Software Requirements Specification

for

MARS Routing System

Version 1.1 | 15 November 2017

Prepared by MARS Team:

Ross Blassingame
Chandler Samuels
Shawn Polson
Robert Ballard
Dean Moser
Josh Jenkins

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Final Draft	Team		

Review & Approval

Requirements Document Approval History

Approving Party	Version Approved	Signature	Date
Marcel Llopis			

Requirements Document Review History

Reviewer	Version Reviewed	Signature	Date
Chandler Samuels			

Contents

Introduction

1.1 Purpose

The purpose of this document is to define and describe the requirements of the product and to spell out the system's functionality and its constraints. The goal of this document is to ensure that both parties agree on the requirements that the software must meet by the intended deadline of April 16, 2018.

1.2 Intended Audience and Reading Suggestions

The document is intended for the employees at NASA's Jet Propulsion Laboratory, including the project's overseer, Mr. Marcel Llopis, and the designers of the software system at the University of Colorado, Boulder - the MARS Software Team.

1.3 Document Conventions

This document specifies the requirements for the project. There are seven (7) numbered sections. Each section consists of multiple chapters, also numbered. Some chapters have additional sub-chapters.

Overall Description

2.1 Product Overview

The Martian Autonomous Routing System (or MARS) is a command-line utility that, given a GeoTIFF format elevation map, a start coordinate corresponding to the elevation map, and a goal coordinate also corresponding to the elevation map, will produce a lists of coordinates describing the optimal path and it's respective distance for the Rover to take to reach the goal coordinate, within the bounds of the Rover's physical capabilities that are given by the user through the command line.

2.2 Product Functions

The major functions of the MARS system include:

- Ability to read and parse a GeoTIFF file
- Acquire Rover specifications, path to elevation map, and algorithm type from user
- Discover optimal paths from either an unlimited scope algorithm or a limited scope algorithm
- Output coordinates along with their respective slopes and distance from start coordinate to goal coordinate

2.3 User Characteristics

The users of the system are intended to be employees at NASA's Jet Propulsion Laboratory, as they will use this software in the design of future Rovers. For this system, the
user is required to have a basic understanding of a Rover's physical capabilities. These physical capabilities include maximum and minimum slopes that a traditional rover can traverse and maximum field of view that a rover can identify.

2.3 User Objectives

The user's objective for this product is to provide optimal routes that a Rover may take given various Rover capabilities. The product should be stable and responsive to incorrect input at the command line.

2.3 General Constraints

**See Caveats section in our Final Research Document located on our GitHub at https://github.com/RossBlassingame/JPL-CUSeniorProjects/blob/master/docsMARS%20Research%20Paper.pdf

System Features

3.1 Path Finding Algorithms

3.1.1 Description and Priority

The algorithms' purpose is to attempt to find the optimal path between a pair of start and goal coordinates. There are two algorithms that the user can choose: an unlimited scope algorithm and a limited scope algorithm. The unlimited scope algorithm assumes that the Rover has the elevation map stored in memory; therefore, it has an unlimited scope. The limited scope algorithm assumes the Rover does not have the elevation map stored in memory and has to explore the terrain through it's given field of view; therefore, it has a limited scope.

3.1.2 Stimulus/Response Sequence

Stimulus: The user chooses one of the two algorithms to run.

Response: The system responds to the user's request by running the desired algorithm.

3.1.3 Functional Requirements

REQ-1: **The algorithm shall parse and read a GeoTIFF elevation map.**

1. User shall provide a valid elevation map in GeoTIFF format.
2. Highly critical
3. Error handling is necessary if requirement cannot be met
4. We foresee technical risks if requirement cannot be met.

REQ-2: **The algorithm shall discover all paths that are within the bounds of both the given GeoTIFF and specified rover capabilities.**

1. Paths should be discovered within the bounds provided by the user and the provided elevation map.
2. Moderately critical
3. Error handling is necessary if requirement cannot be met.
4. We foresee technical risks if requirement cannot be met

REQ-3: **The algorithm shall merge all critical components of system.**

1. Necessary to ensure that all system components are used to provide an optimal path.
2. Highly critical
3. Error handling is necessary if requirement cannot be met.
4. We foresee technical risks if requirement cannot be met.

