MEAM 510: Design of Mechatronic Systems

Lab 4

Aslamah Rahman, Rahul Sharma, Pinak Ajay Kulkarni

February 5, 2019

Contents

1 Functionality

1.1 Minimum functionality

The overall car structure consists of a single base, with a tricycle configuration. Two independently controlled motors on the rear of the base govern the motion and direction of motion of the car frame, which are attached to its corresponding wheel. Two conventional (plastic) toy wheel were used as the driver wheels. The third lone wheel, a caster wheel was attached to the front of the base. Steering has been achieved by differential turning with variable turning radii, enabled via programming. The entire body was covered by means of soft spongy paper to protect the circuitry and add to the lightness of the robot.

Attacking capabilities are introduced by means of a hammer with a rotating base and arm, controlled by a servo each. A switch is placed one the face of the hammer to detect when a hit is made, whose output is directly connected to the weapon pin on the tophat $ESP32$.

The tophat displays the team number and color, and health as per the data provided by the central computer. Healing is achieved by means of 4 phototransistors, whose combined output is provided to the pin on the main $ESP32$ for detecting duration of a detected pulse. When the correct healing frequency is detected, healing animation is performed by pulsing the dead LEDs on the tophat between white and off-white colors.

1.2 Extra functionality

The following additional functionalities were integrated with the robot:

- Rotating and shielding hammer: The hammer base was rotated using a servo and position control was achieved using a potentiometer. The hammer arm also has a 180° range of motion also allowing to perform as a shield when attacked.
- LED matrix display: An LED matrix display of 4 blocks in parallel was used to indicate attacks and during the rest of the game. All throughout the game, the LED matrix displayed happy faces. When an attack is initiated, the LED matrix displayed sad faces.
- Smart autonomous mode: Navigation in autonomous mode is achieved by using an ultrasonic sensor mounted on a servo in the front of the robot. If the sensed distance between the robot and the any obstacle directly in its line of motion is less than 15 cm, the robot continues to move forward. Once it is not, the robot stops moving and the sensor rotates by 45° left and right, measures the distance of the next obstacle in both direction and chooses that which has the farthest obstacle.

2 Electrical Design

2.1 Robot

Keeping in mind the need to separate components based on their current draw, two batteries were used to power the entire robot. A 9V alkaline battery was used to power the two motors through the $SN754410$ Quadruple Half-H bridge and three servos. Separate $LM7805$ voltage regulators are used to get 5 V for internal logic transition for the H-bridge, and for providing 5 V VCC to the LED matrix, ultrasonic sensor and three servos. A 2-cell LiPo with current capacity of 2000 mAh was used to power the three $ESP32$s and the tophat. A common ground rail was established between both power sources and 100 μF capacitors was installed across the batteries to minimize noise. Also a capacitor of 0.1 μF was soldered across each motor and servo terminals to reduce noise from interfering with the rest of the signals in the circuit, especially the I2C communication. All wires were braided and twisted accordingly for maximum resistance to noise from motors, tophat LEDs and LED matrix display.

Connections on the perfboard have been made using IC sockets for the head $ESP32$, H-bridge and voltage regulator, two 2-pos screw terminals for the motors, one 6-pos 90° IDC header for the LiPo, three 3-pos 90° IDC header for the three servos, two 6-pos 90° IDC header for the alkaline battery and power to $ESP32$ for controlling servos and soldering connections as required.

Sensing of healing light was achieved using an four phototransistors connected in a circular fashion with a single output line to be able to sense from multiple positions. The phototransistor array was placed face down to detect light whenever the robot moved over the healing lights. Detection of hammer hit was achieved but connecting a button to the face of the hammer which would be pulled low when the button is pressed. The output of the button was connected to the weapon pin on the tophat $ESP32$.

3 Process & Code Architecture

An overview of the code architecture has been provided in Figure 2. Individual components have been discussed in detail in the following subsections.

3.1 Driving & hammer control

Control of the car is achieved using a custom made joystick, by sending signals using UDP from the $ESP32$ on joystick to $ESP32$ on car over WiFi. A 2-axis analog joystick manufactured by Adafruit was purchased for the same. Control of the servos attached to the base and arm of the hammer was achieved using potentiometers.

