Learning Roomba: Student’s Guide
Module 4 Sensors and Actuators

Drew Housten (dhousten@gmail.com)
1 What is a Sensor?

A sensor is something that can detect the environment. It can help the robot determine where it is, where other objects are, what the present situation looks like, and other factors that will help it make decisions about what to do. Humans have sensors as well. Human sensors include our eyes (sight), ears (hearing), nose (smell), tongue (taste), and skin (touch). We also use internal sensors to know when we are tired, when we are hungry, or when we are in pain. Robots can have a number of similar sensors to human sensors.

2 Types of Sensors

Sensors come in many shapes and forms. Here are some common sensors used on robots.

- Sonar: Distance sensors using an ultra-sonic signal to measure distance between the sensor and the nearest object. These work similar to the way bats “see”.
- Infrared: These sensors are similar to sonar sensors except that they use infrared light signals instead of ultra-sonic sound signals to measure the distance.
- Ladar: These are also another variant of the distance sensors, but use lasers instead of ultra-sonic sound signals.
- Wheel encoders: The encoders measure the angle that a wheel rotates, which can be used, along with the diameter of the wheel, to calculate a rough distance traveled by the wheel.
- Cameras: Cameras provide visual information to the robot.
- GPS: Global Positioning Systems (GPS) help locate the robot by communicating to satellites.
- Microphones: These sensors provide a “hearing” ability to the robot.
- Bump Sensors: These sensors tell a robot when it has touched something in the environment.

3 What is an Actuator?

An actuator is something that the robot can use to interact with the environment. We have already talked about one of the types of actuators in a previous Module, the drive motors. Drive motors are not the only type of actuators though. Similar to sensors, humans also have actuators. Our muscles and vocal chords are just two types of actuators that we have.
4 Types of Actuators

Like sensors, actuators come in many shapes and forms. Here are just a couple actuators typically used on robots.

- Drive motors: Allow the robot to move around by driving wheels.
- Other motors: Motors can be used in other ways to enable picking things up with an arm, steering the wheels, or other activities that need a motor.
- Speakers: Speakers allow a robot to “talk”.
- Lights: Lights on the robot help the robot communicate status or other information as well as to see in dark areas.

5 How can sensors and actuators be used?

How sensors and actuators are used depend on the purpose of the robot as well as the programmer. Actuators can be used to climb stairs or sensors can be used to prevent falling down stairs. Distance sensors can be used to navigate, to build a map, to avoid obstacles, or just to classify objects in the environment. Wheel encoders can help determine where a robot is or can help detect when wheels are slipping. Lights can be used for communication to another robot or could just provide status to humans in the environment. Speakers could play music or could output speech. The list continues indefinitely. One of the challenges of a roboticist is to know when to use actuators and sensors and how to use them.

6 Roomba’s Sensors and Actuators

The Roomba has a number of sensors and actuators available to use. The actuators include two drive motors, a speaker, LEDs, and in some models, vacuum and brush motors. The sensors include wheel encoders, bump sensors, IR wall sensors, an IR receiver, cliff sensors, buttons, wheel drop sensors, several battery and current sensors, and in some models, dirt-detection sensors. Each of the actuators and sensors are described here.

- Drive motors: Drive the Roomba’s wheels. There is one on each side of the Roomba.
- Speaker: Allows the Roomba to play songs or status beeps.
- LEDs: Lights on the top of the Roomba. These can be used to display status information.
- Vacuum/Brush motors: These are motors to control the vacuuming functions of the Roomba. Because the Roomba Create is not meant as a vacuum cleaner, that model does not have these motors.
- Wheel encoders: Can measure how much each wheel has rotated. From this information, the translational and rotational position of the Roomba can be calculated.

- Bump sensors: There are 2 bump sensors in the front of the Roomba. These can tell if the Roomba has collided with an object in the front-left or front-right of the Roomba.

- IR wall sensors: These are infrared distance sensors that can detect the wall on the right side of the Roomba.

- IR receiver: This is an infrared receiver for a non-distance infrared signal from one of the Roomba Virtual Walls (See Figure 1). The Virtual Walls emit a beam that can be used to restrict the Roomba’s travel area (See Figure 2) or for approximate positional information.

- Cliff sensors: The cliff sensors look down and detect when the front of the Roomba has started to go over a cliff.

- Buttons: There are several buttons on the top of the Roomba. A program can detect when one or more of those buttons are pressed.

- Wheel drop sensors: The Roomba has the ability to detect when a wheel has dropped. This usually happens if the Roomba gets picked up or is stuck on something such that a wheel is off the ground.

- Battery/Current sensors: There are several sensors to detect the status of the battery or current to the motors.

- Dirt-detection sensors: In the vacuum models of the Roomba, there are dirt-detection sensors that will detect when the Roomba has hit a high-dirt area.

Figure 1: Roomba Virtual Wall units
7 Decision-Making

After the capabilities and limitations of the available sensors and actuators are understood, decisions need to be made about how to act. There are several different ways to make those decisions. Two approaches are described here: a behavior-based approach and a sense-plan-act approach.

7.1 Behavior-based Robotics

A behavior-based robotics approach uses the sensor data to immediately make actions. The robot would not have any model or memory of the environment and instead relies on the current state of the sensors. Complex behavior emerges by combining simple, reactive components into a larger whole. An example of a reactive component would be something like “if cliff detected, stop motors”. Each component is very simple and quick to react.

