2017 Trinity College

Firefighting Robot Competition

Submitted by:

Woojin Lee, EE

Rene Perez, EE

EE495 Senior Project Seminar

University of Evansville

Advisor: Mark Randall

December, 6, 2016

Table of content

I. Introduction

The Trinity College Firefighting Robot Contest is held every year at Trinity College in Hartford, Connecticut. People from around the world compete in three stages of difficulty. The goal is to build a robot that is able to successfully advance through the different stages in the least amount of time with the least amount of penalties for the honor of winning a prestigious competition. In addition to competing, the competition allows contestants the opportunity to design machines with the potential to save lives. About seventy U.S. firefighters die in action and with the technology that this contest inspires, the options to solving this as a world problem are limitless.

II. Problem definition

A. Stage configurations

The arena consists of 4 separate rooms in an 8ft x 8ft track [1]. In all three stages the robot has to avoid obstacles and extinguish fire. The first level of the arena will be presented with high contrast walls and floors. The second will have additional features such as rugs, wall decorations, and wallpaper to simulate a realistic environment.

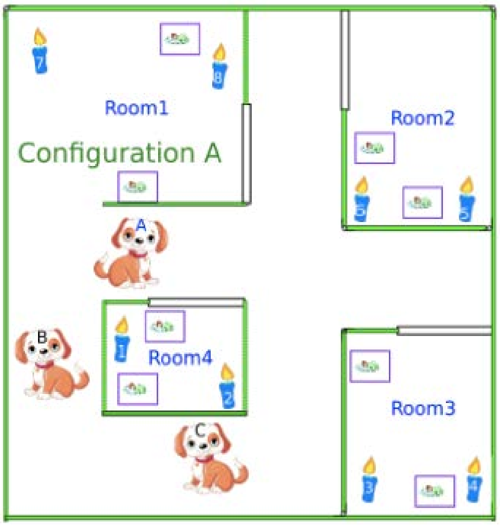

Figure1 Stage 1,2 [1]

The third stage is a combination of two arenas connected by a 1-meter pathway. To get through this last level, extinguish the fire and save the baby. A baby doll will be located on a cradle that has a certain color and pattern on the outside. The robot must use a computer vision based camera to recognize the side of the cradle with the pattern and color, take the baby out of the cradle, and safely move it on a net. During this process, the robot should avoid any contact with obstacles or wall contact.

There are several bonuses and penalties. For example, using something other than air to extinguish the flame is a bonus in as operating mode; running into the dog obstacle which is a penalty. The contest's objective is not only to help apply engineering knowledge, but to motivate high school and some middle school students to apply to STEM majors and see how fun and beneficial it can be.

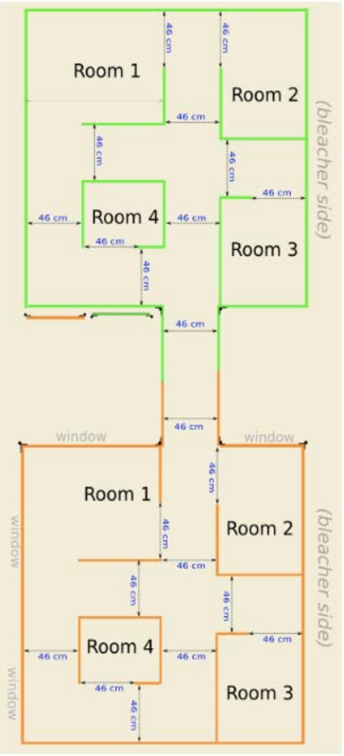

Figure 2. Stage 3 [1]

B. Scoring Guideline

Five trials are allowed per stage. The lowest scores would be computed for the overall performance of the robot [1]. After each factor is computed the Operating score would come out as: Time Score X Room Factor X Operating mode**.** After all stages are competed, the minimum score team would be take the first place.

Scoring methods for stage 1 and 2 are the same, but different in stage 3. In stages 1 and 2, scores are given by judges depending on the three elements: operating modes, measure actual time, and determining room factor. After each trial in the stages, the judge computes the lowest score as the robot's overall total performance in the contest.

1. Operating mode score (MF)

Operating modes are defined as multiplying the operating modes factor by time and there are 6 different operating modes that we could possibly aim to accomplish [1]. In each stage, the operating modes are optional, but to compete in the 3 stage, it is required. Our team has decided to compete all three stages and with all operating modes.

