

## **Controle automatizado de garra robótica para transporte de materiais**

```
#include <Servo.h> // Biblioteca para controle de servomotores
```

```
// ===== SERVOS DO BRAÇO =====
```

```
Servo servoBase, servoAltura, servoProfundidade, servoGarra;
```

```
// ===== SERVOS DOS SENSORES (EXPULSORES) =====
```

```
Servo servo1; // Capacitivo (D4)
```

```
Servo servo2; // Indutivo (D8)
```

```
// ===== JOYSTICKS =====
```

```
const int joyBasePin = A0;
```

```
const int joyAlturaPin = A1;
```

```
const int joyProfundidadePin = A2;
```

```
const int botaoGarraPin = 7;
```

```
// ===== ENTRADAS E SAÍDAS =====
```

```
const int pinoBloqueio = 2;
```

```
const int pinoModoAuto = 12;
```

```
const int pinoPresenca = A3; // Sensor NPN de presença
```

```
const int pinoSensorCap = 9; // Sensor capacitivo
```

```
const int pinoSensorInd = 10; // Sensor indutivo
```

```
const int pinoEsteira = A4;
```

```
// ===== LIMITES DO BRAÇO =====
```

```
const int baseMin = 20, baseMax = 180;
```

```
const int alturaMin = 40, alturaMax = 170;
```

```
const int profundidadeMin = 70, profundidadeMax = 160;
```

```
const int garraAberta = 0, garraFechada = 80;
```

```

// ===== POSIÇÕES INICIAIS =====

int posBase = baseMin;

int posAltura = 160;

int posProfundidade = profundidadeMax;

int posGarra = garraAberta;


// ===== ESTADOS =====

bool estadoGarraFechada = false;

bool botaoAnterior = HIGH;


// ===== MODO AUTOMÁTICO =====

enum EstadoAuto {

    AGUARDAR_OBJETO, ATRASO_INICIO, BUSCAR, PEGAR, LEVAR,

    ENTREGAR, SUBIR, VOLTAR, AGUARDANDO

};

EstadoAuto estadoAuto = AGUARDAR_OBJETO;


// ===== TEMPORIZAÇÕES =====

unsigned long tempoInicioPresenca = 0;

const unsigned long atrasoInicio = 5000;

unsigned long tempoAberturaGarra = 0;

bool aguardandoEsteira = false;


// ===== PONTOS DE TRABALHO =====

int posX_base = 20, posX_altura = 90, posX_prof = 140; // Coleta

int posY_base = 175, posY_altura = 170, posY_prof = 115; // Entrega

int alturaMaxAuto = 170;

int alturaEntrega = 150;

```

```
// ===== EXPULSORES =====
```

```
const int posInicial1 = 75, posAtuado1 = 10;
```

```
const int posInicial2 = 90, posAtuado2 = 10;
```

```
const unsigned long tempoAtuado = 5000;
```

```
bool atuando1 = false, atuando2 = false;
```

```
bool sensor1Liberado = true, sensor2Liberado = true;
```

```
unsigned long tempoInicio1 = 0, tempoInicio2 = 0;
```

```
// ===== FILTRO DE JOYSTICK =====
```

```
int filtrarJoystick(int pin) {
```

```
    int total = 0;
```

```
    for (int i = 0; i < 3; i++) {
```

```
        total += analogRead(pin);
```

```
        delay(2);
```

```
    }
```

```
    return total / 3;
```

```
}
```

```
// ===== MOVIMENTO SUAVE =====
```

```
void moverServoLento(Servo& servo, int& posAtual, int posDestino, int passo = 1, int tempo = 10) {
```

```
    if (posDestino == posAtual) return;
```

```
    if (posDestino > posAtual) {
```

```
        for (int i = posAtual; i <= posDestino; i += passo) {
```

```
            if (digitalRead(pinoBloqueio) == HIGH) return;
```

```
            servo.write(i);
```

```
            delay(tempo);
```

```

    }
} else {
    for (int i = posAtual; i >= posDestino; i -= passo) {
        if (digitalRead(pinoBloqueio) == HIGH) return;
        servo.write(i);
        delay(tempo);
    }
}
posAtual = posDestino;
}

// ==== MOVIMENTO DOS EXPULSORES (SUAVE E SEM ATRASO) ====
void moverServoSensor(Servo &servo, int pino, int posDestino, int tempoMovimento = 6) {
    servo.attach(pino);
    int posAtual = servo.read();
    if (posAtual < posDestino) {
        for (int i = posAtual; i <= posDestino; i++) {
            servo.write(i);
            delay(tempoMovimento);
        }
    } else {
        for (int i = posAtual; i >= posDestino; i--) {
            servo.write(i);
            delay(tempoMovimento);
        }
    }
    servo.detach();
}

```

```

// ===== SETUP =====

void setup() {
    servoBase.attach(3);
    servoAltura.attach(11);
    servoProfundidade.attach(6);
    servoGarra.attach(5);

    pinMode(botaoGarraPin, INPUT_PULLUP);
    pinMode(pinoBloqueio, INPUT_PULLUP);
    pinMode(pinoModoAuto, INPUT_PULLUP);
    pinMode(pinoPresenca, INPUT_PULLUP);
    pinMode(pinoSensorCap, INPUT_PULLUP);
    pinMode(pinoSensorInd, INPUT_PULLUP);
    pinMode(pinoEsteira, OUTPUT);
    digitalWrite(pinoEsteira, LOW);

    Serial.begin(9600);

    servoBase.write(posBase);
    servoAltura.write(posAltura);
    servoProfundidade.write(posProfundidade);
    servoGarra.write(posGarra);
}

// ===== LOOP =====

void loop() {
    bool bloqueio = digitalRead(pinoBloqueio);
    bool modoAuto = (digitalRead(pinoModoAuto) == LOW);
    bool presenca = (digitalRead(pinoPresenca) == LOW);