The following simple example will help to illustrate this concept. This example and the diagrams are borrowed from Valentino Braitenberg’s book “Vehicles: Experiments in Synthetic Psychology”. Our robot starts with a single motor hooked to a single sensor (Figure 3). The sensor will detect light and the stronger the light, the faster the motor goes. It is very simple, but will react to its environment.

Figure 3: Braitenberg’s Vehicle 1 with one motor and one sensor
The vehicle will get slightly more complicated if we have two motors and two sensors. There are two ways to hook up the sensors: left sensor going to the left motor and right sensor going to the right motor (Figure 4) or left sensor going to the right motor and right sensor going to the left motor (Figure 5). The robot will exhibit different behaviors depending on how the sensors are hooked up. In the first example (Figure 4), if a light source is detected on the left of the robot, the left motor will increase speed and the robot will veer away from the light source (remember the Motion Module and swing turns). This robot can be considered cowardly to light sources. In the second example (Figure 5), if a light source is detected on the left of the robot, the right motor will increase speed causing the robot to veer towards the light source. As the vehicle gets closer, the vehicle will increase speed. This robot can be considered aggressive to light sources.

Figure 4: Braitenberg’s Vehicle 2a with two motors and two sensors, hooked up with the left sensor controlling the left motor and the right sensor controlling the right motor

Figure 5: Braitenberg’s Vehicle 2b with two motors and two sensors, hooked up with the left sensor controlling the right motor and the right sensor controlling the left motor

This is a simple example, but it shows that simple, reactive components can be combined to exhibit different behaviors. There is no memory of the environment or any planning, yet the robot can be considered to be cowardly or aggressive. Furthermore, the complexity can continue to increase by combining the same basic components in different ways by inhibiting motor control or detecting additional environmental factors.
7.2 Sense-Plan-Act Systems

On the other-hand, a Sense-Plan-Act (SPA) approach (Figure 6) maintains an internal model about everything that it has observed. The robot would take the sensor input and its memory about the environment and past conditions to form a plan about how to act in the environment. The control flow goes from the environment to the sensors to the planning stage to the actuators back out to the environment.

![Figure 6: Sense-Plan-Act (SPA) Approach](image)

The argument is that some long-term knowledge is necessary in order to make effective decisions. However, planning is typically slower than pure reactive-based approaches, which could prevent effective safety functions. If a cliff is detected but it takes several minutes to make a decision about what to do, the robot may already have gone over the cliff by the time a decision is reached.

8 Sensor and Actuator Code Interfaces

The sensors and actuators that you will have access to through code are provided here with the exception of the drive commands. Those commands are found in Module 3.

- `pause(duration)`: The Roomba will not process any other commands for the specified duration in seconds.
- `setStatusGreenLED(status)`: Turns on or off the green status LED. If status is `true` then the LED will be turned on. If status is `false` then the LED will be turned off.
- `setStatusRedLED(status)`: Turns on or off the red status LED. If status is `true` then the LED will be turned on. If status is `false` then the LED will be turned off.
- `addSongNote(notenum, duration)`: Adds a note to the current song. The song does not play until `playSong()` is called.
- `clearSong()`: Clears the current song
- `playSong()`: The Roomba will play the current song
• `getLeftBump()`: This method will be true if the left bump sensor has been hit.

• `getRightBump()`: This method will be true if the right bump sensor has been hit.

• `getCliffFrontLeft()`: This method will be true if a cliff has been detected in the front left.

• `getCliffFrontRight()`: This method will be true if a cliff has been detected in the front right.

• `getCliffLeft()`: This method will be true if a cliff has been detected on the left.

• `getCliffRight()`: This method will be true if a cliff has been detected on the right.

• `getWall()`: This method will be true if the IR sensor has detected a wall on the right of the Roomba.

• `getVirtualWall()`: This method will be true if the Roomba has detected a virtual wall beam.

• `getLeftWheelDrop()`: This method will be true if the Roomba’s left wheel has dropped down.

• `getRightWheelDrop()`: This method will be true if the Roomba’s right wheel has dropped down.

• `getCenterWheelDrop()`: This method will be true if the Roomba’s front caster wheel has dropped down.

• `getOdometryX()`: This method will return the Roomba’s x-position according to the odometry readings. Position 0 is the start position of the Roomba.

• `getOdometryY()`: This method will return the Roomba’s y-position according to the odometry readings. Position 0 is the start position of the Roomba.

• `waitForVirtualWall()`: This method will wait until a virtual wall has been detected.

• `waitForBump()`: This method will wait until a bump sensor has been hit.
9 Exercises

- Write a program that will have the Roomba drive forward until a virtual wall is detected. Have it then turn around and drive forward until a bump sensor is hit. As soon as that happens, make the roomba stop.

- Write a program to make the Roomba spin until it has been picked up (the wheels drop). As soon as that happens, stop the motors and make the Roomba play a song.

- Write a program to make the Roomba drive forward until it detects something. If an object is detected through the bump sensors, have the Roomba turn around and drive forward for 1 meter. If a virtual wall is detected, have the Roomba turn 90 degrees to the right and stop.

10 Homework

Write some pseudocode for a program that will drive forward forever, except if a wall, cliff, or other object is detected. If so, have the robot stop moving.