EMT2461
Electromechanical Systems – Software Interface
FALL 2024

NEW YORK CITY COLLEGE OF TECHNOLOGY
CITY TECH

PROJECT STATUS REPORT

FINAL PROJECT SUMMARY

REPORT DATE	PROJECT NAME	PREPARED BY
Date: 12/10	Project: Candy Claw Machine	Name: Yusuf Smaili, Ronny Yeap

STATUS SUMMARY

The Candy Claw Machine project uses an Arduino Mega to control it, while we use an aluminum frame paired with some cardboard to

PROJECT OVERVIEW

TASK	% DONE	DUE DATE	STATUS	NOTES
Building The Frame	100	12/9	Aluminum Extrusions were put together.	We built the frame by using aluminum extrusions. Using mounts, we put them together in a rectangular shape.
Pulleys & Belt G2T	0	12/14	We need to order the Pulleys & Belt to have our gantry function.	
Entire Claw Machine	30	12/18	We still need to connect the electronic components, finish up the frame with some cardboard, as well as order pulleys and belts for our movement.	

BUDGET OVERVIEW

CATEGORY	SPENT	% OF TOTAL	ON TRACK?	NOTES
Aluminum Extrusions	$68	58%	Delivered	Used to make frame
Lego Mechanical Parts (REFUNDED)	$35	30%	REFUND PENDING	Does not fit DC MOTOR!!
Technical Computer Parts	$14	12%	Delivered	Used to control the movement

RISK AND ISSUE HISTORY

ISSUE	ASSIGNED TO	DATE
Lego parts may be too small compared to the frame	Yusuf S.	12/4
Lego mechanical part is not what we imagine or missing parts we need	Ronny	12/4
Suspect the end of project size is too big to bring to campus		12/4
UPDATE: LEGO PARTS NEED TO BE RETURNED	Yusuf S.	12/10

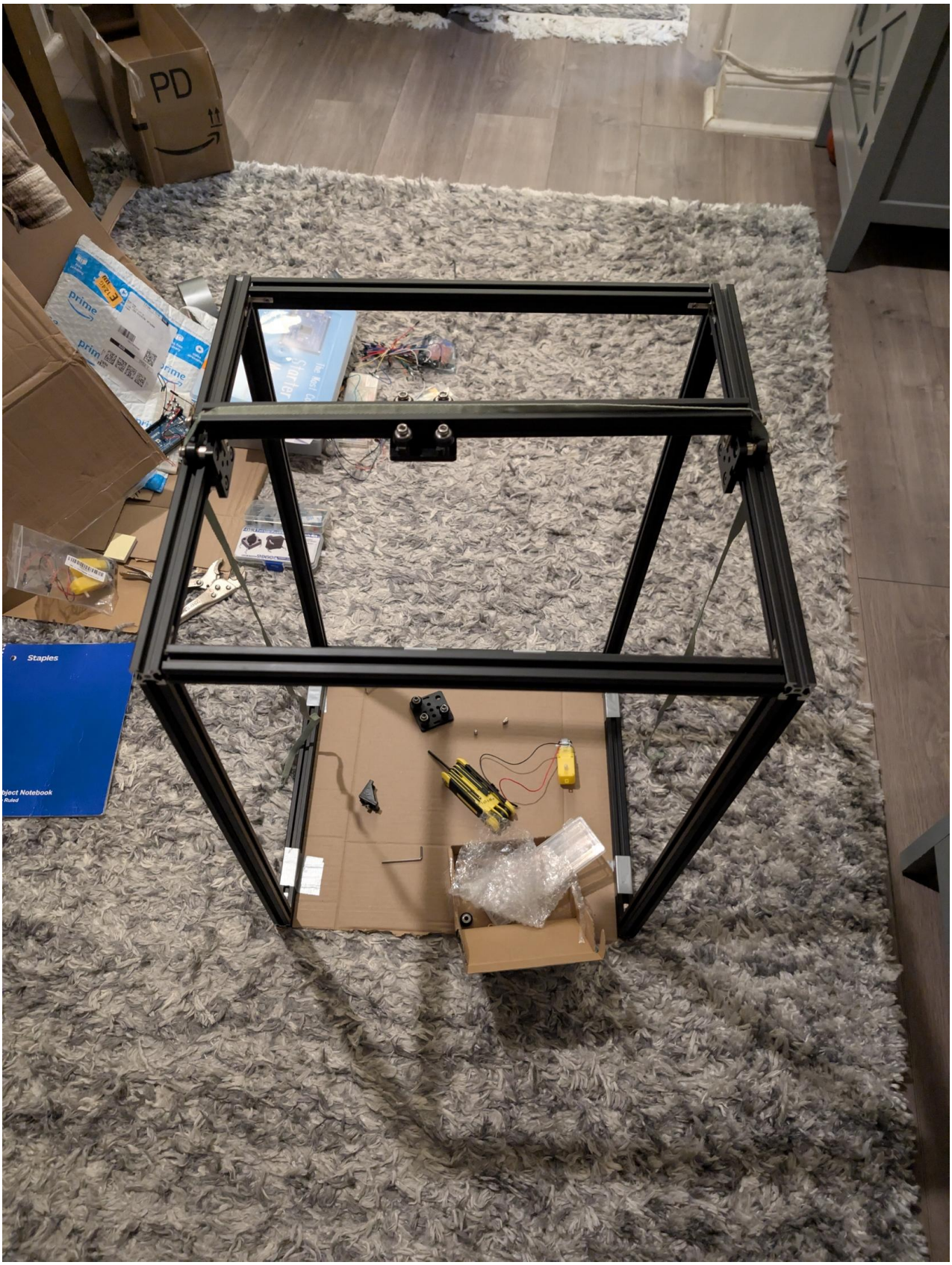
CONCLUSIONS/RECOMMENDATIONS

Honestly, we're not sure if this project is going to be finished, as some things didn't go as planned as planned before. But I promise we're trying to the best of our ability...

