**ENGI 3051 – Software Engineering Design II**

**Fall 2015**

**Stereoscopic 3D Reconstruction System**

Software Test Document

Contributing Members:  
Slim Babay, Antoine Stavro

Date Submitted: November 20th, 2015

# Approval

This document has been read and approved by the following team members responsible for its implementation:

Developers

|  |  |  |
| --- | --- | --- |
| **Print Name** | **Signature** | **Comments** |
| Slim Babay |  |  |

|  |  |  |
| --- | --- | --- |
| **Print Name** | **Signature** | **Comments** |
| Antoine Stavro |  |  |

Client

|  |  |  |
| --- | --- | --- |
| **Print Name** | **Signature** | **Comments** |
| Greg Toombs |  |  |

# Revisions

Table of Contents

[Approval 2](#_Toc436032367)

[Revisions 3](#_Toc436032368)

[Introduction 5](#_Toc436032369)

[Testing Approach 5](#_Toc436032370)

[Glossary 6](#_Toc436032371)

[Test Plan 8](#_Toc436032372)

[Features to be tested 8](#_Toc436032373)

[Features not to be tested 8](#_Toc436032374)

[Network Response time 8](#_Toc436032375)

[Fuzzing or unexpected HTTP requests 8](#_Toc436032376)

[Testing tools and environment 8](#_Toc436032377)

[Unit Test Cases 9](#_Toc436032378)

[Video streaming and image capture 9](#_Toc436032379)

[Camera Calibration and Reconstruction 9](#_Toc436032380)

[3D rendering 10](#_Toc436032381)

[Model Rotation (mouse and keyboard) 10](#_Toc436032382)

[Functional Tests 11](#_Toc436032383)

# Introduction

This software testing document serves to provide a detailed overview of how the software solution will be tested for quality for the client, Greg Toombs. The software product will be evaluated in satisfying the specified required as a 3D capturing and displaying application developed using Java and object-oriented methodologies.

The system will require two cameras that will display a video stream on the server.  The streaming will assist the user to setup the scene and angle of the object and allow them to can take a snapshot. 2 views will display the feed of the camera and a 3rd will display the 3D reconstruction upon capturing. The system will apply prebuild open-source stereoscopic feature algorithms matching to two 2D pictures and use the determined intrinsic and extrinsic factors of separate cameras in reconstructing a 3D model. The object will be sent to the client’s browser to be rendered and allow the user to control the observed angles by the use of their mouse for rotation.

If a user likes the 3D model, they will be able to save it onto their computer with the export button.  Conversely, the import button will allow the user to load and display a previously saved model onto the screen.

The software can be divided into two separate components: a backend that handles the 3-dimensional reconstruction from images acquired by two separate cameras and a frontend web application that provides a graphical user interface and displays the rendering of the 3-dimensional object produced by the backend. Additional detailed step-by-step descriptions of each use case from the previous requirements document are also provided for further explanation.

## Testing Approach

A test driven development approach has been applied in producing this software application with an emphasis on ensuring that system meets it’s design specifications, functional and non-functional requirements. Code has been tested throughout the development stages of the application, to find errors early on in the development process and reduce errors in later stages and time spent debugging problems.

The black box testing technique will be used in order to test the software product, testing is done for 2 major categories:

1. Deriving test cases from classes and boundary value analysis.
2. Functional analysis test cases.

## Glossary

Table - Glossary of terms pertaining to the domain of the project

|  |  |
| --- | --- |
| Term | Description |
| Focal length | The optical distance where light rays converge to form a sharp image of an object to the camera sensor at the focal plane in the camera. The focal length is one of the parameters determining the perspective of the image produced by the camera. |
| Perspective | The visual effect or difference in the apparent sizes of objects that are at different distances from the camera. Given 2 images taken at the same location, changing the focal length of lens will also change the angle of view and magnification of the image. |
| Depth of field | The distance that appears to be in focus in front of and behind the focal point of the lens. This is determined by the aperture, lens focal length, and the distance to the subject. |
| Calibration | Function served to form the scene view by projecting 3D points into the image plane using a perspective transformation of a known model. To simplify the process for the end user, most of this will be done during development with additional adjustments performed based on inter-image correspondence matching  Matrices of intrinsic and extrinsic parameters are produced to offset and normalize factors that would account for errors in stereoscopy. |
| Intrinsic parameters | Internal variables dependent of the source of the image. These include varying factors such a focal length, lens distortion and the principal point offset between different cameras. It creates a mapping between the pixel and camera coordinates in the frame. |
| Extrinsic parameters | External variables dependent on the position and rotation of camera. These include difference perspectives of the scene of the difference cameras and are required for stereoscopy. |
| Object rendering | The process or function of generating an image based on geometry of the 3-dimensional model and the viewpoint or perspective from which it is being viewed. |
| Point cloud | A point cloud is a set of data points (X, Y and Z) of a 3D coordinate system used to form the model and their respective colour. Points represent the external surface of the model and are derived from corresponding matches from images and their calculated location in space. |
| Aperture | A hole or an opening which plays a role in determining how much light reaches the image plane. In the context of this project, it is equivalent to the size of the image sensor of the specified camera. |
| Aspect ratio | Description of the proportional relationship between the width of an image and its height. In the context of this project it is a function of the height divided by its width. |
| Epipolar line | A line vector that corresponds points in one image to another. |
| Feature detection | Process or algorithm used for object recognition by developing key-points from a 2D image. Key points are usually acquired from high-contrast regions of the image, such as edges, with the aim to produce a collection of scale and rotational invariant features that can be used for their localization. |
| Feature matching | Process or algorithm used to for object recognition to find features that exist in more than one image that accounts for translation, scale, rotation, and other geometric distortions. |
| Disparity map | A representation, often rendered as a greyscale illustration, of the apparent pixel or feature differences between two images. |

# Test Plan

## Features to be tested

* Video streaming and Picture capture
* Camera Calibration
* 3D rendering
* Model Rotation (mouse and keyboard)

## Features not to be tested

### Network Response time

The network speed is out of our control.  The rendering should be good, however we won’t know for sure because we are not able to test the reliability of it.

### Fuzzing or unexpected HTTP requests

The System will not be tested for unspecific HTTP requests. It will be assumed that the normal use of the software consists of only utilizing the html form controls and allowing the web application in handling any requests for the user.

## Testing tools and environment

Several tools used for development include Google Chrome (V46) and Jetbrains intelliJ 15 will be used for testing. In addition we’ll be using JUnit to create Test Methods for our classes. Meshlab (V1.3.3) will be used to test the generated 3d model by the server independently of the client and in qualitatively assessing how 3D models are rendered in the client.

# Unit Test Cases

## Video streaming and image capture

**Purpose**

The purpose of this test is to check that the system is able to capture a video stream from each camera and that both frames on the server are synchronized with as little apparent delay as possible on the screen. The system will also be tested to ensure that clicking the capture button will properly take a snapshot of the object from each camera and display the still images in the image frames on the screen.

**Input**

The input will be the video stream from each of the cameras and the user button click.

**Expected Output**

The expected output will be a still picture of the object from two different angles in each of the frames.

**Pass Criteria**

The test will pass if each capture image has a clear images and have an appropriate synchronization.

**Test Procedure**

Move the camera around the object and take multiple pictures.  Also; we will check the sync timing of the system by taking a snapshot of a timer (counting down in real time) then check that each image frame has the same time displayed.  For example, if we take the snapshot of a stopwatch and ensure that the time captured in the image is the same in all frames.

## Camera Calibration and Reconstruction

**Purpose**

The purpose of this test is to check that each camera distortion matrix and fundamental and essential camera matrices determined are correct.

**Input**

The input will be the images taken from the cameras once the capture button has been pressed.

**Expected Output**

The expected output will be a proper representation of the two images combining to create a 2D model.

**Pass Criteria**

The test will pass if the 2D model has been calibrated successfully without any distortion.

**Test Procedure**

Take a snap shot of a chess board with each camera, which will created a 2D representation of the object.  We’ll check for any distortion on the model, and if there is, adjust the distortion matrix to fix it.  Additionally, we will capture an image of a cube and determine a 7 point image from the corners of the object. Using those features, we will attempt to manually calculate a 3x3 distortion matrix for each camera to verify its values.

