Text to Motion Database

Test Report

Brendan Duke
Andrew Kohnen
Udip Patel
David Pitkanen
Jordan Viveiros

April 10, 2017

Contents

1 Preface

1.1 Revision History

Table 1.1: Revision History

Date	**V**ersion	**N**otes
March 14, 2017	0.0	File created
March 23, 2017	0.1	Initial template completed
March 26, 2017	0.2	Completed tables and organized sections
April 7, 2017	1.0	Added more detail to tests and tables
April 9, 2017	1.1	Added automated testing and PCKh figure

1.2 List of Figures

This document contains one figure located within automated testing, labeled 7.1.

List of Tables

2 Introduction

2.1 Purpose of Document

The purpose of this manuscript is to document the testing that has been performed on the Text-to-Motion Database. The testing that has been performed largely follows the test plan and requirements that were provided in the Test Plan and Software Requirements and Specification documents respectively.

2.2 Scope of Testing

Following good design principles, the Text-To-Motion Database has been modularized into 3 conceptual modules. The first module is a web framework that allows users to access key features like uploading media, viewing previous uploads, and running pose estimation through our deep learning algorithm. The database is the second module in the application and is used to store events from the website like account registration or the uploading of new media. The third module is the most expansive and is the deep learning module that is responsible for running pose estimation on a remote server.

Following the application's modular decomposition, the tests have been separated into 3 conceptual tests and automated testing.

The first stage of testing is the minimum viable product, and was done to ensure that a base level of functionality was provided. This is where we test to see that all of the software modules have been created and can function synchronously. In order to ensure a minimum viable product was complete both system and unit testing were utilized.

The second set of tests focused on Solution Constraints testing or Functional tests. In this set of tests, the performance of the deep learning algorithm was rigorously tested and quantified. Building upon the testing of the deep learning algorithm the database was tested to ensure the search capabilities by using specific parameters and test cases.

In the third set of tests, volunteers were asked to test the Non-Functional requirements of the finished product. Examples of these requirements include: usability, design requirements and learnability.

The final set of tests were automated tests ran on the http-server in order to verify the server was posting the correct GET and POST calls. These tests were also used to verify the correctness of the pose estimation of an image.

The final section in this document includes a traceability table that helps to organize and explain how tests are connected to the Software Requirements and Specification document.

3 Preliminary Testing

3.1 Minimum Viable Product

3.1.1 TextToMotion Availability

Description

This test is done to ensure that the TextToMotion Database (brendanduke.ca/) is functioning as intended and available to the public. These tests will be done by first accessing the home page of brendanduke.ca/, and then navigating to other pages within the website. If the pages are available and contain functioning hyperlinks the test will be considered a pass.

Results

The website was accessible through the browser at brendanduke.ca with all buttons, URL's, and hyperlinks being responsive the test has been passed.

Test #	**T**est	**I**nitial State	**E**xpected Output	**A**ctual Output	**R**esult
3.1	Input brendanduke.ca to view the home page	Google.ca	The home page of brendanduke.ca	The home page of brendanduke.ca	Pass
3.2	Add ../Account/Register to the URL or click "Register"	brendanduke.ca	The user registration page	The user registration page	Pass
3.3	Add ../Account/Login to the current URL or click "Login"	brendanduke.ca	The user sign in page	The user sign in page	Pass
3.4	Add ../ImagePoseDraw to the current URL or click "ImagePoseDraw"	brendanduke.ca	The ImagePoseDraw page	The ImagePoseDraw page	Pass
3.5	Add ../Demo to the current URL or click "Demo"	brendanduke.ca	The Demo page with the camera feed	The Demo page with the camera feed	Pass
3.6	Add ../TextToMotion to the current URL or click "TextToMotion"	brendanduke.ca	The TextToMotion page	The TextToMotion page	Pass

3.1.2 Updating the database

Description

The database allows users to upload images to the database through the website through brendanduke.ca/ImagePoseDraw. The test will be considered a pass if an image can be uploaded successfully and the user lands on ImagePoseDraw without error.

Input Data

Throughout this section the input data is referred to by "Input".jpeg and the website location is referenced through ../ImagePoseDraw which represents brendanduke.ca/ImagePoseDraw.

Input	Description
AverageGuy.jpg	A picture of a man standing still and facing the camera
AverageGirl.jpg	A picture of a woman standing still and facing the camera
TomBrady.jpg	A picture of football star Tom Brady

Results

All images uploaded without error and the user was redirected back to brendanduke.ca/ImagePoseDraw after each upload, therefor the test has been passed.

