Autonomous Pool Playing Robot

Requirements Specification

Ernest Selman
selmae@mcmaster.ca
1201291

Eric Le Fort
leforte@mcmaster.ca
1308609Guy Meyer
meyerg@mcmaster.ca
1320231

March 2, 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 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

2 Traceability Matrix

The following traceability matrices will demonstrate that the tests to be performed prove that each of the specified requirements have been tested.

Functional Requirements Traceability Matrix

Req IDs	Reqs Tested	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	F17	F18
Test Cases	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Tested Implicitly																			
1.1...																			

Table 4: Functional Requirements Traceability Matrix

Non-Functional Requirements Traceability Matrix

Req IDs	Reqs Tested	LF1	UH1	UH2	UH3	P1	P2	P3	P4	OE1	MS1	MS2	S1	S2	S3	S4	S5	S6	S7	CP1	L1
Test Cases	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Tested Implicitly																					
1.1...																					

Table 5: Non-Functional Requirements Traceability Matrix

3 Mechanical Components

4 Electrical System

Test ID: 12.1	Sychronious Motion in X Rail	Status: TBT
Description: Verify that X-Rails can synchronously move to the same location at the same speed without getting stuck while loaded		
Pass/Fail Condition: If rail moves adequately and quickly as expected		
Pre-Conditions: None Input: Location along x-direction (i.e. 2000 steps)		
Expected Results: Smooth and consistent motion along axis until position is met. Followed by an immediate stop	Actual Results:	
Post-Conditions: Rails are stationary with no slip.		

Table 6: Sychronious Motion in X Rail

Test ID: 12.2	Motion in Y Rail	Status: TBT
Description:Verify that Y-Rail can move to a location without getting stuck while loaded		
Pass/Fail Condition: If rail moves adequately and quickly as expected		
Pre-Conditions: None Input: Location along y-direction		
Expected Results: Smooth and consistent motion along axis until position is met. Followed by an immediate stop	Actual Results:	
Post-Conditions: Rail is stationary with no slip.		

Table 7: Motion in Y Rail

Test ID: 12.3	End-Effector Orientation	Status: TBT
Description: Verify that EE-Base Motor can orient to a specific angle without getting stuck while loaded		
Pass/Fail Condition: If motor turns adequately and quickly as expected to correct angle		
Pre-Conditions: None Input: Angle of orientation with respect to the x-axis		
Expected Results: Smooth and consistent motion until orientation is met. Followed by an immediate stop	Actual Results:	
Post-Conditions: Motor is stationary.		

Table 8: End-Effector Orientation

Test ID: 13.2	Shooting Mechanism Orientation	Status: TBT
Description: EE is positioned correctly and waiting command to power piston		
Pass/Fail Condition: Piston is settled at correct oreintation, awaiting command to actuate piston		
Pre-Conditions: Motors orient piston to proper orentation Input: Position and orientation components sent to Arduino		
Expected Results: System moves to desired location and waits for piston signal	Actual Results:	
Post-Conditions: Piston can be safely actauted and strike cue ball		

Table 9: Shooting Mechanism Orientation

Test ID: 16.1	Perimeter Coverage	Status: TBT
Description: EE will be moved around the table to ensure that it is able to reach all locations and orientations		
Pass/Fail Condition: EE is capable of completing a full trip around the perimeter without stops		
Pre-Conditions: None Input: Motion command from Arduino		
Expected Results: EE will travel around perimeter of table. Inspection that its location is sufficient for shot-taking is required.	Actual Results:	
Post-Conditions: System awaits next command.		

Table 10: Perimeter Coverage

Test ID: 16.2	Ball Avoidance	Status: TBT
Description: As the EE is moving around the table it much avoid the balls to not interfere with gameplay		
Pass/Fail Condition: Able to move randomly around table without moving rolling or stationary balls		
Pre-Conditions: Ball in motion OR stationary Input: Random motion along table		
Expected Results: EE travels directly over balls and does not make contact	Actual Results:	
Post-Conditions: None		

Table 11: Ball Avoidance

Test ID: 14.1	User Input to Arduino	Status: TBT
Description: User applies input, then the Arduino indicates a message was received		
Pass/Fail Condition: Arduino output to console correct desired status		
Pre-Conditions: None Input: User pressed input button		
Expected Results: Related console output: make shot, cancel, or move, depending on the button pressed	Actual Results:	
Post-Conditions: None		

Table 12: User Input to Arduino

Test ID: n	Module: Ball **Ball Constructor Good Inputs**	Status: TBT
Pass/Fail Conditions: This test is passed 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 13: Ball Constructor Good Inputs

5   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.

5.1   Unit Tests

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

5.1.1   PC Controller Program

Ball Tests

Test ID: n	Module: Ball	Status: TBT
	Ball Constructor Large X	
Pass/Fail Conditions: This test is passed if an IllegalArgumentException is thrown		
Pre-Conditions: None Input: 1.87658, 0.7, 0		
Expected Results: An IllegalArgumentException has been thrown.	Actual Results:	
Post-Conditions: There should not have been a Ball created.		

Table 14: Ball Constructor Large X

Test ID: n	Module: Ball	Status: TBT
	Ball Constructor Large Y	
Pass/Fail Conditions: This test is passed if an IllegalArgumentException is thrown		
Pre-Conditions: None Input: 1, 0.94958, 0		
Expected Results: An IllegalArgumentException has been thrown.	Actual Results:	
Post-Conditions: There should not have been a Ball created.		

Table 15: Ball Constructor Large Y

Test ID: n	Module: Ball	Status: TBT
	Ball Constructor Small X	
Pass/Fail Conditions: This test is passed if an IllegalArgumentException is thrown		
Pre-Conditions: None Input: -1.001, 0.7, 0		
Expected Results: An IllegalArgumentException has been thrown.	Actual Results:	
Post-Conditions: There should not have been a Ball created.		

Table 16: Ball Constructor Small X

Test ID: n	Module: Ball	Status: TBT
	Ball Constructor Small Y	
Pass/Fail Conditions: This test is passed if an IllegalArgumentException is thrown		
Pre-Conditions: None Input: 1, -1.001, 0		
Expected Results: An IllegalArgumentException has been thrown.	Actual Results:	
Post-Conditions: There should not have been a Ball created.		

Table 17: Ball Constructor Small Y

InferenceEngine Tests

PCCommunicator Tests

Test ID: n	Module: Ball	Status: TBT

Ball Constructor Small Value

Pass/Fail Conditions: This test is passed if an IllegalArgumentException is thrown

Pre-Conditions: None

Input: 1, 0.7, -1

Expected Results: An IllegalArgumentException has been thrown.	Actual Results:

Post-Conditions: There should not have been a Ball created.

Table 18: Ball Constructor Small Value

Test ID: n	Module: Ball	Status: TBT

Ball Constructor Large Value

Pass/Fail Conditions: This test is passed if an IllegalArgumentException is thrown.

Pre-Conditions: None

Input: 1, 0.7, 16

Expected Results: An IllegalArgumentException has been thrown.	Actual Results:

Post-Conditions: There should not have been a Ball created.

Table 19: Ball Constructor Large Value

Test ID: n	Module: InferenceEngine	Status: TBT

Updating Table State

Pass/Fail Conditions: This test is passed if all post-conditions are met.

Pre-Conditions: None

Input: A 16-by-2 array of doubles that are valid positions, BallType.STRIPES

Expected Results: None	Actual Results: None

Post-Conditions:
1. Stored BallType is BallType.STRIPES.

2. The stored positions array is the same as the one passed in.

3. The stored best shot is null.

4. The stored table state reflects the positions passed in.

Table 20: Updating Table State

Test ID: n	Module: InferenceEngine	Status: TBT
	Selecting an Optimal Shot	
Pass/Fail Conditions: This test is passed if a reasonable Shot is returned.		
Pre-Conditions: The current table state is not null and the current ball type is not null or BallType.CUE. Input: None		
Expected Results: A reasonable Shot (no bank shots, shooting the right ball, valid x-/y-coordinates).	Actual Results:	
Post-Conditions: The best shot for the current table state is stored.		

Table 21: Selecting an Optimal Shot

Test ID: n	Module: PCCommunicator	Status: TBT
	Read Valid Table State from File	
Pass/Fail Conditions: This test is passed if the output matches the data in the text file.		
Pre-Conditions: None. Input: A text file with 16 ball positions		
Expected Results: The 16 ball positions stored in the text file.	Actual Results:	
Post-Conditions: None.		

Table 22: Read Valid Table State from File

Test ID: n	Module: PCCommunicator	Status: TBT
	Read Table State from Non-Existent File	
Pass/Fail Conditions: This test is passed if a FileNotFoundException is thrown.		
Pre-Conditions: None. Input: None.		
Expected Results: A FileNotFoundException is thrown.	Actual Results:	
Post-Conditions: None.		

Table 23: Read Table State from Non-Existent File

Test ID: n	Module: PCCommunicator	Status: TBT
	Read Table State from File with Invalid Data	

Pass/Fail Conditions: This test is passed if an InputMismatchException is thrown.	
Pre-Conditions: None. Input: A file containing the text "Bad data".	
Expected Results: An InputMismatchException is thrown.	Actual Results:
Post-Conditions: None.	

Table 24: Read Table State from File with Invalid Data

Test ID: n	Module: PCCommunicator	Status: TBT
	Initiating the VR Program	

Pass/Fail Conditions: The test is passed if the VR Program has been run.	
Pre-Conditions: None. Input: None.	
Expected Results: Program is run and TableState.csv has been updated.	Actual Results:
Post-Conditions: TableState.csv contains the results of the VR Program.	

Table 25: Initiating the VR Program

Test ID: n	Module: Shot	Status: TBT
	Shot Constructor Good Inputs	

Pass/Fail Conditions: This test is passed if the Shot is successfully created and stores the correct information.	
Pre-Conditions: None Input: 1, 0.5, 3.5, 1	
Expected Results: A new Shot with an x-coordinate of 1, a y-coordinate of 0.5, an angle of 3.5, and a power of 1.	Actual Results:
Post-Conditions: Shot has been created.	

Table 26: Shot Constructor Good Inputs

Test ID: n	Module: Shot	Status: TBT
Shot Constructor Large X		
Pass/Fail Conditions: This test is passed if an IllegalArgumentException is thrown.		
Pre-Conditions: None Input: 1.87658, 0.5, 3.5, 1		
Expected Results: An IllegalArgumentException is thrown.	Actual Results:	
Post-Conditions: Shot has not been created.		

Table 27: Shot Constructor Large X

Test ID: n	Module: Shot	Status: TBT
Shot Constructor Small X		
Pass/Fail Conditions: This test is passed if an IllegalArgumentException is thrown.		
Pre-Conditions: None Input: -0.001, 0.5, 3.5, 1		
Expected Results: An IllegalArgumentException is thrown.	Actual Results:	
Post-Conditions: Shot has not been created.		

Table 28: Shot Constructor Small X

Shot Tests

SimulationInstance Tests

Test ID: n	Module: Shot	Status: TBT
Shot Constructor Large Y		
Pass/Fail Conditions: This test is passed if an IllegalArgumentException is thrown.		
Pre-Conditions: None Input: 1, 0.94958, 3.5, 1		
Expected Results: An IllegalArgumentException is thrown.	Actual Results:	
Post-Conditions: Shot has not been created.		

Table 29: Shot Constructor Large Y

Test ID: n	Module: Shot	Status: TBT
Shot Constructor Small Y		
Pass/Fail Conditions: This test is passed if an IllegalArgumentException is thrown.		
Pre-Conditions: None Input: 1, -0.001, 3.5, 1		
Expected Results: An IllegalArgumentException is thrown.	Actual Results:	
Post-Conditions: Shot has not been created.		

Table 30: Shot Constructor Small Y

Test ID: n	Module: Shot	Status: TBT
Shot Constructor Large Angle		
Pass/Fail Conditions: This test is passed if an IllegalArgumentException is thrown.		
Pre-Conditions: None Input: 1, 0.5, 6.284, 1		
Expected Results: An IllegalArgumentException is thrown.	Actual Results:	
Post-Conditions: Shot has not been created.		

Table 31: Shot Constructor Large Angle

Test ID: n	Module: Shot	Status: TBT
	Shot Constructor Small Y	
Pass/Fail Conditions: This test is passed if an IllegalArgumentException is thrown.		
Pre-Conditions: None Input: 1, 0.5, -0.01, 1		
Expected Results: An IllegalArgumentException is thrown.	Actual Results:	
Post-Conditions: Shot has not been created.		

Table 32: Shot Constructor Small Angle

Test ID: n	Module: Shot	Status: TBT
	Shot Constructor Large Power	
Pass/Fail Conditions: This test is passed if an IllegalArgumentException is thrown.		
Pre-Conditions: None Input: 1, 0.5, 3.5, 1.001		
Expected Results: An IllegalArgumentException is thrown.	Actual Results:	
Post-Conditions: Shot has not been created.		

Table 33: Shot Constructor Large Power

Test ID: n	Module: Shot	Status: TBT
	Shot Constructor Small Power	
Pass/Fail Conditions: This test is passed if an IllegalArgumentException is thrown.		
Pre-Conditions: None Input: 1, 0.5, 3.5, 0		
Expected Results: An IllegalArgumentException is thrown.	Actual Results:	
Post-Conditions: Shot has not been created.		

Table 34: Shot Constructor Small Power

Test ID: n	Module: SimulationInstance	Status: TBT
Simulation Instance Constructor Good Inputs Not Shooting 8-Ball		

Pass/Fail Conditions: This test is passed if the array of Balls is created, the 8-ball is not the target ball, and the initial velocity of the cue ball is set.
Pre-Conditions: InferenceEngine.myBallType = BallType.SOLID Input: A 16-by-2 array of doubles with at least one ball of type "solid" on the table, 2, 0.4

Expected Results: A SimulationInstance has been created with an array of Balls with positions corresponding to the array, the initial velocity vectors of the cue ball have been set according to the power and angle.	Actual Results:

Post-Conditions: A SimulationInstance has been created.

Table 35: Simulation Instance Constructor Good Inputs

Test ID: n	Module: SimulationInstance	Status: TBT
Simulation Instance Constructor Good Inputs Shooting 8-Ball		

Pass/Fail Conditions: This test is passed if the array of Balls is created, the 8-ball is the target ball, and the initial velocity of the cue ball is set.
Pre-Conditions: InferenceEngine.myBallType = BallType.SOLID Input: A 16-by-2 array of doubles with no balls of type "solid" on the table, 2, 0.4

Expected Results: A SimulationInstance has been created with an array of Balls with positions corresponding to the array, the initial velocity vectors of the cue ball have been set according to the power and angle.	Actual Results:

Post-Conditions: A SimulationInstance has been created.

Table 36: Simulation Instance Constructor Good Inputs

Test ID: n	Module: SimulationInstance	Status: TBT
Simulation Instance Constructor Large Power		

Pass/Fail Conditions: This test is passed if an IllegalArgumentException has been thrown.
Pre-Conditions: None Input: A 16-by-2 array of doubles, 2, 1.001

Expected Results: An IllegalArgumentException has been thrown.	Actual Results:

Post-Conditions: An IllegalArgumentException has been thrown.

Table 37: Simulation Instance Constructor Large Power

17

Test ID: n	Module: SimulationInstance	Status: TBT

Check for Walls

Pass/Fail Conditions: This test is passed if the expected results are equal to the actual results.

Pre-Conditions: None

Inputs:
(0.07070, true)
(0.07072, true)
(0.866, true)
(0.868, true)
(0.980, true)
(0.982, true)
(1.776, true)
(1.778, true)
(0.07070, false)
(0.07072, false)
(0.849, false)
(0.851, false)

Expected Results:	Actual Results:
false	
true	
true	
false	
false	
true	
true	
false	
false	
true	
true	
false	

Post-Conditions: None.

Table 38: Check for Walls

Test ID: n	Module: SimulationInstance	Status: TBT

Get Angle from Coordinates

Pass/Fail Conditions: This test is passed if the expected results are within 0.0001 of the actual results. Notably in the case where x = y = 0, the angle will be $\frac{3}{2}\pi$ which is not technically correct but that does not matter for this project.

