Project Title: System Verification and Validation Plan for Mechtronics Enigeering

Team 32, Wingman, SmartVault
Edward He
Erping Zhang
Guangwei Tang
Peng Cui
Peihua Jin

November 2, 2022

1 Revision History

Date	Version	Notes
2022-11-2	Edward He, Erping Zhang Guangwei Tang, Peng Cui Peihua Jin	Revision 0

Contents

List of Tables

2 Symbols, Abbreviations and Acronyms

symbol	description
SRS	Software Requirements Specifications
MG	Module Guide
MIS	Module Interface Specification

3 General Information

3.1 Summary

The SmartVault project is a Mechatronics system that is able to assist user identify the location of assigned objects in a given area.

3.2 Objectives

This document is intended to develop a systematic plan for testing the functionality of the system.It meant to show the system has met the requirements in both software and hardware aspects mentioned in requirements document.In this document, system test and unit test will be conducted to check the functionality of the system. Functional and non-functional requirements will be separated for testing for both section. By the end of testing process, it can be shown that the system is working properly and available for usage.

3.3 Relevant Documentation

1. Development Plan
2. System Requirements Specification
3. Hazard Analysis
4. Module Interface Specification
5. Module Guide

4 Plan

In this section, it provides an overview and roadmap for the document. This section outlines the members of the verification and validation team for the project SmartVault and their assigned roles. This section describes the methods to verify and validate each component of the project, which includes SRS, Design, implementation, testing tools and the software for SmartVault.

4.1 Verification and Validation Team

Member	Role
Edward He	Static Verification Lead; Back-end Testing
Erping Zhang	Design Verification Lead; Front-end Testing
Guangwei Tang	SRS Verification Lead; Hardware Testing
Peng Cui	Dynamic Verification Lead; Code Inspection
Peihua Jin	Automated Testing Lead; Code Inspection
Spencer Smith	Review of documents and system
Peer Groups	Review of documents and system

Table 1: Verification and Validation Team Members and Roles

4.2 SRS Verification Plan

In order to verify the SRS document, the SRS checklist will be used to make sure that the document included all components on the checklist and are complete. The SRS document will also be reviewed by other groups and based on their feedbacks, changes will be made accordingly. Professor Spencer Smith and the teaching team will also review the document and provide feedbakcs. Requirements Listed in the SRS document will be verified and validated by the methods listed in the following sections.

4.3 Design Verification Plan

To verify the design of the system, MG and MIS checklist will be utilized. The team has decided and implemented a procedure for each individual design made by different team developer. Any design decision is required to be reviewed and verified by at least 3 other members. The design of SmartVault will also be reviewed by peers and instructor.

4.4 Implementation Verification Plan

The implementation of the SmartVault will be done with both dynamic and static verification. For dynamic verification, both non-functional and functional requirements implementation will be tested. The test will consist both system test and unit test to ensure the written code comply with the requirements and functionalities. More details for the dynamic verification refer to

section 5 and 6 of this document. For the static verification, it will be done with code inspection and coding standards. The team will utilize Git and Github to inspect and verify coding. The merge of a new functionality or modification of current implementation has to be reviewed by 2 others member prior to merging to the main branch. Code inspections/walkthrough will be done weekly to ensure it follows the coding standards. SmartVault will be primarily written in Python and team members are required to follow PEP8 standard by using linter Pylint.

4.5 Automated Testing and Verification Tools

- Coding Standard(PEP 8) : pylint
- Unit Testing Frameworks: PyUnit

4.6 Software Validation Plan

Software validation will be mostly conducted by team members. External validation is set to be conducted with consumers(stakeholders) of the product where they can try out the system and report if there is anything they would like to change.

5 System Test Description

5.1 Tests for Functional Requirements

5.1.1 Image Processing and Storage

Manual Testing

1. IPR1-1

   Control: Manual

   Initial State: The system is turned on, the working environment should be empty

   Input: Images of the working environment and a human show up in the environment

Output: Coordinate of the detected human body

Test Case Derivation: The output coordinate of the detected human body should located on the human body in the image

How test will be performed: Turn on the system. Start running the system without the human showed up in the environment. Then the human should move into the environment and the system will put a square shape on the human body in the image.

2. IPR2-1

   Control: Manual

   Initial State: The system is turned on, the working environment should be empty

   Input: Images of the working environment and a hand show up in the environment

   Output: Coordinate of the detected hand and each joints

   Test Case Derivation: The output coordinate of the detected hand and joints should located on the joints in the image

   How test will be performed: Turn on the system. Start running the system without the hand showed up in the environment. Then the hand should move into the environment and the system will put dots on each joints of the hand in the image.

