Testing Report

for

Lunar Rover Mapping Robot Controller

Version 1.0.0

Prepared by Hoang Long Pham
SEP UG-17

School of Computer Science,
The University of Adelaide

2017 October 30

Change History

Version	Dated	Edited By	Change Summary
0.0.1	15/10/2017	Simon Gray	Initial Draft
0.1.0	24/10/2017	Simon Gray	Changing Format
0.2.0	26/10/2017	Simon Gray, Hoang Long Pham and Matthew Kozirev	Verifying Results
1.0.0	30/10/2017	Benjamin Schuh	Final grammar and formatting

Related Documentation

ID	Document Name	Version
1	Software Requirements Specifications	2.3.0
2	Software Design Documentation	2.0.0
3	Software Project Management Plan	2.0.0
4	User Manual	1.0.0

Contents

1 Introduction

This document presents the acceptance test results for the Lunar Rover Mapping Robot.

1.1 Purpose

The purpose of this document is to present the verification results for the requirements as they are listed in the SRS. It will aim to show the results of testing and prove that robot has or hasn't passed the corresponding requirements.

1.2 Scope

This document aims to include all of the test results to verify that the Lunar Rover Mapping Robot Controller successfully fulfils its implemented requirements.

1.3 Document Overview

The document is provided in five sections and contains information about the task of the project and other knowledge that has been identified as required knowledge.
The Second section defines the situations in which the tests were conducted, along with the personnel that were involved in the tests.
The third section lists the tests that were conducted along with the baseline results that were required of the project.
The fourth sections lists how the tests proved certain requirements that were listed in the SRS, along with the test method that was used, and if the test passed. It also includes whether the requirements listed were verified.
The fifth section lists the conclusion of the tests and includes a high level summary of all the results from the tests conducted.

1.4 Acronyms and Abbreviations

Table 1: Acronyms

Acronyms	**Meaning**
GUI	Graphical User Interface
NGZ	No Go Zone
Rover	”LEGO MINDSTORMS Education EV3 Core Set” construction hardware
SDD	Software Design Document
SPMP	Software Project Management Plan
SRS	Software Requirement Specifications
UI	User Interface
DTD	Document Type Definitions
PC	Personal Computer
XML	Extensible Markup Language, format used to display mapping data

2 Test Conditions

The Rover and GUI were tested under the following circumstances:

- Conducted on a computer with standards at least as high as those listed in section 2.3.1 and with the Lego Mindstorms EV3 robot
- Conducted with only a single user, from the group UG-17
- Conducted with only one open network connection
- Conducted with only one instance of the software open
- Conducted after the final version of the code was completed
- Conducted where possible with the JUnit plug in

2.1 Test Personnel

Test Director
Pham, Hoang Long

Test Engineers
Kozirev, Matthew
Khuc, Ngoc Chau
Schuh, Benjamin Thomas
Zaky, Katon Akhmad
Gray, Simon James
Yang, Yong

2.2 Item Under Test

The two items that will be tested are the software for the Rover, as well as the GUI that has control over the Rover in certain circumstances, and which displays relevant information to the Rover's operation.

2.3 Test Equipment

2.3.1 GUI Testing

The GUI testing was conducted on a computer this the minimum requirements of:
Operating System: Windows 7/10 32/64 bit
RAM: 4GB
CPU: AMD A6-6200 APU with Radeno^{TM} HD Graphics
CPU Speed: 2.0GHz
Graphic Card: AMD Radeno HD8400
Storage: 1MB
Java Version: Java SE 1.7.0 80
Eclipse Version: Eclipse Mars 4.2

2.3.2 Rover Testing

The software related to Rover movement was all conducted on the LEGO Mindstorms education EV3, the technologies abilities were as follows:
Display: 178x128 pixel Monochrome LCD
Main Processor: TI Sitara AM1808 (ARM926EJ-S core) @ 300 MHz
Main Memory: 64 MB RAM, 16 MB Flash, microSDHC Slot
USB Host Port: Yes
WiFi: Optional dongle via USB port
Bluetooth: Yes

2.3.3 Code Testing

All software testing where possible has been conducted with JUnit, which is a Java plugin. When a test was required to be conducted in a fashion that is unsuited to JUnit then it will be done at the testers control with physical testing. In the case of the GUI this will be done by manually checking that it fulfils all requirements asked of it. The sensor testing for the Rover was also done in a similar manner, with a user physically testing the sensors and comparing the received data to what was expected.