What we need to do is order the new mechanical parts for the gantry, and if that doesn't go as planned, this project may very well not be able to be completed.

MECHANICAL PARTS:

(Pictures)



(Description):
Lego Mechanical parts on the right

(Status): Parts delivered, Frame built, mechanical part of gantry built.

ELECTRONIC COMPONENTS:

(Pictures)

(Description):
The picture includes the Arduino MEGA, DC Motor circuit boards, Joystick, and DC motors. As well as the Breadboard

(Status): Items delivered but not yet connected.

CODE STATUS:

(Code)

```
#include <Servo.h>

// Define motor pins
#define ENA 5  // PWM pin for Motor 1 (X-axis)
#define IN1 2  // Control pin for Motor 1
#define IN2 3  // Control pin for Motor 1
#define ENB 6  // PWM pin for Motor 2 (Y-axis)
#define IN3 4  // Control pin for Motor 2
#define IN4 7  // Control pin for Motor 2

#define Z_EN 9   // PWM pin for Motor 3 (Z-axis, vertical lift)
#define Z_IN1 8  // Control pin for Motor 3
#define Z_IN2 10 // Control pin for Motor 3

// Define pins for the joystick
const int joystickX = A0;  // X-axis of joystick
const int joystickY = A1;  // Y-axis of joystick
const int buttonPin = 2;   // Joystick button

// Movement thresholds for the joystick
const int threshold = 100;

void loop() {
  // Read joystick values
  int xValue = analogRead(joystickX) - 512; // Center joystick value is ~512
  int yValue = analogRead(joystickY) - 512;
// Read joystick button
  if (digitalRead(buttonPin) == LOW) {
    clawServo.write(clawClosed); // Close claw
    delay(500);                  // Debounce delay
    while (digitalRead(buttonPin) == LOW); // Wait for button release
    clawServo.write(clawOpen);   // Open claw
  }
}

// Servo pin
#define SERVO_PIN 11

Servo clawServo; // Create a servo object

int servoPosition = 0; // Initial servo position (claw closed)
int threshold = 20;    // Potentiometer dead zone threshold
bool returnInProgress = false;

void setup() {
  // Motor pins setup
  pinMode(ENA, OUTPUT);
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
```

```
  pinMode(ENB, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
  pinMode(Z_EN, OUTPUT);
  pinMode(Z_IN1, OUTPUT);
  pinMode(Z_IN2, OUTPUT);

  // Button setup
  pinMode(BUTTON, INPUT_PULLUP);

  // Servo setup
  clawServo.attach(SERVO_PIN);
  clawServo.write(servoPosition); // Start with claw closed

  Serial.begin(9600); // For debugging
}

void loop() {
  int potXVal = analogRead(POT_X); // Read X-axis potentiometer
  int potYVal = analogRead(POT_Y); // Read Y-axis potentiometer
  bool buttonPressed = digitalRead(BUTTON) == LOW;

  // Skip manual control if returning to origin
  if (!returnInProgress) {
    // Control X-axis motor
    if (potXVal > 512 + threshold) {
      moveMotor(ENA, IN1, IN2, HIGH); // Move right
    } else if (potXVal < 512 - threshold) {
      moveMotor(ENA, IN1, IN2, LOW); // Move left
    } else {
      stopMotor(ENA);
    }

    // Control Y-axis motor
    if (potYVal > 512 + threshold) {
      moveMotor(ENB, IN3, IN4, HIGH); // Move forward
    } else if (potYVal < 512 - threshold) {
      moveMotor(ENB, IN3, IN4, LOW); // Move backward
    } else {
      stopMotor(ENB);
    }
  }

  // Handle return to origin and claw open command
  if (buttonPressed && !returnInProgress) {
    returnInProgress = true;
    returnToOriginAndOpenClaw();
    returnInProgress = false;
  }
}
```

```
// Function to control motor movement
void moveMotor(int enablePin, int inPin1, int inPin2, bool direction) {
  analogWrite(enablePin, 200); // Set speed (0-255)
  if (direction) {
    digitalWrite(inPin1, HIGH);
    digitalWrite(inPin2, LOW);
  } else {
    digitalWrite(inPin1, LOW);
    digitalWrite(inPin2, HIGH);
  }
}

// Function to stop a motor
void stopMotor(int enablePin) {
  analogWrite(enablePin, 0);
}

// Function to return to origin (0, 0) and open the claw
void returnToOriginAndOpenClaw() {
  // Move X-axis back to 0
  Serial.println("Returning X-axis to 0...");
  moveMotor(ENA, IN1, IN2, LOW); // Move left
  delay(1500); // Adjust time based on distance and speed
  stopMotor(ENA);

  // Move Y-axis back to 0
  Serial.println("Returning Y-axis to 0...");
  moveMotor(ENB, IN3, IN4, LOW); // Move backward
  delay(1500); // Adjust time based on distance and speed
  stopMotor(ENB);

  // Open the claw
  Serial.println("Opening the claw...");
  clawServo.write(0); // Adjust value for fully open claw
  delay(500); // Wait for claw to open
}
```

(Description): The difference between this template compared to the last one is that it uses a joystick, which we will use to control our claw machine. The last code only vaguely described an input, but now we know for sure that we'll use a joystick.

(Status): Done but needs some tweaks.