Test #	**T**est	**I**nitial State	**E**xpected Output	**A**ctual Output	**R**esult
3.6	Upload TomBrady.jpeg using the ImagePoseDraw functionality	.. /ImagePoseDraw/Create	Land on ../ImagePoseDraw	Landed on ../ImagePoseDraw	Pass
3.7	Upload AverageGirl.jpeg using the ImagePoseDraw functionality	.. /ImagePoseDraw/Create	Land on ../ImagePoseDraw	Landed on ../ImagePoseDraw	Pass
3.8	Upload AverageGuy.jpeg using the ImagePoseDraw functionality	.. /ImagePoseDraw/Create	Land on ../ImagePoseDraw	Landed on ../ImagePoseDraw	Pass

3.1.3 Verifying Pose Estimation

Description

After the user has uploaded an image for pose estimation they should be able to view the pose estimation that has been run on the image. In order to verify that some form of pose estimation has been run on the image the user must view their upload and have visible annotations made to the image to be considered a pass.

An image once uploaded can be found within brendanduke.ca/ImagePoseDraw/Details/N, where N represents the number of the uploaded image. This can also be achieved by searching through the uploads.

Results

Each uploaded image had pose estimation run sucessfuly and resulted in a new image with a skeleton overlay on the limbs so this test was passed.

Test #	**T**est	**I**nitial State	**E**xpected Output	**A**ctual Output	**R**esult
3.9	View TomBrady.jpeg from ../ImagePose-Draw/Details/30	.. /ImagePose-Draw	TomBrady.jpeg with pose estimated limbs	TomBrady.jpeg with pose estimated limbs	Pass
3.10	View AvergaeGirl.jpeg from ../ImagePose-Draw/Details/26	.. /ImagePose-Draw	AvergaeGirl.jpeg with pose estimated limbs	AvergaeGirl.jpeg with pose estimated limbs	Pass
3.11	View AverageGuy.jpeg from ../ImagePose-Draw/Details/27	.. /ImagePose-Draw	AverageGuy.jpg with pose estimated limbs	AverageGuy.jpg with pose estimated limbs	Pass

3.1.4 Search by Name or Description

Description

The web interface has the ability to search through the uploaded images and videos based on their name or information within the associated description. The test will be considered a pass if the name is input in the search bar and image is returned.

Results

All three tests passed as it was successful to search through the entries and find the upload through an associated name or description.

Test #	**T**est	**I**nitial State	**E**xpected Output	**A**ctual Output	**R**esult
3.12	Input "Tom" in the search bar within ../ImagePoseDraw	.. /ImagePoseDraw	A list of figured including the Tom Brady image that was uploaded	A single return of the Tom Brady image	Pass
3.13	Input "Average" in the search bar within ../ImagePoseDraw	.. /ImagePoseDraw	A list of results including the AverageGirl image	Two results, one of which was Average Girl	Pass
3.14	Input "Guy" in the search bar within ../ImagePoseDraw	.. /ImagePoseDraw	A list of results including the AverageGuy image	Two results, one of which was Average Guy	Pass

3.2 Solution Constraints Testing

3.2.1 Deep Learning Methods Test

Description

In order to provide a proper demonstration of the deep learning mechanics that can be associated with pose estimation we will meet with Dr.Taylor to ensure that we are implementing the Bulat et al paper properly. This test will be considered a pass if Dr.Taylor is satisfied with the implementation of the previously mentioned paper.

Results

Test #	**T**est	**R**esult
3.15	Meet with Dr.Taylor in order to verify the integrity of the deep learning implementation within the TextToMotion Database.	Pass

3.2.2 Standard Data Format Test

Description

An automated test that checks if the human pose data used for the project is standard and compatible with existing software libraries.

Results

This test was technically a failure. The data was not converted to a format compatible with libraries. Instead the data is stored in JSON strings, which is a standard format, but not compressed as an ideal format such as HDF5 would be.

4 Functional Requirements

4.1 Input Data

Throughout this section the input data is referred to by "Input".MP4 and the website location is referenced through ../ImagePoseDraw which represents brendanduke.ca/ImagePoseDraw.

Input	Description
E9FY2.MP4	A short video of a woman eating a sandwich
U4XV9.MP4	A short video of a man waking up and getting out of bed
Z1A0Q.MP4	A short video of a man sitting on a stool

4.2 Supported Video Encoding Test

4.2.1 Description

Tests whether the ReadFrames API is able to decode MP4 files. If we are able to run pose estimation on the video, then the ReadFrames API is able to process the frames and the test will be considered a pass.

Results

Each test was successful in using the ReadFrames API to decode the MP4 files and run pose estimation so this test has been passed.

Test #	**T**est	**I**nitial State	**E**xpected Output	**A**ctual Output	**R**esult
4.1	Using the ReadFrames API on E9FY2.MP4, to run pose estimation	ReadFrame API	A compiled E9FY2.MP4 that has been pose estimated	E9FY2.MP4 with pose estimation	Pass
4.2	Using the ReadFrames API on U4XV9.MP4, to run pose estimation	ReadFrame API	A compiled U4XV9.MP4 that has been pose estimated	U4XV9.MP4 with pose estimation	Pass
4.3	Using the ReadFrames API on Z1A0Q.MP4, to run pose estimation	ReadFrame API	A compiled Z1A0Q.MP4 that has been pose estimated	Z1A0Q.MP4 with pose estimation	Pass

4.3 Frame Reading Timestamp Accuracy Test

4.3.1 Description

Tests whether the timestamps on the frames returned by the ReadFrames API match their temporal position in the original video stream. In order for this test to be considered a pass two random timestamps will be verified and if they match it will be considered a success. On a single failed timestamp two new timestamps will be taken until it is a definite failure or success.