	Multiplier Factor	Explanation
Standard	X1.0	Standard mode
Arbitrary Start Location	X0.80	Random starting position
Return Trip	X0.80	Return back to starting position
Non-air extinguisher	X.75	Using other than air to extinguish fire
Furniture	X0.75	Furniture located in arena
Candle location	X0.75	Random candle location

Table 1. Operating mode Multiplier

2. Actual time(AT)

Actual time criteria is the part where the robot is timed in each trial. There will be a maximum of 300 seconds to complete the task and 120 seconds to return to starting position after extinguishing flames. If it takes more than 300 seconds, the judge will assign AT=600 for the score. If it gets stuck in a loop for more than 5 times in a row or does not move for 30 seconds, a score of AT=600 will be assigned. The less AT time received, the better the score.

3. Room Factor(RF)

A total of four rooms in the arena has to be searched for fire. Depending on the amount of rooms searched before finding the fire, the room factor differs. If the flame was out after search of

	Room factor
First room	1.0
Second room	0.85
Third room	0.50
Fourth room	0.35

Table 2. Room factor Multiplier

For example, if the robot found the flame in the second room it had entered, the room factor that it would receive would be RF=0.85.

4. Penalty Points (PP)

Penalty points will be given to the robot that violates the dimension rule of the competition [1]. There are specifically three different movements the robot cannot attempt during the competition. Touching the candle will be 50 points added to the actual time, kicking the dog would be 50 points added and continuous wall contact will be 0.5 points per centimeter.

5. Level 3 mode factors

Although stage three robots are required to handle all operating modes, it has three factor options. The first option is ‘using a computer vision with a camera’. Using a computer vision software to locate the cradle and rescue it to the safe zone will have an operating mode of 0.6. The second option is ‘Hallway option’. Since two arenas are connected by a hallway, teams can choose to have a flat or ramped platform for the hallway. The factor for choosing a ramp would be 0.9. The third option is ‘All Candles Option’ the task for this mode is to extinguish all candle lights in each room within rescuing the baby to the safe spot.

The safe zone would be indicated by a 10x10cm blue square with a 5cm diameter red circle. Outside of the indicated safe zone, it will have a net where the baby must land on.

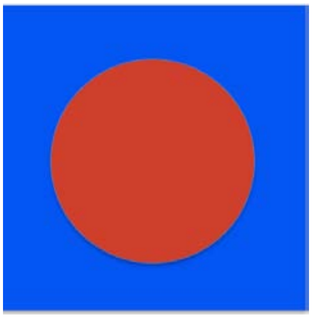

Figure 3. Safe Zone indication

C. Client Requirements

1. The dimension should not be bigger that 31 x 31 x 27 cm [1]
2. It must have a start button to activate the robot and a kill switch button to kill power.
3. The robot must activate at 3.8kHz with a microphone located on top of the robot.
4. A red LED should be present to indicate detection of fire
5. While activated, the robot must not touch obstacles, such as dogs and walls.
6. On top of the robot, it should have clear indication of the component as below.

Figure 4. Layout for components

III. Solution

A. Hardware

To achieve the desired results for this firefighting competition, make sure that everything is accomplished and completely fulfill all the requirements expected from the jurors. Major tasks could be identified as navigating through the platform and avoiding objects, locating the flame, extinguishing the flame, going back to base, locating the baby's cradle, rescuing the baby, and returning him to home base.

For navigation, two high precision DC motors and a motor driver board will be used to control said motors through the MCU 8051 [3]. The motors that will be used are the Pololu 12V DC motors with encoders included and a 100:1 gear ratio gearbox [2]. To effectively calculate how fast the robot will go, it is necessary to consider the radius of the wheels, the weight of the robot, and how much torque it needs to move at a certain speed.

Figure 4 Pololu 12 VDC motor with encoders and a 100:1 gearbox [2]

After all these factors are accounted for, choose and calculate the necessary revolutions per minute. The motors will be controlled by sending PWM signals to achieve different speeds and directions depending on the high time wanted. All this will be accomplished using an H-bridge which allows a set amount of voltage go to either motor, in this case two motors, so a dual h-bridge is needed. The H-bridge that is going to be used is the 2A Dual H-Bridge L298 [6], which is a widely-used H-bridge that can be applied to many different packages like the one in

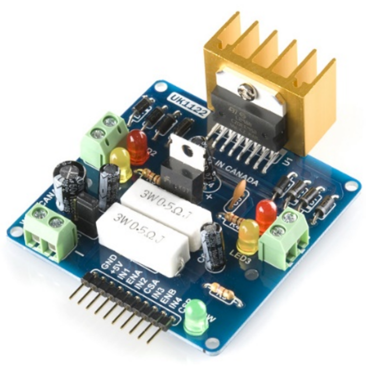

Figure 5 2A Dual H-Bridge L298 [6]

Fig. 2. Some of the features of this board are that it can handle 2A on each motor if needed (which probably will not be the case), it has a good heat sink, directional LEDs and a 5V regulator that can be used throughout the entire circuit and for any other applications that need it and it is cheap.