## 3D rendering

**Purpose**

The purpose of this test is to check that the reconstruction from the 2D to 3D model is done correctly with an appropriate response time.

**Input**

The input will be the render button which will start the reconstruction process.

**Expected Output**

The expect output will be a reconstruction of the object in 3D.

**Pass Criteria**

The pass criteria will be an accurate representation of the object in a 3D model with a smooth texture and the rendering process has a good response time.

**Test Procedure**

For the test procedure, a 2D object will be display on the screen, then the render button will be clicked to begin the reconstruction process from a 2D to 3D model.

## Model Rotation (mouse and keyboard)

**Purpose**

Test that the model rotates when the mouse is clicked on the object and moved along the screen.

Also, we’ll test the rotation of the model with the keyboard.

**Input**

The input will come from the left mouse button being clicked and held then moving it along the screen for the rotation.  The second input will be with the left, right, up, down arrows from the keyboard, and once are they are pressed it will rotate in the appropriate direction.

**Expected Output**

The 3D model will rotate in any direction the by using either the mouse or the keyboard.

**Pass Criteria**

The rotation is done successful with both peripheral and without any lag during the rotation.

**Test Procedure**

Once the 3D model is created, the rotation of the model will be done by left click and holding the mouse then dragged along the screen.   For the keyboard, each arrow button (left, right, up, down) will be clicked and held to check that it rotates in the right direction.

