Testing Management Plan Document

Automated Spinal Segmentation and Remote Monitor Calibration for Surgical Assessment

Project for EN.601.456 Computer Integrated Surgery II

Group 8:
Damiano Marsilli, Arijit Nukala, Jonathan Young

Introduction

This document describes the scope of test procedures and activities that will be utilized to analyze the quality of the three subprojects of our project: spinal keypoint labeling, transfer function, and cloud platform. These tests are intended to be written and performed before production, with subsequent tests being developed for functionality assurance. The tests listed also set emphasis on which potential vulnerabilities and failings may arise and to what specificity.

Purpose

The test plan documents strategies and a high-level framework to guide testing activities and ensure that they are effectively implemented within the team's work. The plan defines the following parameters for testing: scope, milestones, requirements, dependencies, risks, procedures, documentation, and other administrative and functional parameters.

Objective and Tasks

1. Objectives

   The objective of the Test Plan is to outline the testing activities, and respective responsibilities that will ensure a thorough, representative evaluation of our software and the integration of our various submodules.

2. Roles and Responsibilities

Name	Role	Test Responsibility
Damiano Marsili	Spinal Keypoint Lead	Evaluate the accuracy, generalizability, and stability of the spinal keypoint estimation model.
Arijit Nukala	Transfer Function Lead	Evaluate the accuracy, generalizability, and stability of the transfer function. Quantify error-bounds for sensor-drift

Jonathan Young	AWS Pipeline Lead	Evaluate the user-friendliness, processing-time, and stability of the AWS cloud pipeline.

Scope

The scope of this test plan highlights the activities that will be covered in testing. As this project is divided into three intersecting sub-tasks, each of the tasks will be tested in isolation, through model validation, unit tests, and stress-tests respectively. We will also be testing the integration of all of the three sub-tasks as part of the AWS pipeline testing.

Testing Strategy/Levels of Testing

Testing Procedure

A. Spinal Keypoint Labeling

a. Environment

Model Evaluation will be performed locally on a personal workstation with GPU access. The trained models will be instantiated in a Python script. CUDA will be used to perform inference on a GPU to ensure shorter testing times.

b. Features to Test

Model Accuracy - The trained spinal keypoint model will need to be evaluated to ensure keypoint predictions are reasonably accurate. To this end, a subset of labeled data, referred to as the test dataset, is held out during training. This data will be used to validate the accuracy and generalizability of our model. As a performance metric, we will use Mean-Squared-Error between the predicted keypoints and the ground-truth annotations.

B. Transfer Function

a. Environment

Function evaluation will be performed locally on a workstation. The transfer function will initially be evaluated in MATLAB using MATLAB R2022a then translated into Python. There are no strict computing requirements for the transfer function.

b. Features to Test

Transfer Function accuracy - The transfer function will need to be validated such that the general accuracy can be estimated. A limited set of patient data will be analyzed manually to determine the spinal ROM via estimates from video. These estimates will be compared to the function output and analyzed via a Bland-Altman plot.

Transfer Function IMU drift - To measure and observe the level of drift incurred by the IMU that can be removed by the transfer function, multiple accuracy tests will taken at set intervals. The approximate change in accuracy will be measured in each axis and quantified via Mean-Squared-Error.

C. Cloud Pipeline

a. Environment

Unit testing will be performed via a script run on a local instance. The script will attempt to connect to various services and sections of our pipeline on AWS through the boto3 library and verify their functionality. There are no physical requirements, but the local machine that hosts the unit tests must have an appropriate Python version installed alongside the boto3 library.

System testing will be performed on AWS Cloudwatch, which monitors the processes and runtime usage of the various services. This monitorization service provides insights into how the systems and services are behaving. Like unit testing, system testing will have no physical environment restrictions.

Performance testing will also be done via AWS Cloudwatch and virtually, to monitor the runtime of the entire system. Communication with clients, either mentors or doctors, through virtual settings must be established in order to analyze the useability and efficacy of the interface.

b. Features to Test

AWS Cognito - Our team needs to first ensure that clients are able to log in to the system, and perform related tasks to log in such as resetting passwords, changing profile information, etc. From a privacy standpoint, our team also needs to ensure that the access is restricted; clients should have only access to their own data.

AWS SQS - The queue service needs to be monitored to ensure that it is organizing jobs in the right fashion and that jobs are placed on the queue and removed from the queue without issues of duplication.

AWS Cloudwatch - Metrics regarding the queue are utilized to scale the current system - the EC2 cluster. Our team needs to ensure that the scaling is appropriate in terms of provisioning new EC2 instances from within the cluster itself. Moreover, our team needs to ensure that the scaling is cost-effective and that not an excessive amount of EC2 instances are being launched unnecessarily when fewer could have been.

AWS EC2/AMI - Our team needs to ensure that the EC2's runtime is appropriate for our project and that they are able to start independently and process independently of human intervention. Moreover, our team needs to ensure that the AMIs have the right template for all use case scenarios and are easily deployable without failing.

AWS Lambda/Cohesiveness - Lambda's are one component that integrated the various services together alongside other triggers. Our team needs to ensure that the various services as a whole work together to generate an output, without human intervention. Moreover, the pipeline needs to generate useable data and must be able to self-check that the systems and services are working properly and throw a warning if there appears to be a mistake.

Schedule

The tentative testing schedule goes as follows:

Test	Timeframe
[Keypoint Labeling] - Tests to ensure that the key points are correctly identified	4/17-4/19
[Transfer Function] - Tests to ensure that the transfer function is able to map IMU to Computer Vision generated coordinates effectively.	4/17-4/19

[AWS Pipeline] - Tests to ensure that individual services and the pipeline as whole work.	4/17-4/19
Unit Testing	Continuous
System integration functionality testing	4/20-4/23
Performance testing on sample patient test data	4/23-4/28