For the robot to not crash, sensors will be needed that will create the conditions for the changes in high time for the motors to go or not go. Using distance sensors made by SHARP to follow the right wall and whenever that distance grows assume that it is a room and go check it, else if it becomes a small distance from the front sensor assume that it is either a wall or an obstacle which must be avoided (turn 180 degrees if dog obstacle encountered else turn 90) [7]. These sensors are optimal for long distance applications since they measure down to 10cm which is equivalent to 3.1V, anything below 10cm doesn't get fed. To fix this problem, we make boundaries in software to avoid having the robot get too close to structure. This specific task of navigating and maintaining a certain distance from the wall can be accomplished with the use of only one sensor if that was a limit. To make this one sensor work a condition that limits the distance will be needed to steer the robot and follow the right wall. Something along these lines if(wall>4cm): steerRight() else: steerLeft() where "wall" is the distance measured by the sensor. The second task would be to locate the flame and extinguishing it.

Figure 6 GP2Y0A21YK SHARP sensor [7]

To locate and extinguish the flame the robot must be in a room with a lit candle. To detect the flame, use the UVTRON flame detector which is sensitive to ultraviolet emissions [8]. If there is a fire in its vicinity, then it will send pulses of 10 milliseconds to the microcontroller unit. If said fire is available then use a servo with a phototransistor attached to it to pinpoint the location, which will guide the servo towards the brightest spot in the room.

Figure 7 UVTron Flame Sensor [8]

Once the brightest spot in the room is found make the robot go towards the source and then using an EZ versa valve and some air or non-air gas cartridges extinguish the flame (if non-air gas like CO2 or another compound that can extinguish flames is used, there is a bonus in the competition) [9]. A code structure for this part of the track could be if (flame==1): extinguish() , else: leaveRoom(), this is a basic structure of what this task could be performed as.

Figure 8 EZ-1 Versa Valve [9]

Regarding going back to base, encoders included in the two DC motors should be used to track how far the robot has gone and keep a variable that counts turns made to get to the flame, and from there trace back our steps by using this relation between distance and turns to get back to home base.

Figure 9 General Use Servo

To rescue the baby, use computer vision software to identify some color patterns on a box which will be under the cradle with the baby in it. To achieve this, using a camera pi module, which interfaces with a Raspberry Pi MCU, identify the patterns and determine the side of the box that the camera is viewing and depending on that, position the robot to face the right direction to pick up the cradle using servo motors with a little arm on them so that they can lock into the cradle's handles.

Figure 10 Camera Pi module for the Raspberry Pi Platform

B. Software

Regarding software, the team has decided to go with three microcontrollers which are going to be used throughout the three stages of the competition. The three are going to be the ARM Cortex M4 STM34F07VG [4], the AT89CC03 (8052 MCU) [3], and the Raspberry Pi 3 [5]. The first listed will be used as a band pass filter to activate the robot's start routine through sound. The 3.8 kHz starting signal will enter through a microphone and then go through the ARM board and toggle a pin if said signal is in range. Once that happens the brain of the robot, the 8052 platform, will begin working in the order shown in the project diagram.

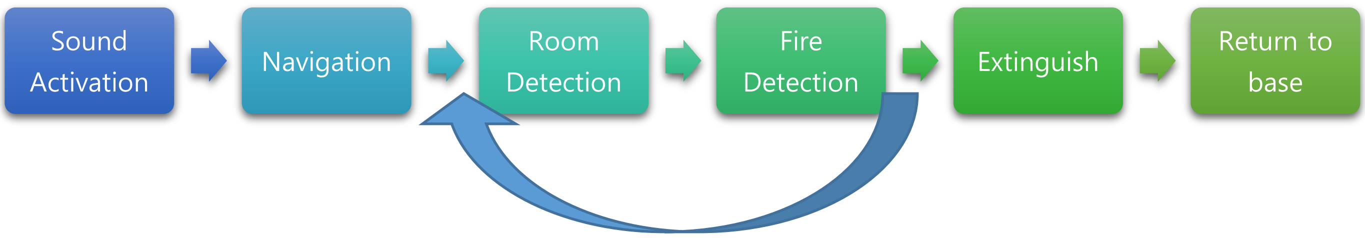

Figure 11 Project Flow diagram

Following the flow chart, sound activation comes first as mentioned previously. The signal comes through a FIR bandpass filter of 30^{th} order. This filter is developed with coefficients from MATLAB and then plugged into the FIR coefficients code for a bandpass filter. If the microphone picks up the 3.8kHz signal it will toggle a pin that is connected to the 8052.