4.3.2 Results

All three tests were able to match timestamps to the original video and the test has been passed.

Test #	**T**est	**I**nitial State	**E**xpected Output	**A**ctual Output	**R**esult
4.4	Use the ReadFrames API on E9FY2.MP4, in order to verify the timestamps	ReadFrame API	Matching timestamps at 5 and 7 seconds	Matching timestamps at 5 and 7 seconds	Pass
4.5	Use ReadFrames API on U4XV9.MP4, in order to verify the timestamps match	ReadFrame API	Matching timestamps at 3 and 10 seconds	Matching timestamps at 3 and 10 seconds	Pass
4.6	Use ReadFrames API on Z1A0Q.MP4, in order to verify the timestamps match	ReadFrame API	Matching timestamps at 6 and 11 seconds	Matching timestamps at 6 and 11 seconds	Pass

4.4 Human Pose Estimation Data Quality Test

4.4.1 Description

Test to ensure the data quality produced by the human pose estimator component was acceptable.

A set of Charades videos will be processed by the human pose estimator, and skeleton animations corresponding to the generated human pose data will be created (this is a scoped part of the software pipeline). A double-blind test will be ran, wherein testers will be shown random mixed sets of the skeleton animations produced by McMaster Text to Motion, together with skeletons from actual motion capture data coming from CMU"s motion capture lab. Testers will indicate whether they think the motion capture data came from actual motion capture, or from the pose estimation software.

The McMaster Text to motion Results should be guessed as accurate at similar rates to the Charades tests.

4.4.2 Results

We provide the ratio of Text-to-Motion images guessed as accurate compared to the Charades Images.

Test #	**T**est	**C**harades/ TextToMotion	**R**esult
4.7	Nick was shown a set of videos and asked to determine which was generated by the Text-ToMotion Database	1/1	Pass
4.8	Drew was shown a set of videos and asked to determine which was generated by the Text-ToMotion Database	7/5	Pass
4.9	Sarah was shown a set of videos and asked to determine which was generated by the Text-ToMotion Database	5/4	Pass

4.5 Database Output Full Range Coverage Test

4.5.1 Description

Tests to be sure all entries in the database can be successfully searched for. The previous input data has been renamed in order to specify names and descriptions which can be found below. In order for this test to be considered a pass each uploaded video will need to be found when searching for the name or description given.

4.5.2 Input Data

Throughout this section the input data is referred to by "Input".MP4 and the website location is referenced through ../ImagePoseDraw which represents brendanduke.ca/ImagePoseDraw.

Input	Description
`Waking_Up.mp4`	Given tags sleeping, boy, man, sleepy, getting up, table
`Eating.mp4`	Given tags sandwich, eating, girl, woman, table
`Stool.mp4`	Given tags man, stool, corner, sitting

Results

The given set of videos appeared in the returned list of videos from the search, so this test has been passed.

Test #	**T**est	**I**nitial State	**E**xpected Output	**A**ctual Output	**R**esult
4.10	Search for the Waking_Up.mp4 within ../ImagePoseDraw with keyword "Waking"	.. /ImagePoseDraw	A list of results including Waking_Up.MP4	A single result of the Waking_up.mp4	Pass
4.11	Search for the Eating.mp4 within ../ImagePoseDraw with keyword "Eat"	.. /ImagePoseDraw	A list of results including Eating.MP4	A single result of the Eat.mp4	Pass
4.12	Search for the Stool.mp4 within ../ImagePoseDraw with keyword "sitting"	.. /ImagePoseDraw	A list of results including Stool.Mp4	A single result of the Stool.mp4	Pass

4.6 Database No False Positives

4.6.1 Description

Tests that the database search does not return any false positives, such as videos or images that do not contain searched words. The same videos from the previous test will be used with the same tags. Thus, we will search with tags other than those provided. If no videos appear, then the test is a success.

Results

Using the nonsensical keywords from our input data, no search results were returned, meaning that the test passed.