Pre-Conditions: None

Inputs:
(1, 0)
(2, 1)
(0, 1)
(-1, 2)
(-1, 0)
(-1, -5)
(0, -1)
(2, -3)

Expected Results:	Actual Results:
0	
0.463647609	
$\frac{\pi}{2}$	
2.034443936	
π	
4.514993421	
$\frac{3\pi}{2}$	
5.300391584	

Post-Conditions: None.

Table 39: Get Angle from Coordinates

TableState Tests

Test ID: n	Module: SimulationInstance	Status: TBT
	Ball-Wall Collision	

Pass/Fail Conditions: This test is passed if the expected results are within 0.0001 of the actual results.	
Pre-Conditions: None Inputs: (5, true) (-1.2, false)	
Expected Results: -4.33 -1.2	Actual Results:
Post-Conditions: None.	

Table 40: Ball-Wall Collision

5.1.2   PC VR Program

5.1.3 μC Program

5.2   System Tests

6   Summary of Results

This section will be completed once the first version of the system is completed and all tests can be run.

Test ID: n	Module: SimulationInstance	Status: TBT

Check if in Pocket

Pass/Fail Conditions: This test is passed if the expected results are equal to the actual results.

Pre-Conditions: None

Inputs:
(1, 0.5)
(0,0)
(0.06, 0.02)
(0, 0.921)
(0.03, 0.92)
(0.924,0)
(0.92, 0.02)
(0.924, 0.921)
(0.95, 0.921)
(1.848,0)
(1.84, 0.04)
(1.848, 0.921)
(1.84, 0.915)

Expected Results:	Actual Results:
false	
true	
false	
true	
false	
true	
false	
true	
false	
true	
false	
true	
false	

Post-Conditions: None.

Table 41: Check if in Pocket

Test ID: n	Module: TableState	Status: TBT
	TableState Constructor Good Inputs	

Pass/Fail Conditions: This test is passed if the TableState is successfully created and stores the correct information.	
Pre-Conditions: None Input: A 16-by-2 array of doubles that hold the position of the balls	
Expected Results: A new TableState with 16 balls in positions corresponding to those passed in.	Actual Results:
Post-Conditions: TableState has been created.	

Table 42: TableState Constructor Good Inputs

Test ID: n	Module: TableState	Status: TBT
	TableState Constructor Too Many Elements	

Pass/Fail Conditions: This test is passed if the TableState is not created.	
Pre-Conditions: None Input: A 17-by-2 array of doubles	
Expected Results: An IllegalArgumentException has been thrown.	Actual Results:
Post-Conditions: TableState has not been created.	

Table 43: TableState Constructor Too Many Elements

Test ID: n	Module: TableState	Status: TBT
	TableState Constructor Not Enough Elements	

Pass/Fail Conditions: This test is passed if the TableState is not created.	
Pre-Conditions: None Input: A 15-by-2 array of doubles	
Expected Results: An IllegalArgumentException has been thrown.	Actual Results:
Post-Conditions: TableState has not been created.	

Table 44: TableState Constructor Not Enough Elements

Test ID: n	Module: TableState	Status: TBT
TableState Constructor Elements Too Small		
Pass/Fail Conditions: This test is passed if the TableState is not created.		
Pre-Conditions: None Input: A 16-by-1 array of doubles		
Expected Results: An IllegalArgumentException has been thrown.	Actual Results:	
Post-Conditions: TableState has not been created.		

Table 45: TableState Constructor Elements Too Small

Test ID: n	Module: TableState	Status: TBT
TableState Constructor Elements Too Large		
Pass/Fail Conditions: This test is passed if the TableState is not created.		
Pre-Conditions: None Input: A 16-by-3 array of doubles		
Expected Results: An IllegalArgumentException has been thrown.	Actual Results:	
Post-Conditions: TableState has not been created.		

Table 46: TableState Constructor Elements Too Large

Test ID: n	Module: TableState	Status: TBT
TableState Deep Copy		
Pass/Fail Conditions: This test is passed if the array of Balls returned have the same values but are not the same Objects.		
Pre-Conditions: A TableState exists in memory. Input: None.		
Expected Results: An array of Balls that have the same positions as those in the TableState.	Actual Results:	
Post-Conditions: None.		

