System Test Plan

For

Turtle Tech

Team members: Christopher Angland, Laurel Dodson, Parker Brown, RoseEllen Hoke, William Wiemann

Version/Author	Date
Version 1	19 October 2022
Version 2	8 December 2022

Table of Contents

1. Introduction

1.1 Purpose

This document is a test plan for TurtleTech System Testing, produced by the System Testing team. It describes the testing strategy and approach to testing the team will use to verify that the application meets the established requirements of the business prior to release.

The purpose of the TurtleTech project is to produce a functioning neural network system on a Nvidia Jetson device that identifies aerial turtle images against non-turtle images in real-time. The neural networking system will only identify one specific animal at a time, it will not be able to differentiate multiple things at once.

1.2 Objectives

- Meets the requirements, specifications and the Business rules.
- Supports the intended business functions and achieves the required standards.
- Satisfies the Entrance Criteria for User Acceptance Testing.
- The neural networking system will identify one specific animal at a time
- The ERAU team will produce a functioning neural networking system by the end of the Fall semester.

2. Functional Scope

The Modules in the scope of testing for the TurtleTech System Testing are mentioned in the list attached below:

- Image Recognition Accuracy
- Image Processing Speed
- Program Size
- Portability to Board
- User Interface Legibility
- User Experience (Ease of Use)

3. Overall Strategy and Approach

3.1 Testing Strategy

Turtle Tech System Testing will include testing of all functionalities that are in the defined functional scope of section 2. The strategy will be evaluating these metrics first independently and then together. The testing method will at first be a white box testing method. The design and structure of the system will be known to the engineer who is running the tests. Eventually, testing will progress to implementing a black box method, hiding the design and structure of the system, and having a tester provide feedback based solely on the results. This will allow for the neural network to improve via machine learning, the working of which is easily suited to black-box testing, and for testers to more easily focus on the important characteristics of ID accuracy.

3.2 System Testing Entrance Criteria

The readiness of the system to begin the process of system testing can be characterized by having a running algorithm that takes in images from outside sources onto the Jetson, and returns sorted images with labels corresponding to the criteria by which images are to be categorized (which is defined in the program – in this case, the identification of turtles). The initial testing of the algorithm will take place on a local machine, but the final testing will take place on the Jetson hardware, after the program is implemented onto the Jetson board.

3.3 Testing Types

3.3.1 Usability Testing

Usability Testing will consist of verifying that analyzed images have selected areas containing likely turtles to test visibility for the user. Additionally, the interface for users to review images must be easily navigated, with the ability to return to a main menu and rapid, easy switching between different sorted images. Access to the images must be simple and launching the program itself must be a process with very few steps, in order to promote ease of use of the program for the user.

To quantify these requirements, the accuracy of the model will be the benchmark for the verification of neural network functionality. Currently the neural network accuracy is around 77%, so improvement of this figure is the goal of testing. For the customer usability of the system, all photos saved photos that contain turtles will be in a single folder and contain timestamps.

Hardware testing at this point has been resolved since the mid-flight shut down issue with the payload is no longer occurring. At first, ground tests were conducted to attempt to replicate the midflight shutdown. The most notable of these ground tests was a stress test in which the aircraft had the upper and lower panels on the payload, was left in the back of a car on a humid day. After this test still failed to cause the shutdown, overheating of the processor was ruled out as the issue. With the issue now being identified as a brownout, a secondary power source (12V LiPo battery) was provided to supply power to the payload. After this change, a test flight was conducted, and the payload did not experience a shutdown.

3.3.2 Functional Testing

The business requirements of the product are to identify turtles and save images of identified turtles. Identification of turtles is handled by the neural network, relying on visual recognition software to identify any turtles found in images provided by the software. After a turtle is recognized, the image is then saved with the timestamp of when the image was processed to be accessed later.

The business rules and conditions of this test are not a priority for this test. The test-bank of photos being provided to the turtle is not private, and currently the product is not being presented to the customer. In the future, the main rule is that the product meets the functional requirements.

3.4 Suspension Criteria and Resumption Requirements

This section covers the reasoning and requirements of circumstances to suspend testing of the system or a portion of it, and the necessary steps that must be taken prior in order to resume testing in such an occurrence.

3.4.1 Suspension Criteria

Testing will be suspended if hardware breakdowns unrelated to our system render the entire UAV into a nonfunctional state, as inflight testing would not be possible under such circumstances. Testing would also be suspended if further requirements are added, or previous requirements must be adjusted for the program to properly fulfill. Tetsing will be suspended if other extenuating circumstances mean the tests must be adjusted to accurately evaluate the product, and in any of these cases, returns to testing and modifications to the test plan are under the jurisdiction of the team.

3.4.2 Resumption Requirements

Testing will be resumed when the original issue requiring the suspension of testing has

been alleviated, resolved, or circumvented by actions of either the team or outside sources. If the hardware has been having a repeated problem and has since been repaired, and is no longer causing the same breakdown, testing can resume. If an incomplete requirement or additional task has been unfulfilled and has been addressed or added to in order to satisfy customer requirements, the testing can resume, and in any of these instances, resumption and potential adjustment to testing will be decided upon by the team in response to the severity of the issue that caused suspension of testing.

