Assignment # 2 – Feb 9 2016 – ECE 674

Saad Sadiq – C11954772

**Q1. Identify and explain the properties that you would associate with the concept of an intelligent agent.**

There are several properties or defining characteristics of intelligent agents. Following are some key properties described as follows

1. **Reactivity**

A reactive system is one that maintains an ongoing interaction with its environment, and responds to changes that occur in it (in time for the response to be useful). Most real world environments are dynamic and it is quite difficult to develop software programs for the dynamic environments: programs must take into account possibility of failure.

1. **Social Ability**

Social ability is the ability to interact with other agents and possibly humans via some kind of agent-communication language to perform sophisticated activities. In real world multi-agent environments we cannot attempt to achieve our goals without taking others into account.

1. **Rationality**

An intelligent agent will not act in such a way as to prevent its goals being achieved. At least to the extent as its beliefs permit an agent will act only towards its objectives.

1. **Reasoning**

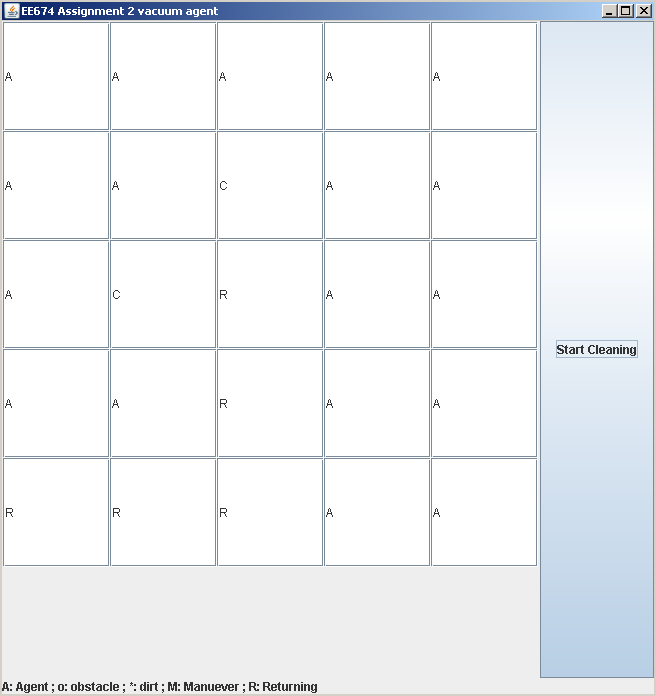
An intelligent agent should have a knowledge base with beliefs on the world and it should have the ability to infer and extrapolate on current knowledge. The agent should have the capacity to make plans. This distinguishes an intelligent agent from a robotic reactive agent.

1. **Autonomy**

Ability to pursue goals in an autonomous way, without direct continuous interaction/commands from the user. This differentiates agents from traditional software programs. Given a vague/imprecise goal, the agent will determine the best way to attain that goal.

**Q2. Scale the vaccum world up to 5x5 and also contains an obstacle which the agent needs to avoid.**

**Program Readme**

The following program has been written in Java with Java Development Kit 7. The instructions to run the program are as follows. I have used Java swing container to visually depict the grid layout and movement of the agent. The code will run until the room is cleaned and the agent returns back to its original position, after that the code will stop.

The code has all the methods and imports all the necessary libraries. To test the code you only need to copy paste the code in your IDE of choice as a class and run it.

**Running from Command Prompt**

1. Run Command Prompt and type

C:\> cd \mywork

C:\mywork> dir

This displays the directory contents.  You should see vaccum.java among the files.

1. C:\mywork> set path=%path%;C:\Program Files\Java\jdk1.8.0\_51\bin

This tells the system where to find JDK programs.

1. C:\mywork> javac HelloWorld.java

This runs javac.exe, the compiler.  You should see nothing but the next system prompt...

C:\mywork> dir

1. C:\mywork> java HelloWorld

This runs the Java interpreter.  You should see the program output:

Hello, World!

