SOFTWARE DOCUMENT

GROUP 4

2nd December 2016
Version 2

Contents

1 BACKGROUND

1.1 Edit History

Alexis Giguere-Joannette: 2016-10-28 document created

Tristan Saumure Toupin: 2016-10-29 work on second half of API research

Alexis Giguere-Joannette: 2016-10-30 work on first half of API research

Jake Zhu: 2016-10-30, Ported to LaTeX

Alexis Giguere-Joannette, Kareem Halabi & Tristan Saumure Toupin: 2016-10-31 work on the flowchart and class hierarchy

Alexis Giguere-Joannette: 2016-10-30 update the architecture section

Tristan Saumure Toupin & Alexis Giguere-Joannette: 2016-11-07 updated the class hierarchy

Jake Zhu: 2016-12-01, Edited the architecture section

Tristan Saumure Toupin & Alexis Giguere-Joannette : 2016-12-01, Edited the javadoc section

2 ARCHITECTURE

2.1 Flowchart

See Software Flowchart V2.pdf

2.2 Class Hierarchy

Two file types of the class hierarchy have been included in the directory: Class Diagram.pdf and Class Diagram.png

3 API RESEARCH

For this section we selected 13 lejos packages that could possibly be useful. We then describe the interfaces, classes and method that we could use during the final project.

3.1 Package lejos.hardware

The hardware package of the leJOS library allows to access all the components related to the brick. It contains many useful interfaces such as the Audio interface to play sounds (beeps) and the Power interface to get information about the EV3 battery and current usage. There are also multiple hardware related classes such as the user interaction with the brick either from the buttons, the bluetooth connection or a WiFi connection.

3.2 Package lejos.hardware.lcd

The LCD package is used to display data on the LCD screen of the EV3 brick. The TextLCD interface is the one used to display text to the screen. The LCD class contains all the methods used for controlling the screen of the brick such as the clear() and drawInt() methods.

3.3 Package lejos.hardware.motor

This package is used to control motors connected to the EV3 brick. It contains methods to control the motors by setting the speed, setting the acceleration, starting and stopping. There are also methods that allows to get tachometer data that can be used to do an odometer. There are also methods to move either forward or backward, to rotate a certain number of degrees and to wait until the rotation is complete.

3.4 Package lejos.hardware.port

There are basically two interfaces used from the port package. One for the motors ports (MotorPort) and the other for the sensors (SensorPort). Both interfaces allows to select from four different ports. In the MotorPort interface, the choices for ports are A, B, C, D and for the SensorPort they are S1, S2, S3, S4.

3.5 Package lejos.hardware.sensor

The sensor package contains all necessary methods to access the sensors connected and get data from it. The SensorMode interface allows to select the mode from the sensor connected. This interface extends the SampleProvider interface from the robotics package that contains the fetchSample() method to retrieve data from the sensor.

3.6 Package lejos.robotics.filter

The filter package contains a lot of filter classes that can be implemented to filter data received from the sensors for example. Most of the filters classes constructors take as input the data from a SampleProvider object.

3.6.1 ConcatenationFilter Class

This class can concatenate two different sources by inputting both their SampleProvider objects. Concatenating two sources of data can be used to use multiple sensors and filter its data from only one concatenated source.

Useful Classes

- MaximumFilter takes as input a SampleProvider object and an integer bufferSize. It returns the maximum values in the most recent samples. The number of samples is from the bufferSize input.
- MinimumFilter takes as input a SampleProvider object and an integer bufferSize. It returns the minimum values in the most recent samples. The number of samples is from the bufferSize input.
- SumFilter takes as input a SampleProvider object and an integer length. It returns the sum of the most recent samples. The number of samples is from the length input.

3.7 Package lejos.robotics.geometry

The geometry package of the leJOS library is used to create shapes such as a point, a line and a rectangle. Creating such shapes provides the ability to use geometry methods from the Shape interface through the different geometry classes. For example, you can create a rectangle area and a method to know if a point is inside the area can be used. Another application of this package is to use the line class to represent a line and use a provided method to compute the point of intersection of two lines. Methods are also available to represent shapes in a two dimensional space. A useful application of these methods would be to represent the entire path of the robot to know if it already went to a destination.

3.8 Package lejos.robotics.localization

3.8.1 Interface BeaconLocator

This interface will allow us to scan the whole map from anywhere using the method locate() which return an ¡Arraylist¿ with angles corresponding with the angle where a beacon has been seen. The angles are measured counter-clockwise: 0 is ahead, 90 is on the left of the robot, 180 is behind and 270 on the right.

3.8.2 Class OdometryPoseProvider

This Class allows the user to keep track of the position of the robot. For instance, the method getPose() will return an object Pose (see class pose under section 3.10 Package lejos.robotics.navigation) which is a Point object (the coordinates x and y) and another float (the heading angle). This Class is basically an odometer.

3.9 Package lejos.robotics.mapping

3.9.1 Class LineMap

This Class creates a map of a closed environment. The measurements stored are represented by Line objects. This class have many method that might be useful for the project such as getBoundingRect() which returns the bounds of the map. Also, the method inside(Point p) returns a boolean which tell if a point is inside the bounds and range(Pose pose) which returns the nearest wall.

3.10 Package lejos.robotics.navigation

3.10.1 Interface MoveListener and Interface MoveProvider

The interface MoveListener has to be implemented in order to update any classes that use the interface MoveProvider. This implementation will provide a movement to a pose when a moving class such as Move or MovePilot is used.

3.10.2 Interface NavigationListener

This is simply to tell a class that a waypoint is reached or that a movement has been completed.

3.10.3 Class Pose

This class provide an object Pose which is composed of a 3 floating points: coordinate x, coordinate y and the heading angle. The coordinates x and y are an object Point. This class will be useful for the odometer as well.

3.10.4 Class Navigator, Class Move and Class MovePilot

These classes were build to make movements. This is the equivalent to the navigation method that we used for the laboratories. They will allow us to turn, rotate, travel a distance, travel to a point and much more.

3.11 Package lejos.robotics.objectdetection

3.11.1 Interface Feature, Interface FeatureDetector and Interface FeatureListener

The interface Feature is basically made to describe an object detected by one or more sensors. The two other interfaces have to be implemented in order to access and create qualities to a scanned object.

3.12 Package lejos.robotics.pathfinding

3.12.1 The classes to find a path

Lejos provides many classes that help find a path to go from point a to b. They are all based on different algorithm which all use a Node object. For instance, the Class NodePathFinder might be used to go to the Green Zone or the Red zone.

3.13 Package lejos.utility

3.13.1 Class DebugMessages

This class can be used to show error messages. Lejos created this to help developpers design algorithms by identifying bugs. This class was created specially for the NXT bricks. Therefore, it might not work properly on the EV3.

3.13.2 Class Timer and Class Stopwatch

These two classes will help coordinate decisions taken by the robot such as fetching the sample by a sensor.

4 Javadoc

NOTE: The javadocs can be found in the doc folder of the DPM-TEAM04-Source folder. The name of the html file is index.html.

5 GLOSSARY OF TERMS