Test #	**T**est	**I**nitial State	**E**xpected Output	**A**ctual Output	**R**esult
4.13	Search for "Raptor" within ../ImagePoseDraw	.. /ImagePoseDraw	A list of results that don't contain the input data	No results were returned	Pass
4.14	Search for "Exponent" within ../ImagePoseDraw	.. /ImagePoseDraw	A list of results that don't contain the input data	No results were returned	Pass
4.15	Search for "Glasses" within ../ImagePoseDraw	.. /ImagePoseDraw	A list of results that don't contain the input data	No results were returned	Pass

4.7 HTTP GET test

4.7.1 Description

A test to ensure the HTTP server can receive GET requests with an image, and return a JSON string of values containing joint positions. We present the values as an array, where each vertex has two components representing the x and y position. If every value is between -0.5 and 0.5 then the data returned is accurate, and the test is a pass. The GET requests were created using curl with the -X GET option and attempting to reach brendanduke.ca:8765

4.7.2 Input Data

Input	Description
Sporty Woman	A picture of a woman standing and waving her hands
Running Man	An illustration of a man running
Sitting Man	A 3d model of a person sitting

4.7.3 Results

Test #	**T**est	**I**nitial State	**E**xpected Output	**A**ctual Output	**R**esult
4.16	Using curl -X GET with parameters of brenden-duke.ca:8765 and the Sporty Woman image	CMD line with curl installed	An array of values between -0.5 and 0.5	See Table 4.1	Pass
4.17	Using curl -X GET with parameters of brenden-duke.ca:8765 and the Running Man image	curl is installed and functional	An array of values between -0.5 and 0.5	See Table 4.2	Pass
4.18	Using curl -X GET with parameters of brenden-duke.ca:8765 and the Sitting Man image	curl is installed and functional	An array of values between -0.5 and 0.5	See Table 4.3	Pass

4.7.4 Output Data

Table 4.1: For test 4.16 (r is for left and l is for right)

Body Part	JSON Values
r ankle	[-0.0736842105263, 0.328947368421]
r knee	[-0.0315789473684, 0.157894736842]
r hip	[-0.0526315789474, 0.0105263157895]
l hip	[-0.0315789473684, 0.0315789473684]
l knee	[-0.0315789473684, 0.157894736842]
l ankle	[-0.0315789473684, 0.157894736842]
pelvis	[-0.0342105263158, 0.0105263157895]
thorax	[-0.0105263157895, -0.223684210526]
upper neck	[0.0105263157895, -0.244736842105]
head top	[0.0105263157895, -0.394736842105]
r wrist	[-0.223684210526, -0.328947368421]
r elbow	[-0.202631578947, -0.244736842105]
r shoulder	[-0.0526315789474, -0.223684210526]
l shoulder	[0.0526315789474, -0.223684210526]
l elbow	[0.181578947368, -0.265789473684]
l wrist	[0.160526315789, -0.371052631579]

Table 4.2: For test 4.17 (r is for left and l is for right)

Body Part	JSON Values
r ankle	[-0.157894736842, 0.244736842105]
r knee	[-0.0736842105263, 0.0947368421053]
r hip	[0.0315789473684, -0.0315789473684]
l hip	[0.0315789473684, -0.0526315789474]
l knee	[0.0526315789474, 0.118421052632]
l ankle	[0.223684210526, 0.0736842105263]
pelvis	[0.0315789473684, -0.0342105263158]
thorax	[-0.0315789473684, -0.157894736842]
upper neck	[-0.0315789473684, -0.157894736842]
head top	[-0.0526315789474, -0.328947368421]
r wrist	[-0.0105263157895, -0.0315789473684]
r elbow	[-0.0315789473684, -0.0736842105263]
r shoulder	[-0.0526315789474, -0.157894736842]
l shoulder	[-0.0526315789474, -0.160526315789]
l elbow	[0.0526315789474, -0.157894736842]
l wrist	[0.0526315789474, -0.0315789473684]

Table 4.3: For test 4.18 (r is for left and l is for right)

Body Part	JSON Values
r ankle	[-0.139473684211, 0.328947368421]
r knee	[-0.221052631579, 0.0526315789474]
r hip	[-0.0315789473684, -0.0105263157895]
l hip	[0.0763157894737, 0.0315789473684]
l knee	[-0.0315789473684, 0.202631578947]
l ankle	[-0.0736842105263, 0.371052631579]
pelvis	[0.0315789473684, 0.0105263157895]
thorax	[0.0947368421053, -0.265789473684]
upper neck	[0.0763157894737, -0.328947368421]
head top	[0.0526315789474, -0.5]
r wrist	[-0.118421052632, -0.0736842105263]
r elbow	[-0.0736842105263, -0.118421052632]
r shoulder	[-0.0105263157895, -0.265789473684]
l shoulder	[0.202631578947, -0.244736842105]
l elbow	[0.178947368421, -0.0736842105263]
l wrist	[0.0315789473684, -0.0552631578947]

4.8 False Positive Test

4.8.1 Description

Tests whether the HTTP server will return a 404 error when attempting to make an empty GET request. To do this we SSL to the server (openssl s_client -connect brendanduke.ca:8765) and when asked for an input, hit enter/return twice to create an empty request.

4.8.2 Results

Test #	**T**est	**I**nitial State	**E**xpected Output	**A**ctual Output	**R**esult
4.19	Sending an empty request with SSL	CMD line with curl installed	A 404 Error	HTTP/1.0 400 Bad Request Server: BaseHTTP/0.6 Python/3.5.2	Pass

5 Non-Functional Requirements

5.1 Usability

5.1.1 Description

In order to determine the usability of the Text-to-Motion database, a small sample of users were asked to use the website to perform some predetermined actions and answer questions afterwards.

