CS1701 GROUP PROJECTS LECTURES AND TUTORIALS

ASSINGMENT 3 – SOFTWARE IMPLEMENTATION

Rooney Mosiana Nsasa (Student)
2315045

GROUP 10A - STEVE COUNSELL

Task 2 Traffic Lights

The purpose of this assignment is to implement the task chosen into the Swiftbot and including core and additional functionalities (if applicable)

Table of Contents

Task 2: Traffic Lights

Description

The purpose of this program is to navigate the Swfitbot via colour coded inputs within the camera. The traffic lights consist of 3 colours: red, green and blue. Once it is within a required distance from the traffic light, it will use the camera to take a capture and convert the image into a pixel thus allowing to assign its RGB value before extracting the 3 colours mentioned prior (red, green and blue) and computing the value which will determine the colour; Once done so, the swiftbot will react appropriately to the colour detected.

FUNCTIONAL REQUIREMENTS

1. The program should process the photo taken by the Swiftbot's camera to determine the colour of the traffic light displayed ahead.

2. **The program** should be implemented through a command line interface (**input** and **output**)

3. **The program** should make appropriate error checks and exception handling, to make sure that the inputted values are within range.

4. Invalid input is rejected; therefore, **the program** should display an error message and ask the user to enter a new input.

5. **The user** should be able to navigate the Swiftbot through the colour codes inputted within the camera (red, green, blue) within 2-5 seconds

6. **The Swiftbot** should respond to the traffic lights within 20m ahead using the camera

7. If the traffic light colour is green, the underlights should be set to green and the Swiftbot should pass the traffic light. After passing the traffic light, **the Swiftbot** should stop for half a second, set the underlights to yellow, and then move forward in its initial speed until the next traffic light is detected.

8. If the traffic light colour is red, the underlights should be set to red and **the Swiftbot** should stop at the traffic light. After half a second, the Swiftbot should start moving forward at its initial speed until the next traffic light is detected.

9. If the traffic light colour is blue, the Swiftbot should stop for half a second, turn the underlights blue and change direction by turning 90 degrees to the left. The robot should then move at a low speed for a second and stop. After half a second, **the Swiftbot** should retrace its movement and

move back to the original path. Then the robot moves forward at a moderate speed until it detects the next traffic light.

10. Before **the program** terminates, it should ask the user whether it should show the log of the execution. If the user responds '**yes' (Button Y)** then it displays the following:

-number of times **Swiftbot** encountered traffic lights

-most frequent traffic light colour **Swiftbot** encountered

-number of times most frequent traffic light was detected

-duration of execution

11. The program should stop once the user presses 'X' on the Swiftbot.

NON-FUNCTIONAL REQUIREMENTS

1. The program should be designed in a way which enhances user experience, for example its ease of use
2. Swiftbot shows messages after the traffic light is displayed (green = "Go!" red = "Stop" blue = "returning to original position" for example)
3. The Swiftbot should respond appropriately to any changes to the traffic lights

The reasoning as to why the additional non-functional requirements are included is because this is what I would expect the system will play out. For example, once it has registered the colour of the traffic light, the bottom LED should turn on to its corresponding colour. But most importantly, design it in a way which enhances user experience as mentioned prior, hence why the program should be implemented via command line user interface making it easier to use.

Some additional non-functional requirements that **could** be included for example, when the Swiftbot's camera reacts to the traffic light and detects its colour, the program could send out a message such as "GO!" when its green, "STOP!" when its red and vice versa. This just adds more realism to the task as the Swiftbot is being navigated by user input, so you'd want it its behaviour to be similar to how cars react to everyday traffic.

NOT ALL OF THE FUNCTIONAL REQUIREMENTS ARE INCLUDED

Instead of asking the user for a log of execution when the user presses button X, it will display a goodbye message and terminate the program immediately. The reasoning for this is because I decided to add an infinite loop for my detection process via ultrasound that will make the swifbot keep moving despite pressing button X. I strive to improve my programming skills in JavaScript so that I can understand how to better handle ways of producing writing code.

1.2 Changes to Algorithm and User Interface Design

Algorithm/UI	Additional or Requirement	Changes made	Reasoning for changes
Algorithm	Requirement	Made a function so that the user presses button A on the Swiftbot to initiate the program	So that the program doesn't start immediately and gives the user time to understand the program they're reading from the UI
UI	Additional	Displays a Welcome message	Greets the user when they enter the program
UI	Requirement	Tells the user to start the program, they should press Button A on the Swiftbot.	Basic explanation of how the task works so that the user can understand with ease.
UI	Additional	Once the program initiates, it will print out "Searching for traffic lights at x cm"	Allows the user to know how far they are until it meets the traffic light from within a certain distance
Algorithm	Requirement	Made a function so that the user presses button X on the Swiftbot to terminate the program	So that the Swiftbot doesn't continuously search for a traffic light until it detects one. The user can exit when they wish to
Algorithm	Additional	If any incorrect buttons are pressed within the swiftbot no action will happen	Apart from button X and Y, no other buttons have to be enabled, making error handling redundant.

1.3 Testing

Test #	Description	Expected	Outcome	Pass/Fail
1	When the user presses Button A on the Swiftbot it prints out text saying "Button A pressed, initializing program……"	It prints out "Button A pressed, initializing program……" once pressed	Has printed out "Button A pressed, initializing program……" once pressed	Pass

2	When button A is pressed, the underlights turn yellow before moving off once button A is pressed.	The underlights turn yellow and the swiftbot starts moving at a low speed	The underlights have turned yellow and the swiftbot moves at a low speed	Pass
3	Once the Swiftbot is within 20cm of the object it reacts to the light accordingly	The Swiftbot will stop once is within a certain range and react to the traffic light	The Swiftbot has stopped and reacted to the traffic light.	Pass
4	If the traffic light = red. The swiftbot changes its lights to red and stops at the light	The under light changes to red and stops at the traffic light	The under light has changed to red	Pass
5	If the traffic light = green, under lights are set to green	Underlights turn green	Underlights has changed to green	Pass
6	If the traffic light = blue, the underlights blink blue	Underlights turn blue and blink	Underlights have turned blue and blinked	Pass
7	The program stops once the user presses button X	Button X is pressed and the program stops	Button X has been pressed and it promptly stops the program.	Pass
8 (Additional)	Before the program terminates it displays “Button X pressed”	Button X is pressed and it prints out a line before closing	Button X has been pressed and it prints out a line before closing	Pass
9 (Additional)	Before the program terminates, it displays a goodbye message	Button X is pressed and it prints out a line before closing	Button X has been pressed and it prints out a line before closing	Pass
10	Before terminating the program, it asks the user if they want to view the log of execution?	Button X is pressed and it prints out a line asking the user if it wants to view the log of execution	Button X has been pressed and it prints out a line before closing	Pass
11	If user presses button “Y” when	Button Y is pressed and it	Button Y can be pressed but it	Fail

	it prints out a log of execution	prints out a log of execution	does not display the log of execution.	
12	If the user presses X then it saves the log onto a file before terminating the program	Button X is pressed and it terminates the program (user can check files afterwards as they are automatically saved)	Button X is pressed and it terminates the program but it doesn't save a file before terminating	Fail
13	Swiftbot passes the traffic light within 2 seconds if its green	Swiftbot passes traffic light within 2 seconds if the traffic light is green	Swiftbot passes the traffic light within 2 seconds once it's green.	Pass
14	Swiftbot stops for a second if the traffic lights are red	Swiftbot stops for a second if the traffic lights are red and goes back to its normal speed.	Swiftbot has stopped for a second before returning to its normal speed	Pass
15	Swiftbot turns 90 degrees and moves before stopping and retracing its path if traffic lights are blue	Swiftbot turns 90 degrees and moves before retracing its path	Swiftbot turns 90 degrees and retraces	Pass

1.4 Planning and Monitoring

Task	Completed? (Y/N)	Date of Completion	Difficulties faced?
Designing the cover page	Yes	1/1/2024	N/A
Updating the implementation summary	Yes	18/1/2024	N/A
Creating the tables for Algorithm design and Testing	Yes	22/1/2024	N/A
Filling both algorithm design and UI design tables with results and improvements	Yes	26/01/2024	Being disingenuous when mentioning extra functionalites

1.4 Source Code Pasting

```
2   import java.awt.image.BufferedImage;
3   import java.io.File;
4   import java.io.IOException;
5
6   import javax.imageio.ImageIO;
7
8   import swiftbot.Button;
9   import swiftbot.ImageSize;
10  import swiftbot.SwiftBotAPI;
11
12  public class TrafficLights {
13     static SwiftBotAPI swiftBot;
14     double distance = swiftBot.useUltrasound();
15
16     public static void main(String[] args) {
17
18           swiftBot = new SwiftBotAPI();
19
20           menu();//menu method
21           System.out.println("Welcome! Press Button A to start");//prints out
    a welcome message
22           buttons();//method for buttons
23
24
25     }
26
27     public static void loop() {//created a method called loop, which goes
    through these functions
28           yellow();
29           swiftBot.startMove(25, 25);
30           detectTrafficLight();//method to detect traffic lights
31
32     }
33
34
35
36
37
38     public static void detectTrafficLight() {//method for detect traffic light
39         try {//try block = a code that throws an exception. if any errors are
    found the block code doesn't execute
40            while (true) {//infinite loop (did try to break it so that the x
    button would exit properly but no time ig
41                double distance = swiftBot.useUltrasound();//reads the distance
    from the sensor
42
43                if (distance >= 20) {//checks if the distance is greater than
    or equal to 20cm
44                    yellow();//method for the colour yellow
45                    swiftBot.startMove(25, 25);//swiftbot moves at a low speed
    (calculated)
```

```
46                  System.out.println("Searching for traffic light at " +
    distance + "cm");//prints out distance in cm
47              } else {
48                  System.out.println("Traffic Light detected at " + distance
    + "cm!");//prints out detection in cm
49                  swiftBot.stopMove();//swiftbot stop movement
50                  RGB();//goes through the rgb method
51                  break;//Exits loop
52              }
53          }
54      } catch (Exception e) {//Catches exceptions
55          e.printStackTrace();
56          System.out.println("ERROR: Failed to detect traffic light.");// Log
    an error message
57          System.exit(0);//exits program
58      }
59  }
60
61
62
63  public static void buttons() {//method for buttons
64      try {
65          swiftBot.enableButton(Button.A, () -> {//when button A is pressed,
    it will print out statements below and go through yellow method
66              System.out.println("Button A Pressed.");
67              System.out.println("Initializing program.....");
68              yellow();//method for the colour yellow
69              swiftBot.disableButton(Button.A);//Disables button A so it
    isn't called again
70              swiftBot.startMove(25, 25);//moves at a low speed
71              detectTrafficLight();//goes through detect traffic light
    method
72          });
73
74
75          swiftBot.enableButton(Button.X, () -> {//when button x is pressed it
    will print a goodbye message before terminating program
76              System.out.println("Button X Pressed.");
77              System.out.println("\r\n"
78                      + "   ______                   __ __
    __\r\n"
79                      + "  / ____/____   ____   ____/ // /_   __  __
    ___   / /\r\n"
80                      + " / / __ / __ \\ / __ \\ / __  // __ \\ / / /
    // _ \\ / / \r\n"
81                      + "/ /_/ // /_/ // /_/ // /_/ // /_/ // /_/ //
    __//_/  \r\n"
82                      + "\\____/ \\____/ \\____/ \\__,_//_.___/ \\__, /
    \\___/(_)   \r\n"
83                      + "                                   /____/
    \r\n"
84                      + "");
85              swiftBot.disableButton(Button.X);//button X disabled so that
    it doesn't get called again
86              System.exit(0);//terminates the program (thread.sleep)
87          });
88
89      } catch (Exception e) {//catches an exception, although only 2 buttons are
    active during the program so it makes error handling redundant
```

```
90          e.printStackTrace();//if the user tries to press other buttons, it