```

```

// ===== AUTOMÁTICO =====

if (modoAuto && bloqueio == LOW) {
    switch (estadoAuto) {
        case AGUARDAR_OBJETO:
            if (presenca) {
                tempoInicioPresenca = millis();
                estadoAuto = ATRASO_INICIO;
            }
            break;

        case ATRASO_INICIO:
            if (!presenca) estadoAuto = AGUARDAR_OBJETO;
            else if (millis() - tempoInicioPresenca >= atrasoInicio)
                estadoAuto = BUSCAR;
            break;

        case BUSCAR:
            moverServoLento(servoBase, posBase, posX_base);
            moverServoLento(servoAltura, posAltura, posX_altura);
            moverServoLento(servoProfundidade, posProfundidade, posX_prof);
            estadoAuto = PEGAR;
            break;

        case PEGAR:
            moverServoLento(servoGarra, posGarra, garraFechada);
            digitalWrite(pinoEsteira, HIGH);
            estadoAuto = LEVAR;
            break;
    }
}

```

case LEVAR:

```
moverServoLento(servoAltura, posAltura, alturaMaxAuto);  
moverServoLento(servoBase, posBase, posY_base);  
moverServoLento(servoProfundidade, posProfundidade, posY_prof);  
moverServoLento(servoAltura, posAltura, alturaEntrega);  
estadoAuto = ENTREGAR;  
break;
```

case ENTREGAR:

```
moverServoLento(servoGarra, posGarra, garraAberta);  
tempoAberturaGarra = millis();  
aguardandoEsteira = true;  
estadoAuto = SUBIR;  
break;
```

case SUBIR:

```
moverServoLento(servoAltura, posAltura, alturaMaxAuto);  
estadoAuto = VOLTAR;  
break;
```

case VOLTAR:

```
moverServoLento(servoBase, posBase, posX_base);  
moverServoLento(servoProfundidade, posProfundidade, posX_prof);  
moverServoLento(servoAltura, posAltura, posX_altura);  
estadoAuto = AGUARDAR_OBJETO;  
break;
```

```
}
```

```
}
```

```

else {

    // ===== MANUAL =====

    int valBase = filtrarJoystick(joyBasePin);
    int valAltura = filtrarJoystick(joyAlturaPin);
    int valProf = filtrarJoystick(joyProfundidadePin);


    int angBaseMax = (posAltura < 160) ? 130 : baseMax;
    int alturaMinDinamico = (posBase > 130) ? 100 : alturaMin;


    int angBase = map(valBase, 0, 1023, baseMin, angBaseMax);
    int angAltura = map(valAltura, 0, 1023, alturaMinDinamico, alturaMax);
    int angProf = map(valProf, 0, 1023, profundidadeMin, profundidadeMax);


    moverServoLento(servoBase, posBase, angBase);
    moverServoLento(servoAltura, posAltura, angAltura);
    moverServoLento(servoProfundidade, posProfundidade, angProf);


    bool botaoAtual = digitalRead(botaoGarraPin);
    if (botaoAnterior == HIGH && botaoAtual == LOW) {
        estadoGarraFechada = !estadoGarraFechada;
        int destino = estadoGarraFechada ? garraFechada : garraAberta;
        moverServoLento(servoGarra, posGarra, destino);
    }
    botaoAnterior = botaoAtual;
}

// ===== EXPULSORES =====

unsigned long agora = millis();
int estadoSensor1 = digitalRead(pinoSensorCap);

```



```

int estadoSensor2 = digitalRead(pinoSensorInd);

// Capacitivo
if (estadoSensor1 == LOW) sensor1Liberado = true;
if (estadoSensor1 == HIGH && sensor1Liberado && !atuando1) {
    moverServoSensor(servo1, 4, posAtuado1, 3);
    tempoInicio1 = agora;
    atuando1 = true;
    sensor1Liberado = false;
}
if (atuando1 && (agora - tempoInicio1 >= tempoAtuado)) {
    moverServoSensor(servo1, 4, posInicial1, 3);
    atuando1 = false;
}

// Indutivo
if (estadoSensor2 == LOW) sensor2Liberado = true;
if (estadoSensor2 == HIGH && sensor2Liberado && !atuando2) {
    moverServoSensor(servo2, 8, posAtuado2, 3);
    tempoInicio2 = agora;
    atuando2 = true;
    sensor2Liberado = false;
}
if (atuando2 && (agora - tempoInicio2 >= tempoAtuado)) {
    moverServoSensor(servo2, 8, posInicial2, 3);
    atuando2 = false;
}

// ===== ESTEIRA =====

```

```
if (aguardandoEsteira && agora - tempoAberturaGarra >= 500) {  
    digitalWrite(pinoEsteira, LOW);  
    aguardandoEsteira = false;  
}  
  
// ===== SERIAL =====  
  
Serial.print("Base: "); Serial.print(posBase);  
Serial.print(" | Altura: "); Serial.print(posAltura);  
Serial.print(" | Prof: "); Serial.print(posProfundidade);  
Serial.print(" | Garra: "); Serial.print(posGarra);  
Serial.print(" | Auto: "); Serial.print(modosAuto ? "SIM" : "NÃO");  
Serial.print(" | Emergência: "); Serial.print(bloqueio ? "ATIVA" : "OK");  
Serial.print(" | Presença: "); Serial.print(presenca ? "OBJETO" : "VAZIO");  
Serial.print(" | Cap: "); Serial.print(estadosSensor1 == HIGH ? "ATIVO" : "OK");  
Serial.print(" | Ind: "); Serial.println(estadosSensor2 == HIGH ? "ATIVO" : "OK");  
  
delay(10);  
}
```