Testing Analyser Document

Team 6: Harshika Jain, Sonu Guru, Subodh Sondkar, Shivam Nayak

# Acceptance Testing

* ACCEPTANCE TESTING is a level of software testing where a system is tested for acceptability.
* The major aim of this test is to evaluate the compliance of the system with the business requirements and assess whether it is acceptable for delivery or not.
* It is the last phase of software testing performed after System Testing and before making the system available for actual use.

In our case, acceptance testing was done by demonstrating the system to the client. All the functionalities were demonstrated to the client. We operated the front end of the web app and uploaded the image for object identification, and then processed it and showed the output file with the marked object. Then, we uploaded the point cloud for noise removal, processed it, and showed the output point cloud with the noise removed, and the point cloud of the noise itself. We processed the noise removed files again and showed the output point cloud of the objects with the objects and backgrounds separated.

# System Testing

* SYSTEM TESTING is a level of testing that validates the complete and fully integrated software product.
* The purpose of a system test is to evaluate the end-to-end system specifications.
* Usually, black box testing method is used at this particular level.
* It is the third level of software testing performed before acceptance testing.

Here, we test the product as a whole. We went through all the flows through the eyes of a user for each use case. We tried all the correct and incorrect input and output paths that a user might give, and raised appropriate responses for the same. Since we know about the product and its underlying code but are testing by using the user’s point of view, we are actually gray box testing.

# Integration Testing

* INTEGRATION TESTING is the process of testing the interface between two software units or modules.
* It’s focus is on determining the correctness of the interface. The purpose of the integration testing is to expose faults in the interaction between integrated units.
* Once all the modules have been unit tested, integration testing is performed.
* Integration Testing is the secondlevel of testing performed System Testing.

Different approaches of integration testing:

1. In bottom-up testing, each module at lower levels is tested with higher modules until all modules are tested. The primary purpose of this integration testing is, each subsystem is to test the interfaces among various modules making up the subsystem. This integration testing uses test drivers to drive and pass appropriate data to the lower level modules.
2. Top Down is an approach to Integration Testing where top-level units are tested first and lower level units are tested step by step after that. This approach is taken when a top-down development approach is followed. Test Stubs are needed to simulate lower level units which may not be available during the initial phases.
3. Big Bang is the simplest integration testing approach, where all the modules are combining and verifying the functionality after the completion of individual module testing. In simple words, all the modules of the system are simply put together and tested. This approach is practicable only for very small systems. If once an error is found during the integration testing, it is very difficult to localize the error as the error may potentially belong to any of the modules being integrated. So, debugging errors reported during big bang integration testing are very expensive to fix.
4. Sandwich is an approach to Integration Testing which is a combination of Top Down and Bottom Up approaches. In top-down approach, testing can start only after the top-level module has been coded and unit tested. In a bottom-up approach, testing can start only after the bottom level modules are ready. This sandwich approach overcomes this shortcoming of the top-down and bottom-up approaches.

Our frontend was made in django. All the four units are independent of each other in terms of code. In terms of data, noise removal, object extraction and object separation use the data in a pipeline fashion. The output data given by noise removal is used as input by object extraction, and that given by it is used by object separation.

As the units are independent in code (i.e. no cross calls), hence the big bang method was used, and merging wasn’t an issue. The debugging reports were taken from the frontend, which showed which unit the error belonged to, as all the units have their separate webpage.

# Unit Testing

* UNIT TESTING is a software testing technique by means of which individual units of software i.e. group of computer program modules, usage procedures and operating procedures are tested to determine whether they are suitable for use or not.
* The purpose is to validate that each unit of the software performs as designed where a unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output.
* Unit Testing of a software product is carried out during the development of an application.It is typically performed by the developer.
* It is the first level of software testing and is performed before Integration Testing.

Our backend has four units namely, object identification, noise removal, object extraction, and object separation. But each unit follows the same process. First, the location of the data in the file system is given. The unit finds the file in the file system. Testing was done for the case of wrong location, and appropriate errors are shown to the user.

Then, the location for the output data to be stored in the file system is given. Here, testing was done for non-existent paths, and for the folders whose access is restricted, an error is shown, otherwise the appropriate folder structure is made, and the output data files are stored there.