Once one of the 8052's pins are toggled by the ARM board, proceed to the second block in the flow chart which is navigation. Navigation relies heavily on sensor reading, the eyes of the robot are the SHARP proximity sensors which are going to be read by using the analog to digital converter chip in the 8052 [7]. The sensors take 5VDC as input and return a voltage depending on the distance. There is going to be a total of three proximity sensors, two on each side and one on the front. By having said sensors in these positions, the conditions that must be set are easy to come up with.

Depending on these conditions and the obstacles present, the robot will turn either 90 degrees or 180 degrees. To drive the motors, use the Programmable counter array in the 8052 and use the PWM mode to change speed control and turn, drive, or stop.

To detect a room line following sensors will be used, these also return an ADC value just like the SHARP proximity sensors, but these go under the robot close to the floor. Room entrances have a white line; this white line will return a different value from black to the line sensor. Once the condition is met (white line) then it means that the robot is in the room.

In the room the robot will check the toggling of a pin from the UVTRON flame sensor, if the bit is toggled over a small period, then there is a fire, if there isn't the robot will proceed to follow the right wall and leave the room. If fire is in the room then the robot will proceed to pinpointing the source, by using a phototransistor which also returns voltage for the ADC to read and guide a servo which will move accordingly. To move the servo the PCA will be used once more.

When the fire source is located, the robot must go up to it and toggle the valve switch which will spray CO_2 gas on the source, the UVTRON flame sensor will check again for fire and if fire is still present then the previous processes will happen, else the robot will leave the room and go back to home base.

To go back to home base the robot must store values for the left and right encoders located on each of the DC motors and then trace back the path taken considering the turns made and the distance between each turn.

IV. Work

Part 1.

Our team is planning on ordering all the parts by the last week of October. The components of the robots that are needed:

- Motor driver(buy)
- Two motors(buy)
- IR sensor(stockroom)
- One set of tires(buy)
- Infrared Proximity Sensor(buy)
- UV Tron flame detector (previous team's)
- Gripper(buy)
- Computer vision camera(buy)
- Versa valve (given from competition)

IR sensors are available in the stock room of University of Evansville, and since UV Tron flame detector is no longer available for purchase it's been decided to use the previous project teams. The rest of the components should be researched and ordered getting ready for test.

Part 2.

In the first week of November testing of the parts. Check if the sensors and motor drivers are working as needed . IR sensors would be made by this period, infrared proximity sensor would give a practical result of distance/voltage aside of datasheet and UV Tron flame detector as well.

Part 3 navigation

After testing all the parts for the robots, navigation code and hardware. Getting the platform of the robot, wheels, motors, and Infrared Proximity sensors together, the result should be a navigating autonomous robot. Encoders will be used to return to the original position after completing the task.

Part 4. Sensors

UV Flame detector and servo for versa valve would be completed and get ready for a test trial. The UV Tron would detect the entire room for fire and an IR sensor would pick out the exact spot of the light. Start with having each section coded and running without navigation first and then get it all together.

Part 5. Sensors

Line sensors are important since it is an indication that the robot has moved into the room. This will be done last since it has been done previously for a similar project in EE454 class. Each room of the arena would have a line indicating that the robot has entered the room

Part 6. PCB Board

PCB will be used for the circuit board. In this stage all parts tested will have been tested with laid out circuitry for the PCB design

Part 7. Building parts

Once the PCB board is ready, the robot will be in one piece for the actual test on the track.

Part 8. 3D printer

The chassis is going to be design with 3d printer to cover the motors and wires so that it would look better.

Part 9. Baby Rescue

It’s the first time that the computer vision based camera section was added for stage 3. planning to work everything that would go through stages 1 and 2 first. For saving the baby use 2 servos to lift the whole cradle for saving the baby.

Number	Assignments	Achievements	Due Date
1	Order Parts	Order parts for project	Oct/4week
2	Test parts	Test each component if it works properly.	Nov/1week
3	Navigation	Encoder, Motors and H-drivers	Nov/1-2week
4	Sensors	UV Flame detector, and servo for versa valve	Nov/4week
5	Sensors	Line Indication sensors	Dec/1week
6	PCB Board	Order PCB board	Dec/3week
7	Building part	Put the parts together as an actual robot	Jan/4week
8	3-d printer	3-D print the chassis	Jan/4week
9	Baby part	actuators, computer vision, and grab the baby	Feb~march
		Get actuators working to grab and hold on to the baby	Feb~march
		Work on computer vision based camera to spot the cradle	Feb~march

Table 3. Schedule by dates.