**Running from an IDE for e.g. Eclipse**

1. Begin by creating a new Java project File> New>Java Project. You can also use the shortcut Alt+Shift+N.
2. Enter the project name “vaccumCleaner” and click Finish
3. Start a new Java class. File > New > Class
4. Enter the class name “vaccum” and click Finish
5. Copy & Paste the code below in the code editor
6. Click Save and then press Ctrl+F11 to run the Code

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** javax.swing.\*;

**public** **class** vaccum **extends** JFrame **implements** ActionListener {

JTextField[][] b = **new** JTextField[5][5];

JButton bt;

**int** face;//0 u;1 d; 2 l; 3 r

**int** i=0;

**int** j=0;

**int** moves;

**void** move() {

//printing the coordinates and facing of the agent

System.***out***.printf("X= %d, Y= %d, Facing= %d \n", i, j, face);

//stepping same as in debugging

JOptionPane.*showMessageDialog*(**this**, "step");

**switch** (face) {

// 4 cases for the up down left right. based on the face

// direction the agent moves 1 step

**case** 0:

// The detector checks 1 block ahead of the agent for wall or previously traversed path

**if** (i + 1 > 4 || b[i + 1][j].getText().equalsIgnoreCase("A")) {

face = 2; // if wall is reached, turn face left

System.***out***.println(face);

// The detector checks 1 block ahead of the agent for dirt

} **else** **if** (b[i + 1][j].getText().equalsIgnoreCase("\*")) {

System.***out***.printf("Cleaning dirt at %d,%d \n", i + 1, j);

b[i + 1][j].setText("C");

i++;

}

// The detector checks 1 block ahead of the agent for obstacle

**else** **if** (b[i + 1][j].getText().equalsIgnoreCase("o")) {

manueverObstacle();

}

//seems like an empty clean block ahead, move the agent forward

**else** {

b[i + 1][j].setText("A");

i++;

}

moves = moves + 1;

**break**;

**case** 1:

**if** (i - 1 < 0 || b[i - 1][j].getText().equalsIgnoreCase("A")) {

face = 3; // if wall is reached, turn face right

System.***out***.println(face);

} **else** **if** (b[i - 1][j].getText().equalsIgnoreCase("\*")) {

System.***out***.printf("Cleaning dirt at %d,%d \n", i - 1, j);

b[i - 1][j].setText("C");

i--;

} **else** **if** (b[i - 1][j].getText().equalsIgnoreCase("o")) {

manueverObstacle();

} **else** {

b[i - 1][j].setText("A");

i--;

}

moves = moves + 1;

**break**;

**case** 2:

**if** (j - 1 < 0 || b[i][j - 1].getText().equalsIgnoreCase("A")) {

face = 1; // if wall is reached, turn face down

System.***out***.println(face);

} **else** **if** (b[i][j - 1].getText().equalsIgnoreCase("\*")) {

System.***out***.printf("Cleaning dirt at %d,%d \n", i, j - 1);

b[i][j - 1].setText("C");

j--;

} **else** **if** (b[i][j - 1].getText().equalsIgnoreCase("o")) {

manueverObstacle();

} **else** {

b[i][j - 1].setText("A");

// b[i][j].setText("c");

j--;

}

moves = moves + 1;

**break**;

**case** 3:

**if** (j + 1 > 4 || b[i][j + 1].getText().equalsIgnoreCase("A")) {

face = 0; // if wall is reached, turn face up

System.***out***.println(face);

} **else** **if** (b[i][j + 1].getText().equalsIgnoreCase("\*")) {

System.***out***.printf("Cleaning dirt at %d,%d \n", i, j + 1);

b[i][j + 1].setText("C");

j++;

} **else** **if** (b[i][j + 1].getText().equalsIgnoreCase("o")) {

manueverObstacle();

} **else** {

b[i][j + 1].setText("A");

j++;

}

moves = moves + 1;

**break**;

}

}

**public** vaccum() {

//container to visually depict the grid layout and movement of the agent

**super**("EE674 Assignment 2 vacuum agent");

bt = **new** JButton("Start Cleaning");

Container content = **this**.getContentPane();

JPanel panel = **new** JPanel();

panel.setLayout(**new** GridLayout(6, 5));

**for** (**int** i = 0; i < 5; i++) {

**for** (**int** j = 0; j < 5; j++) {

b[i][j] = **new** JTextField();

panel.add(b[i][j], j);

}

}

bt.addActionListener(**this**);

//adding legends

content.add(**new** JLabel("A: Agent ; o: obstacle ; \*: dirt ; M: Manuever ; R: Returning"), BorderLayout.***PAGE\_END***);

content.add(bt, BorderLayout.***AFTER\_LINE\_ENDS***);

content.add(panel);

**this**.setSize(660, 700);

**this**.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

}

**public** **void** actionPerformed(ActionEvent be) {

//When the start button is pressed, run the following code

placeAgent();

placeDirt();

placeObstacles();

face = 3;

**int** l[] = **new** **int**[2];

i = l[0];

j = l[1];

moves=1;

System.***out***.println("Starting Cleaning");

