King Fahd University of Petroleum and Minerals
Department of Computer Engineering

INTRODUCTION TO ROBOTICS COE 484

Homework No 2 (Due on March 29, 2008)

Questions	Grading
1	
2	
3	
4	
5	
TOTAL	

Design of a simple reactive bahavior using SimRobot

Simrobot is a simulator that allows simulating an autonomous robot operating in defined field. Simrobot allows programming the use of geometric modeling to define the scene, the field which the robot moving area, the robot, and other field features and objects. A demo was presented to you regarding the Simulator with a simple reactive behavior. The demo was about a moving robot which senses its collision with an obstacle and the robot bounces back whenever its sensor detects such a collision. The robot controller was about moving in one direction and when a collision is predicted through sensor data, the robot reverses its direction of motion. Note that the controller describes one iteration of the above behavior and it is assumed that the iteration is repeated for ever. In this homework the student is requested to build up on the above demo and design a reactive controller. Answer each of the following questions:

1. Study the provided program on WebCT, both the scene description and the controller (C++) code. Write a short description of the program and comment on the parts you find it difficult to understand.

2. Create your own robot of type ”VEHICLE” and use a cylinder object to represent it with the following specification: height of 15 and base diameter of 30. The color should be yellow.

3. Use of C++ code to program the controller, when it finds an obstacles it goes back (bouncing or reversing direction) for about 50 points and then starts rotating to avoid the obstacle. You can create your own obstacle as a test bed of your program.

4. Design the ball to move when the robot hits it, or use another type of sensor and reads the data and implement some action, or any other type of improvements. Explain your idea and testing results in the homework.

5. Submit a short report describing the experiment and your code to the instructor and to the WebCT.

Solution

1. SimRobot is a simulator composed of two main components; one of which is the Controller, the other is the scene. The scene is just a description of the World will be displayed on the GUI on which the robot will appear and move according to the controller program. The scene contains the dimension of each and every object in the World and their colors and materials, which they are made of. Also, it contains the geometric relation between every one. Moreover, the scene classifies every object in one of the following categories: Sensor, Actuator and different object. On the other hand, the controller is more like the brain of everything. It controls the objects and identifies the reaction of each in different aspects of time. Also, it tries to employ some of logic in the scene. The reaction consists of reading some sensors (S), carrying out some computation (X=F(S,…)) on sensor data and others, and outputting the computation result as motion command to the robot (such as Move robot to X).

2. The example that was already given in the homework is well-made. The robots have two different speeds. The bouncing algorithm was simple but effective. A simple sensor was used to implement the bouncing algorithm. Such sensor was monitoring the execution of the movement, sort of feedback from the motors. When a lack of movement was detected by the Sensor the controller try to set a rotation around the Z-Axis and try to inverse the Direction of the movement. Till the movement stops again, where the Controller sets everything back to normal, no rotation and forward Direction of movement.

3. The implementation: it is noticed that the controller get called, from a programming point of view, at every point the object passes 1 point of movement. Therefore, we may implement the bouncing algorithm to count the steps that the robot takes in its movement. Also, we may use the usual sensor used in the given example to detect an obstacle. Then, we may bounce the robot by 50 points backward and try to rotate 90 degrees (to the left or to the right) and going around the object again so it can continue its normal line of movement. Figure 1 and the line drawn on it illustrate this motion.

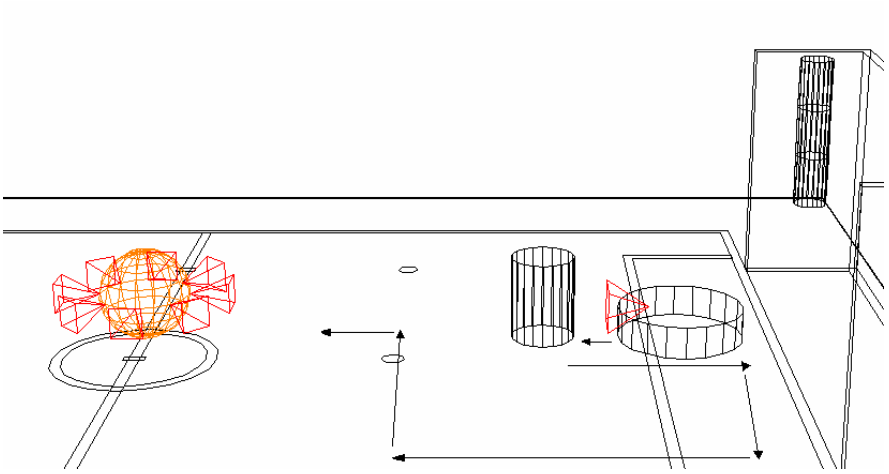

Figure 1: Bouncing Algorithm (Thanks to Mr. Al-Hawaj for the Figure)

To implement such an algorithm, we may use a number of states where every state represents one angle in the line of movement.

Alternative 1: Another strategy for this problem is to use a counter for motion control following the hit. When the robot is not moving due to obstacle, it reveres the direction and reset the counter. It continues moving the reverse direction until the counter reaches the 40 frame. In the next 30 counter frames, the robot start rotating. After the 70^{th} frame, the robot continues moving in the same direction as it was before it had hit the obstacle.

Alternative 2: Some refinements are also possible. For example, the robot may bounce back just until it does not detect the collision any more. Then, it translates to its right or its left by a small amount like 10 points without turning. And then moves forward again while sensing for more collision. The original direction of motion is then maintained. In the case of a collision, the algorithm repeats again. This will allow the robot the least deviation from its original motion direction.

4. The Ball hitting algorithm is more interesting. To detect a hit to the ball we use Ultrasonic sensors that can be created in SimRobot. A set of Ultrasonic sensors may be used to detect the distance between the ball and the object in front of it. Using a threshold value we may program the controller to release the ball. The movement of the ball was

looking like it was hit and then the speed of the ball started to decrease, till it stops. Every step or rotation of the Controller the speed of the ball was decreased by some value till the value becomes zero. There is about eight ultrasonic in the ball to detect exactly where the place of the hit was detected.

5. Note on implementation: Here are the objectives:
    - To create a robot of type “VEHICLE” in SimRobot as a cylindrical yellow shape with height = 15 and diameter = 30.
    - Program the robot that if hit anything it will go back 50 points and then it will try to go around the object.
    - Design a ball that moves when hit.

    Design:

    - The Robot can be designed easily by using both VEICLE and POLYEDER objects. Then the behavior of the robot can be controlled by the controller itself. The robot will move along its x axis as long as there are no obstacles. If the robot hit anything, it will back 50 points and then turns 40 degree. After that it starts moving normally. The following code describes the behavior of the robot.

      ```
      if(!srExecuted3&&start!=0)
      {
      s23test = true;
      }
      if(s23test){
      if (step == 0){
      if (rot != 0){
      SetActorValue(m_apMove3[1],m_srDir[2]);
      rot--;
      }
      else{
      step = 50;
      rot = 40;
      s23 = (s23+rot)%360;
      s23test = false;
      }

      }
      else{
      SetActorValue(m_apMove3[0],-m_srDir[2]);
      step--;
      }
      }
      else{
      SetActorValue(m_apMove3[0],m_srDir[2]);
      }
      ```

    - On the other hand, the ball can also be implemented as VEICLE that has 8 ultrasonic sensors. Sensors are distributed around the ball at each angle of 45 degrees. Those ultrasonic sensors are used to detect if an object has hit the ball and from where it was hit. Then the speed of the ball is decreased each frame to simulate fraction.

      ```
      if(ballmove4)
      ```

```
{
SetActorValue(m_apMove4[1],-m_srDir[4]);
m_srDir[4] = m_srDir[4]-0.009F;
if(m_srDir[4]<=0)
{
ballmove4=false;
m_srDir[4]=1.0F;
}
}
```

- The above code is an example of how the ball behaves when the fourth ultrasonic sensor detects that it was hit, which is on angle 270 degree. The ball will move to the opposite Y axis and starts decreasing its speed.

```
if(ballmove7)
{
SetActorValue(m_apMove4[0],m_srDir[4]);
SetActorValue(m_apMove4[1],m_srDir[4]);
m_srDir[4] = m_srDir[4]-0.009F;
if(m_srDir[4]<=0)
{
ballmove7=false;
m_srDir[4]=1.0F;
}
}
```

- The above code is an example of how the ball behaves when the seventh ultrasonic sensor detects that it was hit , which is on angle225 degree. The ball will move to the opposite Y axis, opposite x axis and starts decreasing its speed