Before the participants were asked to preform any actions they were given a minute to familiarize themselves with the interface, but were not given any guidance or tips from the development team. Once the time was up they were asked to upload an image using their mobile device or desktop. If they used the desktop the user either uploaded an image through a URL or a previously saved file.

While performing the required action the participant's time was recorded and used to determine if a requirement had passed or failed. Upon completion of the task the users were asked to rate the style and design of the website on a scale from 1 to 10, with 1 being the lowest option and 10 being the highest.

5.1.2 Results

The results from the participants can be seen throughout the Non-Functional Requirements along with a pass or fail based on the requirements description.

5.2 Look and Feel Requirements

5.2.1 Colour Scheme

Description

A test to see if the colour scheme of the website is visually appealing. Making the colour scheme of the website visually appealing to a larger audience is important as it will help keep users on the website and ideally provide a reason to reference it to a friend or colleague.

When testing the colour scheme anything above a 6 was considered a pass. We chose anything above a 6 because with this rating it means users enjoyed the colour scheme and may result in them recommending the website to others or returning.

Results

Test #	**U**ser	**I**nitial State	**R**ating	**R**esult
5.1	Nick	brendanduke.ca	7	Pass
5.2	Drew	brendanduke.ca	9	Pass
5.3	Sarah	brendanduke.ca	7	Pass

5.3 Style Requirements

5.3.1 Minimalistic Web Design

Description

The website interface should be minimal and should inform the user of valid actions through visual means. This is a modern requirement in website design, the more minimalistic design you can provide the better it will be for the user experience and any new technologies use to enforce this will help drive traffic.

While testing the minimalistic design anything above a 7 was considered a pass. This required a higher rating then the colour scheme as we believed it would be a key factor in providing the user with an enjoyable experience and help provide a greater amount of uploads if the interface was easily understood.

Results

Test #	**U**ser	**I**nitial State	**R**ating	**R**esult
5.4	Nick	brendanduke.ca	9	Pass
5.5	Drew	brendanduke.ca	9	Pass
5.6	Sarah	brendanduke.ca	8	Pass

5.4 Ease of Use Requirements

5.4.1 Upload/Download

Description

Through the web interface a user should be able to upload a picture using either a mobile phone camera, URL, or saved file. The participant will start on the Home page and be asked to upload an image through one of the methods previously mentioned.

In order for this to be considered a pass it should take the users 45 seconds or less to complete the upload process and click the button. The required time of 45 seconds was

chosen due to the amount of time that was required to find an image, URL, or file while running trials throughout the development.

Results

Test #	**T**est	**I**nitial State	**U**ser	**T**ime	**R**esult
5.7	Upload an image from a URL	.. /ImagePose-Draw/Create	Nick	32 seconds	Pass
5.8	Upload an image from a mobile device	.. /ImagePose-Draw/Create	Drew	38 seconds	Pass
5.9	Upload a file stored within the Desktop	.. /ImagePose-Draw/Create	Sarah	27 seconds	Pass

5.4.2 Text Box Functionality

Description

The user should be able to input a descriptive word or phrase into a text-box from within the web interface in order to search for a video. With a the website being part of a larger database the ability to search for an image that the user just uploaded is an important factor in determining the usability for users.

In order to complete this task the users were asked to search for a specific word and display the results. Any time below 10 seconds will be considered a pass. Each user was allocated 10 seconds because we did not want the action of typing the keyword to be the bottleneck when searching for a database entry.

Results

Test #	**T**est	**I**nitial State	**U**ser	**T**ime	**R**esult
5.10	Search for "Woman"	.. /ImagePose-Draw	Nick	4 seconds	Pass
5.11	Search for "f"	.. /ImagePose-Draw	Drew	3 seconds	Pass
5.12	Saerch for "the"	.. /ImagePose-Draw	Sarah	4 seconds	Pass

5.5 Learning Requirements

5.5.1 Usability Tests

Description

The user should be able to interact with the website without any previous knowledge. They will given a minute to explore the website. After that time the participants were asked to rate the usability on a scale of 1-10.

In order for the result to be considered a pass the ranking must be greater then a 6. This was chosen as having a website that is usable to someone without any prior knowledge to pose estimation or deep learning keeps the users that may use the website much larger and widely accepted.

Results

Test #	**U**ser	**R**ating	**R**esult
5.13	Nick	7	Pass
5.14	Drew	8	Pass
5.16	Sarah	7	Pass

5.6 Politeness and Understandability Requirements

5.6.1 Hiding the Inner Workings

Description

Users should not be able to see the deep learning model and its training when using the pose estimation. When prompted the website should display the correct skeletons without any low-level detail. Once the image was uploaded the participants were asked if they saw anything that seemed out of place or any information on the software used, if they did not it will be considered a pass.

Results

Users indicated that the deep learning model was encapsulated from their view, and hence this test passed.