3. IPR3-1

   Control: Manual

   Initial State: The system is turned on, the working environment should be empty

   Input: Images of the working environment and few objects in the environment

   Output: Coordinate of the detected objects

   Test Case Derivation: The output coordinate of the detected objects should located on the object in the image

How test will be performed: Turn on the system. Start running the system without the objects showed up in the environment. Then the objects should move into the environment and the system will put square on each object in the image.

4. IPR4-1

   Control: Manual

   Initial State: The system is turned on, the working environment should include at least one item.

   Input: Images of the working environment and object in the environment. Different Images should have different location of the object in the environment.

   Output: The re-location mode should be activated

   Test Case Derivation: The re-location mode should be activated when the object location change

   How test will be performed: Turn on the system. Start running the system with the objects showed up in the environment. Then change the location of object in the environment, the system should activated the re-location mode.

5. IPR5-1

   Control: Manual

   Initial State: The system is turned on, the working environment should include at least two items.

   Input: Images of the working environment and at least two different objects in the environment.

   Output: The system should record two different objects into the database

   Test Case Derivation: The database should have the correct record according to the testing cases

   How test will be performed: Turn on the system. Start running the system with the different two objects showed up in the environment. Then check the record in database, it should have two different records of the objects.

Automatic Testing

1. IPR6-1

   Control: Automatic

   Initial State:The system is turned on, camera works properly.

   Input: The automatic testing tool will activate the photo storage function.

   Output: The photos taken by the camera with create date/time

   Test Case Derivation: The photo should have the time information

   How test will be performed: Turn on the system, make sure the camera working as required. Then run the automatic testing tool, the tool will force the system to store the photos with create data/time information with it.

2. IPR7-1

   Control: Automatic

   Initial State:The system is turned on.

   Input: The automatic testing tool will call the data storage module and send the object information to the module.

   Output: The different objects information will be stored in the database with unique ID

   Test Case Derivation: The object information in the database should have unique IDs

   How test will be performed: Turn on the system. Then run the automatic testing tool, the tool will call the data storage module and pass the parameter into the module. Then the module will store the objects information with unique IDs.

3. IPR8-1

   Control: Automatic

   Initial State:The system is turned on.

Input: The automatic testing tool will call the photo storage function and send the photo to the module.

Output: The photos in the data storage module should be in ascending or descending order of time

Test Case Derivation: The photos in the database should be sorted

How test will be performed: Turn on the system. Then run the automatic testing tool, the tool will call the data storage module and pass the photos and parameter into the module. Then the module will sort the photos according to the create time.

4. IPR9-1

   Control: Automatic

   Initial State:The system is turned on.

   Input: The automatic testing tool will call the photo storage function and send the photo to the module.

   Output: The photos in the data storage module should be in ascending or descending order of objects IDs

   Test Case Derivation: The photos in the database should be sorted

   How test will be performed: Turn on the system. Then run the automatic testing tool, the tool will call the data storage module and pass the photos and parameter into the module. Then the module will sort the photos according to the objects IDs.

5.1.2 UI Interface Menu

Manual Testing

1. UIR1-1

   Control: Manual

   Initial State: Turn on the system and User interface show up in the laptop

   Input: User's manipulation to the user interface

Output: The graphical display to the user

Test Case Derivation: After user highlight a certain photo, the item on the display should change to another color.

How test will be performed: User turn on the system and user interface. Operate the system to highlight a object.

2. UIR2-1

   Control: Manual

   Initial State: Turn on the system and User interface show up in the laptop

   Input: User's manipulation to the user interface

   Output: The graphical display to the user

   Test Case Derivation: After user change the sorting method, the system should give the certain response

   How test will be performed: User turn on the system and user interface. Operate the system to change the sorting method.

3. UIR3-1

   Control: Manual

   Initial State: Turn on the system and User interface show up in the laptop

   Input: User change the signal strength of Wifi

   Output: The graphical display to the user

   Test Case Derivation: After user change the Wifi strength, the system should give the certain response

   How test will be performed: User turn on the system and user interface. Switch the network connection to a weak strength Wifi, the system will give an alert.

4. UIR4-1

   Control: Manual

Initial State: Turn on the system and User interface show up in the laptop

Input: User change the unplug the camera to insert a fault

Output: The graphical display to the user

Test Case Derivation: After user unplug the camera, the system should give the response on status identifier

How test will be performed: User turn on the system and user interface. Unplug the camera, the system status identifier will change.

5.2 Tests for Nonfunctional Requirements

5.2.1 Usability

1. APR1-1

   Type: Structural, Manual

   Initial State: The camera and its mount is connected to the laptop and stay stationary on a flat platform.