V. Result

The robot managed to go through stage 1 and stage 2 flawlessly and could put out the candles at varying heights from 15cm to 20cm. Robot started when a 3.8kHz sound was played. Avoided all obstacles effectively and overall had good timing on the track. Robot completed some extra challenges as well. The Versa Valve challenge was completed as well as the variable candle location and using something other than water to extinguish the flame. Stage 3 was not successful due to the fact that developing the pattern recognition code was harder than expected and using the Haar cascade method, would require some sort of experience with other image processing software. Aside from that mishap everything went well and robot managed to perform consistently. The robot's construction has been thought from IEEE standards of both safety and batteries together with environmental solutions. The batteries used were rechargeable nickel metal-hydride batteries which create less of an impact on the environment than disposable ones. The robot is equipped with a kill power switch which would prevent mishaps that may happen so safety was something considered even by rules of the robot competition in which we finished thirteen out of thirty-five.

Matlab code for filter implemented in ARM board:

//Filter is from Dr. Blandford's EE311 class over bandpass filters.

```
fs = 11025;

sb1 = 2800; %Stopband 1 end

pb1 = 3230; %Passband start

pb2 = 3857; %Passband end

sb2 = 4700; %Stopband 2 start

ripP = 0.05; %Passband ripple

ripS = 0.05; %Stopband ripple

F = [sb1 pb1 pb2 sb2 ]; %Frequency array

M = [0 1 0 ]; %Gain array

Err = [ripS ripP ripS]; %Ripple array

idealF = [0 sb1 pb1 pb2 sb2 (fs/2)];

idealM = [0 0 1 1 0 0];

[N F A W] = firpmord(F, M, Err, fs); %Pick out order

N = 30;

b = firpm(N, F, A, W); %Create filter
```

```c
//Firefighting Robot 2017
//Rene Perez and Woo Jin Lee
//October 20, 2016

  #include <at89c51cc03.h>
  void check4fire();//Uses the UVTron to check for fire P3_5
  void move(int ht);//Function used to actually move the servo
  void delay (unsigned char msec);//Debugging delay
  void delayshorten(int d);
  void stopmotor();//All CCAPXH channels are set to 0% hightime
  void forward();//Right (CCAP3H) and Left (CCAP4H) are set to 50%
  void turnright();//Left is 75% while Right is 0%
  void turnleft();//Left is 0% while Right is 75%
  void correct();//If getting to close to wall correct to stay middle
  void correct2();
  void obstacle();//Obstacle detectes so we are going to turn around
  void turn180();//Turn 180 degrees and go forward
  void evacroom1();//After turning off fire return home
  void Set_Everything_Up();//Sets pin functions, clock speed and interpt enabler
  void extinctor();//While there is fire, find brightest spot once proximity is reached and blow off the fire.
  void linecase();//This function positions the robot depending on which room the fire is at a spot where reading fire is easily
                 //accessible.
  unsigned char ReadADC(unsigned char chan);//ADC reading function, input channel wanted
  unsigned char Right, Left, Front, Line, diff,diff2, firesensor;//ADC values for sensors
  unsigned char Setpoint=176;//Set value used to set a threshold and correction factor to follow walls in a straight manner
  int hightime;//Hightime for the servo
  int lineroom=0;//Room line counter, used to keep track of where we are
  int fire=0;//Once this is set, fire has been neutralized
  int k=0;
  int counturn=0;//Variable used to count how many turns there are once we leave lineroom 4 to go into following right wall instead of left.
  unsigned int count=0;//Used to count for  a certain period of time to follow right wall after lineroom 4
  //Note: Save ADC pins for the sensors
  void main (void)
  {
    Set_Everything_Up();//All specific pin setups and PCA setups done here, plus initial conditions.
    while (1)
    {
      //check4fire();
      if (P3_0==0)//If sound activation pin is cleared
      {
        P3_3=1;
        CCAPM0=0x49;//Enable intrpt bt, Match bt ,and Compare bt
        hightime=5600;//hightime for servo
        move(hightime);//Set the servo at mid position
        if (lineroom==0)
        {
//          Right=ReadADC(0x01);//Read P1_1 Wall Sensor
//          Left=ReadADC(0x02);//Read P1_2 Wall Sensor
//          while(Left>Right)
//          {
//            CCAP2H=110;
//            CCAP3H=110;
//            CCAP4H=255;
//            Right=ReadADC(0x01);//Read P1_1 Wall Sensor
//            Left=ReadADC(0x02);//Read P1_2 Wall Sensor
//          }
          for(k=0;k<2;k++)
          {
        for (count=0;count<60000;count++)
        {
            Right=ReadADC(0x01);//Read P1_1 Wall Sensor
            Left=ReadADC(0x02);//Read P1_2 Wall Sensor
```

```
                Front=ReadADC(0x03);//ReadP1_3 Wall Sensor
                diff=Right-Setpoint;
                if (Right>=140 && Right<=176)
                  {
                    CCAP2H=255;
                    CCAP3H=50;
                    CCAP4H=50;
                  }
                  else if (Right>= 176)
                  {
                    correct();
                  }
                  else if(Front>=180)
                  {
                    obstacle();
                    delayshorten(4);
                    stopmotor();
                    delayshorten(2);
                  }
                  else if (Right<140)
                  {
                    turnright();
                  }

          }
         }
       }
        while(1)
        {
        Right=ReadADC(0x01);//Read P1_1 Wall Sensor
        Left=ReadADC(0x02);//Read P1_2 Wall Sensor
        Front=ReadADC(0x03);//ReadP1_3 Wall Sensor
        Line=ReadADC(0x04);//Read P1_4 Line Sensor
        diff=Right-Setpoint;//Correcting factor
