Test Plan

FROG Recognizer of Gestures

Team Better Recognize
Version 2.0
May 3, 2010

Revision Sign-Off

By signing the following, the team member asserts that he/she has read the entire document and has, to the best of his or her knowledge found the information contained herein to be accurate, relevant, and free of typographical error.

Name	Signature	Date
Josh Alvord		
Alex Grosso		
Jose Marquez		
Sneha Popley		
Phillip Stromberg		
Ford Wesner		

Revision History

The following is a history of revisions of this document.

Document Version	Date Edited	Changes
Version 1.0	2/8/10	Initial test plan sections outlined
Version 1.1	4/3/10	Outlined unit testing and requirements traceability matrix
Version 2.0	5/3/10	Major test plan revisions, final iteration

Table of Contents

1. Introduction

1.1 Purpose

This document describes how testing will be performed on the FROG system. It identifies the primary components that will undergo extensive unit testing and describes a suite of test cases to be used for system and integration testing. The test cases were designed to provide adequate coverage of FROG features and requirements.

1.2 Overview

Section 2 – Resource Requirements: Provides the hardware and software requirements for testing and evaluating the FROG system.
Section 3 – Unit Testing: Identifies the primary components of the FROG system and describes the extensive unit testing required of each of these components.
Section 4 – Integration and System Testing: Provides a Requirements Traceability Matrix and describes the test cases developed to provide adequate coverage of the FROG requirements and features.
Section 5 – Testing Deliverables: Description of how testing was to be documented and reported.

2. Resource Requirements

2.1 Hardware

The hardware required for FROG testing includes:

- 4 Sun SPOTs
- 1 Sun SPOT base station with USB connector cable
- 1 Windows XP lab computer (Intel Pentium 4 3.2 GHz, 1 GB RAM)
- 1 iMac or MacBook (to test platform independence)

2.2 Software

The software required to be installed on the testing computer (as described in section 2.1) is as follows:

- Java JRE 6.0
- FROG v1.1 (without Bluetooth)
- Apache ANT (to deploy code to SPOTS)
- Sun SPOT SDK RED (5.0)
- Sun SPOT Manager(initialization of SPOT and Base station API)

The software required to be installed on a Sun SPOT for testing with FROG is:

- FROG SPOT Deployment Code 1.0 (available from Installation directory on project CD)

3. Unit Testing

Unit testing has been divided into four parts: SPOT data acquisition and communication, the graphical user interface (GUI), all algorithms relating to the training and recognition pipelines, and the demo game. All items listed in each section are the minimum testing needed to be considered acceptable for each unit.

- SPOT data acquisition and communication: The SPOT specific code for connecting, disconnecting, data acquisition, and data transfer.
- GUI: All aspects related to the functionality, appearance, and usability of the graphical user interface.
- Mathematical foundation: All components of the training and recognition pipelines that provide the theoretical basis of FROG's gesture recognition.
- Demonstration game: All code within the demo game including loading libraries, animations, execution when gestures are performed, proper recognition, and scoring.

3.1 SPOT Data Acquisition and Communications

- **Communications**
  - **Distance**

    Determine…
    - Maximum possible sending distance by incrementally increasing distance between base station and remote.
    - Optimal distance for sending using the same method. Optimal sending distance would be the range a SPOT can be from the base station where there is no noticeable delay between releasing the "send" button and the host receiving and processing the end of data.
    - If optimal distance is lessened by using multiple SPOTs simultaneously.
  - **Reliability**

    Determine…
    - If objects in the area affect the transmission of data of the SPOT. Try using the SPOT behind doors and desks and around large metal computer cases. Use around cell phones and other transmitting devices.
    - If rapidly pressing and releasing the "send" button causes the SPOT to lock up, send corrupt data, or any other ill effects.
    - How long a SPOT can sit idle before its battery is dead or it unexpectedly disconnects. It is not desirable to unexpectedly disconnect for any other reason than the battery on the SPOT dying. Could leave the SPOT on overnight plugged in so its battery cannot die and see if it is still connected in the morning.

