PROJECT 3

ENPM 690

ROBOT LEARNING

Prof. Dr. Donald Sofge

UNIVERSITY OF
MARYLAND

Kartik Venkat (116830751)

This paper represents our own work in accordance with University regulations.

Contents

1 Question 1

Program a simple robot vehicle in a simulated environment (robot simulation tools and libraries may be used). Your simulated robot should exhibit at least one sensor input (e.g., forward-looking range sensor that returns the distance to the nearest obstacle) and two control outputs (e.g., left and right wheels, or speed and direction of vehicle motion). Show that you can drive your robot around through mouse or keyboard inputs.

Solution:

1.1 Approach

The following approach was used to solve this question:

1. The first step was to select the simulation software. In my solution I have used VREP software developed by Coppeliarobotics.
2. The next step was to choose the language to program the simuation. In my solution I have linked python with vrep using the RemoteApi library.
3. Once python is linked with VREP, the next step is to select the robot and create a VREP environment. In my solution I have used a pre-existing robot model called KUKA YouBot and setup a home like environment.
4. Once the VREP environment is ready, the next step is to create object handles for the four wheels of the YouBot in the python script so that we can later program them as per our needs.
5. After creating the object handles, an algorithm to take keyboard input from the user must be written. For this, the Pynput python library was imported from which the 'key' function was used to detect which arrow keys were pressed and the listener function was used to detect each key press and key release.
6. Also, a threaded function was defined to keep printing the front proximity sensor values.

1.2 Model Development

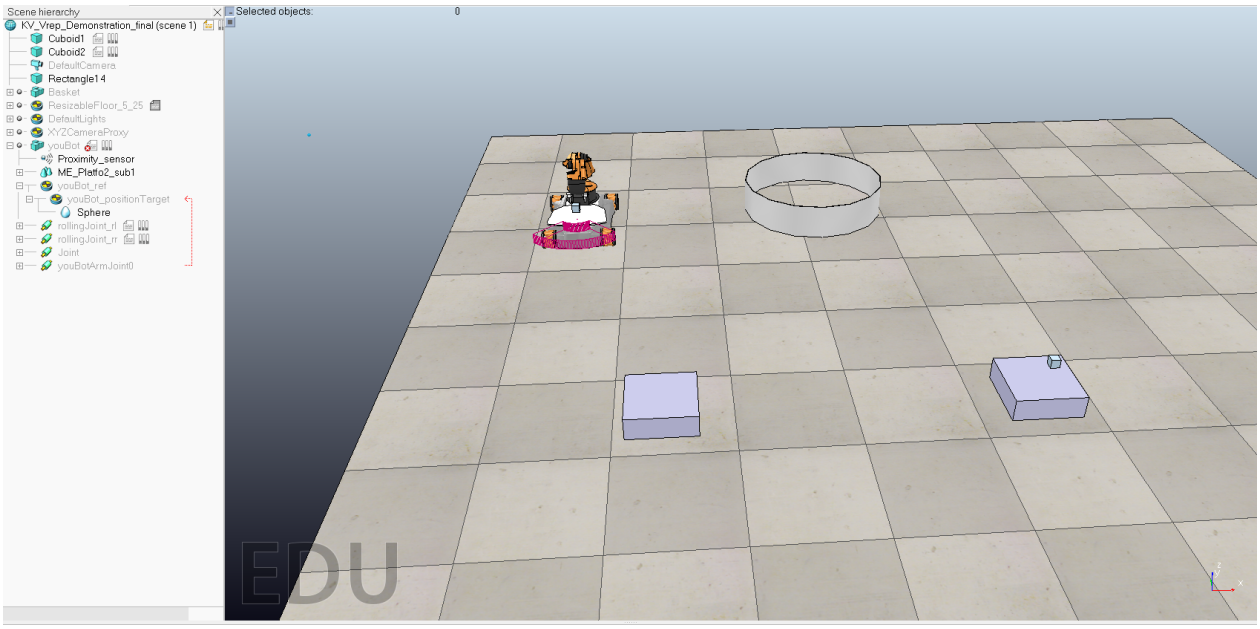

Figure 1: Initial test VREP environment.