          while(fire==1)//If fire is set stop the motors
          {
            stopmotor();
          }
          if (Right>=165 && Right<=180)//Sensor conditions for going forward.
          {
            forward();
          }
          else if (Front>=176)//Sensor conditions for turning around and avoid crash.
          {
            obstacle();
            delayshorten(4);
            stopmotor();
            delayshorten(2);
            if(lineroom==4)
            {
              counturn++;
            }
          }
          else if (Line<=10)//Sensor conditions for stopping and analyzing a room
          {
            stopmotor();
            delayshorten(4);
            lineroom++;//Everytime you see a line make sure to increase counter
            check4fire();//UVTron checks room
            //The following if statements are to exit said areas more efficiently
            if (lineroom==1)
            {
             turnleft();
             delayshorten(4);
             forward();
             delayshorten(6);
            }
            else if (lineroom==2)
```

```
            {
              stopmotor();
              delayshorten(2);
            }
            else if (lineroom==3)
            {
              forward();
              delayshorten(4);
            }

            else
            {
              stopmotor();
            }
          }
          else if (lineroom==4 && counturn==2)
          {
              while(Left<150)
              {
                Right=ReadADC(0x01);//Read P1_1 Wall Sensor
                Left=ReadADC(0x02);//Read P1_2 Wall Sensor
                Front=ReadADC(0x03);//ReadP1_3 Wall Sensor
                diff=Right-Setpoint;
                if (Right>=140 && Right<=176)
                  {
                    CCAP2H=255;
                    CCAP3H=75;
                    CCAP4H=75;
                  }
                  else if (Right>= 176)
                  {
                    correct();
                  }
                  else if (counturn==3)//Condition to get out of this loop and keep following the
right wall
                  {
                    break;
                  }
                  else if(Front>=180)
                  {
                    obstacle();
                    delayshorten(4);
                    stopmotor();
                    delayshorten(2);
                  }
                  else if (Right<140)
                  {
                    turnright();
                  }
              }
              while(fire==0)//Follow left wall
              {
                Left=ReadADC(0x02);//Read P1_2 Wall Sensor
                Front=ReadADC(0x03);//ReadP1_3 Wall Sensor
                Line=ReadADC(0x04);//Read P1_4 Line Sensor
                diff=Right-Setpoint;
                diff2=Left-Setpoint;
                if (Left>=176 && Left<=200)
                  {
                    forward();
                  }
                  else if (Front>=170)
                  {
                    obstacle();
                    delayshorten(3);
                    stopmotor();
                    counturn++;
                    delayshorten(6);
```

```c
                }

                  else if (counturn==3)//Condition to get out of this loop and keep following the right wall
                  {
                    break;
                  }
                  else if (Left> 200)
                  {
                    correct2();
                  }
                  else if (Line<=10)
                  {
                    stopmotor();
                    delayshorten(4);
                    lineroom++;
                    check4fire();
                  }
                  else if (lineroom==5)//If no fire is found in any of the rooms just stop
                  {
                    fire=1;
                    return;
                  }
                  else if (Left<176)
                  {
                    turnleft();
                  }
                  else
                  {
                    stopmotor();
                  }
                }
          }
        else if (Right>= 180)//Sensor conditions for correcting the robot position
        {
          correct();
        }
        else if (Line<=10)//Sensor conditions for stopping and analyzing a room
        {
          stopmotor();
          delayshorten(4);
          lineroom++;//Everytime you see a line make sure to increase counter
          check4fire();//UVTron checks room
          //The following if statements are to exit said areas more efficiently
          if (lineroom==1)
          {
           turnleft();
           delayshorten(4);
           forward();
           delayshorten(6);
          }
          else if (lineroom==2)
          {
            stopmotor();
            delayshorten(2);
          }
          else if (lineroom==3)
          {
            forward();
            delayshorten(4);
          }

          else
          {
            stopmotor();
          }
        }
        else if (Right<165)
```

```
        {
          turnright();
        }
        else if (fire==1)//If fire has been turned off
        {
          return;
        }
        else if (counturn==2)
        {
          break;
        }
        else
        {
          stopmotor();
        }
       }
      }
     }
   }

  void check4fire()//UVTron Flame check
  {
    int i;//counting variable to determine if flame
          //output level is either 5V or GND by 10ms
          //LED will indicate presence of candle
    for(i=0;i<60000;i++)//Measuring a bigger chunk of time in order to determine if 1 or 0
                        //due to pulse width.
    {
      if (P3_5==0)//If UVTron returns a zero there is a flame
      {
        linecase();//Position the robot at a postion that will allow for the fire to be visible to robot
        firesensor=ReadADC(0x00);//Read channel 0 of ADC
        while(firesensor>85)//Condition for the light receiving diode to detect fire, while the conditions true, search for fire 360
        {
          firesensor=ReadADC(0x00);
          CCAP2H=100;
          CCAP3H=100;
          CCAP4H=255;
        }
        stopmotor();
        delayshorten(5);
        Front=ReadADC(0x03);//ReadP1_3 Wall Sensor
        while(Front<160)//Condition to make the robot stop in front or close enough to candle to blow out
        {
          CCAP2H=255;
          CCAP3H=110;
          CCAP4H=110;
          Front=ReadADC(0x03);//ReadP1_3 Wall Sensor
        }
        stopmotor();
        delayshorten(3);
        extinctor(); //Precisely extinguish the flame
        fire=1;
        while(fire==1)//Stop the motors forever after fire is off.
        {
          CCAP2H=255;
          CCAP3H=255;
          CCAP4H=255;
        }
      }
      else //There is no fire in the room, get out move on
      {
            fire=0;
            turn180();
            return;
```

```c
      }
    }
    i=0;//reset counter
  }
  void move (int ht)//Servo Move function Using PCA
  {
    CCAP0L= ht%256;
    CCAP0H= ht/256;
    CR=1;//start timer
  }
  void delay(unsigned char msec)//function for delay
  {
  int j,i;
  for(i=0;i<msec;i++)
   for(j=0;j<1250;j++);
  }
  void delayshorten(int d)//Simple delay function that takes previous function and runs it several times for a prolonged period of time
  {
    int i;
    for(i=0;i<d;i++)
    {
      delay(5000);
    }
  }

  unsigned char ReadADC(unsigned char channel)
  {
      unsigned char sensor;
      ADCON |= 0x20;//Enable ADC
      ADCON &= 0xF8;//Clear channel select
      ADCON |= channel;//Set ADC channel
      ADCON |= 0x08;//Start ADC conversion
      while(!(ADCON & 0x10));//Wait until conversion is done
      sensor =ADDH;
      ADCON &= 0xEF;//Software flag, bt 4 has to be cleared by software
      return sensor;
  }

  void stopmotor()
  {
    CCAP2H=255;
    CCAP3H=255;
    CCAP4H=255;
  }

  void turnright()
  {
    CCAP2H=255;
    CCAP3H=255;
    CCAP4H=25;
  }

  void turnleft()
  {
    CCAP2H=255;
    CCAP3H=25;
    CCAP4H=255;
  }
  void forward()
  {
    CCAP2H=255;
    CCAP3H=75;
    CCAP4H=75;
  }
  void obstacle()
  {
    CCAP2H=5;
```

```c
    CCAP3H=5;
    CCAP4H=255;
  }
  void correct()
  {
    CCAP2H=255;
    CCAP3H=75-diff;
    CCAP4H=75+diff;
  }
  void correct2()
  {
    CCAP2H=255;
    CCAP3H=75+diff2;
    CCAP4H=75-diff2;
  }
  void turn180()
  {
    obstacle();
    delayshorten(6);
    stopmotor();
    delayshorten(2);
  }

  void extinctor()//While there is fire, find brightest spot once proximity is reached and blow off the fire.
  {
    while(P3_5==0)//while there is fire present
    {

      unsigned char firesensor;
      unsigned char oldfiresensor=20;
      CCAPM0=0x49;//Enable intrpt bt, Match bt ,and Compare bt
      hightime=2350;//hightime for servo
    //0.20/(2.5/255) where 2.5 is biggest and 0.20 is the limit
      while (hightime<11750)
      {
        firesensor= ReadADC(0x00);//Read Channel 0 of the ADC
        if (firesensor<=10 && oldfiresensor>firesensor)//comparison to threshold and previous value
        {
        CR=0;//Stop and extinguish fire
        P3_4=0;//Blast with versa valve
        delay(1000);
        }
          else
          {
          oldfiresensor=firesensor;//this will help keep looking for the brightest spot.
          move(hightime);//keep moving the servo until brightest source found
          delay(2);
          hightime=hightime+4;//increase the high time to keep moving from left to right or vice versa xc
          P3_4=1;//Close the valve
          }
      }
    }
    return;
  }
  void linecase()//This function positions the robot depending on which room the fire is at a spot where reading fire is easily
                  //accessible.
  {
          if (lineroom==1)
          {
            CCAP3H=110;
            CCAP4H=110;
            delayshorten(20);
          }
          else if(lineroom==2)
```

```
            {
              CCAP3H=110;
              CCAP4H=110;
              delayshorten(15);
              turnright();
              delayshorten(5);
              CCAP3H=110;
              CCAP4H=110;
              delayshorten(5);
              stopmotor();
              delayshorten(4);
            }
            else if (lineroom==3)
            {
              CCAP3H=110;
              CCAP4H=110;
              delayshorten(25);
              stopmotor();
            }
            else if (lineroom==5)
            {
              CCAP3H=110;
              CCAP4H=110;
              delayshorten(5);
              stopmotor();
            }
            else
            {
              stopmotor();
            }
    }

    void Set_Everything_Up()
    {
      ADCF=0x1F;//P1.0-4 for ADC sensors
      ADCON=0x20;//Enable the ADC function
      ADCLK=0x08;//Moderate speed of clock
      ADCON &= 0xF8;//Clear Channel Select
      CKCON=0x01;//Set to select 6 clock periods per machine cycle. x2 mode
      CMOD=0x01;//PCA overflow it
      CCAPM2=0x42;//Compare, PWM mode P1_5
      CCAPM3=0x42;//Compare, PWM mode P1_6
      CCAPM4=0x42;//Compare, PWM mode P1_7
      IEN0=0xC0;//PCA intrpt bt and enable all
      CR=1;//Intrpt enable
      stopmotor();//Make sure the motors are not stoping
      P3_4=1;//Make sure the versa valve is closed
      P3_3=0;
      Setpoint=176;
    }

    void toggle (void) interrupt 6 using 1 //PCA interrupts bank 1
    {
      if (CF)
      {
        IEN0= 0x00;
        P3_1=1;
        CF=0;
        IEN0=0xC0;
      }
      if (CCF0)
      {
        IEN0= 0x00;
        P3_1=0;
        CCF0=0;
        IEN0=0xC0;
      }
    }
```