Test #	**T**est	**I**nitial State	**U**ser	**R**esult
5.17	Uploading an image from URL	.. /ImagePose-Draw/Create	Nick	Pass
5.18	Uploading an image from mobile	.. /ImagePose-Draw/Create	Drew	Pass
5.19	Uploading an image from desktop	.. /ImagePose-Draw/Create	Sarah	Pass

5.7 Speed and Latency Testing

5.7.1 External Database Connection and Verification Time

Description

The web interface should be able to connect to an external database and store or query items. In order for this test to be considered a pass the confirmation of the image being uploaded would have to occur within 60 seconds so that additional resources are not wasted by the database.

The required time was determined to be 60 seconds because we are remotely connecting to a server within the University of Guelph in order to access their GPU and would add time to the total execution time.

Input Data

Input	Description
URL image	An image of a male
Desktop file	An image of Seth Rogan
Mobile image	An image of Andrew

Results

According to the response times of this test and verification, the database queries were executed fast enough for the test to pass.

Test #	**T**est	**I**nitial State	**E**xpected State	**A**ctual State	**T**ime	**R**esult
5.20	Upload an image from a URL	.. /ImagePoseDraw/Create	ImagePoseDraw with the new image	ImagePoseDraw with the new image	52 seconds	Pass
5.21	Upload a file stored within the Desktop	.. /ImagePoseDraw/Create	ImagePoseDraw with the new image	ImagePoseDraw with the new image	58 seconds	Pass
5.22	Upload an image from a mobile device	.. /ImagePoseDraw/Create	ImagePoseDraw with the new image	ImagePoseDraw with the new image	56 seconds	Pass

6 Other Relevant Testing

Again we test this on the same files we have used in the previous tests: `TomBrady.jpg`, `AverageGirl.jpg` and `AverageGuy.jpg`.

6.1 Precision and Accuracy

6.1.1 Bone and Joint Position

Description

The pose estimation should accurately predict the placement of joints and bones of the person in the provided photo. This will be determined with visual means with an uncertainty range of 20 pixels.

Input Data

Input	Description
`AverageGuy.jpg`	A picture of a man standing still and facing the camera
`AverageGirl.jpg`	A picture of a woman standing still and facing the camera
`TomBrady.jpg`	A picture of football star Tom Brady

Results

We were able to qualitatively confirm that these tests passed for the given input images. A more rigorous "PCKh" metric is used to formally evaluate the performance of our deep learning model on single-person pose estimation.

Test #	Test	Test Location	Tester	Result
6.1	Verify the joint position of TomBrady.jpg	.. /ImagePose-Draw/Details/30	Jordan	Pass
6.2	Verify the joint position of AverageGirl.jpg	.. /ImagePose-Draw/Details/26	Jordan	Pass
6.3	Verify the joint position of AverageGuy.jpg	.. /ImagePose-Draw/Details/27	Jordan	Pass

6.1.2 Deep Learning Model

Description

The **PCKh** metric, used by the MPII Human Pose Dataset, defines a joint estimate as matching the ground truth if the estimate lies within 50% of the head segment length. The head segment length is defined as the diagonal across the annotated head rectangle in the MPII data, multiplied by a factor of 0.6. Details can be found by examining the MATLAB evaluation script provided with the MPII dataset.

If our model can achieve 80% total PCKh then the test is considered a pass.

Results

Our model achieves 85% PCKh, thus this test is a pass.

Test	PCKh Value
r ankle	68.907%
r knee	77.201%
r hip	83.583%
l hip	84.444%
l knee	77.419%
l ankle	69.055%
pelvis	89.776%
thorax	98.071%
upper neck	97.823%
head top	96.557%
r wrist	81.288%
r elbow	87.205%
r shoulder	92.918%
l shoulder	92.828%
l elbow	85.278%
l wrist	80.719%
Total PCKh	**85.195%**

6.2 Reliability and Availability Requirements

6.2.1 Software Availability

Description

The software component of the project should be available at all times. If we can have an event regularly occur then it will be considered a pass. To do this we arranged to have the pose estimation algorithm automatically called on to process a single image at 4 hour intervals, and record the time.

Results

The software successfully processed the image at the specified 4 hour intervals over a period of 2 days.

6.2.2 Website Availability

Description

The software component of the project should be available at all times with the exception of maintenance and migration. When we make a web server call we should receive an HTTP verified response. To do this we will have three users sending a HTTP POST request to the server. If they receive a response the test is considered a pass.

Results

All three users were able to receive responses to their HTTP POST requests, therefore we consider this test a pass.

6.3 Robustness or Fault-Tolerance Requirements

6.3.1 Web Interface Error Handling

Description

The web interface should respond to unhandled exceptions by throwing the corresponding error messages. If an exception is thrown and an error message is displayed then the test is considered a pass.

Input Data

Input	Description
Scenario 1	Try uploading an image while the image storage service (Amazon S3) is not available.
Scenario 2	Trying to search an image on the database while the database server is down.

