Woody: Roborodentia 2011 Robot

Felix Chung, Canh Sy, Hanson Yu
Computer Engineering Program
California Polytechnic State University
San Luis Obispo, CA
June 6, 2011

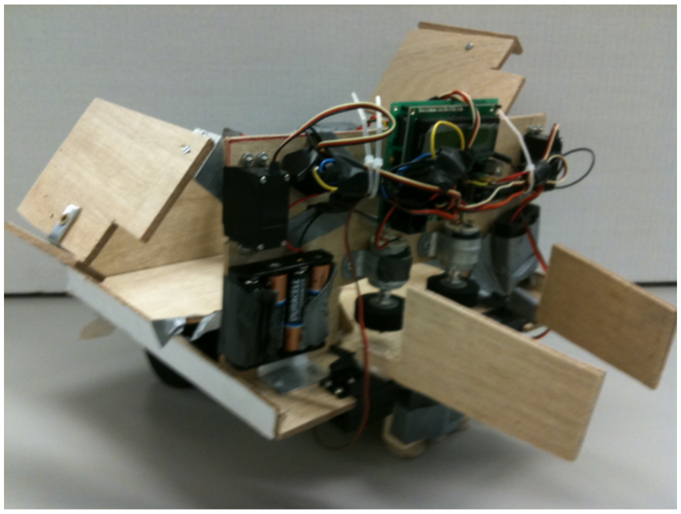


Fig. 1. Picture of Woody

I. Introduction

Woody is a fully autonomous robot built and designed for the 2011 Cal Poly Roborodentia competition. Woodys goal is to score the most points in the competition by navigating the competition field, efficiently collecting ping pong balls from the racks, and shooting the balls into the goal accurately. There was no specific budget set for the project, but costs were kept low by using tools from the Computer Engineering Capstone Lab, inexpensive materials such as wood, and reusable parts from robots from previous Roborodentia competitions.

II. Strategy

Our strategy for our robot was to accomplish three tasks: move, collect, and shoot. We felt that the robot that could do these tasks most efficiently would win the competition.

The strategy for movement was initially to have the base of the robot, which the wheels were attached, to act independently from the second tier of the robot, which contained everything else. We would use two rubber wheels driven by servos in the back of the base. The front of the base would have another wheel to complete a tricycle like base. The base of the robot would be connected to the second tier with a ball bearing, allowing the base to turn without the second tier turning. Walls of the competition field would act as a guide for our robot to keep our robot on course.

Collecting the balls off the two racks would be done by collectors on each side of the robot. The collectors were paddles driven by servos that would spin and knock balls off the racks and into the robots second tier, where many balls could be stored at once. The second tier where the balls were collected into would be tilted at a forward angle, allowing the balls to be guided into the shooters.

Shooting would be done by two motors with rubber wheels that were continuously spinning. We imagined shooters like those of a baseball pitching machine, where ping pong balls would be funneled into the shooters and shot out at a high velocity. Adjustable deflector paddles on both sides of the robot would allow the robot to always shoot balls towards the goal, no matter where the robot was on the field. The deflectors would always adjust and aim the balls towards the goal.

The robot would navigate the field by first collecting the balls on the left, or starting side of the field. After collecting all the balls, the robot would move back to the starting position and repeat this for the re-rack on the starting side. Next the robot would move to the right side of the field, and collect the balls on that rack. After the balls on that side were collected, the robot would try and collect the re-racked balls on the right side again.

III. Electronics Design

Woody was built with the following electronic components:

- Polybot board
- 2 DC motors
- 4 continuous servos
- 2 non-continous servos
- relay switch
- AA battery holder (holds 4 AA batteries)
- AAA battery holder (holds 4 AAA batteries)

Our original concept design is shown at Fig. 2.

All servos are connected directly to the Polyboard board powered by a AAA battery pack, which contains a switch.

The DC motors are connected to a separate independent AA battery pack controlled and operated by a relay switch connected to the Polybot board through the relay jumper near the middle of the board.