Figure 2: Final VREP environment for Question 1

2 Question 2

Add a programmed behavior to your robot, such as following (or avoiding) a light, or wandering, while avoiding collisions with obstacles.

Solution:

2.1 Approach

The following approach was used to solve this question:

1. This part of the assignment was built over the first part model. The main difference between the two models is that in the first part, the robot is tele-operated, whereas in the second part, the robot must wander in the environment whilst avoiding obstacles.
2. Now, in my solution, I have added three ultrasonic disc type sensors in the front, left and right of the robot and created their respective object handles.
3. The next task is to program these sensors for obstacle avoidance. For this, the sensor values are read using the 'simxReadProximitySensor' function from the RemoteApi library.
4. To program the robot to wander in the environment, we defined thresholds for obstacle avoidance and if the robot is not near any obstacle, we make it turn in either left or right direction randomly with a probability of 7.5% each.
5. If the robot detects an obstacle ahead, it will move back and then randomly turn in either left or right direction an then try moving forward again. Similarly, if it detects an obstacle on it's left, it will move towards the right and vice-versa.
6. The speed of the robot can be varied by changing the value of the speed variable in the python script.

2.2 Model Development

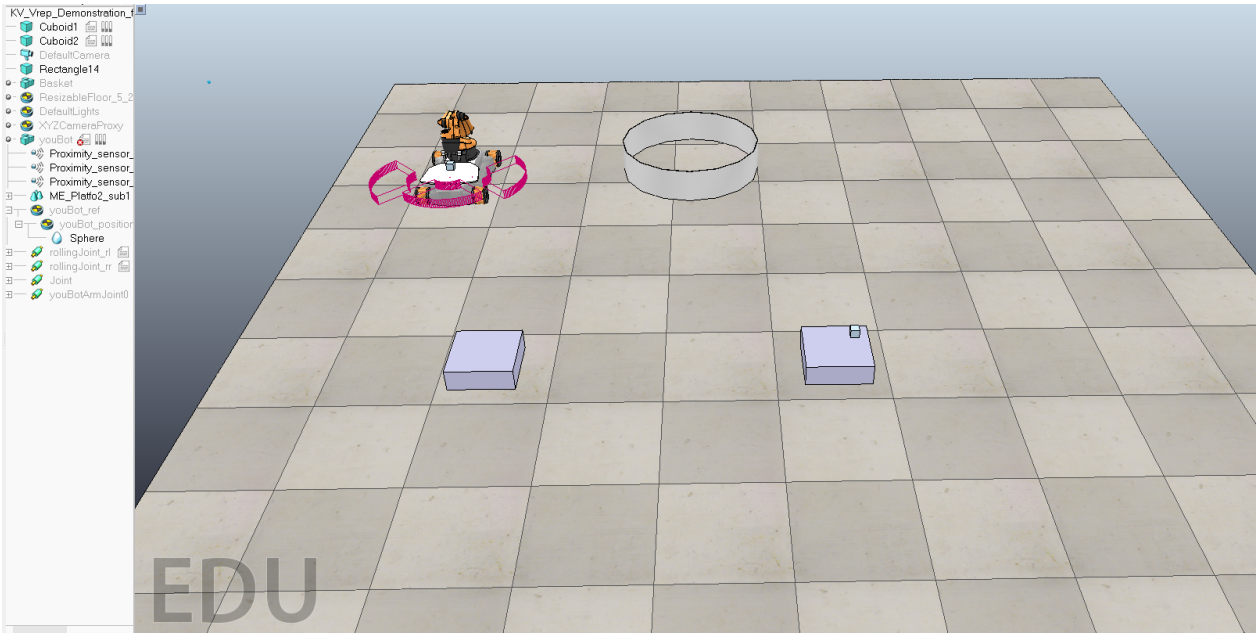

Figure 3: Initial test VREP environment.