**while** (!(b[2][2].getText().equalsIgnoreCase("A"))) { //until the agent reaches the center location

move();

} returnAgent();// call the return function to return agent to charging station/starting position

}

**void** manueverObstacle(){

System.***out***.println("Found Obstacle");

**switch** (face) {

**case** 0: // facing up

System.***out***.println("Manuevering around the obstacle");

j--;; System.***out***.print(j);

i+=2; System.***out***.print(i);

j++; System.***out***.println(j);

b[i][j].setText("A");

**case** 1: // facing down

System.***out***.println("Manuevering around the obstacle");

j++; System.***out***.print(j);

i-=2; System.***out***.print(i);

j--; System.***out***.println(j);

b[i][j].setText("A");

**case** 2: // facing left

j=j-2;

System.***out***.println("Manuevering around the obstacle");

i--; System.***out***.print(i);

j-=2; System.***out***.print(j);

i++; System.***out***.println(i);

b[i][j].setText("A");

**case** 3: // facing right

System.***out***.println("Manuevering around the obstacle");

i++; System.***out***.print(i);

j+=2; System.***out***.print(j);

i--; System.***out***.println(i);

b[i][j].setText("A");

}

}

**void** returnAgent(){

System.***out***.println("Room Cleaned, returning to charging station");

face =2;

**while**(i>0){

b[i][j].setText("R");

i--;

}

face=3;

**while**(j>0){

b[i][j].setText("R");

j--;

}

b[0][0].setText("R");

}

**void** placeAgent() {

outerloop:

**for** (**int** i = 0; i < 5; i++) {

**for** (**int** j = 0; j < 5; j++) {

**if** (b[i][j].getText().equalsIgnoreCase("")) {

b[i][j].setText("A");

**break** outerloop;

}

}

}

}

**void** placeDirt(){

b[0][2].setText("\*");

b[2][1].setText("\*");

b[3][2].setText("\*");

}

**void** placeObstacles(){

b[1][2].setText("o");

}

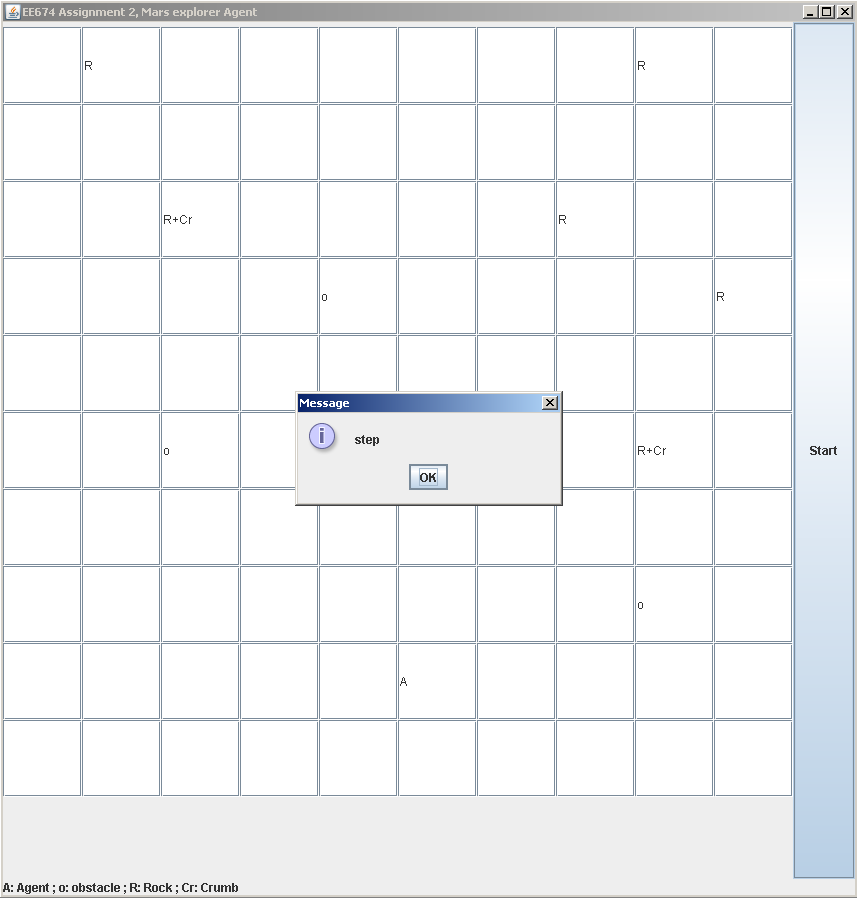
**public** **static** **void** main(String[] args) {

**new** vaccum().setVisible(**true**);

}

}

**Q3. Implement the behavior of one agent in the Mars explorer example. Can be implemented using subsumption architecture or logic based architecture.**

The following program has been written in Java with Java Development Kit 7. The instructions to run the program are as follows. The code is set to run forever hence to stop the code you can kill the command prompt or click on the stop button in Eclipse IDE.