Upon receiving the signals from joystick, they are parsed back to an integer value by the respective microncontroller and used to control the velocity and direction for the motors and positions for the servos. Steering is achieved using differential drive, with two motors attached to the rear wheels and a caster wheel in front. To make a turn, custom turning radius is achieved by adjusting the duty cycle of the motors based on the X-axis (x) and Y-axis (y) values, which ranges from -255 to 255 where absolute value gives the magnitude and sign given direction, provided by the 2-axis joystick as per the equations below. The buffer range for not making any turns is PWM values between -50 and 50.

To turn forward right,

$$DC_{motor\ 1} = -\frac{255}{205}x + \frac{255^2}{205} \tag{1}$$

$$DC_{motor\ 2} = |y| \tag{2}$$

To turn forward left,

$$DC_{motor\ 1} = y \tag{3}$$

$$DC_{motor\ 2} = \frac{255}{205}x + \frac{255^2}{205} \tag{4}$$

$$\tag{5}$$

To turn backward right,

$$DC_{motor\ 1} = y \tag{6}$$

$$DC_{motor\ 2} = -\frac{255}{205}x + \frac{255^2}{205} \tag{7}$$

$$\tag{8}$$

To turn backward left,

$$DC_{motor\ 1} = |y| \tag{9}$$

$$DC_{motor\ 2} = \frac{255}{205}x + \frac{255^2}{205} \tag{10}$$

The hammer is given two degrees of freedom by means of two servos. The input from the controller received via WiFi is parsed back to the required angle between 0° and 180° that the hammer should rotate to.

3.2 Sensing healing light

Healing light detection was performed using the main $ESP32$ itself. Light sensing was done using a phototransistor whose analog input was captured using the ADC conversion, which will only be initiated if the measured ADC value is beyond a certain threshold. This check has been included to avoid overloading the code by the healing function when sufficient light is not detected. The duration of a pulse received was calculated by measuring the time duration between a consecutive rising and falling of the pulse, which was then converted to its corresponding frequency. The detected frequency was sent to the tophat $ESP32$ by setting the appropriate byte in the I2C bus.

3.3 LED matrix animation

LED matrix animation was done to indicate an attack attempt by the robot. Control of the LED matrix was shifted to the servo $ESP32$ to avoid inteference with the game by having it on the main $ESP32$. Throughout the game, the LED matrix displayed a happy face. When an attack was initiated, which is recognized by the hammer arm angle being less than 90°, the LED matrix displayed a sad face. To control the LED matrix, the *LedControl.h* library in Arduino was used, where the location of LEDs to be lit up for a certain frame of an animation is given by means of a binary array.

3.4 Communication architecture

Communication for driving the robot and rotating and oscillating the hammer is being relayed to its respective $ESP32$ via WiFi from the controller $ESP32$. Data from the tophat $ESP32$ is being relayed to the main $ESP32$ through I2C communication protocol. Data to the display LED matrix as explained in the previous section is being relayed from the $ESP32$ via SPI protocol.

4 Video Links

- `https://photos.app.goo.gl/kPeR4WrNTYGhZ3iF9`

5 Retrospective

6 Appendix

6.1 Bill of Materials

Item	Quantity
NodeMCU ESP32	4
Motors	2
Wheels	2
Capacitor (0.1 μF)	5
Capacitor (100 μF)	1
LM7805 voltage regulator	4
9V alkaline battery	1
2-cell LiPo battery (2000 mAh)	1
Caster wheel	1
Servos	3
SPDT switch	1
Joystick	1
Rotary potentiometer	2
Ultrasonic sensor	1
Phototransistors	4
Resistors (47 kΩ)	4
LED matrix	1
Perfboard	4
Standoffs	
Bolts, nuts, washers	
Acryilic board (4" and 8")	NA
2-pos, 3-pos, 6-pos headers	5
Total	

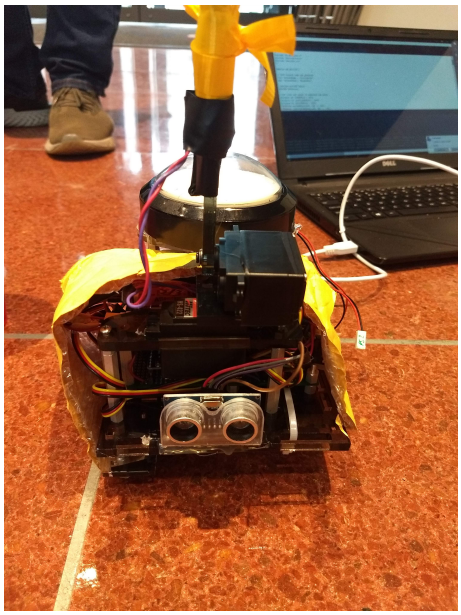
6.2 Electrical circuits

All electrical circuits and communication architectures are demonstrated in Figure 3.

