TED UNIVERSITY

CMPE 492

Virtuanance

Test Plan

Members

Hakan Ahmet Tekin, Cem Tırpanlı, Ali Can Keskin, Doğukan Terzi

Supervisor

Tolga Kurtuluş Çapın

Jury Members

Aslı Gençtav, Venera Adanova

Table of Contents

1. Introduction

Virtuanance is a virtual reality technology that aims to help with remote maintenance by establishing a digital environment that connects peers (professionals and technicians). Users can build a digital environment around reality using a Head-Mounted Device (HMD) and use the features to complete maintenance tasks. Both audial and visual contact between users will be supported by the project. The project allows users to import 3D models of machinery that needs maintenance so that they can communicate with a digital model during operations to quickly recognize and fix problems.

These models will be entirely interactive, with smaller sections that can be separated from one another. The models would also be used to provide visual cues to the technician to help him locate locations quickly. To identify physical equipment and provide the same visual cues on top of the actual machine, tracking algorithms will be used. During the audial service, users would be able to connect with one another. Users' hands will be sensed and used as input devices for improved usability, and we plan to eliminate all hand-held controllers to enable users to seamlessly communicate with the real and digital worlds.

The test plan has been created so that every member of the team has common acknowledgment and sense of the test that will be applied to the project. With this document we aim to identify the test deliverables and what is deemed in and out of scope clearly for all the members of the team.

2. Objectives and Tasks

2.1. Objectives

The main purpose of this test plan is to deliver a fully working software that is satisfying all the functionalities that it was intended to. We aim to successfully find out and eliminate all bugs before initial release. There will be no special tool that will be used for testing and debugging tools that are provided by the IDEs (Visual Studio & Visual Studio Code) will be used for debugging.

While testing every member of the team will be both the customer and the tester in this project. Testing with AR/VR devices will be conducted by one of Cem TIRPANLI or Hakan Ahmet TEKİN. This is because we are experiencing a pandemic and the devices can be given to these two members of the team. The tests that do not require any VR/AR can be conducted by any member of the project.

We aim to assure that the project meets all of the requirements and satisfies the use case scenarios. The primary objective of ours is to be able to deliver a software that satisfies all the quality metrics that we had mentioned on previous reports.

2.2. Tasks

With this document, first, we will analyze the project and identify the components of the software that will be tested. Secondly, we will be determining the scope of the testing. After that, we will be identifying and planning the testing strategies that are suitable for this project. Then, we will make the environment to be ready for testing. These tasks will help us deliver a project that will meet the quality and other requirements that we had already identified in previous reports.

3. Scope

3.1. In Scope

a) Security: By using network package capturers such as Fiddler and WireShark, we will trace the network packages transferring between client and servers to determine if the data transfer is observable by third party or not.

b) Reliability: System will be tested under several conditions to ensure it will be working on various environmental status and will not be bothered unless there are major obstacles interfering with the system from outer environment.

c) Performance: There will be stress tests performed on the system to ensure that system will at least work in a capable way if it is performing major computational processes.

d) Usability: There will be “black-box” tests performed to certain individuals to ensure that the UI and the software itself is easy to understand and easy to use.

e) Network: All the connections and the data transfer between the clients will be observed if there is any package loss high enough to interfere with the running of the software.

f) System Integrity: As different libraries and components are used in the software, their integrity and performance while running together will be tested and they will be kept under maximum performance.

g) Interactions: Most of the UI factors will be tested in a “white-box” testing manner so that the testers will know what button and what interaction will trigger the specified event. After the factors will be tested, there will be “black-box” testing phase to ensure the capability of buttons.

3.2. Out Of Scope

a) Scalability will not be tested.

b) Server performance will not be tested.

c) Local and web storage related complexities will not be tested.

4. Testing Strategy

4.1. Unit Testing

Unit testing will be conducted by the developers during the development process. It will be done in order to ensure that the necessary functionalities and the code coverage has been achieved. Recognition and Tracking, Connection, Communication, File Importing and Sharing, Leap Motion Controls are some of the parts of the software that will be put to Unit Testing. The purpose of this tests are making sure that all components are working as intended on their own and they are ready to be tested in integration testing. Ideally all of the known bugs should be eliminated. But it is acceptable if the remaining bugs are not making one or more functionalities completely useless. So if a bug is only visual or will be faced when doing something that is out of ordinary use of the program, then it may be ignored if there is not enough time to spend eliminating that bug before the final release.

4.2. System Testing

There will be black box and white box tests performed on the software to ensure it works in the intended way. The tests will be done after the fully integration of the components.

White Box Testing: After the software is finalized in terms of fully integrating all the components together, developers will test the software according to written scenarios. After all the scenarios are complete, it will be considered as white box testing is complete.

Sample scenario: Technician A does not know how to fully fix the component B. After plugging their HMD device and leap motion, they connect to the expert C. Expert C, by watching the environment from technician A's eyes, creates a visual feedback on showing what to do to technician A. After receiving the visual feedback from expert C, technician A terminates the connection and unplugs the HMD device and leap motion.

