Çankaya University

Faculty of Engineering

Computer Engineering Department

Test Plan Document

Ceng 408

Innovative System Design and Development II

Advisor: Dr. Serdar ARSLAN

Prepared By	ID
Batuhan DİLEK	**201911023**
Görkem KARABAY	**201811039**
İbrahim Efe ERER	**201911403**
Zeynep Tulya AYTEKİN	**201811008**
Hasan Mert YILDIRIM	**201911071**

1. Introduction

1.1 Version Control

Version No	Description of Changes	Date
1.0	First version	April 09, 2023
2.0	Second version	April 15, 2023
3.0	First release	May 17, 2023
4.0	Second release	June 8, 2023

1.2 Overview

GenEye is a Deep Learning based, real time image processing smartphone application that helps Visually Impaired Users to find some objects and warn Users about obstacles on their way. The use cases of the project had been stated on the SRS document. The characteristics of the GenEye will be tested on the system in various settings.

1.3 Scope

In this document, we included the test plan of the use cases, test cases according to the test plan, and test design specifications. Project's features will be tested in different settings and environments.

1.4 Terminology

Acronym	Definition
SRS	Software Requirement Specification
SDD	Software Design Description
GUI	Graphical user interface
Machine learning	Machine Learning is a branch of Computer Science and Artificial Intelligence which tries to imitate human learning using algorithms.
API	Application Programming Interface

2. Features to be Tested

In this section we will provide a brief explanation of test cases for the characteristic features of our project. For each of these test cases we will provide a test design specification given at the end of this document.

2.1 Application's UI Interactions

GenEye's UI has 2 different types of interactions. First, choosing the intended mode on touching the system. Secondly, telling the intended object's name to the voice input screen of the application.

2.2 Correct Labeling

GenEye is a project that directs users to objects or makes users avoid obstacles. Therefore, it is vital to correctly label found objects. In our application, thresholds of the found objects include a name tag. Tester should seek for objects or obstacles to see if the application is labeled correctly. We tested the labeling throughout the development step. The program should find and correctly label objects and obstacles. For the sake of object detection, if there are no data on the given input, the program should notify the user that there is no such object. For obstacle detection, the program should inform the user about what kind of obstacle is in front of them.

2.3 Voice Input/Output

At the beginning the user will say the name of the item they are looking for, If there is an object of interest in the frame GenEye produces a voice alert to notify the user about the presence of the object. And after the desired object is on the screen, it will describe the user to the desired object. When the user is selecting the obstacle detection mode GenEye will inform the user about the obstacles on their paths, and the user should be able to provide voice input to select the mode of interest and get the outputs according to it.

2.4 Performance Tests

2.4.1 Latency

In Obstacle Detection and Object detection modes GenEye's system needs time for its calculations on image processing due to this system's notification of image can be varied in time interval. While we tested this latency test we also observed different events, such as; Testing with different threads, different objects, multiple objects, etc.. Our tests showed us that our latency is 0.4 seconds between frames, this latency can be observed in User's view as 0.8 to 1.1 seconds.

2.4.2 Ram Usage

As we mentioned before, GenEye needs to use multiple parts of User's smartphone, such as; camera, microphone, speakers. All these hardware usage and inner system analysis in image processing also affect smartphones' RAM. In our tests we tested with multiple environments that have different hardware parts. GenEye's Ram uses different amounts of RAM in different modes. In the main menu, we saw 5 mb of RAM but in Obstacle Detection and Object Detection modes this results increases to 237 mb.

2.5 Android Version Test

During our progression in GenEye, we use various emulators and android smartphones to test our project. GenEye is made with the idea of multiple Android versions. our application can work with Android 6 (Android Marshmallow) as most previous version and Android 15 (Vanilla Ice Cream).

3. Features not to be tested

After our conversations with Professor Gül TOKDEMİR, we did our tests with visually impaired people. We asked the Professor's suggestion to visually impaired people and they said it was not necessary. The suggestion was that while guiding them the location of the object they wanted, GenEye would also tell them the obstacles on their path. But she said this feature is unnecessary since there are walking sticks. Besides, one of the unnecessary features is to calculate the distance of any object. The solution to this is simple, an estimated proximity calculation can be made according to how far below the screen any object is. Another unnecessary attribute is unusual items. Thanks to modern design and architecture, the strangest possible variation of all kinds of items has been produced, making it difficult for GenEye to recognize them. In addition, it is unnecessary to measure the estimation sensitivity of the application if there is more than one type of search on the screen or more than one of any type.

4. Item Pass/Fail Criteria

To be able to get success on this project, GenEye should detect objects and obstacles, and notify the user when the detected object is found and obstacles are close to User. If any of the features are mentioned in Section 2, we will consider the test as a success.