2.4 Example JUnit Test for XML Parsing

2.4.1 Test 4.001: Access All Elements of an XML Document

If the child element of the root element may be accessed with the XMLParser class, then this implies that it is possible to access all sub-elements of the document. So, the first test checked if the root element and one of the sub-elements was accessible.

2.4.2 Test 4.002: Access the Attribute of a Tag

The next test checked if the value of an attribute can be accessed. To test if any attribute value can be accessed, the test checked if the attribute of the sub-element of the root element can be accessed.

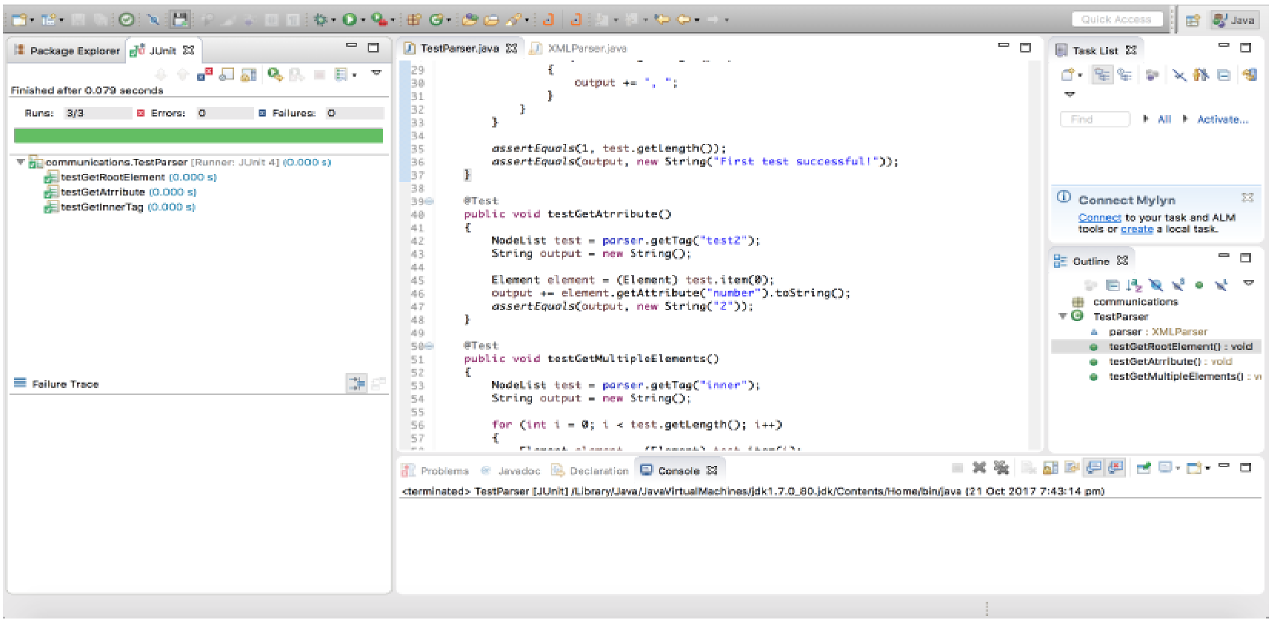
2.4.3 Test 4.003: Access All Instances of a Certain Tag

The final test checked if all elements of a tag name can be accessed, this would occur in the case that the document featured multiple instances of a tag.