# Functional Tests

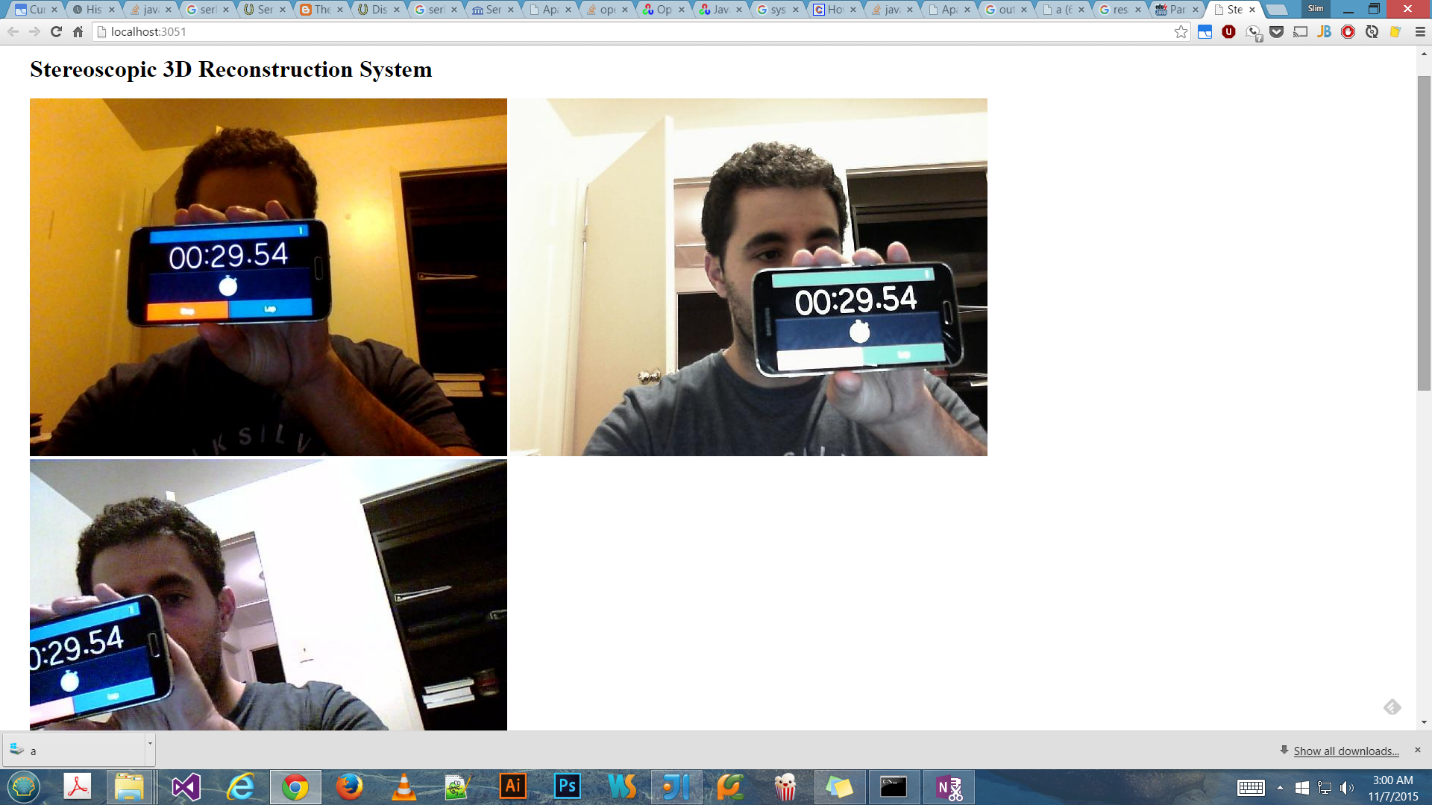
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Description | Preconditions | Postconditions | Procedure |
| 1. Camera calibration | Verify that camera calibration works and produces meaningful calibration matrices | 1.1. Video stream of each camera shown to user.  1.2. Chessboard is placed in the view of each camera | 1.1. Camera matrix is a 3x3 matrix with entries that are not all zero  1.2 Fundamental and essential matrices containing rotational and translation vectors of each camera are not all zero | 1. If values haven’t been set for each camera the system will search for a chessboard pattern in each camera feed  2. If the chessboard pattern is found it will begin a self-calibration procedure and capture data points as the board is moved around in the display.  3. The system will overlay a notification indicating the camera has been calibrated. |
| 2. 3D reconstruction and rendering | Verify that the system can export a 3D point cloud or mesh object and that the system can display a 3D object | 2.1 Cameras have been calibrated  2.2 Object has been placed in front of both cameras | 2.1 Point cloud has created and contains more than 1 value  2.2 client is sent mesh object  2.3 3D mesh is rendered and displayed in browser | 1. Capture button is clicked  2. Ensure 3D object is visible in browser  3. Rotate to confirm proper 3D interpolation |
| 3. Exporting | Verify that the system can send the client the exported 3D Model | 3.1 3D object has been reconstructed  3.2 export button has been enabled indicating client has the 3D model identification information | 3.1 a .obj file begins to download on client | 1. Reconstruct an object  2. Click Export |

# Appendix: Test Logs

## Log for Video streaming and image capture

**Test results:**

Stopwatch shows the system passed in displaying synchronized frames from each captured camera. The streaming had acceptable delay.

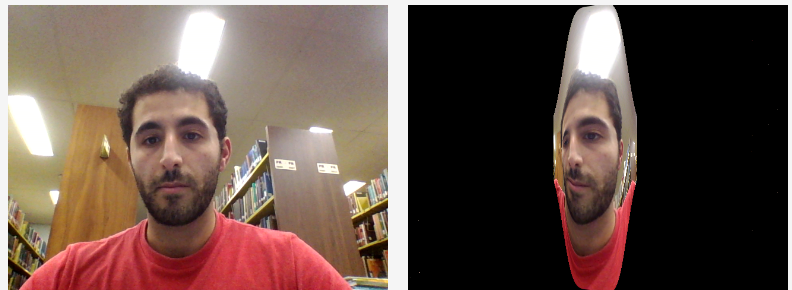


## Log for Camera Calibration

**Test results:**

Chessboard pattern was automatically detected by system and the information was used to produce distortion coefficients and camera matrices. Calibration failed as the rectified image was increasingly distorted.





**Incident reports:**

Chessboard was printed on a piece of paper which was structurally no solid. The material flexibility would allow bending that would be incorrectly perceived as lens distortion and cause the rectified image to change undesirably.

(This page left intentionally blank)