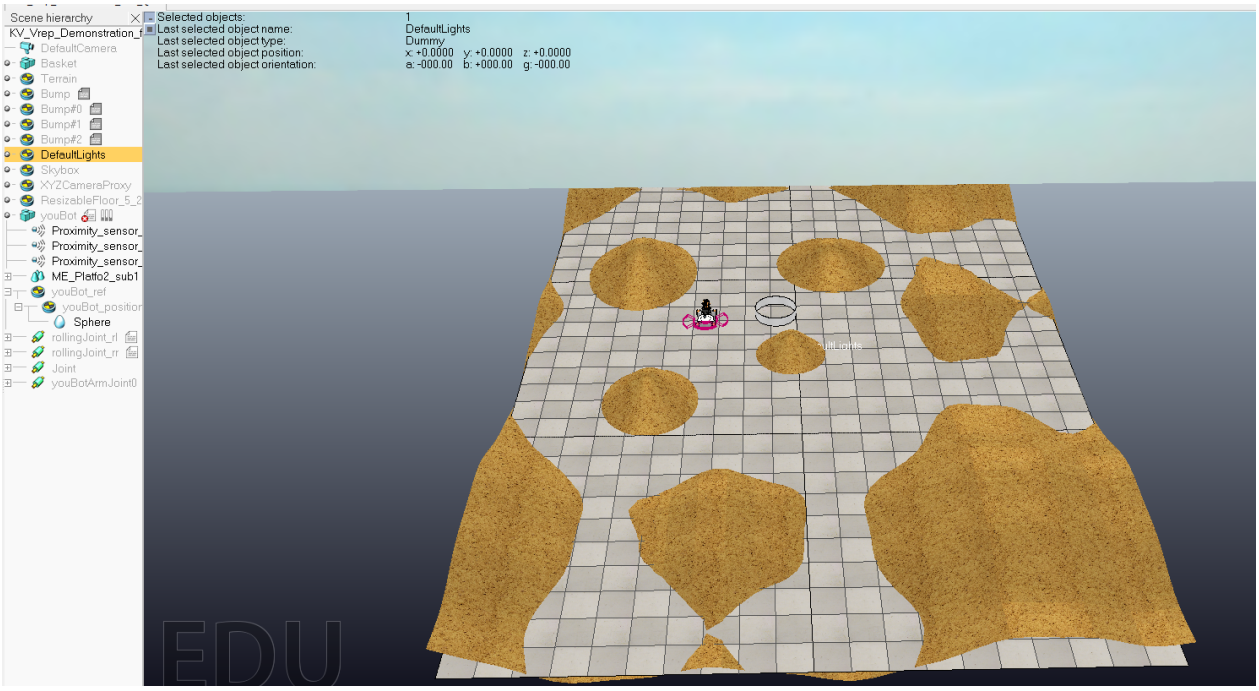

Figure 4: Second stage VREP environment for Question 2.