6.3 CAD drawings

6.4 Datasheet for non-standard components

- Servos: `https://www.dfrobot.com/wiki/index.php/Hitec_HS422_Servo_(SKU:SER0002)`
- LED Matrix: `https://datasheets.maximintegrated.com/en/ds/MAX7219-MAX7221.pdf`
- Ultrasonic sensor: `https://www.mouser.com/ds/2/813/HCSR04-1022824.pdf`



(a) Top view

(b) Side view

(c) Rear view

Figure 1: Robot structure

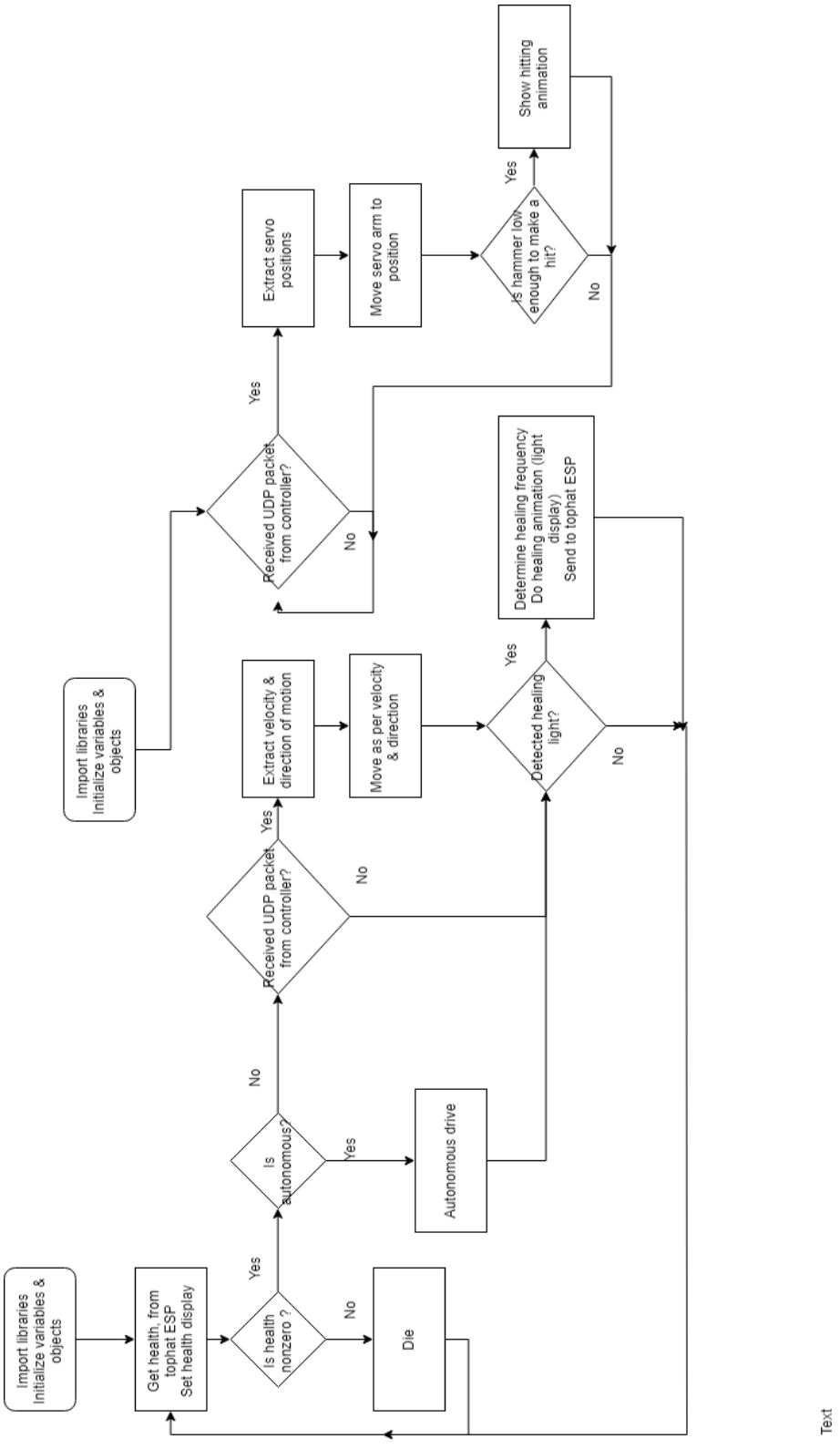


Figure 2: Process architecture

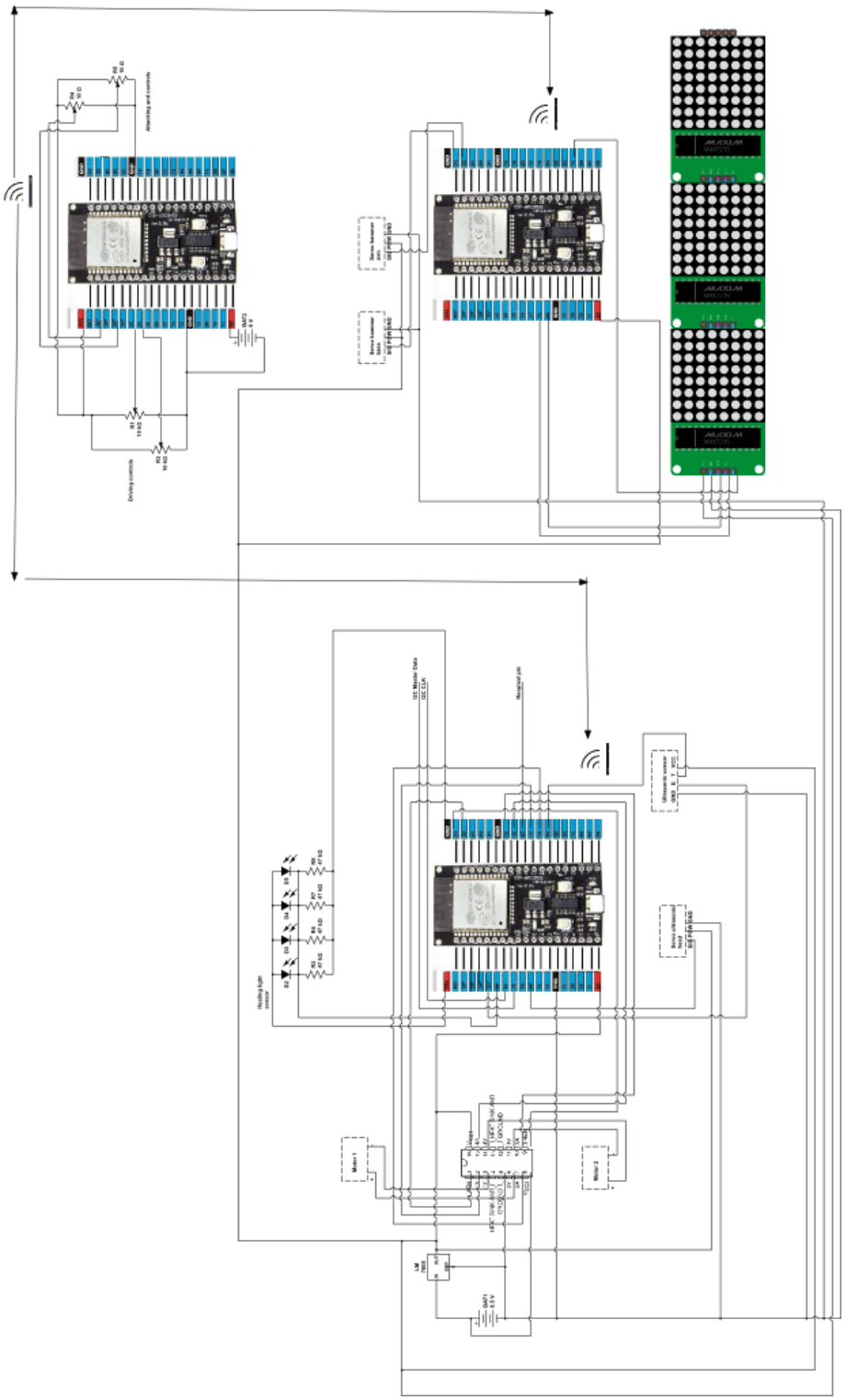


Figure 3: Circuit diagram