Software Validation Specification
for
Line FollowerChallenge Robot DD48879

Prepared by:

Jiří Klepl
Lukáš Rozsypal

Department of Distributed and Dependable Systems
Faculty of Mathematics and Physics, Charles University

Prague, 18. 4. 2021

Contents

Chapter 1

Introduction

This Software Requirements Specification (SRS) document describes the requirements of a Line Follower Challenge Robot. The SRS begins with an introductory section describing its overall purpose, scope, and defining terms and acronyms that will be utilized therein. The following section will describe the software product in detail, including functions, constraints, and user characteristics. Then, a detailed list of requirements will be provided, after which those requirements will be modeled using a domain model, data dictionaries, use case diagrams, sequence diagrams and system state diagrams. In the final three sections, a product prototype is described, references for this document are listed, and a point of contact for the project is provided.

1.1 Purpose

The purpose of this document is to specify the means of achievement of victory in the Line Follower Challenge.

1.2 Scope

The software described in this SRS document is a control system for the Line Follower Challenge Robot DD48879 (CSR/DD48879). It is designed as an embedded system to be used in the Line Follower Challenge. The major goals of the system are to achieve victory in the Line Follower Challenge. The CSR/DD48879 system will attempt to follow the line at the maximum speed maintainable by the device. The system can use a combination of sensors and camera to detect the line and is able to adjust its own speed via Kobuki motors.

1.3 References

1. Robotic Day, "Line Follower: The robot follows black line on the track as fast as possible", 2016
2. Kobuki Team, "Kobuki User Guide", 2016
3. Dodger67 et al, Physical disability, 2021, available online:
   https://en.wikipedia.org/wiki/Physical_disability

Chapter 2

Overall Description

2.1 Stakeholders and their goals and needs

1. **Investors**: Investors from the *Department of Distributed and Dependable Systems, Faculty of Mathematics and Physics, Charles University* want to win the competition as a part of their marketing strategy, their satisfaction is dependent solely on the achievement of victory
2. **Competition organizers**: They want to promote robotics among general public via creating highly competitive challenges. At the same time, one of their main concerns is safety of all competitors, spectators and organizers present during the competition. They will be satisfied by a highly performant and safe system.

2.2 Operating Environment

The operating environment is the place of the competition. After previous iterations of the challenge, it is expected to resemble a table with lines drawn on top of it. The robots participating in the challenge are expected to stay on the table without exiting the area close to the line. The robots are required to be completely autonomous during the challenge without receiving any information from outside sources.

2.3 Constraints

There are multiple constraints defined by the rules of the Line Follower Challenge (see the rules [1]). Here we will list the most important ones when making design decisions about the system.

- The objective of the challenge is to finish each run as fast as possible
- The robot shall be completely autonomous

2.4 Assumptions and references

- The challenge will not be excessively long or it will be with a supply of power for the robot and its system

Chapter 3

Software validation testing specification design

3.1 General

The specification of the testing design follows the Software Requirements Specification which should be delivered either with this document or in precedence of this document to each of the addressees of this document.

3.2 Test designs

- **Test design *STOP-BUTTON***

  General:

  This test design establishes procedures for the validation on the stop button functionality as specified in the Software Requirements Specification (as **SRS_001**).

  Features to be tested:

  This test design will test the following features:

  – Stop button presence
  – Stop button accessibility
  – Stop button response
  – Stop button response dependability
  – Stop button response time

  Approach refinements:

  – STOP-BUTTON-01 Visually verify the presence of the stop button, and verify it is located on top side of the robot.
  – STOP-BUTTON-02 Attempt to press the button on a powered-off robot. Verify it is easily accessible even to a physically disabled person (as defined by [3]).
  – STOP-BUTTON-03 Try pressing the button on a powered-on robot under each of the following conditions:
    ∗ The robot is located on the start line but it has not yet received any start signal
    ∗ The robot is following the line
    ∗ The robot lost the line and it is spiraling

    * The robot is rounding an Object

  Verify the robot stops all movement within 100ms and it does not start until given a new start signal.

  - STOP-BUTTON-04 Try pressing the button on a powered-on robot under each of the following potentially unanticipated conditions:
    * The robot is placed on a chessboard
    * The robot is placed on a glass table
    * The robot is placed on a monochromatic table with no lines
    * The robot fell off a table placed at 1000 mm to 1100 mm height

    Verify that the robot's response does not differ from the response in the test STOP-BUTTON-03
  - STOP-BUTTON-05 Repeat each of the tests STOP-BUTTON-03 and STOP-BUTTON-04 7 times.