3.2 Command Line Interface

3.2.1 Description and Priority

The purpose of the command line interface is to prompt the user for information required to run the algorithm successfully. The command line interface is responsible for being the point of contact between the algorithm and the user. The command line interface is of high priority because it is responsible for initiating the algorithm.

3.2.2 Stimulus/Response Sequence

Stimulus: The command line interface will prompt the user for various inputs including Rover specifications, path to an elevation map, and type of algorithm.

Response: Upon valid entry, the command-line interface shall call the algorithm and start the program.

3.2.3 Functional Requirements

REQ-1: **The user shall be able to initiate the algorithm through the command line.**

1. The Command Line Interface shall call the algorithm upon initiation of the start sequence.
2. Highly critical
3. Error handling is necessary if requirement cannot be met.
4. We foresee technical risks if requirement cannot be met.

REQ-2: **The user shall be able to choose an algorithm of his/her choice through the command line.**

1. The command line shall present valid algorithm types for the user to choose from.
2. Highly critical
3. Error handling is necessary if requirement cannot be met.
4. We foresee technical risks if requirement cannot be met.

REQ-3: **The user shall be able to enter rover specifications through the command line.**

1. More details are specified in section 3.4 Rover Specifications. The command line shall prompt the user to enter rover specifications.
2. Highly critical
3. Error handling is necessary if requirement cannot be met.
4. We foresee technical risks if requirement cannot be met.

REQ-4: **The user shall be able to enter a path to an elevation map through the command line.**

1. The command line shall prompt the user to enter a path to a valid elevation map.
2. Highly critical
3. Error handling is necessary if requirement cannot be met.
4. We foresee technical risks if requirement cannot be met.

REQ-5: **The user shall be able to choose a previously stored Rover through the command line.**

1. The command line shall provide a list of previously stored Rovers and allow a user to choose from the list
2. Not critical
3. Basic error handling is necessary if requirement cannot be met.
4. We do not foresee any technical risks if requirement cannot be met.

3.3 GeoTIFF Parsing

3.3.1 Description and Priority

The purpose of the GeoTIFF parsing tool is to accept a valid user-specified GeoTIFF file, and be able to readily and frequently generate elevation data for arbitrary coordinates from this GeoTIFF. This feature is High Priority, as it is required for both creating and testing the Algorithms.

3.3.2 Stimulus/Response Sequence

Stimulus: The user will provide a path to an elevation map through the command line.

Response: The system will check to make sure the path is valid and the file found at the path is an elevation map in valid GeoTIFF file format.

3.3.3 Functional Requirements

REQ-1: **The system shall alert the user if an invalid path is provided.**

1. The system shall check that the path given is valid.
2. Highly critical
3. This requirement is an extension of REQ-4 in section 3.2 Command Line interface. Error handling is necessary if requirement cannot be met.
4. We foresee technical risks if requirement cannot be met

REQ-2: **The system shall alert the user if an invalid elevation map is provided.**

1. The system shall check that the elevation map provided is in GeoTIFF format.
2. Highly critical
3. This requirement is an extension of REQ-4 in section 3.2 Command Line interface. Error handling is necessary if requirement cannot be met.
4. We foresee technical risks if requirement cannot be met

3.4 Rover Specifications

3.4.1 Description and Priority

The algorithm is designed to be dynamic so that the algorithm will adjust its route based on the specifications of the Rover. Therefore, the rover specifications functionality is a crucial feature for the customizability of the product. Since the path finding algorithm relies on rover specifications as input, this feature is High Priority.

3.4.2 Stimulus/Response Sequence

Stimulus: The user provides the rover specification through the command line.

Response: The system will respond by loading the specifications into the algorithm. The algorithm will then use these specifications as parameters when finding an optimal path from the starting coordinate to the goal coordinate.

3.4.3 Functional Requirements

REQ-1: **The system shall prompt the user for rover specifications through the command line.**

1. The user will be required to enter: field of view(in meters), maximum traversal slope, minimum traversal slope, start position(latitude/ longitude), and goal position(latitude/longitude)
2. Highly critical
3. This requirement is an extension of REQ-3 in section 3.2 Command Line interface. Error handling will be necessary if user enters in correct data type.
4. If requirement cannot be met, the system will need to provide the user with default specifications for the rover.