- **Speed**

  Determine…
    - The maximum sample rate that does not cause noticeable delays or slow downs on the host.
    - If having four SPOTs connected simultaneously affects transmission speed. Is there a noticeable delay in release of the "send" button and the host receiving and processing the end of data

- **Acceleration**
  - **Range**

    Determine…
      - If accelerometer reads 1.0 on each axis when left at rest in various orientations.
      - What the maximum observed acceleration is in any direction. Whip arm as fast as possible and looking for an artificial wall that samples never exceed.

- **Onboard Filtering**
  - **Performance**

    Determine…
      - If enabling filter degrades performance. This can be determined by observing noticeable delays in releasing the "send" button and the host receiving and processing the end of data.
      - If a combination of high sample rate and filtering can cause performance to degrade further. Set sample rate as high as 150Hz and observe with and without Filtering enabled.
  - **Accuracy**
      - If filters are removing vectors correctly. This can be checked by using enabling the exact same filter on FROG as well and using the Log to see if FROG eliminated any vectors that the SPOT missed.

3.2 GUI

GUI items to be tested

- GUI fulfills all requirements as specified in the FROG SRS
- Trace all paths through all GUI components
- Graphical displays (2D and 3D) are showing the correct sets of data/ graphs
- When a new session is loaded, GUI must clear any existing data
- GUI driven event handling corresponds to the correct events
- Input data is being received and tracked appropriately
- GUI has a minimum bounds set for window resizing
- All features of FROG accessible regardless of number of gestures saved
- Terminal items are correctly logging when selected

3.3 Mathematical Foundation of FROG

- **Filtering**
  - Comparison of computational results with by-hand results for a small set
  - Logging filtering behavior and magnitudes involved in computation to see exactly what is being filtered out and if they are behaving correctly
  - In particular, varied the acceleration data artificially around the threshold to make sure there was a "switch-over" from filtering to non-filtering

- **Quantizing**
  - Visual inspection of graphical k-means results – view data in combination with resulting means to verify a semblance of effectiveness
  - Verify k-means++ initializing differently
  - Verify iterative movement of means
  - Verify operation using different k-values (k-means and k-means++)
  - Verify operation with size of input less than the k-value
  - In particular, set up an initial condition in which the initial k=14 means are directly in the center of a cluster and verify that the means did not evolve (they were already the cluster centers at initialization, thus leading to no change)

- **HMM**
  - Verify maintenance of a statistical model; that is, view the evolution of the HMM and make certain that stochastic properties hold (maintained probability 1 in proper places)
  - Verify that our algorithm produced the same results as that of Wiigee's algorithm
  - Produce training that yields near 90% average accuracy for gesture recognition

- **Classifier**
  - Verify through tracking input/output that correct HMM match probabilities are being received and particularly that match probabilities are in a valid probability range
  - As with HMM, verify ability to obtain near 90% average accuracy for gesture recognition

3.4 Demonstration Game

Below is a list of all demonstration mode items to be tested broken down into key components of the game, demonstration mode must adhere to recognition requirements with 4 players

- **UFO items to be tested**
  - A newly created UFO targets a random cow
  - Once a UFO has begun abducting a cow, it will not continue movement until the cow has been abducted and has disappeared
  - UFO tractor beam only shows when picking up a cow
  - The correct UFO is destroyed when a correct gesture is performed

  - Cow is released from abduction if UFO is destroyed and is still visible on screen
  - UFOs stay within game bounds

- **Cow items to be tested**
  - Cows fall back to ground level if freed
  - Cows stay within game bounds
  - Cow does not hide at the end of its abduction path
  - Game ends when last cow has been successfully abducted

4. Integration and System Testing

4.1 Requirements Traceability Matrix

System and acceptance testing will be performed, in part, using a suite of test cases that provide adequate coverage of the software requirements of FROG. The Requirements Traceability Matrix is shown in Table 4-1. A detailed definition of each test case is given in section 4.2.

