Testing plan

Project Title: Real-time Data Visualization Stimuli

1. Introduction

1.1. Project Overview

This project aims to enhance VPN Laboratory's software by developing a real-time data visualization function to display neuron activity, improving research efficiency and accuracy. The project includes developing a real-time data visualization function on a GUI extension of the existing software and developing an integrated software solution that minimizes manual operations, guarantees seamless functions, and improves efficiency and responsiveness. The goal is to automate processes and streamline the research process to contribute to the evolution of advanced autonomous driving technology and improve artificial vision system accuracy and reliability.

1.2. Scope of the test plan document

The scope of this document is to make a testing plan to ensure that the software meets the specified functional and non-functional requirements. The document will identify the testing activities, tools, and resources needed to verify the software's functionality, performance, and security. This document includes an introduction to the project, the test strategy, and the test execution. The plan will also identify the schedule for the testing activities, potential risks, and contingency plans to mitigate the risks. The testing plan will guide the testing process, ensure alignment with project objectives, and track and report on the testing progress and results.

2. Test strategy

2.1. Test scope

The objective of the tests is to ensure that the developed software performs the intended features. The testing procedure is structured into 4 distinct phases. Phase 1), Unit Testing, concentrates on certifying the proper and intended operation of the software's critical individual components, such as buttons and graphic display algorithm. Phase 2), Integration Testing, aims to detect defects and verify that the combined components function effectively as a cohesive system. We will utilize the bottom-up approach for this phase, starting with the bottom-level components (e.g. drop-down list, position of text fields) and gradually advancing upwards (e.g. data storage and GUI windows), simulating or stubbing higher-level components as necessary. Phase 3), System testing, involves live testing on a lab computer for the functionality of a real-time data analysis visualization program which ensures that the developed software can accurately analyze and present scripts containing data captured by the lab's acquisition machine during each experiment. Phase 4), Acceptance Testing, involves customers assessing the system to gauge its readiness for developers' approval and eventual implementation in their respective environments.

2.2. Testing report

The outcomes of the tests will be meticulously documented, capturing intricate details of each conducted test, testing environment, input data, expected and current output. The results for each test will be evaluated throughout the expected output and the current output. Comprehensive details can be found in the testing report table, located in the Appendix section.

2.3. Test assumptions

1. Each essential individual component functions as anticipated.
2. Integration tests can effectively visualize data from testing files.

3. During the initial system/deployment testing, some compatibility errors may be encountered; however, the developed software is capable of processing real data on the lab's acquisition machine.
4. During the final (second) system/ deployment testing, compatibility errors and other identified bugs are fixed and the software is ready for deployment on the lab's machines.

3. Test execution

3.1 Defect testing

Defect testing has been an integral part of the entire project development process. Tests are conducted to uncover errors during the development phase. For example, one defect was found while working on displaying data trial by trial. The data interpolation function displayed an error message and crashed the entire program. The test case used only one trial file, which illustrated how an edge case could lead to the unavailability of the interpolation function. To address this issue, we opted for a collaborative approach by engaging in peer reviewing and group discussions and solved the problem by conducting a different data processing method.

3.2 Unit testing

During the Unit Testing phase, several key components will be scrutinized.

In the stage of Milestone 1, we will test the units we built for debug mode as follows:
1) Verifying **independency of the debug mode** without the Psychtoolbox.
2) Verifying the functionality of the **created layout and the push button** on the GUI.
3) Checking that the generated heatmap data from the testing file is loaded line-by-line and stored with each click of **the push button**.

In the stage of Milestone 2, the units will be tested as follows:
1) Assessing the functionality of the **drop-down menu** to ensure different scripts can be selected and switched seamlessly after each choice.
2) Verifying if the **second script** (temporal frequency) could be matched and implemented with the same layout from the previous script after being chosen.
3) Testing the newly added **refresh button** to ensure the data saved from the last script is cleared on the GUI when switching to different scripts.
4) Confirming that the **debug mode** is activated only after the corresponding button is clicked.

3.3 Integration testing

In the testing phase, our team will adopt a bottom-up approach for integration testing. We will begin with the lower-level modules of the software and gradually move towards testing the top-level modules. To start, we will focus on integrating units 1-3 during the implementation stage. This involves verifying the software's ability to accurately read data and display it in the designated layout as a heatmap. The heatmaps should be updated each time the push button is clicked. Following that, a week later, we will conduct integration testing for units 1-4 during the optimization stage. This testing will ensure the functionality of the dropdown menu and refresh button with different scripts. Once these integration tests are completed, we will proceed to perform an overall integration testing process using real data on the lab machine. This step ensures that all software components integrate seamlessly and meet the specified requirements.

3.4 System / Deployment testing

We will conduct two rounds of system/deployment tests. The first test is scheduled for May 17, 2023, and will be a black-box live test, carried out during a real experiment analyzing and visualizing captured neuron data from files in collaboration with a PhD student who will be one of the end users without having knowledge of the internal workings of the system in the lab. During this test, we will identify compatibility issues and potential bugs related to functionality, usability, performance, and security while running the developed software on the lab's acquisition machines. The second system test will take place one week after the first, during which we aim to resolve all defects identified during the initial test and prepare the software for deployment on the lab's machines.

3.5 Acceptance testing

Before acceptance testing, we will establish clear acceptance criteria with our clients, such as displaying real-time graphs in the expected locations. In addition, the team will ensure a capable software environment to perform the expected tests in the client's laboratory. During acceptance testing, clients will use real input data on the lab machine to evaluate whether the software meets the required specifications for deployment. As the development team, we will provide technical assistance to the clients and actively and promptly address any defects that may occur during the acceptance testing phase to ensure quality standards and expected performance are met in the customer environment.

4. Testing plan schedule

To ensure the quality of our software, we have planned a series of tests over the next few weeks. Unit testing and integration testing for units in Milestone 1 will be conducted on May 10, 2023, followed by the first system/deployment testing on May 17, 2023. A week later, we will perform unit testing and integration testing for the units in Milestone 2, followed by the second system/deployment testing. Finally, acceptance testing will be conducted to ensure that the software meets the required specifications and performs as expected. These tests are crucial to ensuring that the software is of high quality and meets the needs of our users.

5. Risks

Table 1 lists the identified risks along with their potential impact on the project. Each risk has a defined trigger, which is the event that causes the risk, and a mitigation plan to alleviate the risk:

No	Risk	Impact	Trigger	Mitigation Plan
1	Data Loss	High	Software crash leading to the loss of valuable experimental data	Saving data in a hard drive before using it for any purpose
2	Delay	Medium	Potential delay of project deliverables resulting from miscommunication and wasted time spent on insignificant source files.	Incorporate time and communication management strategies and utilize mid-term breaks to expedite the project's progress

Table 2. Potential risks and mitigation strategy

Test Report Table (Appendix)

Environment	Status	Test description	Input	Expected output	Current output
1	Done by 9 May, 2023	Unit 1. Test if the program can execute without connecting to a laboratory Psychtoolbox machine.	N/A	The program executes smoothly and exits gratefully without connecting to the Psychtoolbox machine.	As expected
1	Done by 9 May, 2023	Unit 2. Heatmap layout and push button displays/functions correctly	N/A	The layout is shown on the upper-right corner, and the push button can display the testing graph upon clicking.	As expected
1	Done by 9 May, 2023	Unit 3. Display heatmap in mentioned layout by clicking the push button	Receptive field data	Upon clicking the push button, the software retrieves data and generates a heatmap, which is then displayed within the GUI's layout.	As expected
1	Done by 10 May, 2023	Integration test of Milestone 1	Receptive field data	Software reads data correctly and displays in the layout created as heatmap, the heatmaps should be updated once per click of push button.	As expected
2	Done by 17 May, 2023	1st system/deployment testing	Live data	The software captures live data and stores them in the GUI before displaying real-time data visualisation in the expected GUI position.	As expected
1	Done by 22 May, 2023	Unit testing and integration testing of Milestone 2	Receptive field data	Software should be able to read trial files one by one and display different types of graph according to user selections. The data stored should be reset when switching between different graphs.	As expected
2	Done by 24 May, 2023	2nd system/deployment testing	Live data	Same as above, but read in real-time laboratory data from Psychtoolbox through a data machine, and other identified issues from 1st system testing need to be addressed	As expected
1/2	Done by 31 May, 2023	Acceptance testing	Receptive field data/Live data	Software should be able to perform both milestones 1 and 2 requirements. The software is expected to be accepted by the clients.	As expected
Environment 1 - Mac Os v10.14.6 (18G103) and Matlab vR2021a Environment 2 - VPN Laboratory Machine				Responsible: Haoxian Wu, Yueran Wu Reviewer: Lin Cen Lin Jia Yi, Shengbin Wu	