The code has all the methods and imports all the necessary libraries. To test the code you only need to copy paste the code in your IDE of choice as a class and run it.

**Running from Command Prompt**

1. Run Command Prompt and type

C:\> cd \mywork

C:\mywork> dir

This displays the directory contents.  You should see vaccum.java among the files.

1. C:\mywork> set path=%path%;C:\Program Files\Java\jdk1.8.0\_51\bin

This tells the system where to find JDK programs.

1. C:\mywork> javac HelloWorld.java

This runs javac.exe, the compiler.  You should see nothing but the next system prompt...

C:\mywork> dir

1. C:\mywork> java HelloWorld

This runs the Java interpreter.  You should see the program output:

Hello, World!

**Running from an IDE for e.g. Eclipse**

1. Begin by creating a new Java project File> New>Java Project. You can also use the shortcut Alt+Shift+N.
2. Enter the project name “marsExplorer” and click Finish
3. Start a new Java class. File > New > Class
4. Enter the class name “agent” and click Finish
5. Copy & Paste the code below in the code editor
6. Click Save and then press Ctrl+F11 to run the Code

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** java.util.Random;

**import** javax.swing.\*;

**public** **class** vcai **extends** JFrame **implements** ActionListener {

JTextField[][] b = **new** JTextField[10][10];

JButton bt;

**int** face;//1 u;2 d; 3 l; 4 r

**int** i;

**int** j;

**int** moves;

**int**[][] signalStrength = { { 55,52,51,42,41,32,31,22,21,10}, // The signal strength matrix

{ 60,61,52,51,42,41,32,31,22,21}, // This matrix is used by the agent when it is returning back

{ 65,62,61,52,51,42,41,32,31,22}, // after collecting the rock

{ 70,71,62,61,52,51,42,41,32,31},

{ 75,72,71,62,61,52,51,42,41,32},

{ 80,81,72,71,62,61,52,51,42,41},

{ 85,82,81,72,71,62,61,52,51,42},

{ 90,85,82,81,72,71,62,61,52,51},

{ 95,90,85,82,81,72,71,62,61,52},

{100,95,90,85,80,75,70,65,60,55},

};

**void** Search() {

Random rn = **new** Random();

**int** n = 4;

**int** nface = Math.*abs*(rn.nextInt() % n);

System.***out***.printf("X= %d, Y= %d, Facing= %d \n",i,j,face);

JOptionPane.*showMessageDialog*(**this**, "step");

**switch** (face) {

**case** 0:

**if** (moves%5==0 || i + 1 > 9 || b[i + 1][j].getText().equalsIgnoreCase("R+Cr")) {

face = nface;

System.***out***.println(nface);

} **else** **if**(b[i + 1][j].getText().equalsIgnoreCase("R")){

b[i][j].setText(""); i++;

dropCrumb();

returnHome();

b[i][j].setText("A");

} **else** **if**(b[i + 1][j].getText().equalsIgnoreCase("o")){

b[i][j].setText("");

manueverObstacle();

}**else** {

b[i + 1][j].setText("A");

b[i][j].setText("");

i++;

}

moves++;

**break**;

**case** 1:

**if** (moves%5==0 || i - 1 < 0 || b[i - 1][j].getText().equalsIgnoreCase("R+Cr")) {

face = nface;

System.***out***.println(nface);

} **else** **if**(b[i -1][j].getText().equalsIgnoreCase("R")){

b[i][j].setText(""); i--;

dropCrumb();

returnHome();

b[i][j].setText("A");

} **else** **if**(b[i - 1][j].getText().equalsIgnoreCase("o")){

b[i][j].setText("");

manueverObstacle();

}**else** {

b[i - 1][j].setText("A");

b[i][j].setText("");

i--;

}

moves++;

**break**;

**case** 2:

**if** (moves%5==0 || j - 1 < 0 || b[i][j - 1].getText().equalsIgnoreCase("R+Cr")) {

face = nface;

System.***out***.println(nface);