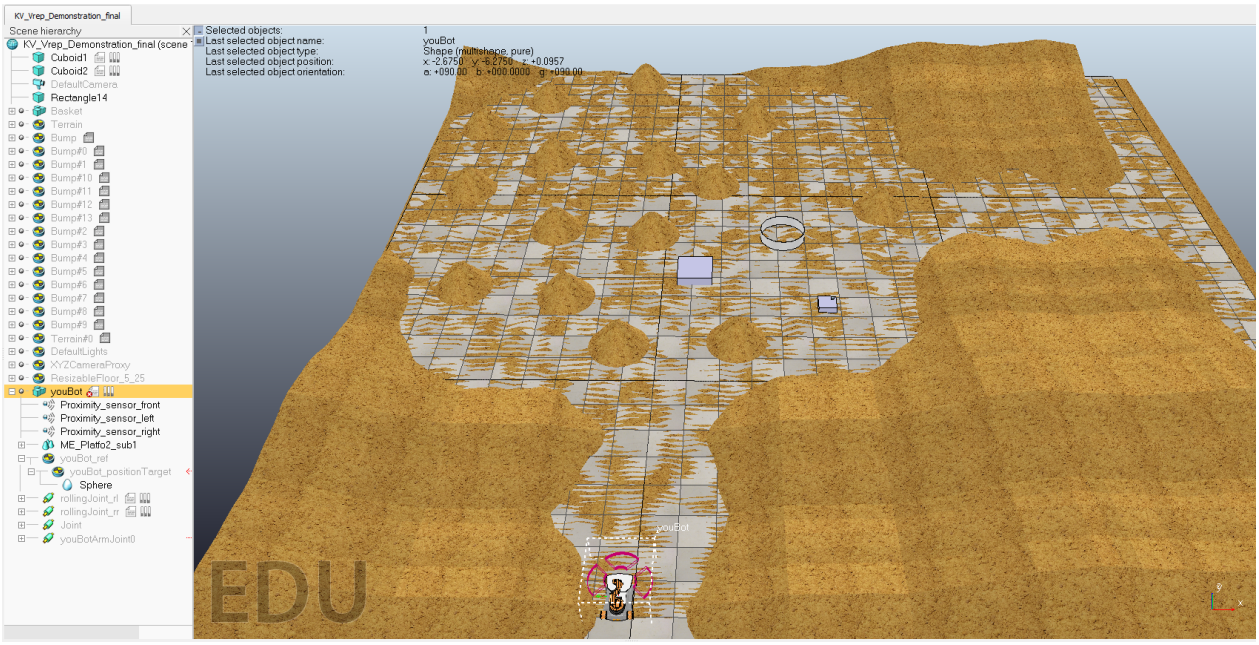

Figure 5: Final stage VREP environment for Question 2.

3 Challenges Faced

The following challenges were faced during the development of this program.

1. Initially I began programming the simulation using MATLAB.The problem faced in this method was mainly while taking keyboard inputs from the user. At first I tried using the 'input' function but that required pressing Enter key after every input which is not much fun. Then I tried using 'getkey' MATLAB plugin to take user inputs. The code worked using this method but there was a lot of lag. Finally, I decided to use Python instead.

2. The next problem faced was while interfacing python and VREP, the problem was due to a mismatch in the port number. This problem occurred when certain objects were deleted from the VREP environment. The problem was fixed by changing the port number to 19997.

3. Another problem faced was the program not detecting the RemoteApi.dll file. This problem occurred because the file being used was for a 32-bit system whereas my system was 64-bit. The problem was fixed by adding the correct file to the directory.

4. The sensor was returning a large negative value instead of zero when no object was in range of the sensor. It was solved by assigning the variables a fixed values if the sensor value was less than or equal to zero

4 Appendix

4.1 README

ENPM 690 Robot Learning

@ Author Kartik Venkat

4.1.1 Question 1:

Instructions to run the code:

1. Using PyCharm or any other IDE:
   Open KV_Vrep_Demonstration_final_Q1.ttt VREP file and Project3_Q1 python file and

   Run the VREP file first and then the Project3_Q1.py Python file.

2. Using Command Prompt: First run the KV_Vrep_Demonstration_final_Q1.ttt VREP file. The run the following command in command prompt: python ...PATH...$\backslash BehaviourRobot\backslash Project3\backslash Question1\backslash project3\_Q1.py$.

Once both files are running and properly linked, press UP,DOWN,LEFT,RIGHT Arrow Keys on the keyboard to control the robot movement.

Special Instructions:

1. Install all package dependencies like pynput before running the code.
2. Update pip and all the packages to the latest versions.
3. Change the port number from 19999 to 19997 or vice versa in case the connection to vrep fails
4. Make sure you have the following files in your directory:
   (a) vrep.py
   (b) vrepConst.py
   (c) simpleTest.py (or any other example file)
   (d) The appropriate remote API library: "remoteApi.dll" (Windows)(64-bit/32-bit), "remoteApi.dylib" (Mac) or "remoteApi.so" (Linux)
   (e) Go to this path directory to find the remoteApi.dll
   C:Files$\backslash V-REP3\backslash V-REP_PRO_EDU\backslash programming\backslash remoteApiBindings\backslash lib\backslash lib$.

.

4.1.2 Question 2:

Instructions to run the code:

1. Using PyCharm or any other IDE:
   Open KV_Vrep_Demonstration_final_Q2.ttt VREP file and project3_q2 python file and
   Run the VREP file first and then the project3_q2.py Python file.

2. Using Command Prompt: First run the KV_Vrep_Demonstration_final_Q1.ttt VREP file. The run the following command in command prompt:
   python ...PATH...$\backslash BehaviourRobot\backslash Project3\backslash Question2\backslash project3\_q2.py$.

Once both files are running and properly linked, the robot will wander around on its own whilst avoiding any obstacles in its path.

Special Instructions:

1. Install all package dependencies like pynput before running the code.
2. Update pip and all the packages to the latest versions.
3. Change the port number from 19999 to 19997 or vice versa in case the connection to vrep fails
4. Make sure you have the following files in your directory:
   (a) vrep.py
   (b) vrepConst.py
   (c) simpleTest.py (or any other example file)
   (d) The appropriate remote API library: "remoteApi.dll" (Windows)(64-bit/32-bit), "remoteApi.dylib" (Mac) or "remoteApi.so" (Linux)
   (e) Go to this path directory to find the remoteApi.dll
   C:Files$\backslash V-REP3\backslash V-REP_PRO_EDU\backslash programming\backslash remoteApiBindings\backslash lib\backslash lib$.

.