REQ-2: **The system shall store rover specifications in a configuration file.**

1. The system will be able to record rover specifications so that the user can easily access previous rover specifications.
2. Not critical
3. This requirement is an extension of REQ-5 in section 3.2 Command Line interface. Error handling will be necessary if user enters in correct data type.
4. We do not foresee any technical risks if requirement cannot be met.

REQ-2: **The system shall allow the user to access and modify prior Rover specifications through the command line.**

1. The system will enable the user to access previously created Rover specifications and modify those specifications upon launching the algorithm.
2. Not critical
3. This requirement is an extension of REQ-5 in section 3.2 Command Line interface. Error handling will be necessary if user enters in correct data type.
4. We do not foresee any technical risks if requirement cannot be met.

3.5 Output

3.5.1 Description and Priority

The system shall be able to produce a valid CSV file and terminal output containing a sequence of coordinates that corresponds to the calculated path taken by the Rover. The output should also include rover specifications, type of algorithm, calculated slopes of each coordinate and distance traveled from start coordinate to goal coordinate. This feature is of high priority because it is required for both creating and testing the algorithm.

3.5.2 Stimulus/Response Sequence

Stimulus: The user starts the program through the command line.

Response: The system will respond with a terminal output and a CSV file when the algorithm completes.

3.5.3 Functional Requirements

REQ-1: **The system shall provide the user with a CSV document upon algorithm completion.**

1. The system will create a CSV document to store information given from the algorithm.
2. Moderately critical
3. Error handling will be necessary if document cannot be created.
4. If requirement cannot be met, the system will revert to standard terminal output.

REQ-2: **The system shall provide the user with terminal output upon algorithm completion.**

1. The system will print information given from the algorithm to the terminal.
2. Highly critical
3. Error handling will be necessary if requirement cannot be met.
4. We foresee there to be technical risks if this requirement cannot be met.

REQ-3: **The system shall write the optimal path found to both methods of output.**

1. The system will write coordinates of the path to both the CSV file and terminal screen.
2. Highly critical
3. This requirement is an extension of REQ-1 and REQ-2. Error handling will be necessary if requirement cannot be met.
4. We foresee there to be technical risks if this requirement cannot be met.

REQ-4: **The system shall write rover specifications to both methods of output.**

1. The system will write rover specifications used to discover the optimal path to both the CSV file and terminal screen.
2. Highly critical
3. This requirement is an extension of REQ-1 and REQ-2. Error handling will be necessary if requirement cannot be met.
4. We foresee there to be technical risks if this requirement cannot be met.

REQ-5: **The system shall write distance of path found to both methods of output.**

1. The system will write the calculated distance of the path, if found, to both the CSV file and terminal screen.
2. Highly critical
3. This requirement is an extension of REQ-1 and REQ-2. Error handling will be necessary if requirement cannot be met.
4. We foresee there to be technical risks if this requirement cannot be met.

REQ-6: **The system shall write each variance in slope of the path found to both methods of output.**

1. The system will write each variance in slope of the path to both the CSV file and terminal screen.
2. Highly critical
3. This requirement is an extension of REQ-1 and REQ-2. Error handling will be necessary if requirement cannot be met.
4. We foresee there to be technical risks if this requirement cannot be met.

Interface Requirements

4.1 Interface Requirements

4.1.1 GUI

The graphical user interface allows for a user to click on the mars_map.tiff to create a starting and ending coordinate.

4.1.2 CLI

The user is expected to run the program through the command line. The product implements a command line interface of the type commonly found on Unix based systems, MS-DOS, and Apple DOS.

4.1.3 API

There are no APIs for this product

4.2 Hardware Interfaces

The program accesses the secondary storage (e.g., hard disk) of the computing system on which the program runs. Access to the secondary storage and other hardware is managed by the operating system.

4.3 Software Interfaces

The MARS system may be used to import and export data. This functionality is built into

Other non-functional requirements

5.1 Performance

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

5.2 Security

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

5.3 Safety

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product's design or use. Define any safety certifications that must be satisfied.>

Software Quality Attributes

6.3 Reliability

6.4 Maintainability

6.5 Portability

6.6 Extensibility

6.7 Reusability

Appendix

7.1 Appendix A: