Autonomous Pool Playing Robot

Requirements Specification

Eric Le Fort
leforte@mcmaster.ca
1308609

February 28, 2017

Contents

List of Tables

Date	Revision #	Comments	Authors
27/02/2017	0	- Initial document creation	Eric Le Fort

Table 1: Revision History

1 Introduction

This document will provide a specification of a test plan for an automated pool playing robot and report on the results of that plan.

1.1 Overview

This document breaks down the required testing for each domain of the system. It begins with the hardware aspect, then moves to the electrical side and then finishes with software. Each section will provide a traceability matrix to map the requirements to tests that check their completion and then go into further detail to describe each test case. Lastly, a summary of the results of testing will be provided to conclude the document.

1.2 Purpose

The aim of this document is to illuminate any design flaws, software bugs, or other issues in the system. Once these issues are discovered, the engineering team will be able to work on eliminating them or minimizing their frequency and consequences.

1.3 Naming Conventions & Definitions

This section outlines the various definitions, acronyms and abbreviations that will be used throughout this document in order to familiarize the reader prior to reading.

1.3.1 Definitions

Table 2 lists the definitions used in this document. The definitions given below are specific to this document and may not be identical to definitions of these terms in common use. The purpose of this section is to assist the user in understanding the requirements for the system.

Table 2: Definitions

Term	**Meaning**
X-axis	Distance along the length of the pool table
Y-axis	Distance across the width of the pool table
Z-axis	Height above the pool table
End-effector	The end of the arm that will strike the cue ball
θ	Rotational angle of end-effector
Cue	End-effector
Personal Computer	A laptop that will be used to run the more involved computational tasks such as visual recognition and the shot selection algorithm
Camera	Some form of image capture device (e.g. a digital camera, smartphone with a camera, etc.)
Table State	The current positions of all the balls on the table
Entity	Classes that have a state, behaviour and identity (e.g. Book, Car, Person, etc.)
Boundary	Classes that interact with users or external systems
Double	Double-precision floating point numbers

1.3.2 Acronyms & Abbreviations

Table 3 lists the acronyms and abbreviations used in this document.

Table 3: Acronyms and Abbreviations

Acronym/Abbreviation	**Meaning**
VR	Visual Recognition
PC	Personal Computer
μC	Micro-Controller
CRC	Class Responsibility Collaboration
TBT	To Be Tested

Test ID: n	Module: Ball	Status: TBT
Ball Constructor Good Inputs		
Pass/Fail Conditions: This test will be successful if all the fields inside of Ball are correctly initialized.		
Pre-Conditions: None Input: 1, 0.7, 0		
Expected Results: A new ball with x-coordinate 1, y-coordinate 0.7, and the value 0.	Actual Results:	
Post-Conditions: A new Ball object should be available.		

Table 4: Ball Constructor Good Inputs

2 Mechanical Components

3 Electrical System

4 Software System

The software system is comprised of four main components: a control system running on an Arduino microcontroller, an automated image capture application running on an Android smartphone, as well a visual recognition program and smart shot selection program running on a PC. On top of the typical suite of unit tests to verify correctness of methods, rigorous system testing will also be crucial to adequately test this system.

The following traceability matrix will demonstrate that the tests to be performed prove that specified requirements have been met.

4.1 Unit Tests

This section will provide a plethora of test cases which should prove correctness of the program. Each individual class will be tested in order to make finding specific test cases easier.

4.1.1 PC Controller Program

Test ID: n	Module: Ball	Status: TBT
Ball Constructor Large X		
Pass/Fail Conditions: This test will be successful if an IllegalArgumentException is thrown		
Pre-Conditions: None Input: 1.87658, 0.7, 0		
Expected Results: An IllegalArgumentException stating that the x-coordinate is out of range.	Actual Results:	
Post-Conditions: There should not have been a Ball created.		

Table 5: Ball Constructor Large X

Test ID: n	Module: Ball	Status: TBT
Ball Constructor Large Y		
Pass/Fail Conditions: This test will be successful if an IllegalArgumentException is thrown		
Pre-Conditions: None Input: 1, 0.94958, 0		
Expected Results: An IllegalArgumentException stating that the y-coordinate is out of range.	Actual Results:	
Post-Conditions: There should not have been a Ball created.		

Table 6: Ball Constructor Large Y

Test ID: n	Module: Ball	Status: TBT
Ball Constructor Small X		
Pass/Fail Conditions: This test will be successful if an IllegalArgumentException is thrown		
Pre-Conditions: None Input: -1.001, 0.7, 0		
Expected Results: An IllegalArgumentException stating that the x-coordinate is out of range.	Actual Results:	
Post-Conditions: There should not have been a Ball created.		

Table 7: Ball Constructor Small X

Test ID: n	Module: Ball	Status: TBT
Ball Constructor Small Y		
Pass/Fail Conditions: This test will be successful if an IllegalArgumentException is thrown		
Pre-Conditions: None Input: 1, -1.001, 0		
Expected Results: An IllegalArgumentException stating that the y-coordinate is out of range.	Actual Results:	
Post-Conditions: There should not have been a Ball created.		

Table 8: Ball Constructor Small Y

Test ID: n	Module: Ball	Status: TBT
Ball Constructor Small Value		
Pass/Fail Conditions: This test will be successful if an IllegalArgumentException is thrown		
Pre-Conditions: None Input: 1, 0.7, -1		
Expected Results: An IllegalArgumentException stating that the value is out of range.	Actual Results:	
Post-Conditions: There should not have been a Ball created.		

Table 9: Ball Constructor Small Value

Test ID: n	Module: Ball	Status: TBT
Ball Constructor Large Value		
Pass/Fail Conditions: This test will be successful if an IllegalArgumentException is thrown		
Pre-Conditions: None Input: 1, 0.7, 16		
Expected Results: An IllegalArgumentException stating that the value is out of range.	Actual Results:	
Post-Conditions: There should not have been a Ball created.		

Table 10: Ball Constructor Large Value