4.2 CODE Question 1:

```
1
2  # coding: utf-8
3
4  # ENPM 690 Robot Learning
5
6  # Assignment 3
7
8  @ author: Kartik Venkat
9
10
11
12 import sys
13
14 from pynput.keyboard import Key, Listener
15
16 import vrep
17 from threading import Thread
```

```
from time import sleep

flag = False

Speed = 4
# Use multi-threading to print the sensor values in the command window.
def threaded_function():
    while True:
        errorCode, detectionState1, detectedPoint1, detectedObjectHandle,
    detectedSurfaceNormalVector = vrep.simxReadProximitySensor(
            clientID, front_sensor, vrep.simx_opmode_streaming)
        if flag:
            print (detectionState1)

        print ("Front Sensor Reading :{}\n".format(int(detectedPoint1[2]*100) if
    detectionState1 else "Nothing in Range"))


        sleep(0.5)
        print ("_____________________________________________")


def on_press(event):
    if event == Key.up:
        if flag:
            print ("UP")
        vrep.simxSetJointTargetVelocity(clientID, wheel1, Speed, vrep.simx_opmode_streaming)
        vrep.simxSetJointTargetVelocity(clientID, wheel2, Speed, vrep.simx_opmode_streaming)
        vrep.simxSetJointTargetVelocity(clientID, wheel3, Speed, vrep.simx_opmode_streaming)
        vrep.simxSetJointTargetVelocity(clientID, wheel4, Speed, vrep.simx_opmode_streaming)
        print ("~~~~~~~~~~~~~~~~~ Move Forward ~~~~~~~~~~~~~~~~~")
    elif event == Key.right:
        if flag:
            print ("RIGHT")
        vrep.simxSetJointTargetVelocity(clientID, wheel1, Speed, vrep.simx_opmode_streaming)
        vrep.simxSetJointTargetVelocity(clientID, wheel2, -Speed, vrep.simx_opmode_streaming
    )
        vrep.simxSetJointTargetVelocity(clientID, wheel3, Speed, vrep.simx_opmode_streaming)
        vrep.simxSetJointTargetVelocity(clientID, wheel4, -Speed, vrep.simx_opmode_streaming
    )
        print ("~~~~~~~~~~~~~~~~~ Turn Right ~~~~~~~~~~~~~~~~~")
    elif event == Key.left:
        if flag:
            print ("LEFT")
        vrep.simxSetJointTargetVelocity(clientID, wheel1, -Speed, vrep.simx_opmode_streaming
    )
        vrep.simxSetJointTargetVelocity(clientID, wheel2, Speed, vrep.simx_opmode_streaming)
        vrep.simxSetJointTargetVelocity(clientID, wheel3, -Speed, vrep.simx_opmode_streaming
    )
        vrep.simxSetJointTargetVelocity(clientID, wheel4, Speed, vrep.simx_opmode_streaming)
        print ("~~~~~~~~~~~~~~~~~ Turn Left ~~~~~~~~~~~~~~~~~")
    elif event == Key.down:
        if flag:
            print ("DOWN")
```

```
66          vrep.simxSetJointTargetVelocity(clientID, wheel1, -Speed, vrep.simx_opmode_streaming
    )
67          vrep.simxSetJointTargetVelocity(clientID, wheel2, -Speed, vrep.simx_opmode_streaming
    )
68          vrep.simxSetJointTargetVelocity(clientID, wheel3, -Speed, vrep.simx_opmode_streaming
    )
69          vrep.simxSetJointTargetVelocity(clientID, wheel4, -Speed, vrep.simx_opmode_streaming
    )
70          print ("~~~~~~~~~~~~~~~~~ Move Backward ~~~~~~~~~~~~~~~~~")
71
72
73  def on_release(key):
74      if flag:
75          print ("{} released".format(key))
76      vrep.simxSetJointTargetVelocity(clientID, wheel1, 0, vrep.simx_opmode_streaming)
77      vrep.simxSetJointTargetVelocity(clientID, wheel2, 0, vrep.simx_opmode_streaming)
78      vrep.simxSetJointTargetVelocity(clientID, wheel3, 0, vrep.simx_opmode_streaming)
79      vrep.simxSetJointTargetVelocity(clientID, wheel4, 0, vrep.simx_opmode_streaming)
80
81
82  print ('Setting Up Simulation Environment...')
83  vrep.simxFinish(-1)  # just in case, close all opened connections
84  clientID = vrep.simxStart('127.0.0.1', 19997, True, True, 5000, 5)  # Connect to V-REP
85
86  if clientID != -1:
87      print ('Connection Established to remote API server')
88      returnCode, wheel1 = vrep.simxGetObjectHandle(clientID, 'rollingJoint_fr', vrep.
        simx_opmode_blocking);
89      print(returnCode)
90      print(wheel1)
91      returnCode, wheel2 = vrep.simxGetObjectHandle(clientID, 'rollingJoint_fl', vrep.
        simx_opmode_blocking);
92      print(returnCode)
93      print(wheel2)
94      returnCode, wheel3 = vrep.simxGetObjectHandle(clientID, 'rollingJoint_rr', vrep.
        simx_opmode_blocking);
95      print(returnCode)
96      print(wheel3)
97      returnCode, wheel4 = vrep.simxGetObjectHandle(clientID, 'rollingJoint_rl', vrep.
        simx_opmode_blocking);
98      print(returnCode)
99      print(wheel4)
100
101     returnCode, front_sensor = vrep.simxGetObjectHandle(clientID, "Proximity_sensor",
102                                                         vrep.simx_opmode_oneshot_wait)
103
104     thread = Thread(target=threaded_function)
105     thread.start()
106     #  Collect events until released
107     with Listener(
108             on_press=on_press,
109             on_release=on_release) as listener:
110         listener.join()
111 else:
```

```
    print ('Failed to connect to API server')
    sys.exit("Connection failed")
```

