Lab 3

November 25th, 2019

Bonus points. For this lab, you can earn up to **2 bonus points** for code quality. To earn full credit, comply with the Java style guide (available through StudIP) and upload your solution to the Git repository of your team. Upload your solution before **December 6th, 23:59h** – later changes will not be considered.

We will only evaluate the solution developed in **Task 3.3**. After examination of the code quality, we will mark flaws directly in the code and add the bonus point credit to your repository. We will notify you via StudIP once you can find the evaluation in your repository.

Introduction. The following task will introduce you to the available sensors of the robots and the corresponding interfaces of the LeJOS API. All sensors will help your robot navigating through the maze during the final challenge, so make sure you know how to use them and be aware of their limitations.

Task 3.1 : Color Sensor

Sensor principle. Centered at the front, the robot is equipped with a color sensor. It uses the ambient light or an LED to measure the intensity of the reflected light of a surface on three channels: red, green and blue. In addition to separate RGB values, it provides a classification of the detected color.

Program description. Write a program that continuously determines the color of the surface in front of the sensor. Use the color ID to print the detected color on the LCD. Examine the behavior of the sensor with differently colored walls – which colors are detected robustly, which might cause problems? Alter the distance between sensor and wall – what is the distance range for correct readings? Write this value down, since you need it for further tasks.

Hints. The class `EV3ColorSensor` provides access to the capabilities of the sensor. Instead of providing the values directly, the color sensor uses the class `SensorMode`. For RGB readings use the lines

```
EV3ColorSensor colorSensor = new EV3ColorSensor(SensorPort.S1);
SensorMode color = colorSensor.getRGBMode();
```

and for the color identification mode use the method `getColorIDMode()`.

To obtain sensor readings, use the method `fetchSample(...)` of the sensor mode class. You have to create a floating point array first to use it as parameter for `fetchSample(...)`. Keep in mind that you have to choose the right size for the array. The sensor mode provides the length of its samples by the method `sampleSize()`. For further capabilities, you are referred to the API.

Task 3.2 : Ultrasonic Sensor

Sensor principle. To prevent your robot from colliding with objects in front of it, it is equipped with an ultrasonic sensor for distance measurement. The robot transmits acoustic waves above the spectrum audible by humans, detects the reflected wave, and uses the time-of-flight to obtain a distance measurement.

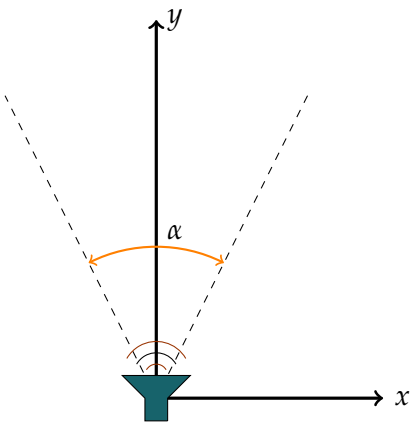

Figure 1: Field-of-view of the ultrasonic sensor; z-dimension is omitted for simplicity.

Program description. Write a program that reads the distance provided by the ultrasonic sensor continuously. Print the distance in cm to the LCD.

If available, use a measuring stick to confirm accurate readings. Check the detectable distance range of the sensor. What happens if you place objects closely? Explain this behavior.

Since the sensor uses directed speakers and microphones, its "field-of-view" is cone-shaped (as illustrated in Fig. 1). Determine the angle α empirically by moving objects at constant y (e.g. constant distance) in x direction (e.g. into our out from the cone) Can you imagine problems with this behavior at junctions?

Bonus: Collect at minimum ten samples, calculate the standard deviation of them, and print it to the LCD. Compare it for high and low distances.

Hints. LeJOS provides the class `EV3UltrasonicSensor` for communication with the sensor. Again, values are not provided directly but through class `SampleProvider`. Use the lines

```
EV3UltrasonicSensor distanceSensor = new EV3UltrasonicSensor(SensorPort.S4);
SampleProvider distance = distanceSensor.getDistanceMode();
```

for initializing the distance senor. Afterwards, the same methods as explained in Task 3.1 can be used.

Task 3.3 : Approaching a wall

Now you combine reading the color sensor, reading the distance sensor and driving the wheel motors, c.f. Task 2.3. Write a program that moves the robot forward in a straight line until an obstacle is detected. Approach the obstacle until you reach a distance at which you can determine the color of the obstacle accurately. Display the current distance to the obstacle to the LCD while moving and print the color to the LCD when you approached the obstacle.

Task 3.4 : Bonus: Gyro sensor

Sensor principle. On top of your robot, you find a gyroscope which provides readings about changes in angular velocity while moving of the robot. It can thus be used to determine the turning angle of your robot during movement.

Program description. Write a program that queries the sensor continuously and outputs the current turning angle to the display. Drive one wheel motor to turn the robot on the spot. Calculate the number of full turns, i.e. 360° and output it to the LCD.

Hints. Use the class `EV3GyroSensor` of LeJOS for communication with the sensor. To get the accumulated angle since last reset of the sensor, use the method `getAngleMode()` which returns a `SampleProvider`. You can use the same methods as in previous tasks to fetch the sensor readings.