- **Test design *CALM***

  General:

  This test design establishes procedures for the validation on the non-annoyance of the robot as specified in the Software Requirements Specification (as **SRS_420**).

  Features to be tested:

  This test design will test the following features:

  - Objective visual annoyance
  - Objective aural annoyance
  - Subjective visual annoyance
  - Subjective aural annoyance
  - Subjective olfactory annoyance

  Approach refinements:

  - CALM-01 Place the robot on a table from the previous iteration of the challenge and make it move randomly on top of it at its maximum speed. Take a recording from fixed position centered at the table within 2 meters and at least one meter higher than the surface of the table. The camera shall record in standard 720px/480px@60. Take another recording under the same conditions except the robot shall be covered under an opaque blanket. Verify the former at most 125% of the latter.
  - CALM-02 Measure the audio volume of the robot from the distance of at least 1m and at most 2m. Verify that the audio volume does not exceed 60dB.
  - CALM-03 Ask 10 independent observers representative of the organizers and the audience of the challenge whether they consider the visual appearance of the robot annoying. Verify that at most one of the participants is annoyed by the robot's visual appearance.
  - CALM-04 Ask 10 independent observers representative of the organizers and the audience of the challenge whether they consider the sound of the robot annoying. Verify that at most one of the participants is annoyed by the robot's sound.
  - CALM-05 Ask 10 independent observers representative of the organizers and the audience of the challenge whether they consider the smell of the robot annoying. Verify that at most one of the participants is annoyed by the robot's smell.

- **Test design *STATE-TRANSPARENCY***

  General:

  This test design establishes procedures for the validation on the indication of the state and its unambiguity as specified in the Software Requirements Specification (as **SRS_421** and **SRS_422**).

  Features to be tested:

  This test design will test the following features:

  - Reporting when the robot is ready to start
  - Reporting when the robot is following the line
  - Reporting when the robot has lost the line
  - Reporting when the robot is rounding an Object
  - Reporting when the robot has reached the finish line
  - Reporting when the robot is manually stopped
  - Reporting when the robot has detected an internal error
  - Reporting when the robot has fallen off the table
  - Reporting when the robot has detected there is no valid path to the finish line

  Approach refinements:

  - STATE-TRANSPARENCY-01 Verify the robot correctly reports when it is ready to start.
  - STATE-TRANSPARENCY-02 Verify the robot correctly reports when it is following the line.
  - STATE-TRANSPARENCY-03 Verify the robot correctly reports when it has lost the line.
  - STATE-TRANSPARENCY-04 Verify the robot correctly reports when it is rounding an Object.
  - STATE-TRANSPARENCY-05 Verify the robot correctly reports when it has reached the finish line.
  - STATE-TRANSPARENCY-06 Verify the robot correctly reports when it is manually stopped.
  - STATE-TRANSPARENCY-07 Verify the robot correctly reports when it has detected an internal error.
  - STATE-TRANSPARENCY-08 Verify the robot correctly reports when it has fallen off the table.
  - STATE-TRANSPARENCY-09 Verify the robot correctly reports when it has detected there is no valid path to the finish line.

- **Test design *CLIFF-SENSOR-REACTION***

  General:

  This test design establishes procedures for the validation on the reliability of the cliff sensors and the robot's reaction to their triggerment as specified in the Software Requirements Specification (as **SRS_003**).

  Features to be tested:

  This test design will test the following features:

  - Reliability of the cliff sensors
  - Robot's reaction to the triggerment of the cliff sensors

  Approach refinements:

- CLIFF-SENSOR-REACTION-01 Test the robot in each of the following conditions:

  - The robot is following the line which leads to the edge of the table
  - The robot lost the line and it is spiraling 10 cm from the edge of the table
  - The robot is rounding an Object centered on the edge of the table

  Visually verify that the robot has not fallen of the table.

- CLIFF-SENSOR-REACTION-02 Test the robot in each of the following potentially unanticipated conditions:

  - The robot is placed on a chessboard table
  - The robot is placed on a glass table
  - The robot is placed on a monochromatic table with no lines

  Visually verify that the robot has not fallen of the table.

3.3 Non Functional Requirements

1