi}\n", array[j], array[j], array[j], array[j], array[j], array[j], array[j], array[j], array[j], array[j]);
        for (int k = 0; k < 52; k++) printf("\\ProvideDocumentCommand{\\tx%c%c}{O{}}{\\text{%c%c}}\n\\ProvideDocumentCommand{\\tb%c%c}{O{}}{\\textbf{%c%c}}\n\\ProvideDocumentCommand{\\bf%c%c}{O{}}{\\relax\\ifmmode\\mathbf{%c%c}\\else\\text{\\(\\mathbf{%c%c}\\)}\\fi}\n\\ProvideDocumentCommand{\\rm%c%c}{O{}}{\\relax\\ifmmode\\mathrm{%c%c}\\else\\text{\\(\\mathrm{%c%c}\\)}\\fi}\n", array[j], array[k], array[j], array[k], array[j], array[k], array[j], array[k], array[j], array[k], array[j], array[k], array[j], array[k], array[j], array[k], array[j], array[k], array[j], array[k]);
    }
}
```