PROJECT 1 REPORT

Introduction to intelligent systems

Satyam Patel
A99053252

Academic Integrity

Academic Integrity Policy: Integrity of scholarship is essential for an academic community. The University expects that both faculty and students will honor this principle and in so doing protect the validity of University intellectual work. For students, this means that all academic work will be done by the individual to whom it is assigned, without unauthorized aid of any kind. By including this in my report, I agree to abide by the Academic Integrity Policy mentioned above.

Problem 1

MATLAB Review

i.)

```
A = [2 59 2  5
    41 11 0  4
    18 2  3  9
    6  23 27 10
    5  8  5  1]

B = [0 1 0 1
     0 1 1 1
     0 0 0 1
     1 1 0 1
     0 1 0 0]
```

ii.)

```
C =

    [0    59     0     5

     0    11     0     4

     0     0     0     9

     6    23     0    10

     0     8     0     0]
```

iii.)

```
ans =

    88
```

iv.)

Max:

```
row =
     1
col =
     2
V =
    59
```

Min:

```
row =
     1  2  3  5  3  1  2  3  4  5  5
col =
     1  1  1  1  2  3  3  3  3  3  4
V =
     0
```

v.)

```
D =
     0     0     0     0
     0   -48     0    -1
     0   -59     0     4
     6   -36     0     5
     0   -51     0    -5
```

vi.)

```
Max:
row =
     4
col =
     1
V =
     6
Min:
row =
     3
col =
     2
V =
   -59
```

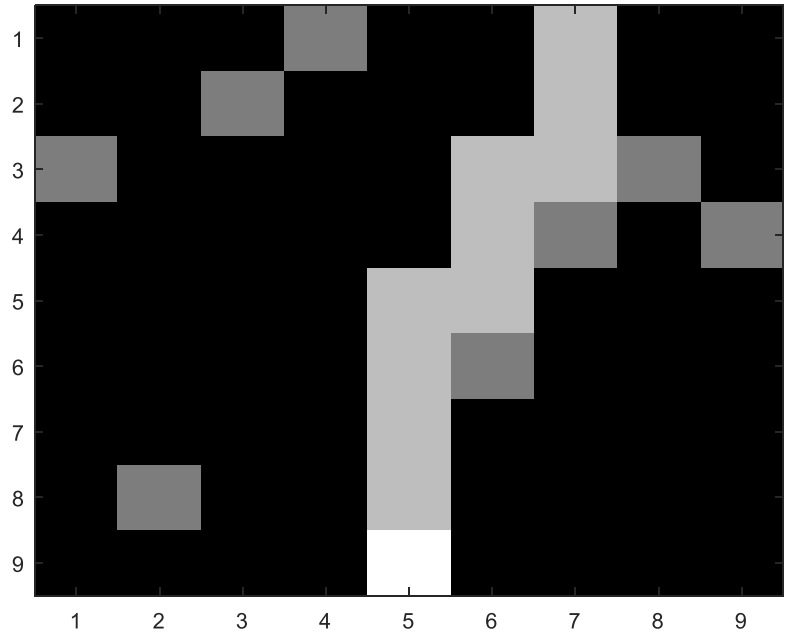
Problem 2

Robot obstacle avoidance

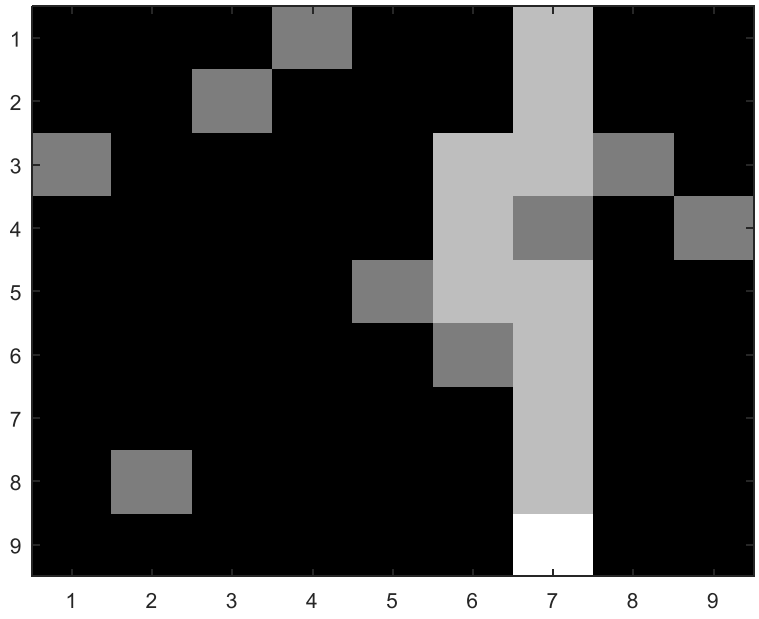
i.) Initially, the robot script does not exhibit the properties of intelligence. Using the **loc** variable to keep track of its position, the robot can move by adding vectors of the form **[a b].** For example, adding the vector [1 0] moves the robot down by 1 unit. However this is all the robot can do. When placing an obstacle at **[4 7]** we see that the robot makes no attempt at circumvention. This is due to no logical statements to deal with obstacles in its path. As a result this robot cannot be classified as intelligent since it cannot traverse all possible traversable maps.

ii.) The addition of the **if statement** improves the functionality of the robot to some extent. It allows for the robot to avoid obstacles directly beneath it as long as there is an empty path to the left. This increases its intelligence since it can now avoid obstacles of this nature.

Press any key to continue. Ctrl+C (or Cmd+C) to stop simulation.

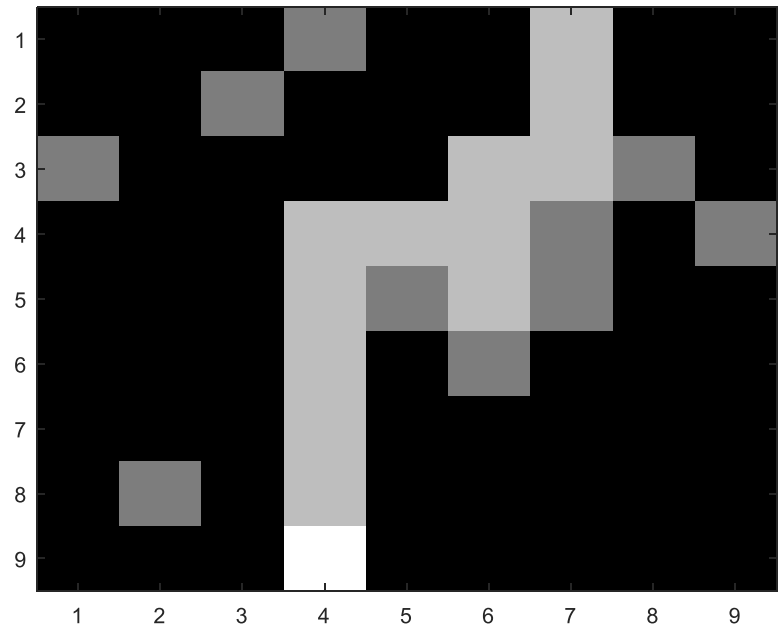


Press any key to continue. Ctrl+C (or Cmd+C) to stop simulation.



iii.)

Press any key to continue. Ctrl+C (or Cmd+C) to stop simulation.

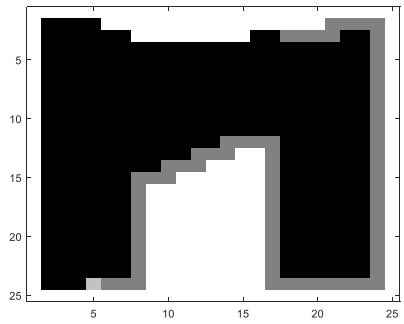


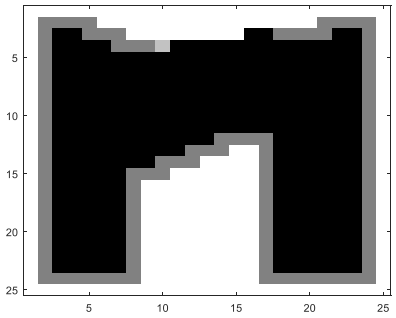
iv.)

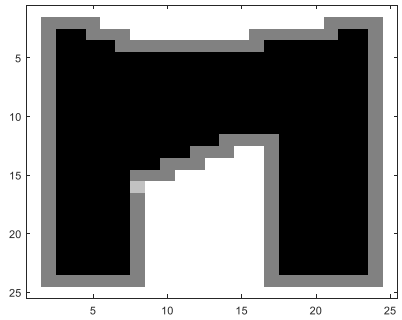
Problem 3

Edge traversal

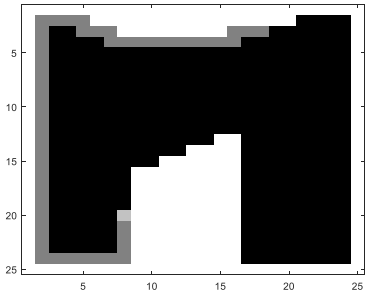
Clockwise:

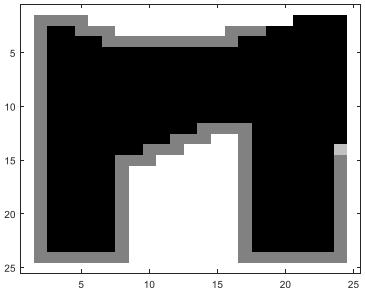


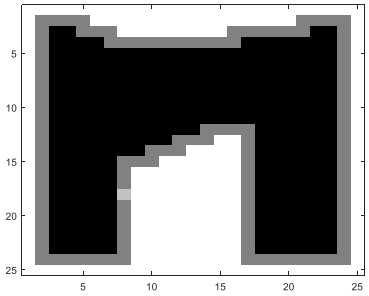




Anti-clockwise:







Problem 4

Report on Cosmo the intelligent toy robot

Science fiction has always expressed an infatuation with artificial intelligence throughout its time. From Asimov's I-Robot to Pixar's Wall-e the concept has always been a crucial element in most futuristic stories. Although we are still far from creating truly intelligent or so called "strong A.I." systems in real life, researchers and engineers are constantly breaking new grounds. One particularly exciting application and consumer product is Cosmo the intelligent toy robot made by the startup Anki.

