CS 241 Final Project (Bear Aware Smart Latch)

Solomon Himelbloom

April 28, 2021

- This project is fine on the course homepage.
- For full-sized images, see the link below.
- README + Repository: `https://github.com/TechSolomon/cs241`

1 Problem Solving

- During this project, I set out to solve the problem of wildlife interactions with humans in Alaska. One common problem without outdoor trash cans (especially roll carts) is that they can easily get knocked over or tampered with when left outside for long durations. Numerous issues arise for moose, coyotes, bears, and other animals as habitat changes can adjust their behavior through interactions with people in the surrounding areas.
- My initial minimum viable product (MVP) for this project was to build a "smart" outdoor trash can/roll cart latching mechanism that watches for changes in angle/location and alerts the owner about any intruders since the last visit. After further discussions and iterations, one issue that would arise is with the power supply; in some instances, this makes sense to use a battery, but in others, it makes more sense to plug it in. Ideally, the system should dock to a power supply;

however, a “knocked over alarm” or latch would be sufficient for an MVP.

2 Technical Approach

- Some supplies that I needed include an Arduino Uno/Nano, a Passive Infrared Sensor (PIR) to connect with an accelerometer to report changes in angle, a mini breadboard, and a servo motor to send lock and unlock commands. Stretch goals for this project included logging location changes through mapping and data visualization, in addition to a do not disturb mode to bypass the active alert system during user-specified dates and times.
- For the initial project prototype, I decided to go with an Arduino Uno, a mini solderless breadboard (4x4 points), a standard servo (TowerPro SG-5010), a water sensor, and the RGB LED (K851264) from the lab kit. In the future, the servo should upgrade to a more powerful locking mechanism. Upon taking the dimensions of a traditional roll cart, I decided that a large pizza box design would be a good substitute for the lid; adding more size constraints yielded a much smaller, “customized” Adafruit cardboard box to house the electronics.

3 Project Code & Physical Design

- For the physical aspects of my design, everything fits inside the Adafruit box with two mini breadboards tying everything together. Black wires signify ground (three-pin connections total), red is for power (5V), and the other wires match their respective breakout for carrying data.
- Upon taking a look at my project code, all features encapsulate the instructions function. The program header describes the servo, water sensor, and RGB LED pin utiliza-

tion to mirror my wiring setup. The locking and unlocking commands control the servo motor through their respective functions, while getWaterLevel (looping function) and showColor (table-driven programming) run repeatedly to send and receive the most accurate end-user information.

4 Conclusion

- Lessons learned with this project included scope creep, time management, reading documentation, and ordering researched parts. My favorite aspect would have to be taking what we learned in the lab and experimenting with the interactions between multiple components.
- For future improvements, I hope to conduct additional outdoor weather testing to explore the limitations of water and sub-zero temperatures. In addition to including an accelerometer magnetometer (only need a STEMMA QT / Quiic JST SH 4-pin connection cable), an independent power supply (with solar power or rechargeable batteries), and IFTTT/Twilio SMS integration as the next iteration in the *Bear Aware Smart Latch* system.