Requirements	Test Cases						
	1	2	3	4	5	6	7
Gen-01 – 3D acceleration data from mobile devices for training & recognition	x	x	x	x	x	x	x
Gen-02 – Device independent, provide for development of plug-ins	x	x	x	x	x	x	x
Gen-03 – Console window with user selectable info	x	x	x	x	x	x	x
Tra-01 – Only one device connected for training at a time	x	x	x	x		x	
Tra-02 – Save & load training sessions for reuse		x	x	x			
Tra-03 – Intuitive method of training system	x	x	x	x		x	
Tra-04 – Display of available trained gestures of a library		x	x	x		x	
Tra-05 – Add or delete gestures from file			x	x		x	
Tra-06 – Support idle state/dir. equiv filters. Readily add additional filters.			x	x		x	
Tra-07 – Vector quantizing & HMM training. Kmeans & Kmeans ++				x			
Tra-08 – Support modification of parameters (K) in quantizing				x			
Tra-09 – Support modification of parameters (states) in HMM				x			
Tra-10 – Real-time 3D graphing of acceleration data		x	x	x		x	
Tra-11 – Display & log sys status, acceleration data, execution results	x	x	x	x		x	
Rec-01 – Connect & recognize gestures on 4 devices simultaneously					x		x
Rec-02 – Each device/user can load a previous trained library					x		x
Rec-03 – Provide recognition feedback to the user				x	x	x	x
Rec-04 – Display & log sys status, acceleration data, execution results				x	x	x	x
Eva-01 – Only one device connected for evaluation at a time			x			x	
Eva-02 – Real-time feedback of recognition performance			x			x	
Eva-03 – Select gesture(s), sample size, and evaluation order			x			x	
Eva-04 – Display & log sys status, acceleration data, execution results			x			x	
Dem-01 – Allow 4 devices to connect and play							x
Dem-02 – Client approved demo program						x	x
Dem-03 – Track user’s score for evaluation of system/user performance						x	x
Dem-04 – Pre-trained gestures required. Message if not available							x
Pr-01 – Communication speed to support 4 users					x		x
Pr-02 – Recognition performed within 10 ms for traditional HMM			x	x	x	x	x
Pr-03 – SPOT filtering shall not hinder data retrieval or communication		x	x	x	x	x	x
Sqr-01 – 80% recognition accuracy			x	x	x	x	x
Sqr-02 – Designed for device specific plug-ins	x	x	x	x	x	x	x
Sqr-03 – Multi-platform framework	x	x	x	x	x	x	x

Table 4-1. Requirements Traceability Matrix

4.2 Definition of Test Cases

- **Test Case # 1 – General startup/minimal functionality/shut-down**
  - Primary items tested:
    - Minimal startup/shut-down
    - Closes correctly with/without base or motes connected
    - Main panel functionality called
    - Validate receiving 3D acceleration data
  - Test operations:
    - No base connected, close correctly, valid log
    - Base connected, remotes off, close correctly, valid log
    - Base connected, remotes on, close correctly, valid log
    - Main panel functional
      - Info, Close, Train, Recognize, Evaluation, Demo buttons functional
    - Validate receiving 3D acceleration data
      - Discover/Connect/No filtering/New gesture/3D data on console and 2D plot
      - Collect in each of 3 orientations to validate X,Y,Z acceleration due to gravity

- **Test Case # 2 – Connections/create gestures/delete instances/save and load sessions**
  - Primary items tested:
    - Discover/connect/reconnect sequence
    - Remains connected extended period of time (20 min)
    - Remains connected @ 15 ft
    - Varying sample rates
    - Filters applied or not
    - Create new named gestures with multiple instances
    - Delete instances
    - Save and reload sessions
  - Test operations:
    - 1 base/1 mote
    - Configure/Discover/Connect/Calibrate/Disconnect/Discover/(re)Connect
    - Sample Rate 1Hz/No filtering/New named Gesture/No Pic
    - Collect 5 instances/Capture one 2D plot for comparison/Delete instance 3/Refresh
    - Leave connected extended period of time
    - Move away distance of 20 ft and verify stays connected
    - Add new named gesture/with pic
    - Configure/Filter on device/DE set .2/ Idle set 1.2
    - Sample Rate 100Hz/Collect 6 instances/Capture one 3D plot for comparison/Delete first and last instance

        - Save session
        - Close down
        - Restart FROG
        - Load session previously saved session
        - Verify gestures and instances
        - Compare data of selected instance with 2D plot previously captured
        - Compare data of selected instance with 3D plot previously captured