2.4.4 JUnit Example Test for XML Parsing

Table 2: JUnit Test Results

Test Number	**Result of Test**
4.001	Pass
4.002	Pass
4.003	Pass



```
Finished after 0.079 seconds

Runs:  3/3    Errors:  0    Failures:  0

communications.TestParser [Runner: JUnit 4] (0.000 s)
    testGetRootElement (0.000 s)
    testGetAtrribute (0.000 s)
    testGetInnerTag (0.000 s)
```

```java
                {
                    output += ", ";
                }
            }
        }

        assertEquals(1, test.getLength());
        assertEquals(output, new String("First test successful!"));
    }

    @Test
    public void testGetAtrribute()
    {
        NodeList test = parser.getTag("test2");
        String output = new String();

        Element element = (Element) test.item(0);
        output += element.getAttribute("number").toString();
        assertEquals(output, new String("2"));
    }

    @Test
    public void testGetMultipleElements()
    {
        NodeList test = parser.getTag("inner");
        String output = new String();

        for (int i = 0; i < test.getLength(); i++)
        {
```

```
<terminated> TestParser [JUnit] /Library/Java/JavaVirtualMachines/jdk1.7.0_80.jdk/Contents/Home/bin/java (21 Oct 2017 7:43:14 pm)
```

Figure 1: JUnit Example Test

Table 3: Summary of Requirements Verified

Requirement ID	Priority	Verify Method	Test Procedure ID	Test Results	Requirement Result	Status
R0007	High	T	4.001 4.002 4.003	Pass Pass Pass	Pass	Verified

2.5 General Conditions

Each Test Procedure details the individual test steps and expected results with individual tests being grouped into logical Test Procedures.

All testing has also included regression testing, ensuring that anything previously developed and tested will still work with the newest version of the software.

3 Test Procedures/Test Activity

3.1 Test Procedure Outline

There are four different types of test scenarios that will be performed to asses the capabilities of the produced code, which are:

1. **Defined Scenarios:**
   A situation that the item in question will be expected to encounter in a normal working environment
2. **Undefined Scenarios:**
   When a situation is encountered that wouldn't be expected through the duration of a normal working environment
3. **Sensor Tests:**
   Tests that were taken to ensure that the hardware used gave accurate values for sensors along with motor movement accuracy with rotation and distance
4. **Code Tests:**
   Tests that were performed on the software/code to verify that it would operate as expected

3.2 Baseline

Testing the physical items to ensure that the hardware will emulate the expected output, for example if the motors are sent a command to move forward 10cm they will move the expected distance.

Table 4: Hardware Testing

Sensor Type	**Test to Verify**
Ultrasonic	3.001
Light	3.002
Gyroscope	3.003
Bump	3.004
Colour	3.005
Hardware Test	**Test to Verify**
Motor Accuracy	3.006
Rotation Accuracy	3.007

3.3 Tests Conducted

The following table lists all tests performed to validate requirements for hardware or code.
The columns listed refers to:
Column 1: Identification of the test performed
Column 2: Short summary of what the test entailed
Column 3: The requirement the test aimed to prove (If applicable)
Column 4: The Result of the Test

Table 5: Summary of Conducted Tests

Test Identification	**Test Summary**	**Related to Requirement**	**Result**
1.001	Detect Map Features (Object/Target) as the Rover is moving	R0001 R0002 R0003	Fail
1.002	Detect the radiation area around the target	R0001 R0002	Pass
1.003	Detect map features (Lines/Coloured Areas) as the Rover is moving	R0002 R0003	Pass
1.004	Reflect detected features and objects in GUI map in real time	R0002 R0028	Pass
1.005	Detect and avoid areas of danger (craters, user NGZ, etc) in the map in automatic mode	R0003	Pass
1.006	Detect and avoid areas of danger (craters, user NGZ, etc) in the map in manual mode	R0003	Fail
1.007	Rover can move to a designated location (coordinates) on the map	R0005	Pass
1.008	GUI can determine goal coordinates from user click on map	R0010	Pass
1.009	Rover and GUI can establish and utilise 2-way communication	R0038	Pass
1.010	Route communications data to the appropriate modules, (Rover mode, Map features, Rover position, GUI button presses)	R0038	Pass
1.011	GUI direction input from user	R0011	Pass
1.012	Rover responds to arrow pressed	R0011	Pass
1.013	GUI mode input from user	R0012	Pass
1.014	Check Rover behaves according to the user selected mode	R0012	Pass
1.015	Manually/Autonomously move the rover around the survey area and record features of interest	R0018	Pass
1.016	Able to display all objects on the map, apart from ultrasonic sensor	R0006	Partial Pass
1.017	GUI is able to import the map data from an external file	R0007	Pass

