Receiver code

#include <SoftwareSerial.h>

#include <Servo.h>

// Bluetooth communication

SoftwareSerial btSerial(0, 1); // RX | TX

struct PacketData {

  byte lxAxisValue;       // Left joystick X-axis

  byte lyAxisValue;       // Left joystick Y-axis

  byte rxAxisValue;       // Right joystick X-axis

  byte ryAxisValue;       // Right joystick Y-axis

  byte wristRollValue;    // Wrist roll

  byte wristPitchValue;   // Wrist pitch

  byte carSpeedValue;     // Speed control

  byte carTurnValue;      // Additional data for car left/right

};

PacketData data;

// Servo motors for the 5DOF robotic arm

Servo baseServo;

Servo shoulderServo;

Servo elbowServo;

Servo gripperServo;

Servo wristRollServo;

Servo wristPitchServo;

// Motor driver pins

const int motorLeftForwardPin = 7;

const int motorLeftBackwardPin = 8;

const int motorRightForwardPin = 4;

const int motorRightBackwardPin = 13;

const int motorLeftEnablePin = 5;  // PWM for left motor speed

const int motorRightEnablePin = 6; // PWM for right motor speed

// Minimum and maximum angles for each joint

const int baseMinAngle = 0;

const int baseMaxAngle = 180;

const int shoulderMinAngle = 0;

const int shoulderMaxAngle = 90;

const int elbowMinAngle = 60;

const int elbowMaxAngle = 150;

const int gripperOpenAngle = 180;

const int gripperCloseAngle = 0;

const int wristRollMinAngle = 0;

const int wristRollMaxAngle = 180;

const int wristPitchMinAngle = 90;

const int wristPitchMaxAngle = 180;

// Current servo positions

int baseAngle = 90;

int shoulderAngle = 45;

int elbowAngle = 105;

int wristRollAngle = 90;

int wristPitchAngle = 135;

bool gripperClosed = true;

unsigned long lastRecvTime = 0;

void setup() {

  // Attach servos

  baseServo.attach(9);

  shoulderServo.attach(10);

  elbowServo.attach(11);

  gripperServo.attach(12);

  wristRollServo.attach(3);

  wristPitchServo.attach(2);

  // Set initial positions

  baseServo.write(baseAngle);

  shoulderServo.write(shoulderAngle);

  elbowServo.write(elbowAngle);

  gripperServo.write(gripperCloseAngle);

  wristRollServo.write(wristRollAngle);

  wristPitchServo.write(wristPitchAngle);

  // Initialize motor pins

  pinMode(motorLeftForwardPin, OUTPUT);

  pinMode(motorLeftBackwardPin, OUTPUT);

  pinMode(motorRightForwardPin, OUTPUT);

  pinMode(motorRightBackwardPin, OUTPUT);

  pinMode(motorLeftEnablePin, OUTPUT);

  pinMode(motorRightEnablePin, OUTPUT);

  btSerial.begin(38400);

}

void loop() {

  String dataString;

  if (btSerial.available()) {

    dataString = btSerial.readStringUntil('\n');

    sscanf(dataString.c\_str(), "%d,%d,%d,%d,%d,%d,%d,%d",

           &data.lxAxisValue,

           &data.lyAxisValue,

           &data.rxAxisValue,

           &data.ryAxisValue,

           &data.wristRollValue,

           &data.wristPitchValue,

           &data.carSpeedValue,

           &data.carTurnValue);

    // Gradually adjust servo angles based on joystick values

    baseAngle = map(data.lxAxisValue, 0, 254, baseMinAngle, baseMaxAngle);

    shoulderAngle = map(data.lyAxisValue, 0, 254, shoulderMinAngle, shoulderMaxAngle);

    elbowAngle = map(data.ryAxisValue, 0, 254, elbowMinAngle, elbowMaxAngle);

    wristRollAngle = map(data.wristRollValue, 0, 254, wristRollMinAngle, wristRollMaxAngle);

    wristPitchAngle = map(data.wristPitchValue, 0, 254, wristPitchMinAngle, wristPitchMaxAngle);

    // Motor speed control

    int motorSpeed = map(data.carSpeedValue, 0, 254, -255, 255) \* 0.7; // Reduced speed

    int turnValue = map(data.carTurnValue, 0, 254, -255, 255) \* 0.7;  // Reduced turn

    // Forward or backward movement with turn control

    if (motorSpeed > 0) { // Forward movement

      analogWrite(motorLeftEnablePin, constrain(motorSpeed - turnValue, 0, 255));

      analogWrite(motorRightEnablePin, constrain(motorSpeed + turnValue, 0, 255));

      digitalWrite(motorLeftForwardPin, HIGH);

      digitalWrite(motorLeftBackwardPin, LOW);

      digitalWrite(motorRightForwardPin, HIGH);

      digitalWrite(motorRightBackwardPin, LOW);

    } else if (motorSpeed < 0) { // Backward movement

      analogWrite(motorLeftEnablePin, constrain(-motorSpeed - turnValue, 0, 255));

      analogWrite(motorRightEnablePin, constrain(-motorSpeed + turnValue, 0, 255));

      digitalWrite(motorLeftForwardPin, LOW);

      digitalWrite(motorLeftBackwardPin, HIGH);

      digitalWrite(motorRightForwardPin, LOW);

      digitalWrite(motorRightBackwardPin, HIGH);

    } else { // Turn in place

      if (turnValue < 0) { // Turn left

        analogWrite(motorLeftEnablePin, -turnValue);

        analogWrite(motorRightEnablePin, -turnValue);

        digitalWrite(motorLeftForwardPin, LOW);

        digitalWrite(motorLeftBackwardPin, HIGH);

        digitalWrite(motorRightForwardPin, HIGH);

        digitalWrite(motorRightBackwardPin, LOW);

      } else if (turnValue > 0) { // Turn right

        analogWrite(motorLeftEnablePin, turnValue);

        analogWrite(motorRightEnablePin, turnValue);

        digitalWrite(motorLeftForwardPin, HIGH);

        digitalWrite(motorLeftBackwardPin, LOW);

        digitalWrite(motorRightForwardPin, LOW);

        digitalWrite(motorRightBackwardPin, HIGH);

      } else { // Stop

        digitalWrite(motorLeftForwardPin, LOW);

        digitalWrite(motorLeftBackwardPin, LOW);

        digitalWrite(motorRightForwardPin, LOW);

        digitalWrite(motorRightBackwardPin, LOW);

      }

    }

    // Control the gripper

    if (data.rxAxisValue < 128 && !gripperClosed) {

      gripperServo.write(gripperCloseAngle);

      gripperClosed = true;

    } else if (data.rxAxisValue > 128 && gripperClosed) {

      gripperServo.write(gripperOpenAngle);

      gripperClosed = false;

    }

    // Move servos to new positions

    baseServo.write(baseAngle);

    shoulderServo.write(shoulderAngle);

    elbowServo.write(elbowAngle);

    wristRollServo.write(wristRollAngle);

    wristPitchServo.write(wristPitchAngle);

    lastRecvTime = millis();

  } else {

    unsigned long now = millis();

    if (now - lastRecvTime > 1000) {

      // Stop all motors, but maintain their last positions (except gripper)

      digitalWrite(motorLeftForwardPin, LOW);

      digitalWrite(motorLeftBackwardPin, LOW);

      digitalWrite(motorRightForwardPin, LOW);

      digitalWrite(motorRightBackwardPin, LOW);

      gripperServo.write(gripperClosed ? gripperCloseAngle : gripperOpenAngle);

    }

  }

}