- **Test Case # 3 – Creating/deleting instances and sets/eval stats/saving&reloading**
    - Primary items tested:
        - Creating multiple gestures and multiple instances
        - Edit - delete instances, change picture, refresh
        - Delete gestures
        - Evaluate selected or all gestures, random or sequential order
        - Saving gestures and evaluation statistics
        - Load saved gesture file
    - Test operations:
        - 1 base/1 mote
        - Connect/change filter/change sampling rate
        - Create multiple gestures (w/o images) with multiple instances
        - Edit/delete instances/change picture/refresh
        - Delete a gesture
        - Evaluation mode and verify gestures present
        - Generate stats on gestures (both selected and all, random and sequential)
        - Save gestures/exit FROG
        - Restart/load gestures/verify gestures and instances correct
        - Verify previous stats saved in Evaluation
        - Add additional gesture
        - Add additional stats
        - Save session/exit FROG/Reload/Validate gestures and stats
        - Clear stats/save session/exit FROG/reload/validate gestures and stats cleared

- **Test Case # 4 – Training parameters/selective logging of results/adding/deleting**
    - Primary items tested:
        - Data logging options
        - Log file saved
        - KMeans and KMeans++
        - Quantizer K value and threshold
        - HMM threshold and number of states
        - Add/delete gestures
        - Train/retrain
        - Save and reload session
    - Test operations:
        - 1 base/1 mote

        - Connect/filter/sampling rate/create new gestures
        - Test combinations of KMeans parameters and thresholds
        - Test combinations of HMM parameters and thresholds
        - Validate training output with log as permitted (K, number states, threshold values)
        - Validate logging correctly in Training, Recognition, and Evaluation
        - Delete/add gestures
        - Retrain
        - Save sessions/exit FROG
        - Load sessions
        - Recognition accuracy and time

- **Test Case # 5 – Recognition/logging**
    - Primary items tested:
        - 4 mote performance
        - Loading multiple FROG session files
        - Selective data logging
    - Test operations:
        - 1 base/4 motes
        - Discover/Connect/Disconnect various motes multiple times
        - Create new gestures with one mote
        - Load gesture files with other 3 motes
        - Perform gesture recognition with each mote – one at a time
        - Perform gesture recognition with 4 motes simultaneously
        - Observe performance of recognition

- **Test Case # 6 – Train/Recognize/Evaluate/Demo with current session**
    - Primary items tested:
        - All modes with current session
        - Add/delete and retrain gestures
    - Test operations:
        - 1 base/1 mote
        - Discover/Connect/Filter/Change Sampling
        - Train multiple instances for all game gestures
        - Recognition on current session
        - Return to Train to delete a gesture and retrain
        - Evaluate on current session
            - Selected gestures in random order
            - Selected gestures in sequential order
        - Return to Train to delete a gesture and retrain
        - Demo with 1 user using current session

- **Test Case # 7 – Recognition and Demo**
    - Primary items tested:

    - Recognition interface
    - Demo setup
    - Game play and scoring
- Test operations:
    - 1 base/4 motes
    - Load 4 sessions trained with game gestures (at least 2 different)
    - Recognition mode with 4 users
    - Obtain recognition results from each of the motes
    - Exit Recognition mode, enter Demo mode
    - Load 1 session lacking correct game gestures, verify warning
    - Load 4 sessions trained with game gestures (at least 2 different)
    - Connect 4 motes
    - Validate scoring by playing single player at a time
    - Restart game
    - Play with all 4 players simultaneously
    - Verify game play
    - Verify game exit and FROG exit

5. Testing Deliverables

Unit testing of code began as each code component was completed. Individual units were initially tested by the person(s) writing the code. Later others in the group were involved with a limited amount of unit testing for some of the components.

System testing was performed by group members other than the person(s) creating the code. Additionally, user acceptance testing was performed by the project sponsor, Dr. Payne. Due to the tightness of the schedule, there was only a limited amount of acceptance testing performed. However, several members of the group were very committed in their testing efforts and performed extensive, thorough integration and system testing, thus providing a well-tested product even with the limited acceptance testing.

Although it was planned to provide a Test Item Report as shown in Appendix A, time limitations prevented extensive use of documented testing.

Appendix A

Below is the format for the Test Item Report to be used for testing requirements of the FROG system.

Test Item Report

Date:

Tester:

Item(s) to be Tested: (What specifically is tested? Is it a unit or integration and system test?)

Method Used: (How specifically will you be testing the item, what steps will you take?)

Results: (What were the specific results of the test?)

Conclusion: (What does this tell you, is the item complete and if not possible reasons why?)