Midterm Project

Drexel University

To: Dr. Christopher Peters

From: Ananiya Deneke

Date: 01/11/2022

Re: ECE 303 Lab Memo

Midterm Project

Purpose

The purpose of this midterm project was to design a simple battleship game with serial communications between MATLAB and Arduino.

Parts Required

Hardware:

Breadboard, Analog Module, Arduino

Software:

MATLAB, Arduino IDE

Functionality

In this experiment, we will be designing a simple battleship game using MATLAB and Arduino. The game starts by pressing the start button on our GUI and it will start by displaying the user grid and prompt the user to place their ship. After that, the CPU will randomly place its ship of the same size on a different grid. The ships are placed horizontally or vertically on the grid and can occupy any number of grid boxes. Once it displays the CPU grid, it will prompt the user to guess where the enemy ship is. If the user guesses right, the corresponding grid box of the CPU containing the ship turns red and indicates that it's a "HIT" and if the user guesses wrong, the corresponding grid box turns white and indicate a "MISS". Once you guess where the CPU ship was placed and sink it (turn red), you win the game. We will be using the Joystick module in our Arduino kit to move the cursor around the grid boxes

Experiment

For the hardware part of this experiment, we connected a Joystick module to the Arduino. The Joystick module had 5 pins. One pin was connected to GND, the second one was connected to +5V, the third and fourth ones were connected to Analog Input pins A0 and A1. The last pin was the switch pin, and it was connected to digital output Pin 2,

For the software part of the experiment, I started by writing Arduino code on the Arduino IDE that maps the input from the Joystick to a smaller range so we can be more accurate when reading its values. Since the Joystick module has two inputs, specifically horizontal and vertical axes, we read those inputs and stored them in variables that we output and pass onto MATLAB through serial port.

Midterm Project

In MATLAB, I started by setting up a GUI with a start button, when the start button is pushed the game starts and we execute the main block of code. After setting up the Start Button in the GUI, we plot two grids, one is for the user to place ships and the other grid is for the CPU to randomly place its ship. The user places the ships using the values read from Joystick using the serial port communication and the CPU places its own ship under certain constraints given by if-statements, namely, The Player and CPU should have the same type and number of ships, ships can only be placed horizontally and vertically (not diagonally) , ships cannot exceed the bounds of the grid (limited play space) and ships cannot overlap on the same grid point After ships are placed, we construct a matrix the keeps track of which grids the ships are placed in and we prompt the user to guess where the ships are located, if the user guesses the right grid box it will turn red and indicate a "HIT" and a corresponding 1 will be assigned to the tracker matrix, otherwise it will turn white and indicate a miss. When all the grid boxes that contain ships turn red the game is over. We used the tracker matrix to keep track of which grid boxes have ships, if the ships in those boxes have been hit, and if all boxes with ships have been hit.

Conclusion

The purpose of this experiment was to design a simple battleship game using serial communications between Arduino and MATLAB. I wrote the main block of code for the game using MATLAB and used the Arduino IDE to setup and use the input Joystick for serial communication. Using MATLAB, I set up two grids, one for the user to place their ship and the other one to monitor the boxes they've sunk on the CPU. The CPU and the user have similar ships and the CPU's ship is randomly generated under certain constraints for valid positions. Once all grid boxes with ships are hit, the game is over.

Appendices

Arduino code:

```
int VRx = A0;
int VRy = A1;
int SW = 2;

int xPosition = 0;
int yPosition = 0;
int SW_state = 0;
int mapX = 0;
int mapY = 0;

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

  pinMode(VRx, INPUT);
  pinMode(VRy, INPUT);
  pinMode(SW, INPUT_PULLUP);

}

void loop() {
  if(Serial.available()>0){
  mapX= Serial.parseInt();
  mapY= Serial.parseInt();
  xPosition = analogRead(VRx);
  yPosition = analogRead(VRy);
  SW_state = digitalRead(SW);
  mapX = map(xPosition, 0, 1023, 0, 5);
  mapY = map(yPosition, 0, 1023, 0, 5);
  Serial.println(mapX);
  Serial.println(mapY);


 delay(100);
  }
  }
```