References

1. Trinity College. (2016, November 28). Rules Firefighting Robot. Retrieved August 6, 2016, from Trinity College, http://www.trincoll.edu/events/robot/rules.html

2. Corporation2016Pololu. (2001). 100: 1 metal Gearmotor 37Dx57L mm with 64 CPR encoder (no end cap). Retrieved December 6, 2016, from Pololu, https://www.pololu.com/product/1446/specs

3. Atmel. AT89C51CC03. Retrieved December 6, 2016, from Atmel AT89C51CC03, http://csserver.evansville.edu/~blandfor/EE354/AT89C51CC03.pdf

4. ARM M4 Cortex. (2015, September 29). ARM M4 Cortex. Retrieved December 6, 2016,from http://www.st.com/content/ccc/resource/technical/document/datasheet/ef/92/76/6d/bb/c2/4f/f7/DM00037051.pdf/files/DM00037051.pdf/jcr:content/translations/en.DM00037051.pdf

5. Raspberry Pi. Raspberry Pi 3 Model B. Retrieved December 6, 2016, from http://uk.rs-online.com/webdocs/1521/0900766b81521bab.pdf

6. L298 H. L298 Dual H-bridge Bidirectional Motor Driver Board. Retrieved December 6, 2016, from http://www.canakit.com/Media/Manuals/UK1122.pdf

7. SHARP. (2005). GP2Y0A21YK Optoelectronic Device. Retrieved December 6, 2016, from http://www.sharpsma.com/webfm_send/1208

8. UVtron R9454 w/ breakout board. (2013, June). Retrieved December 6, 2016, from http://www.hamamatsu.com/resources/pdf/etd/C10423_TPT1023E.pdf

9. Versa Valves. Versa Valves E Family. Retrieved December 6, 2016, from

http://www.versa-valves.com/sites/default/files/catalogs/LIT-E-USA_042016.pdf#page=10