STINT Test Manual

STINT Project

Software Engineering

Winter 2014

University of Western Australia

Crawley, WA 6009

Stint Members (TEAM B):

Dean Cook
Ashwin D'Cruz
Cameron Johnson
Kieran Hannigan
Marcus Pham
Alex Tonkin

Table of Contents

Objectives

This document describes the tests for STINT. The primary test will be to test the functionality of the STINT program to complete the task of selecting stints during a hockey match. Secondary tests are to be testing for correctness and optimality. Minor tests will also be done to check the functionality of the construction of the field and functionality of each of the algorithms separately.

Test Materials

Tests are to be conducted in similar environments, as a typical user will use the program in. Tests are conducted in a Microsoft Windows operating system alongside Catapult Sprint.

Test Summary

Functions to be tested are:

A. Valid input into Catapult Sprint
B. Correct stint start and finish times
C. Acceptable processing times
D. Correct construction of the playing field
E. Correct operation of heuristics separately
F. Error checking for correct user inputs

Testing Strategy

The main testing for STINT is manual inspection for the correctness of the system. Most tests will involve testers manually checking the output data of the tests and determining whether the data is within acceptance levels.

Pure testing java methods are to be created to implement the tests. These methods will be placed in an outside class, separate to the main program with arguments to test each test individually.

Tests are to be done more than 5 times each on different files to check for consistency with results.

Test A

STINT is a program that is to be used in conjunction with Catapult Sprint. STINT is to create .vid files that are to then be used by Catapult Sprint to do further analysis, but effectively reducing repetitive human processing time. Hence, STINT needs to be able to create correctly formatted files that are readable by Catapult Sprint.

This test is to test whether the files outputted by STINT are correctly formatted for use in Catapult Sprint.

Test Specification

The requirements for correct completion of the test includes that the Catapult Sprint is capable of reading in the files created by STINT and display correctly.

Test Description

- **Means of Control:** Output data from STINT is to be input into Catapult Sprint.
- **Input Data**: The .vid file outputted by STINT is to be inputted into Catapult Sprint to test for complete functionality of the .vid file in Catapult Sprint.
- **Input Commands**: Manual input into Catapult Sprint. Catapult Sprint will be asked to process the .vid file and analyze the data.
- **Output Data:** Visually inspect the data has been inputted correctly.
- **System Messages**: No error messages to be seen in Catapult Sprint when attempting to open and process the .vid file.

Procedures: 1. Manually open Catapult Sprint
2. Select .vid file for processing
3. Visually inspect Catapult Sprint for correct display

Test Analysis Report

- **Function:** Positive tests are defined as Catapult Sprint being able to accept the .vid file created by STINT and not throw any error messages to the user. Testing is to be done >10 times on different .vid file outputs. Catapult Sprint should be able to use the .vid file as if Catapult Sprint did the processing.
- **Performance:** Test will show that the STINT created .vid file can be used in the same manner as a .vid file created by Catapult Sprint
- **Data Measures:** File needs to be 100% compatible with Catapult Sprint.

Test B

STINT needs to be able to accurately determine the start and completion times of stints within a match. From the data provided to the program, start/finish times of a match, a specified ground and a .csv containing GPS/accelerometer data, STINT is to make accurate predictions as to the start and finish times of separate stints and then output this as a .vid file.
This test is for correctness of choosing the start and completion times of stints by STINT.

Test Specification

This test is to ensure that STINT is not creating unusable .vid files that have separate stints that are too far outside the actual times of a player being active on the pitch. This test will test the times made by STINT using an external testing method to the main program that will produce a .txt file detailing the variance between the stint times.
Acceptance for this test is for STINT to select all the times for the stints within 5 seconds of a humanly modified .vid file.

Test Description

- **Means of Control**: A correctness testing method/class
- **Input Data**: A human created .vid file using Catapult Sprint, A STINT created .vid file for the same player.
- **Input Commands**: Run correctness testing method with 2 .vid input files.
- **Output Data** – Outputted text file that shows the variance between the 2 inputs. File gives the separate times of the difference between the stint times.
- **System Messages** – No system errors

Procedures:
1. Process a given file in STINT
2. Process same .csv file in Catapult Sprint manually
3. Run correctness method
4. Visually inspect the outputted text file.

Test Analysis Report

- **Function:** Positive test result is defined as the outputted times as shown in the .txt file are all within 5 seconds of a user created .vid file. Users are to visually inspect he output file to ensure there are no times that are greater than 5 seconds. Additionally averages of the times are also given at the end of the text file.
- **Performance**: The test will show the difference in times for the stints as well as allowing users to observe the average difference between the times.
- **Data Measures:** Outputted .txt file should only show times less than 5 seconds and have an average well below 5 seconds.

Test C

STINT is to be a program that reduces the processing time of selecting the start and finish times of stints within a match. Currently, the task takes about 10 minutes per match for an experienced analyst to manually select the start and finish times of stints for further analysis in Catapult Sprint.

This test tests the time performance of STINT, that is, how long the program takes to process a given match.

Test Specification

This test will ensure optimality is maintained when creating the program and allow for a balance between time performance and actual performance of STINT.

Requirements for acceptable pass are defined, as the reported processing time for a match is 10mins. The program however, should aim for a much lower value.