Cosmo is a robotic vehicle around the size of a Tonka truck. Much like a tank it has 2 motors and continuous track to move in all planar directions. Sensing of the environment is achieved via a camera and lots of image processing. In addition to this, to make the robot more relatable, a display is attached to the front of its body which shows animated eyes that correspond to its emotional states. The robot can also interact with its environment using a lifter arm much like a forklift. According to its website, emotions are handled by an "emotion engine" which is a learning algorithm that updates emotional information over time. This allows the robot to have less static intractability with the user and in the end makes it more lifelike. I would assume that the biggest challenge in the entire system is in designing the vision system. Since the only real positional tracking is done via the camera, the onboard computer has to do lots of image processing. This is also coupled with facial recognition for the emotion engine.

I think the main challenge to overcome before the robot will be widely used is its limitation of interaction with the environment. Granted that this may not be its purpose, it would be more appealing to buy if the robot had a more versatile object interaction system. On the ethical side of things, I see no big challenge in its future. From first impressions the robot is just an entertaining toy for kids to play with. There really is no reason to believe that a toy would pose any severe ethical issues. It could be argued that the emotional engine could learn to be annoying and mean but that is very unlikely.

Appendix

code

Problem 1:

```
%% problem 1
% (i) initialize matrices
A = [2 59 2 5
    41 11 0 4
    18 2 3 9
    6 23 27 10
    5 8 5 1]
B = [0 1 0 1
     0 1 1 1
     0 0 0 1
     1 1 0 1
     0 1 0 0]
 % (ii) pointwise multiply
 C = A.*B
 % (iii)
 C(2,:)*C(5,:)'
 % (iv)
 [row,col] = find(C == max(max(C)))
 V = max(max(C))
 [row,col,] = find(C == min(min(C)))
 V = min(C)
 % (v)
 D = [C(1,:)-C(1,:);C(2,:)-C(1,:);C(3,:)-C(1,:);C(4,:)-C(1,:);C(5,:)-C(1,:)]
 % (vi)
 [row,col] = find(D == max(max(D)))
 V = max(max(D))
 [row,col,] = find(D == min(min(D)))
 V = min(min(D))
```

Problem 2:

```
% If there is an object to the South, move a different direction
    % START
    if haveIBeenHereBefore(loc, nextStep)
        nextStep = loc(end,:) + [0 -1];
    end

    if detectObject(loc, obj, 'S')
        nextStep = loc(end,:) + [0 -1];
    end
    if detectObject(loc, obj, 'W') && detectObject(loc, obj, 'S')
        nextStep = loc(end,:) + [0 1];
    end
    if detectObject(loc, obj, 'W') && detectObject(loc, obj, 'S') && detectObject(loc,
obj, 'E')
        nextStep = loc(end,:) + [-1 0];
    end

    % STOP
```

Problem 3:

```
if dir == 0
        if (sensorInput(1,2) == 1 &&
sensorInput(2,3)==0)||(sensorInput(1,3)==1&&sensorInput(1,2)==0&&sensorInput(2,3)==0)
            newPos(2) = newPos(2) + 1; % Move Right
            %newPos(1) = newPos(1) + 1; % Move Down
        end
        if sensorInput(1,2) == 0 && (sensorInput(1,1)==1||sensorInput(2,1)==1)
            newPos(1) = newPos(1) - 1; % Move Up
        end

        if (sensorInput(3,2) == 0 &&
sensorInput(2,3)==1)||(sensorInput(3,2)==0&&sensorInput(2,3)==0&&sensorInput(3,3)==1)
            newPos(1) = newPos(1) + 1; % Move Down
        end
        if sensorInput(2,1)==0 && (sensorInput(3,2)==1||sensorInput(3,1)==1)
            newPos(2) = newPos(2) - 1; % move left
        end
    elseif dir == 1
        if (sensorInput(1,2) == 1 &&
sensorInput(2,1)==0)||(sensorInput(1,1)==1&&sensorInput(1,2)==0&&sensorInput(2,1)==0)
            newPos(2) = newPos(2) - 1; % Move Left
        end
        if (sensorInput(3,2) == 0 &&
sensorInput(2,1)==1)||(sensorInput(3,1)==1&&sensorInput(3,2)==0&&sensorInput(2,1)==0)
            newPos(1) = newPos(1) + 1; % Move Down
        end
        if (sensorInput(2,3) == 1 &&
sensorInput(1,2)==0)||(sensorInput(1,3)==1&&sensorInput(1,2)==0&&sensorInput(2,3)==0)
            newPos(1) = newPos(1) - 1; % Move Up
        end
        if sensorInput(2,3) == 0 && (sensorInput(3,2)==1||sensorInput(3,3)==1)
            newPos(2) = newPos(2) + 1; % Move Right
        end
    end
```