5. Test Design Specifications

5.1. Hardware Tests (HT)

5.1.1. Screen Interaction (HT.SI)

User should use GenEye with ease and to achieve this application should understand users' tap inputs.

Test Case ID	Requirement	Priority	Description
HT.SI		High	Providing screen interaction

5.1.2. Notification System (HT.NS)

One of the key features of GenEye is the user notification system. Application should notify the user in Object Detection mod and Obstacle Detection mod when it finds the objects.

Test Case ID	Requirements	Priority	Description
HT.NS		High	Providing notifications

5.1.3. Camera API (HT.CAM)

GenEye has to have a camera connection to get real time video input to work in Object and Obstacle detections.

Test Case ID	Requirements	Priority	Description
HT.CAM		High	Interacting with camera API

5.1.4. Thread Control (HT.TH)

GenEye has its own thread control system which user can change if they want to take more frequent outputs.

Test Case ID	Requirements	Priority	Description
HT.TH	First Release	Low	Regularizing thread control

5.1.5. Voice Input API (HT.VI)

In this project most of our targeted users are visually impaired people. Therefore, we should consider their input choices, such as voice input.

Test Case ID	Requirements	Priority	Description
HT.VIAPI	First Release	High	Providing voice input

5.2. User Interface (UI)

5.2.1. Choosing Modes (UI.CHO)

User should make a choice between object detection and obstacle detection modes. When the Object Detection Mode button is pressed the application needs an input in order to be configured for searching an object.

Test Case ID	Requirements	Priority	Description
UI.CHO		High	Provides mode choosing

5.2.2. Changing Modes (UI.CHA)

User should be able to switch between the modes by pressing the return button of the android phone.

Test Case ID	Requirements	Priority	Description
UI.CHA		Medium	Provides mode changing

5.3. Object Detection Mode (OBDM)

5.3.1. Detecting The Object (OBDM.DTO)

GenEye has the property which is responsible for detecting objects. This property should detect the intended object which is given by the user.

Test Case ID	Requirements	Priority	Description
OBDM.DTO		High	Detecting objects.

5.3.2. Labeling Detected Objects (OBDM.LDO)

GenEye has the property which is responsible for labeling the intended object. This property should determine the intended object's label.

Test Case ID	Requirements	Priority	Description
OBDM.LDO		High	Providing labeling property

5.3.3. Notification for Object's Position (OBDM.NOP)

GenEye has the property which is responsible for giving notification about the object's location. This property should give notification whilst the determined object is residing on the screen.

Test Case ID	Requirements	Priority	Description
OBDM.NOP		High	Providing notification for object's location

5.4. Obstacle Detection Mode (OSDM.DTO)

5.4.1. Detecting The Obstacles (OSDM.DTO)

GenEye has the property which is responsible for detecting the obstacles. This property should detect the obstacles regarding their level of threat for the user.

Test Case ID	Requirements	Priority	Description
OSDM.DTO		High	Providing detection for current obstacles

5.4.2. Labeling Detected Objects (OSDM.LDO)

GenEye has the property which is responsible for labeling the obstacles. This property should determine the labels of the detected obstacles.

Test Case ID	Requirements	Priority	Description
OSDM.LDO		High	Provides labeling for objects

5.4.3. Notification for Obstacles Position (OSDM.NOP)

GenEye has the property which is responsible for giving notification about the locations of the obstacles. This property should give notification whilst the determined obstacle is residing on the screen. This should be applied regarding their level of threats.

Test Case ID	Requirements	Priority	Description
OSDM.NOP		High	Notifies for locations of the obstacles

6. Detailed Test Cases

Test Case ID	HT.SI
Purpose	Getting input from User to choose or change modes, change processor settings.
Requirements	-
Priority	High
Estimated Time Needed	5 minutes
Dependency	No Dependency
Setup	Open GenEye application OR Open “Object Detection” mode OR Open “Obstacle Detection” mode
Procedure	1. tap any button on main menu, “Object Detection” mode and “Obstacle Detection” mode 2. Wait for buttons intended change
Cleanup	Exit the application.

Test Case ID	HT.NS
Purpose	Providing notification for the guidance of the user both for object detection and obstacle detection modes
Requirements	-
Priority	High
Estimated Time Needed	5 minutes
Dependency	No Dependency
Setup	Open “Object Detection” mode OR Open “Obstacle Detection” mode
Procedure	1. Choose an object 2. Find the Object 3. Try different angles of the screen 4. Wait for notification OR 1. Find an obstacle 2. Try different angles of the screen 3. Wait for the notification
Cleanup	Exit the application.

Test Case ID	HT.CAM
Purpose	Interacting with camera interface
Requirements	-
Priority	High
Estimated Time Needed	5 minutes
Dependency	Smartphone has a camera module
Setup	Open “Object Detection” mode OR Open “Obstacle Detection” mode
Procedure	1. Open “Object Detection” or “Obstacle Detection” mode 2. Choose an object (Depends to the chosen mode) 3. Wait for the camera screen to open
Cleanup	Exit the application.