Table 6: Summary of Conducted Tests

Test Identification	Test Summary	Related to Requirement	Result
1.018	GUI is able to export the map data to an external file	R0007	Pass
1.019	GUI is able to change the color scheme of map elements	R0008	Pass
1.020	Mark NGZ's on the GUI of a chosen dimension	R0009	Pass
1.021	Designate a targeted location on the GUI map for the Rover to move to	R0010	Pass
1.022	Button on GUI is able to make Rover go forward	R0011 R0029	Pass
1.023	Button on GUI is able to make Rover turn right	R0011 R0029	Pass
1.024	Button on GUI is able to make Rover go backward	R0011 R0029	Pass
1.025	Button on GUI is able to make Rover turn left	R0011 R0029	Pass
1.026	Option to change between automatic mode and manual mode on GUI	R0012 R0029	Pass
1.027	Option on GUI to return to deployment location	R0013	Fail
1.028	Emergency stop button on GUI stops all Rover movement	R0014	Pass
1.029	GUI status bar displaying Rovers current action	R0015	Pass
1.030	Legend is located on the **right** side of the GUI	R0016	Pass
1.031	GUI has a help option that will bring up the User Manual	R0019	Pass
1.032	GUI is able to remove NGZ's that have been created by the user	R0021	Pass
1.033	There is an option to click on the status bar to show all previous actions of the Rover with the time stamp they were done	R0023	Pass
1.034	The GUI will be able to display the immediate surrounding of the Rover on the map	R0027	Pass
1.035	The results from the Rovers position and heading are displayed on the Rover map	R0028	Pass
1.036	GUI is able to send commands to the Rover which the Rover will then act upon	R0029	Pass
1.037	User is able to zoom in on the GUI to make the map bigger	R0030	Pass
1.038	User is able to zoom out on the GUI to make the map smaller	R0030	Pass
1.039	The user is able to move the maps visual area around from the initial location	R0031	Pass
1.040	The GUI is able to adjust the size of the interface based off the screen resolution	R0035	Partial Pass
1.041	The GUI is able to send the Rover map data and current operation mode	R0038	Fail
1.042	The Rover is able to send data to the GUI	R0039	Pass

Table 7: Summary of Conducted Tests

Test Identification	**Test Summary**	**Related to Requirement**	**Result**
3.001	The ultrasonic sensor gives expected readings	N/A	Pass
3.002	The light sensor gives expected readings	N/A	Pass
3.003	The gyro sensor gives expected readings	N/A	Pass
3.004	The bump sensor gives expected readings	N/A	Pass
3.005	The colour sensor gives the expected reading	N/A	Pass
3.006	The motor's will move the Rover the expected amount	N/A	Pass
3.007	The Rover will accurately rotate the desired amount	N/A	Pass

4 Test Results

4.1 Requirements Verified

The following table details the results of tests to verify requirements from the SRS, and contains the following information:

Column 1 - Shows the ID if the requirement as listed in the SRS
Column 2 - Requirement priority level
Column 3 - The verification: method, Demonstration, Analysis, Test, Inspection

- Demonstration (D): Manipulating the product or system as its intended to operate to verify the expected results
- Analysis (A): Verification using models, calculations and testing equipment. This allows the a user to make a predictive statements about the expected performance, often used for predicting software/hardware limitations
- Test (T): Using a controlled and predefined set of inputs, data or stimuli to ensure that the product or system will produce a very specific and predefined output as specified by the requirements
- Inspection (I): Using one or more of your five senses to examine the product or system

Column 4 - The test procedure identifier uniquely identifies the individual procedure that must successfully be executed to pass the requirement
Column 5 - Summarise the test outcome of the particular requirement

1. Pass: The requirement has been proved through the corresponding tests
2. Fail: The requirement is unable to be proven through the corresponding tests
3. Partial Pass: The requirement is able to be partially verified, however its unable to completely fulfil the entire requirement
4. N/A: The requirement hasn't been tested

Column 6 - Status of Requirement

1. Verified: Tests have been conducted on requirement, it has either passed or failed
2. Unverified: Tests haven't been performed to check capability to prove requirement