A schematic of the design can be seen in the Appendix.

IV. Mechanical Design

Our original concept design is shown at Fig. 2.

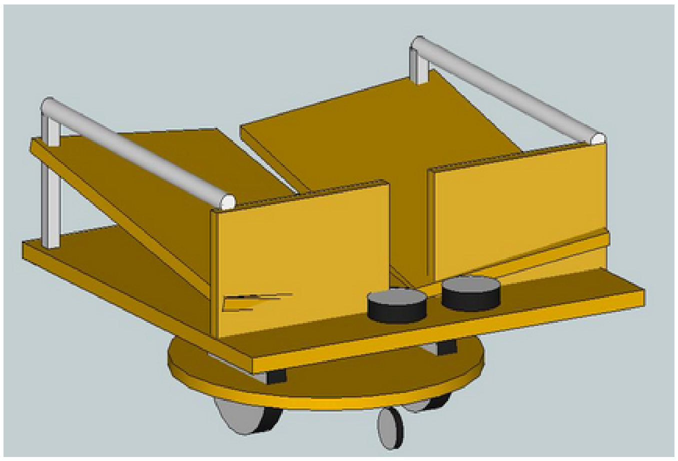


Fig. 2. This is the original concept design.

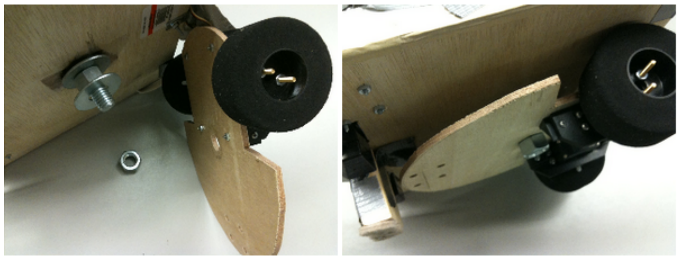


Fig. 3. Base of Woody

Woody is primarily made out of wood. Its frame consist of a 11 x 11 inch piece of wood supported by a circular base which connect the 2.5 inch diameter wheels. The 11 x 11 inch piece of wood is connected to the base through a 3 inch bolt down the center of the two pieces. The top of the 11 x 11 inch surface, where the ping pong balls would be held, needs to be less than 4 inches as to stay below the 4 inch high trough where the ping pong balls are racked. This is shown at Fig. 3.

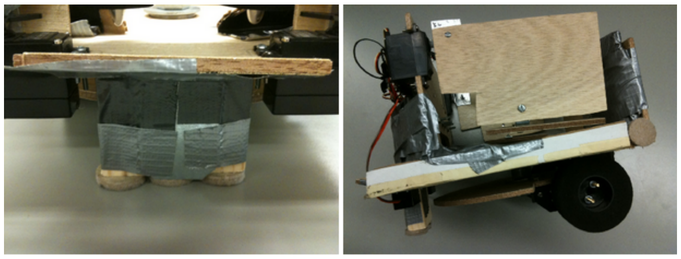


Fig. 4. Front and tilt of Woody

The front of Woody is supported by a piece of wood, which includes sliders to reduce friction between Woody and the competition arena. The block of wood also allows for Woody to have a slight tilt which creates a downward rank of the ping pong balls to follow. This is shown at Fig. 4.

Continuous spinning servos operate the claw used to collect the ping pong balls. The tip that creates the L shaped wooden paddle is necessary as to reach over and around the ping pong balls Woody is collecting. A mechanical drawing of this collecting method is shown at Fig. 5 and 6.