4. Execution Plan -

4.1 Execution Plan

The execution plan will detail the test cases to be executed. The Execution plan to ben will be put together to ensure that all the requirements are covered. The execution plan will be designed to accommodate some changes if necessary, if testing is incomplete on any day. All the test cases of the projects under test in this release are arranged in a logical order depending upon their inter dependency.

Requirement (From SRS)	Test Case Identifier	Input	Expected behavior	Pass / Fail
5.2 Verify the voltage of the Jetson	1.1	A variable voltage device	The Jetson operates at voltages between 12v and 18v	TBA
3.3 Verify the Dependencies of TensorFlow on Local machine	1.2	A python environment	Verify that the integrity of TensorFlow dependencies is okay	Pass
4.1.1 Identify Turtle from image on local machine	1.3	A drone image	Neural Network Identified a turtle above a 50% rate	Pass
4.4.1 Identify turtle tracks from image on local machine	1.4	A drone image	Neural Network identified turtle tracks above a 50% rate	Pass
4.1.1 Identify Turtle from image on local machine	1.5	A drone image	Neural Network Identified a turtle above a 75% rate	Pass
4.4.1 Identify turtle tracks from image on local machine	1.6	A drone image	Neural Network identified turtle tracks above a 75% rate	Pass
3.3 Verify script functionality on local machine	1.7	Shell script	Test script on a folder, iterating through each item	TBA
3.3 Functionality of test environment	1.8	Nvidia Jetson Nano	Power on successful	Pass

3.3 Verify the Dependencies of TensorFlow on test environment	1.9	Environment existing on Jetson Nano	Verification script successful	Fail
4.1.1 Identify Turtle from image on test environment	1.10	Image containing a Turtle	Neural Network Identified a turtle above a 75% rate	TBA
4.1.1 Identify Turtle Tracks from image on test environment	1.11	Image containing turtle tracks	Neural Network identified turtle tracks above a 75% rate	TBA

5. Traceability Matrix & Defect Tracking

5.1 Traceability Matrix

List of requirements, corresponding test cases

Requirement LOW: System requirements Specification, 5.2: The Jetson shall output a voltage between 12v and 18v.

Test Cases: Check for the correct voltage range

Requirement CRITICAL: System requirements Specification, 3.3: The program shall verify the dependencies of TensorFlow

Test Cases: Verify the dependencies of TensorFlow are correct

Requirement CRITICAL: System requirements Specification, 4.1.1: The program shall identify a turtle when a drone image is given

Test Cases: A turtle is identified above a 50% rate

Requirement CRITICAL: System requirements Specification, 4.4.1: The program shall identify turtle tracks when a drone image is given

Test Cases: Turtle tracks are identified above a 50% rate

5.2 Defect Severity Definitions

Critical	The defect causes a catastrophic or severe error that results in major problems and the functionality rendered is unavailable to the user. A manual procedure cannot be either implemented or a high effort is required to remedy the defect. Examples of a critical defect are as follows: • System abends • Data cannot flow through a business function/lifecycle • Data is corrupted or cannot post to the database
Medium	The defect does not seriously impair system function can be categorized as a medium Defect. A manual procedure requiring medium effort can be implemented to remedy the defect. Examples of a medium defect are as follows: • Form navigation is incorrect • Field labels are not consistent with global terminology
Low	The defect is cosmetic or has little to no impact on system functionality. A manual procedure requiring low effort can be implemented to remedy the defect. Examples of a low defect are as follows: • Repositioning of fields on screens • Text font on reports is incorrect

6. Environment

6.1 Environment

The testing environment for the system we are testing is going to be the Nvidia Jetson Nano Development Kit. Eventually, once tested, the model will be deployed on the Nvidia Jetson Xavier. The testing environment will be running a version of Python 3.6, the same Python version we are developing with, and the Xavier has on-board. This is to keep a consistent test environment to ensure there are no abnormalities when deploying the code. In addition, we will include a requirements.txt file, which will have all the libraries and versions used in our environment.

7. Assumptions

This test plan for an image based neural network requires base assumptions in order to more accurately define a successful test. For an isolated software test, all assumptions are based around the external inputs and outputs available to the software being correct for the true application of the project.

The inputs are the power for the computing system, and the images provided to the software. An obvious requirement for software to run is having power for the processor the software is being executed on. This has been an issue in the past, however solutions for this problem are being implemented in the real application of the project so it shouldn't be an issue. The images being provided to the software for this test are from a training set specifically for this project. It is to be assumed that these images that are being inputted to the neural network are of the same resolution that the real project will be receiving, and the contents of the image are the correct target that the software must identify.

An output for the system being tested also must be installed properly so that the capability to measure the success of the software exists. Currently, the planned output for the system is storing saved images onto an SD card. Without this storage device, the system would not have the allocated storage to save the output of the software.

8. Risks and Contingencies

Since this is a pure software test, there are seemingly no real risks to any person or property when testing the product. However, precautions should still be taken to protect the property of the Turtle Tech team when testing software on their processors.

With software, it is still possible to crash processors, corrupt and overrun memory, along with other issues such as data leaks. Although it is highly unlikely to cause these issues intentionally, there is still a risk of some of these problems occurring. The main risk prevention strategies are to test code in emulators before running them on the processors, and to practice proper coding techniques.

9. Appendices