5 Appendix

5.1 Source Code

```
// Solomon Himelbloom
// CS 241 Final Project
// Bear Aware Smart Latch

#include <Servo.h>

// Servo Motor
Servo latch;
const int servoPin = 9;
int pos = 0;

// Water Sensor Data
const int analogInPin = A0;
int sensorValue = 0;

// RGB LED (red, green, & blue) Pins
const char ledPins[3] = {2, 3, 4};
int ledTime = 0;

// Default Color Presets
int red[4] = {500, 0, 0};
int green[5] = {0, 500, 0};
int blue[6] = {0, 0, 500};
int low[7] = {250, 500, 500};
int medium[8] = {500, 500, 250};
int high[9] = {500, 250, 500};

void getServoState() {
  // 0 degrees to 180 degrees.
  for (pos = 0; pos <= 180; pos += 1) {
    latch.write(pos);
    delay(15);
  }

  // 180 degrees to 0 degrees.
  for (pos = 180; pos >= 0; pos -= 1) {
    latch.write(pos);
    delay(15);
  }
}

// Close from open position = 0 degrees to 45 degrees.
void lock() {
  for (pos = 0; pos <= 45; pos += 1) {
    latch.write(pos);
    delay(15);
  }

  int denied[10] = {500, 0, 0};
  showColor(denied);
}

// Open from closed position = 45 degrees to 0 degrees.
void unlock() {
  for (pos = 45; pos >= 0; pos -= 1) {
    latch.write(pos);
    delay(15);
```

```
  }

  int granted[6] = {0, 500, 0};
  showColor(granted);
}

// Report current levels on the sensor.
void getWaterLevel() {
  sensorValue = analogRead(analogInPin);

  Serial.print("Water Level = ");
  Serial.println(sensorValue);

  delay(1000);
}

// KEY: color[0]==red, color[1]==green, & color[2]==blue.
void showColor(const int *color) {
  int totalDelay = 0;
  for (int colorIndex = 0; colorIndex < 3; colorIndex++) {
    for (int pinIndex = 0; pinIndex < 3; pinIndex++) {
      digitalWrite(ledPins[pinIndex], colorIndex == pinIndex);
    }
    delayMicroseconds(color[colorIndex]);
    totalDelay += color[colorIndex];
  }

  for (char p : ledPins) {
    digitalWrite(p, 0);
  }

  // Takes 3 milliseconds.
  delayMicroseconds(3000 - totalDelay);
}

// Smooth shift between custom RGB LED colors.
void getCurrentColor() {
  int color[3] = {ledTime / 10, 0, 500 - ledTime / 10};
  showColor(color);
  ledTime++;
  if (ledTime > 5000) ledTime = 0;
}

void setup() {
  Serial.begin(9600);

  Serial.print("Bear Aware Smart Latch\n");
  Serial.print("CS 241 Final Project by Solomon Himelbloom\n");
  Serial.print("-=-=-=-=-=-=-=-=-=-=-\n");
  Serial.print("Ready for commands.\n");
  Serial.print("*: Clears buffer.\n");

  for (char p : ledPins) pinMode(p, OUTPUT);

  // Attaches the servo pin to the latch object.
  latch.attach(servoPin);
}

String buffer;

void clearBuffer() {
  buffer = "";
}

// Takes project commands from user input.
```

```
void instructions() {
  while (Serial.available()) {
    char c = Serial.read();
    buffer += c;

    switch (c) {
      // Manual Buffer Clear
      case '*':
        Serial.print("Clearing Buffer...\n");
        Serial.print("Current Buffer:" + buffer + "\n");
        clearBuffer();
        Serial.print("Buffer Cleared!\n");

      case 'p':
        if (buffer == "help") {
          // Print a brief summary of supported commands.
          Serial.print("\nHelp Menu:\n");
          Serial.print("*: Clears buffer.\n");
          Serial.print("ok: Checks the Arduino status.\n");
          Serial.print("debug: Checks the Servo & LED status.\n");
          Serial.print("water: Checks the water sensor.\n");
          Serial.print("i: Locks the Smart Latch.\n");
          Serial.print("o: Unlocks the Smart Latch.\n");
          clearBuffer();
        }

      case 'k':
        //  Useful for checking if the Arduino is still alive.
        if (buffer == "ok") {
          Serial.print("OK\n");
          clearBuffer();
        }

      case 'g':
        // Debugs the Servo Motor (full rotation) & RGB LED (color shifting).
        if (buffer == "debug") {
          getServoState();
          getCurrentColor();
          clearBuffer();
        }

      case 'r':
        // Check the current water level.
        if (buffer = "water") {

          for (int checkDepth = 0; checkDepth < 10; checkDepth++) {
            getWaterLevel();

            if (sensorValue = 0) {
              Serial.println("[WATER EMPTY]");
              showColor(red);
            }

            else if (sensorValue > 0 && sensorValue <= 300) {
              Serial.println("[LOW WATER LEVEL]");
              showColor(low);
            }

            else if (sensorValue > 300 && sensorValue <= 500) {
              Serial.println("[MEDIUM WATER LEVEL]");
              showColor(medium);
            }

            else if (sensorValue > 500) {
```

```
              Serial.println("[HIGH WATER LEVEL]");
              showColor(high);
            }
          }

          clearBuffer();
        }

      case 'i':
        // Locks the Smart Latch.
        if (buffer == "i") {
          lock();
          Serial.print("Roll Cart: [LOCKED]\n");
          clearBuffer();
        }

      default:

      case 'o':
        // Unlocks the Smart Latch.
        if (buffer == "o") {
          unlock();
          Serial.print("Roll Cart: [UNLOCKED]\n");
          clearBuffer();
        }

    }
    delay(1000);
  }
}

void loop() {
  instructions();
}
```

5.2 Attached Images