Test Description

- **Means of Control:** Data is fed manually into STINT with an additional timer java method to display the processing time of the program.
- **Input Data**: The .csv file to be inputted into the testing methods which implements STINT and displays the processing time for the program.
- **Input Commands:** Input into the timer test method with the .csv file as an argument.
- **Output Data**: Display time taken in command line/text file
- **System Messages**: Displays the time taken

Procedures: 1. Input .csv into STINT with input command test time
2. Manually inspect value outputted by the timer to assess performance

Test Analysis Report

- **Function:** A positive test result is defined, as the entire processing time for a match is less than 10 minutes. Users are to inspect the time taken for the outputted processing time for a match to be less than 10 minutes.
- **Performance:** This will show the time taken for the entire process of converting the .csv to an appropriate .vid file is within the acceptable times.
- **Data Measures:** Data visually inspected on the window to show that the time taken for the processing time is less than 10 minutes.

Test D

The Stint editor is supposed to take two GPS coordinates and the aspect ratio of a given field as provided by the user. This information is then used in conjunction with the player's GPS data, specifically their location, to determine during which time periods they were on the field and when they were off the field. This data feeds into the main algorithm to determine stint times.
The editor is set to take both GPS coordinates without insisting that the user input the coordinates in any order, for example greater latitude first. This test is designed to ensure that the order of the two coordinates does not affect the final stints outputted.

Test Specification

This particular test will ensure the quality of the stints produced and more and ensure that the order of the GPS coordinates inputted by the user does not affect the stints produced in any way.
This test will be testing methods from the GPS related classes. To be more precise, it will be testing the robustness of the rotate function in the GPSCoordinate class. The function was designed to rotate a point about a point to its 'left'. We wish to ensure that the inverse of this will also work; that a point can rotate correctly about a point towards its right.
In addition, we wish to ensure that after this transformation all player GPS coordinates will also rotate accurately and that the analyser will still precisely determine if a player is on the field or not.
To test this, we will input two GPS coordinates and their related aspect ratio for a field. Then the order in which these GPS coordinates were inputted will be reversed and if the program functions correctly, the outputted stints will be exactly the same.

Test Description

- **Means of Control:** Data will be manually entered.
- **Input Data:** Information for all players for a tournament will be entered along with the GPS coordinates of the field they played on. On the second run through, the same player data will be entered but the order the GPS coordinates are inputted will be reversed.
- **Input Commands:** The program will be tasked to process all the player data.
- **Output Data:** Stints for each player.

Test Analysis Report

- **Function:** This test will demonstrate that all functions related to the GPS class work regardless of the order of the GPS coordinates were entered.
- **Performance:** This test will prove that regardless of the order of input of GPS coordinates, the stints produced will be identical.
- **Data Measures**: The stints produced will be compared and if they are identical, the test would have shown that the GPS class is as robust as planned.

Test E

The main processing of the matches in STINT is in the heuristics used to determine active periods for the player. The 3 pieces of data that are used to analyse stints are start/finish time of the match, GPS data and accelerometer data.

These tests are to work independently of each other and then a final balancer is used to determine whether a player is actually active with respect to the data given.

This series of tests is to test the functionality of the heuristics used independently and show that the program is selecting active times with respect to the heuristic only.

Test Specification

These tests will ensure correct operation of the methods used to select active times. Separate methods are used to test each method. Each heuristic should be able to work independently and predict start and finish times of the stints based on the given data.

Test Description

- **Means of Control:** Data to be entered manually into the test method which executes the main program to only run the specified heuristic.
- **Input Data:** User is to input the .csv file to be processed.
- **Input Commands:** The program will output a .vid file that shows the predicted stints using only one heuristic.
- **Output Data:** A .vid file that can be opened on Catapult Sprint that only uses one heuristic to determine stints.

Test Analysis Report

- **Function:** This test will check for the correct functionality of the given heuristic methods
- **Performance:** This test will show that the method works correctly independently of other heuristics.
- **Data Measures**: The stint will be opened on Catapult Sprint to show the correct times for stints with respect to the given heuristic.

Test F

Users are able to input data into STINT via the graphical user interface provided. The inputs required to run STINT are: Grounds Selection, Start Time, Finish Time and the directory of .csv files to be processed.

Manual input would bring extra room for error. Hence checks to make sure the user is entering correct data is required.

This test is to ensure that the input data is inputted correctly, and checks for the functionality of a checker that prompts the user when there is an irregularity in the input data.

Test Specification

These tests include making sure the user prompts appear in each of the following cases: Invalid grounds, invalid start time, invalid finish time, abnormal game period, incorrect .csv file and a warning for the user for large/deep folders.

Test Description

- **Means of Control:** Data will be manually entered into the program
- **Input Data:** Input data is to be the various manual input into the graphical user interface, each to test for the appearance of the checkers.
- **Input Commands:** Run STINT with strange/incorrect user inputs.
- **Output Data:** Pop-up windows or prompts to appear when incorrect input is received
- **System Messages**: System will halt processing and prompt user to change input.

Test Analysis Report

- **Function:** This test will make sure all the user interface prompts are appearing when required.
- **Performance:** This test will prove that prompts always appear when a user inputs strange data. Users are able to change the data or proceed with given data.
- **Data Measures**: Acceptance for the test involves prompts always appearing when bad inputs are received.