Test Case ID	HT.TH
Purpose	Rearranging the threads
Requirements	-
Priority	Low
Estimated Time Needed	5 minutes
Dependency	Smartphones have multiple threads in their system.
Setup	Open “Object Detection” or “Obstacle Detection” mode.
Procedure	1. Open chosen mode. 2. Open slider menu which below of the screen 3. Change the number of threads with minus and plus buttons.
Cleanup	Exit the application.

Test Case ID	HT.VI
Purpose	Getting User's voice input
Requirements	Open "Object Detection" mode.
Priority	High
Estimated Time Needed	10 minutes
Dependency	Smartphone has to have microphone module
Setup	Open "Object Detection" mode
Procedure	1. Press the microphone button. 2. Say intended objects name 3. Wait for the feedback a. try again on failed try
Cleanup	Exit the application.

Test Case ID	UI.CHO
Purpose	Choosing the intended mode
Requirements	-
Priority	High
Estimated Time Needed	5 minutes
Dependency	-
Setup	Open The application
Procedure	1.Choose intended mode
Cleanup	Exit the application.

Test Case ID	UI.CHA
Purpose	Changing between modes
Requirements	-
Priority	Medium
Estimated Time Needed	5 minutes
Dependency	-
Setup	Choose a mode
Procedure	1. Press smartphone's back button
Cleanup	Exit the application.

Test Case ID	OBDM.DTO
Purpose	Detecting the intended object on the screen
Requirements	Open "Object Detection"
Priority	High
Estimated Time Needed	1 hour
Dependency	HT.CAM
Setup	Open "Object Detection" or "Obstacle Detection" mode
Procedure	1. Choose intended object 2. Find the object 3. Await for boundary box around the intended object on the screen
Cleanup	Exit the application.

Test Case ID	OBDM.LDO
Purpose	Finding the right item in object detection.

Requirements	Image process model
Priority	High
Estimated Time Needed	15 minutes
Dependency	Starting “Object Detection” mode
Setup	1. Open “Object Detection” mode 2. Choose an object 3. Find the an object
Procedure	1. Find the intended item 2. Wait for boundary box around the object on the screen 3. Check the Label on the boundary rectangle
Cleanup	Exit the application.

Test Case ID	OBDM.NOP
Purpose	Output a notification about the intended object’s position.
Requirements	● OBDM.LDO ● HT.NS
Priority	High
Estimated Time Needed	45 minutes
Dependency	-
Setup	Open intended mode.
Procedure	1. Find intended object or obstacle 2. Try different angels of the screen 3. Await for the object's position notification.
Cleanup	Exit the application.

Test Case ID	OSDM.DTO
Purpose	Detecting obstacles on the screen
Requirements	Object detection model
Priority	High
Estimated Time Needed	30 minutes
Dependency	HT.CAM
Setup	Open “Object Detection” mode
Procedure	1. find an obstacle 2. wait for obstacle’s boundaries on the screen
Cleanup	Exit the application.

Test Case ID	OSDM.LDO
Purpose	Labeling right obstacles on the screen
Requirements	-
Priority	High
Estimated Time Needed	1 hour
Dependency	OSDM.DTO
Setup	Open “Obstacle Detection” mode
Procedure	1. Find different types of obstacles 2. Test for boundaries on the screen
Cleanup	Exit the application.

Test Case ID	OSDM.NOP
Purpose	Notifying the User about obstacles position
Requirements	-
Priority	High
Estimated Time Needed	30 minutes
Dependency	● HT.CAM ● OSDM.NOP ● OSDM.DTO
Setup	Open “Obstacle Detection” mode
Procedure	1. Find an obstacle 2. Try different angels on the screen 3. Await for notification about obstacle’s position
Cleanup	Exit the application.

7. Test Results

Test Case ID	Priority	Result
HT.SI	High	Pass
HT.NS	High	Pass
HT.CAM	High	Pass
HT.TH	Low	Pass
HT.VIAPI	High	Pass
UI.CHO	High	Pass
UI.CHA	Medium	Pass
OBDM.DTO	High	Pass
OBDM.LDO	High	Pass
OBDM.NOP	High	Pass
OSDM.DTO	High	Pass
OSDM.LDO	High	Pass
OSDM.NOP	High	Pass

8. References

[1] GenEye SRS:

https://github.com/CankayaUniversity/ceng-407-408-2022-2023-Surroundings-Detection-System-for-Visually-Impaired-People-GenEye/wiki/Software-Requirement-Specification

[2] GenEye SDD:

https://github.com/CankayaUniversity/ceng-407-408-2022-2023-Surroundings-Detection-System-for-Visually-Impaired-People-GenEye/wiki/Software-Design-Description