Results

For Scenarios 1 and 2, appropriate exceptions were thrown and error pages displayed to the user. Therefore this test is considered to have passed.

6.4 Capacity Requirements

6.4.1 Multiple Connections

Description

The web interface should be able to serve multiple connections. If the interface can support 5 connections at once it is considered a pass.

Input Data

Input	Description
Andrew	The connection to the website corresponding to tester Andrew
Brendan	The connection to the website corresponding to tester Brendan
Udip	The connection to the website corresponding to tester Udip
Jordan	The connection to the website corresponding to tester Jordan
Dave	The connection to the website corresponding to tester Dave

Results

The 5 users were able to successfully connect to the website, and run queries simultaneously without experiencing significant slow down or waiting.

6.4.2 Database Capacity

Description

The database should contain at least 5GB of data in order to facilitate growth.

Results

The database successfully handled us uploading a total of 5.1GB of data in the form of pictures of various qualities. These pictures were of randomly acquired photos from Google, which were combined to form the data used in this testing.

6.5 Scaling of Extensibility Requirements

6.6 Operational Environment Requirements

6.6.1 Linux Friendly TensorFlow

Description

The web interface should be run on a Linux friendly server that can access the TensorFlow model either directly or indirectly. By creating an interface that successfully runs on a Apache or nginx server this test will be considered a pass.

Results

Our service is successfully running on a production Linux server using nginx.

6.6.2 Export Types

Description

The project should be able to export multiple types of media (JPEG, PNG, etc) in order to support all major operating systems. We will use `TomBrady.jpg` for this test.

Results

This test was a guaranteed pass due to the fact that our website stores images as a base64, meaning it can be converted into any type.

Test #	**T**est	**R**esult
6.4	Export as PNG	Pass
6.5	Export as JPEG	Pass
6.6	Export as BMP	Pass

7 Automated Testing

The majority of the tests preformed on the TextToMotion Database have been manual due to the state of deep learning and pose estimations requirement of human verification. The server and values within the deep learning algorithm had some level of automation to them as PCKh values were generated in order to verify the confidence maps, and server tests could be run to ensure there was a connection open with the tf-http-server.

7.1 Server Tests

To automate the tests a python client was created that could send POST requests to the HTTP server. The purpose of this test is to ensure that the JSON string values it returns are within the acceptable range ($-0.5 to 0.5$) on a large range of inputs. A small scale of this can be seen within section 4.7 with regards to how the data is output.

The FLIC - Frames Labeled in Cinema - dataset is a set of 5000 pictures taken from popular Hollywood movies. The automated test we ran involved sampling 100 pictures from the FLIC database and sending 100 post requests to the HTTP server. One post request for each picture sampled from the FLIC dataset using the python client we coded. Each HTTP post returned a JSON response which checked with an 'if' statement that the returned values are within the allowed range. If the value was not within the allowed range then a 'fail' message would be printed on the terminal of the screen. The expected output of this test was that none of the 100 POST requests would cause a 'FAIL' message to printed on the screen. The result of running this test 3 separate times yielded the expectation of no fail message.

Table 7.1: Automated Tests of HTTP Server

Test #	Expected Result	Observed Result	Pass/Fail
Test 1	Pass message on console	Pass message on console	PASS
Test 2	Pass message on console	Pass message on console	PASS
Test 3	Pass message on console	Pass message on console	PASS

7.2 Optimization Tests

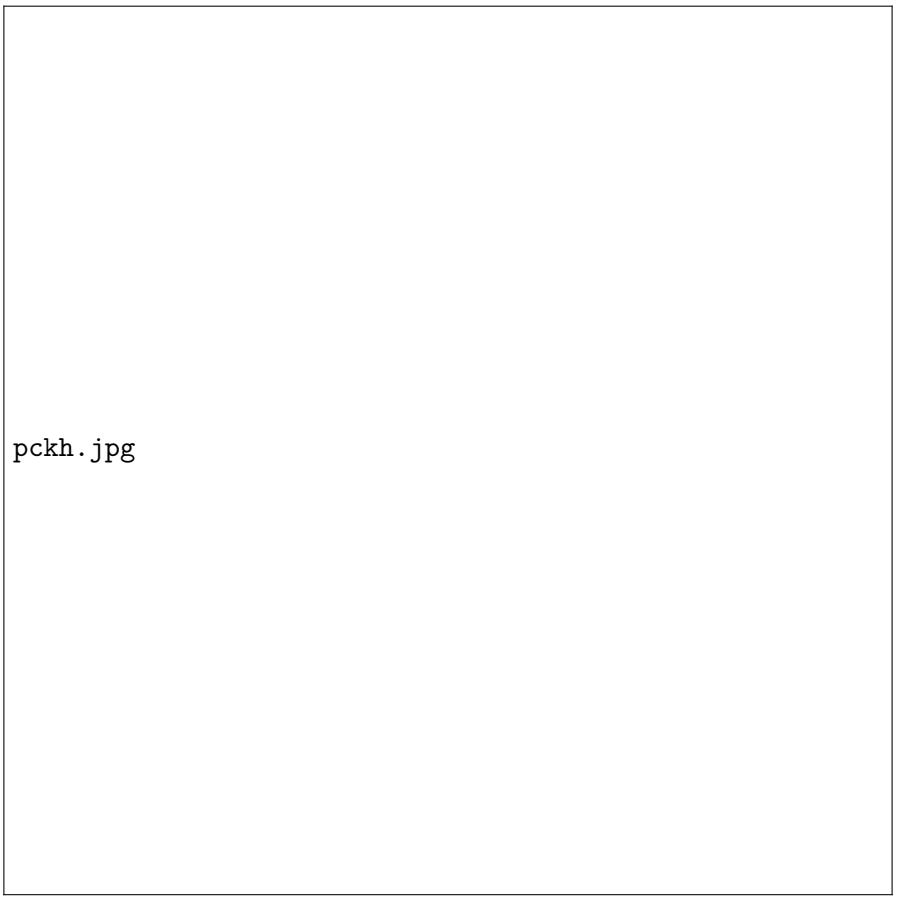
In this section the improvement of the pose estimating neural network is documented as the optimization algorithm is performed. The training procedure to improve the performance of the network involves starting the network randomly. The network has parameters in it and the training procedure iteratively changes these parameters to optimize the network's performance.