} **else** **if**(b[i][j-1].getText().equalsIgnoreCase("R")){

b[i][j].setText(""); j--;

dropCrumb();

returnHome();

b[i][j].setText("A");

} **else** **if**(b[i][j-1].getText().equalsIgnoreCase("o")){

b[i][j].setText("");

manueverObstacle();

}**else** {

b[i][j - 1].setText("A");

b[i][j].setText("");

j--;

}

moves++;

**break**;

**case** 3:

**if** (moves%5==0 || j + 1 > 9 || b[i][j + 1].getText().equalsIgnoreCase("R+Cr")) {

face = nface;

System.***out***.println(nface);

} **else** **if**(b[i][j + 1].getText().equalsIgnoreCase("R")){

b[i][j].setText(""); j++;

dropCrumb();

returnHome();

b[i][j].setText("A");

} **else** **if**(b[i][j + 1].getText().equalsIgnoreCase("o")){

b[i][j].setText("");

manueverObstacle();

}**else** {

b[i][j + 1].setText("A");

b[i][j].setText("");

j++;

}

moves++;

**break**;

}

}

**public** vcai() {

**super**("EE674 Assignment 2, Mars explorer Agent");

bt = **new** JButton("Start");

Container content = **this**.getContentPane();

JPanel panel = **new** JPanel();

panel.setLayout(**new** GridLayout(11, 10));

**for** (**int** i = 0; i < 10; i++) {

**for** (**int** j = 0; j < 10; j++) {

b[i][j] = **new** JTextField();

panel.add(b[i][j], j);

}

}

bt.addActionListener(**this**);

content.add(**new** JLabel("A: Agent ; o: obstacle ; R: Rock ; Cr: Crumb"), BorderLayout.***PAGE\_END***);

content.add(bt, BorderLayout.***AFTER\_LINE\_ENDS***);

content.add(panel);

**this**.setSize(860, 900);

**this**.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

}

**public** **void** actionPerformed(ActionEvent be) {

placeAgent();

face = 3;

**int** l[] = **new** **int**[2];

i = l[0];

j = l[1];

moves =1;

placeObstacles();

placeRocks();

System.***out***.println("Starting Exploration");

**while** (**true**) {

Search();

}

}

**void** placeAgent() {

outerloop:

**for** (**int** i = 0; i < 10; i++) {

**for** (**int** j = 0; j < 10; j++) {

**if** (b[i][j].getText().equalsIgnoreCase("")) {

b[i][j].setText("A");

**break** outerloop;

}

}

}

}

**void** manueverObstacle(){

//b[i][j].setText("M");

**switch** (face) {

**case** 0: // facing up

System.***out***.println("Manuevering around the obstacle");

j--;; System.***out***.print(j);

i=i+2; System.***out***.print(i);

j++; System.***out***.println(j);

b[i][j].setText("A");

**break**;

**case** 1: // facing down

System.***out***.println("Manuevering around the obstacle");

j++; System.***out***.print(j);

i=i-2; System.***out***.print(i);

j--; System.***out***.println(j);

b[i][j].setText("A");

**break**;

**case** 2: // facing left

System.***out***.println("Manuevering around the obstacle");

i--; System.***out***.print(i);

j=j-2; System.***out***.print(j);

i++; System.***out***.println(i);

b[i][j].setText("A");

**break**;

**case** 3: // facing right

System.***out***.println("Manuevering around the obstacle");

i++; System.***out***.print(i);

j=j+2; System.***out***.print(j);

i--; System.***out***.println(i);

b[i][j].setText("A");

**break**;

}

}

**void** placeObstacles(){

b[4][2].setText("o");

b[2][8].setText("o");

b[6][4].setText("o");

}

**void** placeRocks(){

b[7][2].setText("R");

b[4][8].setText("R");

b[9][1].setText("R");

b[7][7].setText("R");

b[6][9].setText("R");

b[9][8].setText("R");

}

**void** dropCrumb(){

b[i][j].setText("R+Cr");

}

**void** returnHome(){

System.***out***.println("Returning Rock back to Mother Ship");

**while**(j>0) {

**if**(signalStrength[i][j] < signalStrength[i][j-1]) {

System.***out***.print("Checking Signal; Traversing Horizontally --");

System.***out***.print(i);

System.***out***.println(j);

j=j-1;

}

}

**while**(i>0){

System.***out***.print("Checking Signal; Traversing Vertically -- ");

i--;

System.***out***.print(i);

System.***out***.println(j);

}

}

**public** **static** **void** main(String[] args) {

**new** vcai().setVisible(**true**);

}

}