   Input/Condition: Launch the program normally and provide a physical impact during functioning

   Output/Result: No electronic components should be exposed and the shell should stay still

   How test will be performed: Test is proceeded during a normal working period where a physical impact will be applied to the hardware. The impact level is trying to simulate the damage that could possibily be done during normal functional period including dropping from the desk, over twisting the camera. After the impact, the appearance will be checked to see if the non-functional requirements APR1 and APR2 are satisfied. There will be also be a survey question regarding if the product is physically safe in the sight

2. EUR1-1

   Type: Functional, Manual

Initial State: The program are stored properly in a USB and hardware parts are ready to be assembled

Input: Users are asked to launch the program and connect the hardware

Output: Users are able to successfully finish the set up and ready to use the system without any assistance

How test will be performed: A group of people who do not have any electronics and coding background are asked to set up the system. Brief instructions regarding the functionality of the system will be given in advance and users will be asked to launch the program and connect the hardware. A survey question regarding the difficulty of setting up the system will be asked at the end of test. A score from 0 to 10 will be used to show user's experience. The majority of users should have no serious difficulty finishing the set up.

3. EUR2-1

Type: Functional, Manual

Initial State: The program are set up properly and ready to use

Input: Users are asked to use the program to find assigned object

Output: Users are able to enter inputs into correct fields and proceed the function

How test will be performed: A group of people who do not have any electronics and coding background are asked to use the system with brief instructions being told in advance. The number of people who successfully find the assigned object with the program will be noted. A survey question regarding the difficulty of using the program will be asked at the end of test.A score from 0 to 10 will be used to show user's experience. The majority of users should be able to clearly see input boxes and enter corresponding inputs.

5.2.2 Performance

1. SLR1-1

Type: Functional, Manual

Initial State: The program are set up properly and ready to use, no further settings are required.

Input: Information of the object is entered properly

Output: The response time of the system to show the location of the object should be less than 5 second

How test will be performed: Users will be asked to use the system to find different objects in the same assigned area and tester will record the time the system response. Then users will enter the same searching information while the objects are placed into another area (objects are all placed into the assigned area). Distance and number of items in the environment will be adjusted for each test condition. For objects with the largest distance and placed into the messiest environment , the response time must be below 5 seconds.

2. SCR3-1

   Type: Functional, Manual

   Initial State: The program are set up properly and ready to use, no further settings are required.

   Input: Information of two objects at the edge of the assigned area will be entered

   Output: The rotation speed where the camera rotate from one edge to the other is slow enough

   How test will be performed: Users will be asked to enter information of two objects at the edge of the assigned area one after another. As the area is assigned and the response time is restricted to less than 5 seconds, the speed that the camera rotate from edge to edge will be the maximum speed can be applied. The maximum rotation speed must be slow enough such that it will not cause any security concern.

3. PAR1-1

   Type: Functional, Manual

   Initial State: The program are set up properly and ready to use, no further settings are required.

Input: The targer object will be moved one small step at a time

Output: The location value displayed should always be whole number

How test will be performed: Users will be asked to enter the information of one object and the tester will adjust the location of the object one small step at a time and observe the location value displayed in the program. Any value displayed in the program must be whole number and time value must be displayed with an accuracy of minute.

4. RAR1-1

   Type: Functional, Manual

   Initial State: The program are set up properly and ready to use, no further settings are required.

   Input: Part of the object to be searching for will be placed outside the assigned area

   Output: The device should return NOT FOUND if the portion of the object showing in the camera is not representative enough

   How test will be performed: User will be asked to enter the information of one object where part of it is not in the camera. As the rotation angle is restricted, the camera must not over-rotate to an unexpected angle. If the partial information is not representative, the program should return NOT FOUND.

5. RFR2-1

   Type: Functional, Manual

   Initial State: The program are set up properly and ready to use, no further settings are required.

   Input: wrong parameters will be entered into input boxes

   Output: The program will return error messages

   How test will be performed: User will be asked to enter wrong parameters in corresponding input boxes. For example, input number in color box. The program must present a error message notifying the user that wrong parameters are entered and clear all the input boxes.

5.3 Traceability Between Test Cases and Requirements

Requirements	Tests
IPR1	IPR1-1
IPR2	IPR2-1
IPR3	IPR3-1
IPR4	IPR4-1
IPR5	IPR5-1
IPR6	IPR6-1
IPR7	IPR7-1
IPR8	IPR8-1
IPR9	IPR9-1
UIR1	UIR1-1
UIR2	UIR2-1
UIR3	UIR3-1
UIR4	UIR4-1
APR1,APR2,SCR1	APR1-1
EUR1,LER1,LER2	EUR1-1
EUR2,UPR1,ACR1	EUR2-1
SLR1	SLR1-1
SCR3	SCR3-1
PAR1,PAR2,PAR3	PAR1-1
RAR1	RAR1-1
RFR2	RFR2-1

6 Proof of Concept Testing

Before any serious development of the game starts, a proof-of-concept test would be carried out to show the undertaking is feasible. The remaining of this section describes the proof-of-concept test in detail.