Table 47: TableState Deep Copy

Test ID: n	Module: System	Status: TBT
	Aligned Shot	

Pass/Fail Conditions: This test is passed if the target ball is sunk by the machine 50% of the time and the shot should be made within 90 seconds (as per the *Summary and Goals* document).
Pre-Conditions: Machine must not be currently moving or taking a shot. There are three balls on the table, the cue ball, the target ball, and the eight ball. The cue ball, target ball, and one of the pockets are aligned near perfectly along an imaginary line. The eight ball is not in a position to interfere with motion of the balls along that line.   Input: *Take Shot* button pressed.

Expected Results: Only the target ball should be sunk.	Actual Results:

Post-Conditions: The eight ball should remain on the table. The target ball should be sunk. There are no requirements for the cue ball, but bonus points if it remains on the table.

Table 48: Aligned Shot

Test ID: n	Module: System	Status: TBT
	Angled Shot	

Pass/Fail Conditions: This test is passed if the target ball is sunk by the machine 50% of the time and the shot should be made within 90 seconds (as per the *Summary and Goals* document).
Pre-Conditions: Machine must not be currently moving or taking a shot. There are three balls on the table, the cue ball, the target ball, and the eight ball. There should be a shot that can be made with a modest angle that will sink the target ball. The eight ball is not in a position to interfere with expected motion of the balls.   Input: *Take Shot* button pressed.

Expected Results: Only the target ball should be sunk.	Actual Results:

Post-Conditions: The eight ball should remain on the table. The target ball should be sunk. There are no requirements for the cue ball, but bonus points if it remains on the table.

Table 49: Angled Shot

Test ID: n	Module: System	Status: TBT
	Shot Cancelled Before Motion	

Pass/Fail Conditions: This test is passed if the machine does not move.
Pre-Conditions: None.   Input: *Take Shot* button pressed,   Then *Cancel* button pressed before machine moves.

Expected Results: The machine should cancel the instruction and not move.	Actual Results:

Post-Conditions: The machine should not have moved or be moving.

Table 50: Shot Cancelled Before Motion

Test ID: n	Module: System	Status: TBT
	Shot Cancelled During Motion	

Pass/Fail Conditions: This test is passed if the machine ceases movement within 2 seconds.	
Pre-Conditions: None. Input: *Take Shot* button pressed, Then *Cancel* button pressed while machine is moving.	
Expected Results: The machine should cease movement.	Actual Results:
Post-Conditions: The machine should not be moving.	

Table 51: Shot Cancelled During Motion

Test ID: n	Module: System	Status: TBT
	Move Request (To Zero X-Coordinate)	

Pass/Fail Conditions: The machine moves to the zero x-coordinate within 20 seconds.	
Pre-Conditions: Machine's y-rail is located closer to the large x-coordinate. Input: *Move* button pressed	
Expected Results: The machine should move to the zero x-coordinate of the table.	Actual Results:
Post-Conditions: The machine should be located at the zero x-coordinate.	

Table 52: Move Request (To Zero X-Coordinate)

Test ID: n	Module: System	Status: TBT
	Move Request (To Largest X-Coordinate)	

Pass/Fail Conditions: The machine moves to the largest x-coordinate within 20 seconds.	
Pre-Conditions: Machine's y-rail is located closer to the zero x-coordinate. Input: *Move* button pressed	
Expected Results: The machine should move to the largest x-coordinate of the table.	Actual Results:
Post-Conditions: The machine should be located at the largest x-coordinate.	

Table 53: Move Request (To Largest X-Coordinate)

Test ID: n	Module: System	Status: TBT
	Check For Political Correctness	

Pass/Fail Conditions: All interviewees agree that there are no direct references to any religious or political groups.	
Pre-Conditions: None. Input: 20 colleagues will be asked to give their opinion on whether the system created has no direct reference to any religious or political groups.	
Expected Results: Colleagues decide that there are no direct references to any religious or political groups.	Actual Results:
Post-Conditions: None.	

Table 54: Check For Legality and Political Correctness