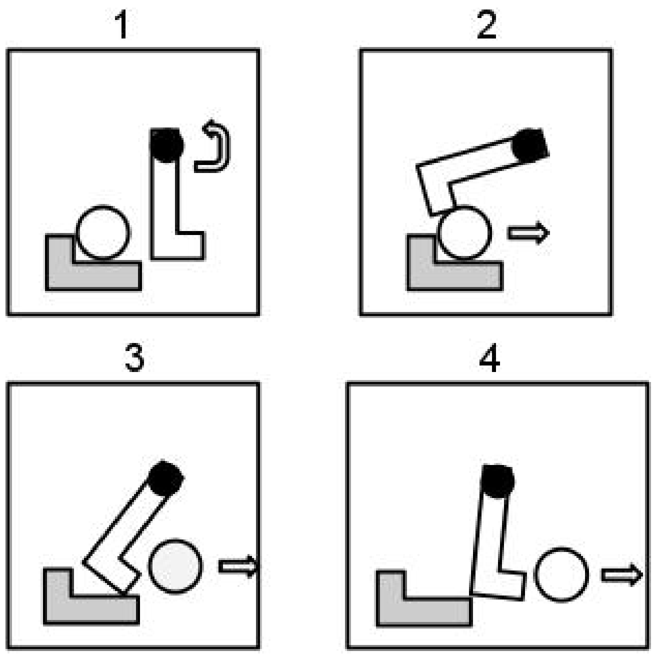


Fig. 5. Theory of collecting balls

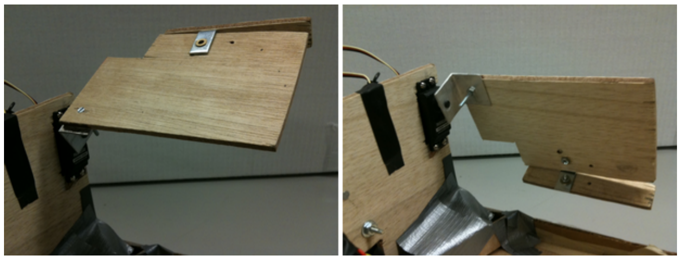


Fig. 6. Collectors

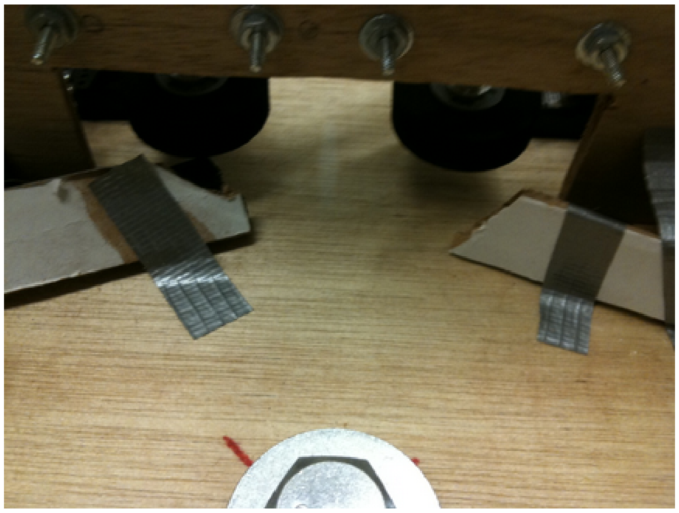


Fig. 7. Funnel

Along with the downward tilt of Woody, two cardboard pieces help to create a funnel for the ping pong ball to follow into the shooters. This is shown at Fig. 7.

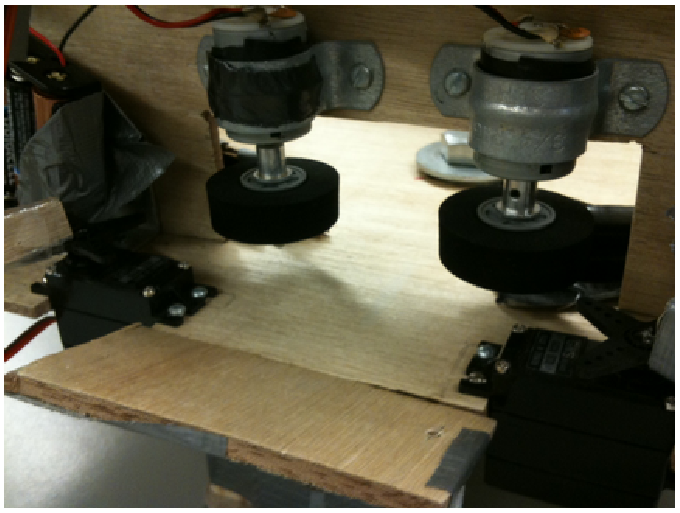


Fig. 8. Shooters

Metal pipe fasteners support the DC motors that operate the spinning wheels. This is shown at Fig. 8.

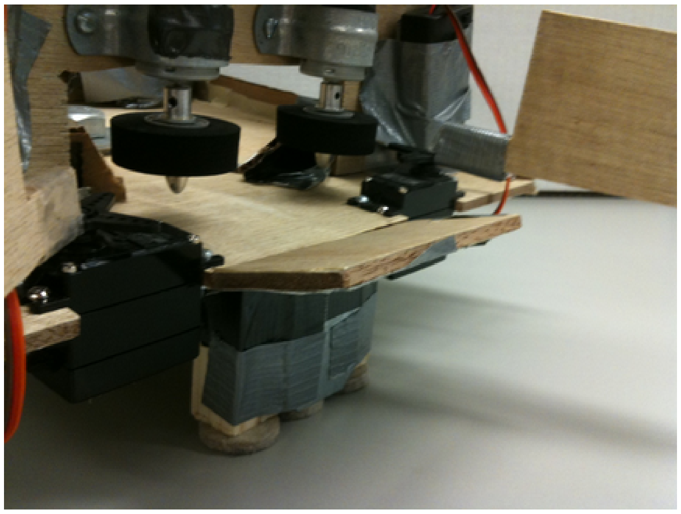


Fig. 9. Ramp for shooter

A bent metal strip supports a wooden ramp in order deflect the ping pong balls at an upward angle, rather than towards the ground. This is shown at Fig. 9.

The ping pong balls are also directed left or right by wooden paddles attached to non-continuous moving servos. This is shown at Fig. 10.

The bulk of the components, which includes the PolyBot board, servos, battery packs, relay, and DC motors, are secured to the front panel of Woody. This is shown at Fig. 11.

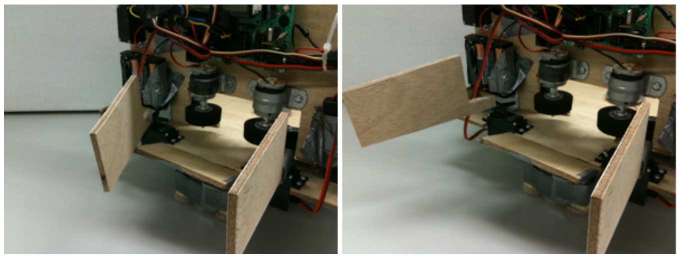


Fig. 10. Wood paddles for aiming

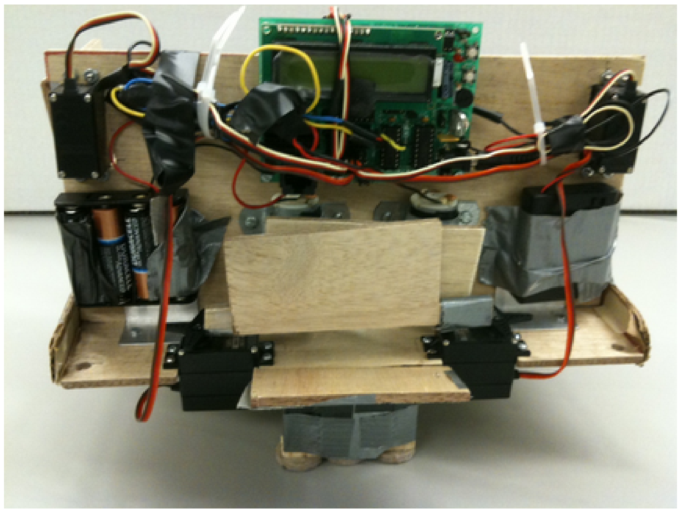


Fig. 11. Front panel showing components

V. Software Design

Our software design was based on timing and positioning of servos and motors. We used the polybot library and its functions, so we can just call functions from the library to control motors and servos. Woodys initial routine was opening up the two front panels for aiming. Since the rules state that the robot cannot exceed 12x12x12 dimensions, we had the two front panels set up as shown in Figure 14. In order for the ball to shoot, we had to open the panels, but we could not open them both at the same time. So In our software, we first move one of the panels to a tested position where the ball would score a goal. Then the next panel would move to its position.

After the two panels open up, we programmed Woody to move forward and stop when it reaches a position where it can start collecting balls. When Woody stops, we turn on the motors first by sending a signal to the relay switch. Then we turn on one of our collectors by turning on the servo controlling the collector. When the shooting motors and the collector servo is turned on, we begin a loop to control how Woody moves forward while collecting and shooting.

Since our design was based on timing, we had a loop that iterates a certain amount of time (we programmed it to iterate 8 times). Because Woody tends to stray a little bit when moving in a straight line, we programmed Woodys movement inside the loop. Inside the loop, we have Woody move forward a certain amount of seconds by turning on both servo wheels. Then we add a short delay between turning off both motors. So we first turn off the left motor wheel, then we added a short delay (300ms) before we turn off the right motor wheel. This makes it so that Woody would stay moving in a straight line when moving forward.

When the loop ends, Woody would be near the end of the trough and from there it begins to move back to its starting position. This is easily done by reversing the direction of the servo motor wheels. Before moving back, we turn of the collector servo, while keeping the shooting motors on in case there are still some balls left. Once it gets back to its starting position, it repeats moving forward and performing the collection and shooting routine. This is done three times to ensure that all the balls are empty in the trough.

Once Woody repeats the routine three times, it tries to move to the other side of the course. In order to do that, we had to perfectly time the servo wheels for Woody to pivot and move toward the other end. In our design, we turned on one wheel servo and delayed a couple seconds in order for Woody to have the correct degree to begin moving toward the other side. When the pivoting ends, Woody begins moving backward toward the other side and stops at a given time. Once it stops moving backward, it begins pivoting back to a position where Woody is facing forward again. Then it moves backward and stops when it reaches the bottom corner of the course. Then it begins the moving forward and performing the collection and shooting routine for the other side.

VI. Budget

TABLE I
Parts Costs

Part	**Cost**
PolyBot Board	$65
4 Continuous Servos	$15 x 4
2 Non-continuous Servos	$15 x 2
2 DC Motors	$7 x 2
2 2.3mm Prop Collet Hub	$3 x 2
Relay Switch Omron G5LE	$1
AA Battery Holder	$2
AAA Battery Holder	$2
Wood	$5
Misc. (screws, nuts, adhesives)	$20
Total	**$205**

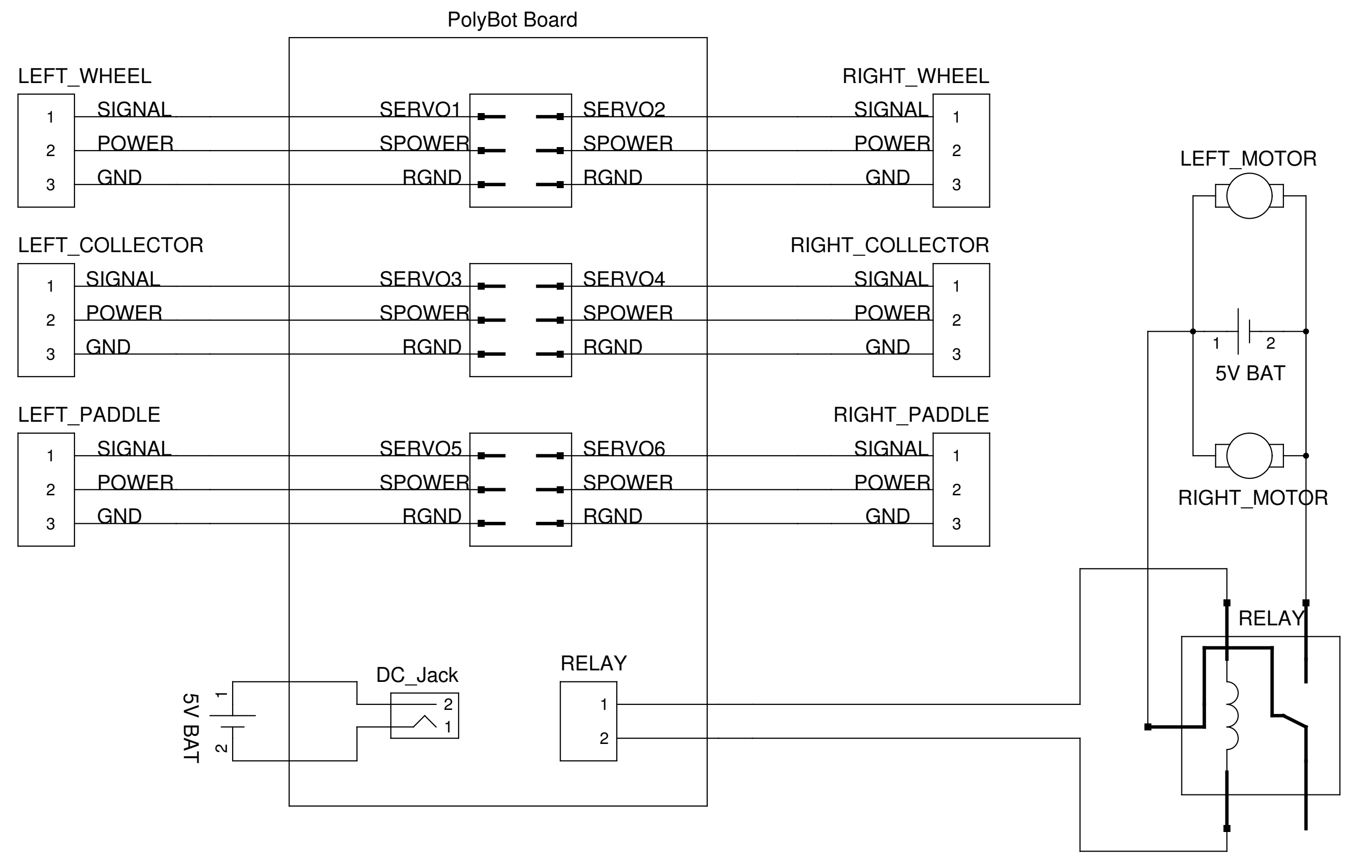
VII. What We Would Change

Many complications arose with our robot as the competition neared, and thus we were unable to perfect our robot. Our initial strategy of having the base move independently did not work because the design was poor. We could not get the robot to turn or move straight how we wanted. We decided to eliminate the robots ability to turn, which allowed the robot to move straight correctly. If we had to do this over, we would use omni-wheels for the movement of the robot.

Another complication involved the timings of movement on the software. Our robots algorithm was hardcoded to move and function based on timings, but these timings varied based on the voltage of the batteries we used. The voltage of the batteries varied and changed every time the robot was on. All of these variations caused inconsistencies and difficulties with our algorithm. If doing this competition again, we would use switches attached to the side of our robot to tell us where we are instead of unreliable timings.

VIII. Conclusion

Woody did not perform how we wanted, as we could only collect balls from one side of the field. However, the collection was consistent on the one side. The balls we were able to collect were able to be shot into the goal very efficiently. We were able to finish in 4th place of the competition, which we were very happy and excited about.



TITLE WOODY	
FILE:	REVISION:
PAGE OF	DRAWN BY: Canh Sy

```c
#include "globals.h"
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>

#define LEFT_WHEEL 0
#define RIGHT_WHEEL 1

void correct() {
    set_servo_position(0, 0);
    delay_ms(100);
    servo_off(1); servo_off(0);
}

void moveForward() {
    set_servo_position(LEFT_WHEEL, 255); // left
    set_servo_position(RIGHT_WHEEL, 0); //right
}

void moveAdjust() {
    moveForward();
    delay_ms(750);
}

void moveBackward() {
    set_servo_position(RIGHT_WHEEL, 255);
    set_servo_position(LEFT_WHEEL, 0);
}

void turn() {
    set_servo_position(LEFT_WHEEL, 0);
    set_servo_position(RIGHT_WHEEL, 0);

    delay_ms(5000);
    servo_off(LEFT_WHEEL); servo_off(RIGHT_WHEEL);
    delay_ms(1000);

    moveBackward();
    delay_ms(12250);
    servo_off(LEFT_WHEEL); servo_off(RIGHT_WHEEL);

    set_servo_position(LEFT_WHEEL, 255);
    set_servo_position(RIGHT_WHEEL, 255);
    delay_ms(11000);

    moveBackward();
    delay_ms(5000);

    moveForward();
    delay_ms(5000);
}

void stop() {
    servo_off(0);
    servo_off(1);
}

void spinCollector(int sel, int pos) {
    set_servo_position(sel, pos);
}

void moveForwardCollect(int servo, int count) {
    int i;
    delay_ms(700);
    set_servo_position(5, 38);

    spinCollector(servo, 255);
```

```c
    delay_ms(1400);

    for(i = 0; i < 8; i++) {
        if(servo == 3) {
            moveForward();
            delay_ms(650);

            servo_off(LEFT_WHEEL);
            delay_ms(400);
            servo_off(RIGHT_WHEEL);
            delay_ms(1000);
        } else {
            moveForward();
            delay_ms(700);
            servo_off(LEFT_WHEEL);
            servo_off(RIGHT_WHEEL);
            delay_ms(1000);
        }
    }

    if(count > 0) {
        moveForward();
        delay_ms(750);
    }

}

int main(void) {
    initialize();
    int i = 0;
    int j = 0;
    delay_ms(1000);

    while(1) {
        // Routine for left trough
        for(i = 0; i < 3; i++) {
            set_servo_position(4, 0);
            delay_ms(700);
            set_servo_position(5, 38);

            delay_ms(1000);
            set_servo_position(4, 100);

            moveForward();
            delay_ms(1150);
            servo_off(LEFT_WHEEL); servo_off(RIGHT_WHEEL);

            relay_on();

            moveForwardCollect(3, i);

            servo_off(3);

            set_servo_position(4, 100);
            set_servo_position(5, 100);

            if(i < 2) {
                for(j = 0; j < 8; j++) {
                    moveBackward();
                    delay_ms(1000);
                    servo_off(LEFT_WHEEL);
                    delay_ms(300);
                    servo_off(RIGHT_WHEEL);
                    set_servo_position(4, 0);
                    set_servo_position(5, 38);
                }
            }

            servo_off(3);
        }

        servo_off(LEFT_WHEEL); servo_off(RIGHT_WHEEL);
        servo_off(3);

        set_servo_position(4, 250);
        set_servo_position(5, 20);
        turn();

        // Routine for right trough
        for(i = 0; i < 3; i++) {
            set_servo_position(4, 250);
            delay_ms(700);
            set_servo_position(5, 150);

            delay_ms(1000);
            set_servo_position(4, 100);

            moveForward();
            delay_ms(1250);
            servo_off(LEFT_WHEEL); servo_off(RIGHT_WHEEL);

            relay_on();

            moveForwardCollect(2, i);

            servo_off(2);

            set_servo_position(5, 100);
            set_servo_position(4, 150);

            if(i < 2) {
                for(j = 0; j < 8; j++) {
                    moveBackward();
                    servo_off(LEFT_WHEEL);
                    servo_off(RIGHT_WHEEL);
                }
            }

            servo_off(2);
        }

        turn();
    }

    return 0;
}
```