6.1 Significant Risks

The successful completion of the project depends on overcoming the following significant risks:

1. In order to satisfy the functionality of human detection, the camera must be able to rotate and follow the movement of the user. The risk is to make the user located in the center of the screen all the time. Otherwise, the observation may behave poorly and unclearly if the user is out of the screen or located at the edge of the screen.

2. SmartVault is used to keep the items of users or to find out when and what gets lost or missed. The most important part of the project is to identify the differences through the compare between two images.

6.2 Demonstration Plan

The prototype will try to test human detection firstly. Three cases will be applied for the proof-of-concept testing.

1. test 1
   Type: Functional, automatic
   Initial State: User stays in the center of the screen
   Input: Null
   Output: The camera does not rotate, and the user is identified with a yellow rectangle shown in the screen.

2. test 2
   Type: Functional, dynamic, automatic
   Initial State: User stays at the edge of the screen
   Input: Null
   Output: The user is identified with a yellow rectangle shown in the screen, then the camera begins to rotate until the user is in the center of the screen.

3. test 3
   Type: Functional, automatic
   Initial State: User is out of the screen

Input: Null
Output: The camera does not rotate, and the user cannot be identified on the screen.

4. test 4
   Type: Functional, dynamic, automatic
   Initial State: User is in the screen
   Input: User randomly walk around the room
   Output: The camera rotates and follows the user, and the user is identified with a yellow rectangle shown in the screen. In addition, the location of the rectangle also moves depending on the movement of the user.

The prototype will try to test item identification secondly. Three cases will be applied for the proof-of-concept testing.

1. test 1
   Type: Functional, static, automatic
   Initial State: Null
   Input: Two identical images with no difference.
   Output: Nothing to do with SmartVault.

2. test 2
   Type: Functional, static, automatic
   Initial State: Null
   Input: Two images with the same item, but the location of the item is changed.
   Output: The difference of the location is identified. SmartVault will record these change and upload to the local file.

3. test 3
   Type: Functional, static, automatic
   Initial State: Null
   Input: Two images with different items, but the locations of the items are the same.

Output: The difference of the items is identified. SmartVault will record these change and upload to the local file.

4. test 4
   Type: Functional, static, automatic
   Initial State: Null
   Input: A image with 6 small items.
   Output: SmartVault will identify each item with a yellow rectangle around it. In other words, there should be 6 rectangles in total.

5. test 5
   Type: Functional, dynamic, automatic
   Initial State: Null
   Input: A image with 1 small item and it will keep moving during the test.
   Output: SmartVault will identify this item with a yellow rectangle around it. In addition, the rectangle will move depending on the movement of the object

7 Unit Test Description

7.1 Unit Testing of Internal Functions

In order to build unit tests for the internal functions of the program, certain modules which can return values could be tested. This will involve choosing proper method and giving them input values. Through comparing what they are expected to output and the actual outputs, a series of unit tests can be created. These unit tests will choose both proper inputs and those could generate exceptions. The acceptable range for the unit test could be determined through testing the output with exceptions. There is no need for stubs or drivers to be applied for the testing. Since the tests have been already done by the individual classes. Some coverage metrics will be shown to measure how much of the code had been covered for testing purposes. The goal is to reach as much as possible to ensure that all functions related

to various modules get tested adequately. The expectation of the coverage percentage of unit tests is set to 80 percent.

7.2 Unit Testing of Output Files

In order to create unit tests for the output files, the quality and accuracy of the output images will be considered for testing. This will involve creating certain situation and compare the actual results with the expected ones. For the saved images in the output files, they will be rated depending on how similar they are with the expected images. Only those output images being rated with a score larger than 95 percent will be considered as qualified. On the other hand, if the score does not meet 95 percent, the test case fails and the developer team should analysis the reason behind it and redo the test case. In addition, some extreme cases will be applied to the unit testing process, to measure the maximum and minimum acceptable range for the unit tests.

8 Appendix

8.1 Usability Survey Questions

1.Do you feel unsafe when you see the product in the first sight?(for example, any exposed electronics and sharp corner that makes you feel uncomfortable

0 —— 1 —— 2 —— 3 —— 4 —— 5 —— 6 —— 7 —— 8 —— 9 —— 10
0 represents very unsafe 10 represents very safe

2.Do you have difficulty when setting up the system (launch the program and connect the hardware)?

0 —— 1 —— 2 —— 3 —— 4 —— 5 —— 6 —— 7 —— 8 —— 9 —— 10
0 represents very difficult 10 represents very easy

3.Do you have difficulty when using the program? (knowing where to put corresponding inputs and found interface is easy to understand)

0 —— 1 —— 2 —— 3 —— 4 —— 5 —— 6 —— 7 —— 8 —— 9 —— 10
0 represents very difficult 10 represents very easy