    won't do anything as they aren't active
91       }
92    }
93
94
95
96
97
98
99
100
101
102
103
104
105      public static void menu() {// creates menu for user interaction
106
         System.out.println("**********************************************************
    ***********\r\n" + "\r\n" + "\r\n"
107                           + "\r\n"
108                           + " _____                __   __  _           __  _
    _      _          \r\n"
109                           + "/__   \\ _ __  __ _  / _| / _|(_)  ___    / / (_)
    __ _ | |__  | |_  ___ \r\n"
110                           + "  / /\\/| '__|/ _` || |_ | |_ | | / __|  / /  | | /
    _` || '_ \\ | __|/ __|\r\n"
111                           + " / /   | |  | (_| ||  _||  _|| || (__  / /___| ||
    (_| || | | || |_ \\__ \\\r\n"
112                           + " \\/    |_|   \\__,_||_|  |_|  |_| \\___| \\____/|_|
    \\__, ||_| |_| \\__||___/\r\n"
113                           + "
    |___/                  \r\n"
114                           + ""
115                           + ""
116                           + ""
117                           +
    "*********************************************************************");
118
119      }
120
121      public static void yellow() {//method for colour yellow
122
123            int[] colourToLightUp = { 255, 0, 255 };//rgb value for colour
    yellow
124
125            try {
126                  swiftBot.fillUnderlights(colourToLightUp);//sets all
    underlights to the colour (yellow) immediately
127            } catch (IllegalArgumentException e) {
128                  e.printStackTrace();
129            }
130
131            try {
132                  Thread.sleep(2000);//Sleep for 2000 milliseconds (2 seconds)
133            } catch (InterruptedException e) {
134                  e.printStackTrace();
135            }
136
```

```
137
138    }
139
140
141
142
143
144    public static void RGB() {//method for RGB
145
146           BufferedImage img =
    swiftBot.takeStill(ImageSize.SQUARE_48x48);//takes an image
147          if(img == null || img == null){// Checks if the image is null
148              System.out.println("ERROR: Image is null");//if so it prints an
    error message
149              System.exit(5);
150          }
151          else{
152              // else, Save the bwImage to a directory.
153              try {
154                          ImageIO.write(img, "png", new
    File("/home/pi/colourImage.png"));//Saves the image into a file
155                  } catch (IOException e) {
156                          e.printStackTrace();
157                  }
158
159              try {
160                          Thread.sleep(1000);//sleeps for 1 millisecond
161                  } catch (InterruptedException e) {
162                          e.printStackTrace();
163                  }
164          }
165
166          int p = 0;//Initialises variable p
167           for (int x = 0; x < img.getWidth(); ++x) {//iterates image width
168                  for (int y = 0; y < img.getHeight(); ++y) {//iterates image
    height
169                           p = img.getRGB(x, y);//gets the RGB value of the pixel
170                  }}// Extract RGB components from the pixel value
171                          int r = (p >> 16) & 0xFF;
172                          int g = (p >> 8) & 0xFF;
173                          int b = p & 0xFF;
174
175                          //processes the dominant colour
176                          switch (getColorType(r, g, b)) {//switch statement
    based on dominant colour
177                          case RED://if dominant colour = red
178                                 red();//calls method for colour red
179                                 loop();//calls loop method
180                                 break;
181                          case BLUE://if dominant colour blue
182                                 blue();//calls method for colour blue
183                                 loop();//calls loop method
184
185
186                                 break;
187                          case GREEN://if dominant colour = green
188                                 green();//it returns the method for colour green
189                                 loop();//calls loop
190
```

```
191
192
193                         break;
194                     default://default case (null)
195
196                        }
197                 }
198
199
200
201     // Method to determine the dominant colour
202     private static ColorType getColorType(int r, int g, int b) {
203           if (r > g && r > b) {// if red is greater than green and blue, it