MATLAB code:

```
        % Button pushed function: STARTGAMEButton
        function STARTGAMEButtonPushed(app, event)
close all
clear all
figure
axis equal off
hold on
running= 1; % true if game is running, false otherwise
n= 5; % grid size
ships= {2,3}; %boxes occupied by ship
arduino = serialport("COM3",9600,"Timeout",15);
pause(1)
int flag = 0;
% Draw grid
for a=0:n-1
    for b=0:n-1
        fill([a a+1 a+1 a a],[b b b+1 b+1 b],[ 0.5843 0.8157 0.9882
    end
end
    titleText = 'User grid pick ship position';
    title(titleText, 'FontSize', 14');
    %Pick user ship, maximum size of ship is 3
    while ( flag <= 3)
    flush(arduino)
    write(arduino,2,'string');
    pause(0.5)
    c = read(arduino,4,'string');
    d = read(arduino,4,'string');
    flush(arduino)
    c = str2double(c);
    d = str2double(d);
    C= ceil(int8(c));
```

```
    D= n-ceil(into(d))+1;
    ship_slection= zeros(n);
    if ship_slection(C,D) == 0
         ship_slection(C,D)=1;
   end
     if(ship_slection(C,D) ~= 0)
        fill ([C-1 C C C-1 C-1],[n-D n-D n-D+1 n-D+1 n-D], 'k');
     end
     flag = flag + 1;
    end
   pause(1)

figure
axis equal off
hold on
for i=0:n-1
    for j=0:n-1
        fill([i i+1 i+1 i i],[j j j+1 j+1 j],[ 0.5843 0.8157 0.9882]
    end
end

% matrix that keeps track if square is occupied, a
check_slection= zeros(n);
place_cpu_ship = ceil(rand*2); %1 means vertical, 2 means horizontal
[numShips d] = size(ships);

% generate cpu ship position
for ship_chosen= 1:numShips
    shipSize= ships{ship_chosen,2};

    %generate valid position
    valid = 0;
    while valid == 0
        if place_cpu_ship == 2
            x= ceil(rand*(n-shipSize));
            y= ceil(rand*n);
            if sum(check_slection(x:x+shipSize-1,y)~=0) == 0
                valid = 1;
            end
        elseif place_cpu_ship == 1
            x= ceil(rand*n);
            y= ceil(rand*(n-shipSize)); n
            if sum(check_slection(x,y:y+shipSize-1)~=0) == 0
                valid = 1;
            end
```

```
        end
    end

    % place ship
    if place_cpu_ship == 2
        for remaining_squares=0:shipSize-1 %remaining squares of the sh
            check_slection(x+remaining_squares,y) = ship_chosen;
        end
    elseif place_cpu_ship == 1
        for remaining_squares=0:shipSize-1 %remaining squares of the sh
            check_slection(x,y+remaining_squares) = ship_chosen;
        end
    end
end

checker= zeros(n);
% 0 means covered, 1 means uncovered
count = 0; %initialise number of moves to be 0
titleText='Battleship';
countText= sprintf('Moves Made: %d',count);
while running== 1
    title({titleText, countText}, 'FontSize', 14');
    [x,y] = ginput(1);
    X= ceil(x);
    Y= n-ceil(y)+1;
    % user clicks on a square
        if checker(X,Y)==0 % square is covered, then uncover it
            checker(X,Y)= 1;
            count = count + 1;
            if check_slection(X,Y)~=0 % if square contains part of a sh
                titleText= 'Hit!';
                countText= sprintf('Number of Moves Made: %d',count);
                ships{check_slection(X,Y),2}= ships{check_slection(X,Y)
                fill ([X-1 X X X-1 X-1],[n-Y n-Y n-Y+1 n-Y+1 n-Y],'r');
            else % if square does not contain part of a ship
                titleText= 'Miss!';
                countText= sprintf('Number of Moves Made: %d',count);
                fill ([X-1 X X X-1 X-1],[n-Y n-Y n-Y+1 n-Y+1 n-Y],'w');
            end
        end

        %check if any ship has been sunk
    sumS= 0;
    for ship_chosen= 1:numShips
```

```
   for ship_chosen= 1:numShips
       if ships{ship_chosen,2}==0
           ships{ship_chosen,2} = -1; %-1 means that ship has been sur

       end
       sumS= sumS + ships{ship_chosen,2};
       if sumS== -1 * numShips     %all ships have been sunk
           titleText= 'YOU WIN!';
           title({titleText, countText}, 'FontSize', 14');
           running = 0;
       end
   end
nd
nd
elete(arduino)
```