Synthetic Data Pipeline for Pose Estimation
Software Test Document

Team Members:
William Stern - western2019@my.fit.edu
Stephane Baruch - vjani2018@my.fit.edu
Nathan Pichette - npichette2018@my.fit.edu
Hanibal Alazar - halazar2019@my.fit.edu

Advisor:
Dr. Ryan White

1. Introduction

Objective:

The objective of this test plan is to explain the conditions and procedures necessary to ensure our synthetic data pipeline works properly. To do this we will list multiple testing ideas along with methods for completion and expected results.

1.1.1.

TC001 - Satellite model Input	
Description: Our API will allow for the user to choose from a list of satellite models which they want to be rendered into their video clip **Goal:** The system will pull the satellite object file from our catalog and insert it properly into blender with correct coloration and model features **Precondition:** User has access to the satellite model data and a proper path to the file is loaded into the system	
Test Steps: 1. Start application 2. Navigate to satellite dropdown 3. Choose specific satellite model 4. Input basic conditions for motion, light, and background 5. Press complete button 6. Find and examine rendered clip	**Expected Results:** 1. Rendered clip will show the correct satellite model with texture within the predetermined environment scenario

1.1.2.

TC002 - Satellite Flight path	
Description: The API will allow for the user to input a complex path that a satellite model will follow in a rendered clip that is returned to the user. **Goal:** Our pipeline will take in input for a complex path when a rendered clip is produced the satellite model should correctly follow the input. **Precondition:** User has access to the satellite model data and correctly formatted path to submit to the API	
Test Steps: 1. Start application 2. Navigate to flight path	**Expected Results:** 1. Rendered clip will show the satellite model moving correctly through

3. Enter a complex flight path 4. Input basic conditions for model used, light, and background 5. Press complete button 6. Find and examine rendered clip	predetermined environment conditions

1.1.3.

TC003 - Rotation	
Description: The API will allow for the user to add rotation in any direction to the satellite model. The rotation will be in addition to the path traveling. **Goal:** Our pipeline will take in input for a rotational pattern and apply that pattern to a satellite model while it travels along a path **Precondition:** User has access to the satellite model data	
Test Steps: 1. Start application 2. Navigate to rotation 3. Enter rotation direction 4. Input basic conditions for model used, light, background, and path 5. Press complete button 6. Find and examine rendered clip	**Expected Results:** 1. Rendered clip will show the satellite model rotating correctly while traveling on path through predetermined environment conditions

1.1.4.

TC004 - Lighting position and strength	
Description: The API will allow for the user to customize the location of the light source as well as the brightness **Goal:** Our pipeline will take in input for a light source's location and strength, a rendered clip is produced and the satellite model should be correctly lit based on specifications **Precondition:** User has access to the satellite model data and understanding of lightsource positioning in 3D space	
Test Steps: 1. Start application 2. Navigate to lighting section 3. Enter lighting location and strength	**Expected Results:** 1. Rendered clip will show the satellite model moving along path with correct shadowing and brightness

4. Input basic conditions for model used, path, and background 5. Press complete button 6. Find and examine rendered clip	

1.1.5.

TC005 - API Usability Test	
Description: This will test to make sure that the API will be usable by the target user. **Goal:** When given access to the API the user should be able to generate a sample video. **Precondition:** The user should be familiar with computer and computer programming. The user will also have access to documentation showing how the API works.	
Test Steps: 1. Give user API access and documentation about how to run the API 2. Instruct user to use API to develop a certain scenario 3. Observe what user does 4. Have one of us developers try to generate the same scenario 5. Compare user scene versus developer created string	**Expected Results:** 1. The user should be able to develop scenes that are close to what one of the developers can generate when given documentation.

1.1.6.

TC006 - Machine Learning Test	
Description: This will test to make sure that the generated videos are able to teach the machine learning model to how estimate pose **Goal:** When given generated video scenes for training the NETS lab should be able to estimate pose using their simulated test. **Precondition:** The user should have access to a computer with a GPU. The user should also have access to a test bench for satellite pose estimation.	
Test Steps: 1. Generate a lot of scenes and corresponding poses 2. Train the neural network using those scenes 3. Test the neural network using the	**Expected Results:** 2. The neural network should be able to correctly generate pose based on synthetic training data

satellite test machine.	

1.1.7.

TC007 - Multiple System Test	
Description: This will test to make sure the software will be compatible with different hardware configurations and operating systems. **Goal:** The program should be able to generate the same exact videos when given the same configuration file on different systems. **Precondition:** The user should be familiar with computer and computer programming. The user will also have access to documentation showing how the API works.	
Test Steps: 1. Prepare a Windows PC, a Mac PC, and a Linux os. 2. Install the software onto the operating systems. 3. Give the software the same configuration files and generate videos. 4. Compare the output videos	**Expected Results:** 1. The generated videos should be exactly the same when using something like cosine similarity to compare them.

1.1.8.

TC008 - Setup Test	
Description: Since this program is used on multiple computer systems, on laptops, and on the cloud, it would be great if it was easy to install and setup. **Goal:** The user should be able to install the software within 10 minutes on any operating system. **Precondition:** The user should be familiar with computer and computer programming.	
Test Steps: 1. Start timer 2. Download all the software and software requirements 3. Install the program 4. End timer when you can start generating a video	**Expected Results:** 1. The user should be able to install and start generating videos within 10 minutes on any common operating system.

1.1.9.

TC009 - Stress Test	
Description: The program should be able to warn users if the input will take a very long time to generate, and if there are any problems. **Goal:** When given parameters for a scene generation, the program should warn users if the generation will take more than 1 hour. The program should also warn users if there are potential problems such as intersecting objects. **Precondition:** The user should be familiar with computer and computer programming.	
Test Steps: 1. Generate a configuration file that should still generate but takes 5 hours or more and has intersecting objects. 2. When run using the configuration file, the API should start running but should warn users about potential problems.	**Expected Results:** 1. The users should be given a warning telling them what might go wrong with the generation. The warning should also give line numbers so the user can investigate and fix the problem.