    will return the colour type red
204                 return ColorType.RED;
205           } else if (b > r && b > g) {// if blue is greater than red and
    green, it will return the colour blue
206                 return ColorType.BLUE;
207           } else if (g> r && g > b) {// if green is greater than red and blue,
    it returns colour green
208                 return ColorType.GREEN;
209           }
210           else {
211                 return null;//else it returns nothing if no dominant colour is
    detected
212           }
213     }
214
215     // Enum = a special class to represent a group of constants like the
    variables for my colours below
216     enum ColorType {
217           RED, BLUE, GREEN
218     }
219
220
221     public static void red() {//method for red
222
223           System.out.println("STOP!");//will print out STOP! when called
224
225           swiftBot.stopMove();//Swiftbot stops movement
226
227           int[] colourToLightUp = { 255, 0, 0 };//rgb value for colour red
228
229
230           try {//fills the underlights red instantly
231           swiftBot.fillUnderlights(colourToLightUp);
232           } catch (IllegalArgumentException e) {
233                 e.printStackTrace();
234           }
235
236           try {
237                 Thread.sleep(2000);//thread sleep = pauses the execution for a
    certain period of time in this instance 2 seconds
238           } catch (InterruptedException e) {
239                 e.printStackTrace();
240           }
241
242
243
```

```
244
245
246
247
248
249     }
250
251     public static void green() {//method for colour green
252
253             System.out.println("GO!");//prints out GO! when called
254
255
256             swiftBot.startMove(100, 100);//Swiftbot moves at top speed
257
258
259             int[] colourToLightUp = { 0, 0, 255 };//rgb value for green
260
261
262             try {//underlights turn green
263                     swiftBot.fillUnderlights(colourToLightUp);
264             } catch (IllegalArgumentException e) {
265                     e.printStackTrace();
266             }
267
268             try {
269                     Thread.sleep(2000);//thread sleep = pauses the execution for a
    certain period of time in this instance 2 seconds
270             } catch (InterruptedException e) {
271                     e.printStackTrace();
272             }
273
274
275
276     }
277
278     public static void blue() {
279             int[] colourToLightUp = { 0, 255, 0 };
280             System.out.println("i see blue");
281
282             try {
283               swiftBot.fillUnderlights(colourToLightUp);
284             } catch (IllegalArgumentException e) {
285                     e.printStackTrace();
286             }
287
288             try {
289                     Thread.sleep(500);//thread sleep = pauses the execution for a
    certain period of time in this instance half a second
290             } catch (InterruptedException e) {
291                     e.printStackTrace();
292             }
293             swiftBot.disableUnderlights();//turns off all underlights (trying to
    make the lights blink by turning on/off)
294
295             try {
296                     swiftBot.fillUnderlights(colourToLightUp);
297                     } catch (IllegalArgumentException e) {
298                             e.printStackTrace();
299                     }
```

```
300
301                 try {
302                       Thread.sleep(500);//thread sleep = pauses the execution
    for a certain period of time in this instance half a second
303                 } catch (InterruptedException e) {
304                       e.printStackTrace();
305                 }
306
307           swiftBot.disableUnderlights();//turns off all underlights
308
309           try {
310                   swiftBot.fillUnderlights(colourToLightUp);
311                 } catch (IllegalArgumentException e) {
312                       e.printStackTrace();
313                 }
314
315                 try {
316                       Thread.sleep(2000);//thread sleep = pauses the
    execution for a certain period of time in this instance 2 seconds
317                 } catch (InterruptedException e) {
318                       e.printStackTrace();
319                 }
320
321
322                 swiftBot.move(50 ,50, 1000);//moves forward before making a
    retracing motion to the left and continues on its path
323                 swiftBot.move(-50, -50, 100);
324                 swiftBot.move(0, -100, 700);
325
326
327
328
329
330
331    }
332
333 }
```