4.3 CODE Question 2:

```
# coding: utf-8

# ENPM 690 Robot Learning

# Assignment 3

#@ author: Kartik Venkat



import sys
import vrep
from time import sleep
from random import randint

debug = False

wheel1 = 2
wheel2 = 2
wheel3 = 2
wheel4 = 2
Turning = False



Speed = 6

print ('Program started')
vrep.simxFinish(-1)  # just in case, close all opened connections
clientID = vrep.simxStart('127.0.0.1', 19997, True, True, 5000, 5)  # Connect to V-REP

if clientID != -1:
    print ('Connected to remote API server')

    returnCode, wheel1 = vrep.simxGetObjectHandle(clientID, 'rollingJoint_fr', vrep.
    simx_opmode_blocking);
    print(returnCode)
    print(wheel1)
    returnCode, wheel2 = vrep.simxGetObjectHandle(clientID, 'rollingJoint_fl', vrep.
    simx_opmode_blocking);
    print(returnCode)
    print(wheel2)
    returnCode, wheel3 = vrep.simxGetObjectHandle(clientID, 'rollingJoint_rr', vrep.
    simx_opmode_blocking);
    print(returnCode)
    print(wheel3)
    returnCode, wheel4 = vrep.simxGetObjectHandle(clientID, 'rollingJoint_rl', vrep.
    simx_opmode_blocking);
    print(returnCode)
    print(wheel4)
```

```
47
48      returnCode, front_sensor = vrep.simxGetObjectHandle(clientID, "Proximity_sensor_front",
49                                                          vrep.simx_opmode_oneshot_wait)
50      returnCode, left_sensor = vrep.simxGetObjectHandle(clientID, "Proximity_sensor_left",
51                                                          vrep.simx_opmode_oneshot_wait)
52      returnCode, right_sensor = vrep.simxGetObjectHandle(clientID, "Proximity_sensor_right",
53                                                          vrep.simx_opmode_oneshot_wait)
54
55      while True:
56          returnCode, detectionState1, detectedPoint1, detectedObjectHandle,
        detectedSurfaceNormalVector = vrep.simxReadProximitySensor(
57              clientID, left_sensor, vrep.simx_opmode_streaming)
58          left_distance = int(100 * detectedPoint1[2])
59          if left_distance <= 0:
60              left_distance = 100
61          if debug:
62              print (detectionState1)
63          returnCode, detectionState2, detectedPoint2, detectedObjectHandle,
        detectedSurfaceNormalVector = vrep.simxReadProximitySensor(
64              clientID, right_sensor, vrep.simx_opmode_streaming)
65          right_distance = int(100 * detectedPoint2[2])
66          if right_distance <= 0:
67              right_distance = 100
68          if debug:
69              print (detectionState2)
70          returnCode, detectionState3, detectedPoint3, detectedObjectHandle,
        detectedSurfaceNormalVector = vrep.simxReadProximitySensor(
71              clientID, front_sensor, vrep.simx_opmode_streaming)
72          front_distance = int(100 * detectedPoint3[2])
73          if front_distance <= 0:
74              front_distance = 100
75          if debug:
76              print (detectionState3)
77
78          print ("Front Reading :{}\n".format(int(detectedPoint3[2] * 100) if detectionState3
        else "Nothing in Range"))
79          print ("Right Reading :{}\n".format(int(detectedPoint2[2] * 100) if detectionState2
        else "Nothing in Range"))
80          print ("Left Reading :{}\n".format(int(detectedPoint1[2] * 100) if detectionState1
        else "Nothing in Range"))
81
82
83
84          if left_distance < 28:
85              print ("~~~~~~~~~~~~~~~~~ Obstacle detected Turn right ~~~~~~~~~~~~~~~~~")
86              wheel1_speed = Speed / 4
87              wheel2_speed = -Speed / 4
88              wheel3_speed = Speed / 4
89              wheel4_speed = -Speed / 4
90          elif right_distance < 28:
91              print ("~~~~~~~~~~~~~~~~~ Obstacle detected Turn left ~~~~~~~~~~~~~~~~~")
92              wheel1_speed = -Speed / 4
93              wheel2_speed = Speed / 4
94              wheel3_speed = -Speed / 4
```

```
            wheel4_speed = Speed / 4
        elif front_distance < 30:
            print("Move Back...")
            wheel1_speed = -Speed / 4
            wheel2_speed = -Speed / 4
            wheel3_speed = -Speed / 4
            wheel4_speed = -Speed / 4

            vrep.simxSetJointTargetVelocity(clientID, wheel1, wheel1_speed, vrep.simx_opmode_streaming)
            vrep.simxSetJointTargetVelocity(clientID, wheel2, wheel2_speed, vrep.simx_opmode_streaming)
            vrep.simxSetJointTargetVelocity(clientID, wheel3, wheel3_speed, vrep.simx_opmode_streaming)
            vrep.simxSetJointTargetVelocity(clientID, wheel4, wheel4_speed, vrep.simx_opmode_streaming)
            sleep(1)
            if 50 > randint(0, 100):
                print("~~~~~~~~~~~~~~~~~ Random Turn Right ~~~~~~~~~~~~~~~~~")
                wheel1_speed = Speed / 4
                wheel2_speed = -Speed / 4
                wheel3_speed = Speed / 4
                wheel4_speed = -Speed / 4
            else:
                print("~~~~~~~~~~~~~~~~~ Random Turn Left ~~~~~~~~~~~~~~~~~")
                wheel1_speed = -Speed / 4
                wheel2_speed = Speed / 4
                wheel3_speed = -Speed / 4
                wheel4_speed = Speed / 4


        elif front_distance == 100:
            if 25 > randint(0, 100):
                if 30 > randint(0, 100):
                    print("~~~~~~~~~~~~~~~~~ Random Turn Right ~~~~~~~~~~~~~~~~~")
                    wheel1_speed = Speed / 4
                    wheel2_speed = -Speed / 4
                    wheel3_speed = Speed / 4
                    wheel4_speed = -Speed / 4


                elif 30 > randint(0, 100):
                    print("~~~~~~~~~~~~~~~~~ Random Turn Left ~~~~~~~~~~~~~~~~~")
                    wheel1_speed = Speed / 4
                    wheel2_speed = -Speed / 4
                    wheel3_speed = Speed / 4
                    wheel4_speed = -Speed / 4

                else:
                    print("~~~~~~~~~~~~~~~~~ Random Move Forward ~~~~~~~~~~~~~~~~~")
                    wheel1_speed = Speed / 4
                    wheel2_speed = Speed / 4
                    wheel3_speed = Speed / 4
                    wheel4_speed = Speed / 4
```

```
145
146             else:
147                 wheel1_speed = Speed / 4
148                 wheel2_speed = Speed / 4
149                 wheel3_speed = Speed / 4
150                 wheel4_speed = Speed / 4
151                 # LeftMotorSignal = Speed
152                 # RightMotorSignal = Speed
153         else:
154             wheel1_speed = Speed / 4
155             wheel2_speed = Speed / 4
156             wheel3_speed = Speed / 4
157             wheel4_speed = Speed / 4
158             # LeftMotorSignal = Speed
159             # RightMotorSignal = Speed
160
161         vrep.simxSetJointTargetVelocity(clientID, wheel1, wheel1_speed, vrep.
    simx_opmode_streaming)
162         vrep.simxSetJointTargetVelocity(clientID, wheel2, wheel2_speed, vrep.
    simx_opmode_streaming)
163         vrep.simxSetJointTargetVelocity(clientID, wheel3, wheel3_speed, vrep.
    simx_opmode_streaming)
164         vrep.simxSetJointTargetVelocity(clientID, wheel4, wheel4_speed, vrep.
    simx_opmode_streaming)
165
166
167         sleep(0.5)
168         print ("_____________________________________________")
169 else:
170     print ('Failed connecting to remote API server')
171     sys.exit("Connection failed")
```