Column 7 - Any comments about the concerning requirement or tests performed to verify the requirement

Table 8: Summary of Requirements Verified

Requirement ID	**Priority**	**Verify Method**	**Test Procedure ID**	**Test Results**	**Requirement Result**	**Status**	**Comment**
R0001	High	D	1.001 1.002	Fail Pass	Fail	Verified	
R0002	High	D	1.003 1.001 1.004	Pass Pass Pass	Partial Pass	Verified	Automatic survey not completed, Manual works as expected
R0003	High	D	1.005 1.006 1.001	Pass Fail Fail	Fail	Verified	
R0004	High	D	1.007	Fail	Fail	Verified	
R0005	High	D	1.007	Pass	Pass	Verified	
R0006	High	I	1.016	Partial Pass	Partial Pass	Verified	
R0007	High	D	1.017 1.018 R0041	Pass Pass fail	Partial Pass	Verified	
R0008	High	I	1.019	Pass	Pass	Verified	
R0009	High	I	1.020	Pass	Pass	Verified	
R0010	High	D	1.008 1.007 1.021	Pass Pass Pass	Pass	Verified	
R0011	High	D	1.011 1.012 1.022 1.023 1.024 1.025	Pass Pass Pass Pass Pass Pass	Pass	Verified	
R0012	High	D	1.013 1.014 1.026	Pass Pass Pass	Pass	Verified	
R0013	High	I	1.027	Fail	Fail	Verified	
R0014	High	I	1.028	Pass	Pass	Verified	
R0015	High	D	1.029	Pass	Pass	Verified	
R0016	High	I	1.030	Pass	Pass	Verified	

Table 9: Summary of Requirements Verified

Requirement ID	**Priority**	**Verify Method**	**Test Procedure ID**	**Test Results**	**Requirement Result**	**Status**	**Comment**
R0017	High	D		N/A	N/A	Unverified	
R0018	High	D	1.015	Fail	Fail	Verified	Requires auto movement
R0019	Medium	I	1.031	Pass	Pass	Verified	
R0020	Medium	D		N/A	N/A	Unverified	
R0021	Medium	I	1.032	Pass	Pass	Verified	
R0022	Low	I		N/A	N/A	Unverified	
R0023	Medium	I	1.033	Pass	Pass	Verified	
R0024	Low	I		N/A	N/A	Unverified	No Implementation
R0025	Low	I		N/A	N/A	Unverified	No Implementation
R0026	Low	I		N/A	N/A	Unverified	No Implementation
R0027	Medium	I	1.037	Pass	Pass	Verified	
R0028	High	I	1.035	Pass	Pass	Verified	
R0029	High	D	1.036	Pass	Partial Pass	Verified	
R0030	High	I	1.037 1.038	Pass Pass	Pass	Verified	
R0031	Medium	I	1.039	Pass	Pass	Verified	
R0032	Medium	I		N/A	N/A	Unverified	No Implementation
R0033	Low	I		N/A	N/A	Unverified	No Implementation
R0034	Low	I		N/A	N/A	Unverified	No Implementation
R0035	Medium	I	1.040	Partial Pass	Partial Pass	Verified	Not all elements are able to correctly adjust
R0036	Low	D		N/A	N/A	Unverified	No Implementation
R0037	Low	D		N/A	N/A	Unverified	No Implementation
R0038	High	D	1.009 1.010 1.041	Pass Pass Partial Pass	Fail	Verified	
R0039	High	I	1.042	Pass	Pass	Verified	

5 Conclusion and Recommendations

This report outlines the activities conducted to demonstrate that the software has been thoroughly tested.

- Verified: Requirements that have been verified to either pass or fail
- Unverified: Untested or Requirements that haven't been addressed
- Passed: Requirements that have been verified that they have been completed
- Partial Pass: Able to partially fulfil a requirement
- Failed: Requirements that haven't been able to pass required tests
- Total Requirement Count: Total amount of requirements to be tested

Table 10: Summary of Requirements Verified

Status	**Requirements**
Verified	28
Unverified	11
Passed	18
Partial Pass	4
Failed	6
Total Requirement Count	**39**