4.3. Integration Testing

4.3.1. Data Transfer

As an integration part, one of the data transfers of the project is whether the data is transferred to each other when the connection between professional and technician is established. The integrity and accuracy of the data from technician to professional should be tested in the integration part. Data transmission between Professional and technician in the project is required to maintain a machine. Because the technician will do business in line with the directions from professional. The data transferred to each other on how to test is few and specific. When all these transfer modules are manually tested and seen to work, it means that the data transfer of integration test is positive

4.3.2. Hardware

There are many hardware elements in the project. In this test, it will be seen whether these hardware can work as a part of a whole and whether they can work in harmony with each other. Big bang testing will be performed to perform Hardware components testing. By bringing all the developed modules together, when they work together, the accuracy of the modules that make sense will be ensured, and the per unit method accuracy will be ignored.

4.4. Performance Testing

4.4.1. Definition

The performance testing will try to examine how fast the application can run during various tasks. If a task can be performed smoothly (without creating any lag or stuttering), the tests will be considered successful.

For Virtuanance, performance tests will focus on the performance of both technician and expert sides. The points for determining success are:

1) Is the VR experience smooth enough that user’s eyes do not notice any lag?
    a. Virtuanance should be running at minimum 90 frames per second to appear smooth in VR headsets. And at minimum 30 fps in normal displays. Virtuanance will be run with and without VR and a connection will be established between them.
2) Does LeapMotion Controller receive and translate input fast enough?
    a. Ideally, the leap motion refresh rate should be synchronized with the program’s fps (i.e. 90hz for 90 fps). But 60hz refresh rate for LeapMotion Controller will be considered successful as it creates a smooth experience.

3) Is there a clear network connection without delays?
    a. There will be constant data transfer between a technician and expert's clients. These transfers should not block execution of Virtuanance in any way.

4) How fast is object tracking in detecting sudden position changes?
    a. Vuforia engine is being used to track objects. It should be able to track medium sized objects without slowing down the execution.

4.4.2. Methodology

1) The first part will be done during a typical scenario of experts helping a technician. The VR user (technician) should be able to move his head for seeing the digital world smoothly without experiencing any kind of stuttering.
2) The LeapMotion Controller user should be able to use their hands to interact with Virtuanance's world. This must happen without any delays, it should happen as if it is in real-time.
3) The network connection will be tested by sending large files (of a digital model) over the connection. Also, numerous indicators (drawings, animations etc.) will be sent during a connection to test network performance.
4) The Vuforia engine will be tested to see its performance on tracking single and multiple objects at the same time using various markers.

5. Requirements

5.1. Hardware

1) VR Headset
2) Network Related Hardware (Modem, Router, Ethernet etc.)
3) A Computer (for usage without VR)
4) A Computer with relatively faster GPU and CPU (minimum NVIDIA GeForce GTX 970, Intel Core i5-4590) (for VR usage [1])
5) LeapMotion Controller
6) Video Camera

6. Features To Be Tested

There are several features determined at the beginning of the project. The first of these are tracking features. If we separate all these issues into topics, these are:

- Hardware-Software Integration Features
- Tracking Features
- UI Features
- Network and Communication Features

Hardware-Software Integration Features will be tested on issues such as the hardware components in the project can work by serving a common purpose, and the compatibility of software with hardware. In Tracking Features, under the Vuforia Engine, the system that works with the object recognition, marker or model base will be tested whether the object is compatible with the real object that is maintain. Both design and maintain tools need to be tested as UI Features. Since there are 2 different users in the project, there will be different UI designs for each and different tools to use these UIs. Testing all these tools is necessary to prove that the system is working correctly, as well as giving errors in any case. Last feature topic, Network and Communication Features, data transfer and communication between users should be tested. Since this project is a remote maintenance project, a clear communication must be established between professional and technician. Therefore, communication and changes in the object to be maintained are tested, and a user change should be communicated to the other user in a clear way.

7. Features Not To Be Tested

We can divide the features that do not to be tested in the project into hardware and software.

7.1. Hardware

HTC Vive is used as VR glasses in the project. Since we do not have other VR glasses other than the HTC Vive, it cannot be tested whether it will work with other glasses. Software to support other glasses has been integrated in the project. However, the results are not known as it has not been tested. On the other hand, there are many components in the project as Hardware. In the absence of any of these, what users will encounter will not be tested. The people who will use the project will need to have the hardware equipment to be written in the Final Report. However, since the project is a VR project, it is necessary to use the project with a computer to operate the VR glasses easily. It will not be tested on low-end computers and a version suitable for these computers will not be released.

7.2. Software

The project has been made suitable for windows users as an operating system. Since different operating systems will not be supported, how both hardware components and software components should work in other operating systems will not be tested. Therefore, people who will use the project are expected to have the latest version of the windows operating system, and other hardware components are expected to be installed and running on the device where they will run the project such as Leap motion Controller. Another software feature not to be tested is the case that the Tracking algorithm does not work due to the enormous size of the objects. This situation will be tested, and the project will be run with the objects that comply with the object size rule given in the "Final Report", since the objects that are too small or too large to comply with the tracking algorithm will cause the test phase of the project to be long and even the auxiliary tracking systems used.

Last, "SteamVR" application will be used to run the VR glasses in the project. Different VR integration programs will neither to be applicable to the project nor to be tested .

8. References

[1] https://www.vive.com/eu/support/vive/category_howto/what-are-the-system-requirements.html