In Figure 7.1 the change in the performance of the network is shown at each iteration in the training algorithm. The performance of the network is measured using a metric called PCKh. This metric is discussed in the Deep Learning section of this manual.

After each epoch interval of training (an interval where each labelled picture in the dataset is analyzed) a measurement of the PCKh value was documented. The PCKh value was calculated separately for each of the joints. However in this graph only 6 joints are displayed. The joints that are shown well represent the performance of the network because they are joints from all different areas of the body (upper and lower body, limbs, head and mid-section). The performance of the network is shown in **??** starting from its random initialization to the 56 epoch.

The network was trained using the MPII Human Pose Dataset consists of around $17,000$ jpeg images. So 17000 pictures had to be passed through the network for each iteration. Clearly for the training procedure to be considered successful the value of the PCKh value should increase steadily and as rapidly as possible as training proceeds. A value above a PCKh of 0.8 for any joint is considered a success. For this training procedure it is clear that all the joints made it above this value except the right knee.

Figure 7.1: Progression of Network Optimization



8 Traceability

Table 8.1: Traceability Matrix for Test-Requirement Relationships

Test Section	**D**escription	**R**equirement
3.1.1	Reliability and Availability of the web interface	Req # 27, 28
3.2.1	Solution Constraints and accuracy of deep learning	Req # 1
3.2.2	Solution Constraints for data rich storage	Req # 2
6.6.1	Implementation Environment to support operating systems	Req # 3
4.2	Functional Requirement that ensures video can be processed	Req # 7
4.3	Functional Requirement for joint positioning relative to timestamps	Req # 8
4.5	Functional Requirement that each media upload has associated text	Req # 9
4.6	Functional Requirement that no false postiives are returned while searching	Req # 7
5.2	Appearance Requirement for the look and feel	Req # 12
5.3	Style Requirement to implement a minimalistic design	Req # 13
5.4.1	Ease of Use Requirement that an average user can use the website	Req # 14
5.4.2	Ease of Use Requirement for text box functionality	Req # 16, 24
5.5	Learning Requirement that a user can 'pose estimate' without prior knowledge	Req # 17, 18
5.6	Politeness and Understandability Requirement to hide the inner workings of deep learning	Req # 20
5.4.2	Ease of Use Requirement for text box functionality	Req # 16
5.7	Speed and Latency Requirement to minimize resource cost	Req # 22, 23

6.1.1	Precision and Accuracy of bone joints positioning	Req # 25, 26
6.1.2	Precision and Accuracy of the deep learning model	Req # 1, 26
6.2.2	Reliability and Availability of the web interface	Req # 27, 28
6.3	Robustness or Fault-Tolerance Requirements to handle errors within theweb interface	Req # 29
6.4.1	Capacity Requirement for multiple connections	Req # 31
6.4.2	Capacity Requirement to populate the database with images and video	Req # 32
6.6.2	Productization Requirement for multiple export types	Req # 37

8.1 Modules

Similarly, the following is a traceability table explicitly relating test cases to modules:

Table 8.2: Tests and Modules Relationships

Test #	Module
3.1(all subsections), 3.2.1, 4.4-4.6, 5(all sections), 6.2(all subsections) - 6.4(all subsections), 6.6.2	ASP.NET and DB
3.1.3, 4.3, 5.8.2, 6.1, 6.5	TensorFlow Models
3.13, 3.2.2, 4.1-4.2, 5.8.2, 6.2(all subsections), 6.6.1	Python HTTP Server

9 Changes After testing

The first of our major change was to the web interface. Though our testers reviewed it favourably – there were numerous references – but we were also given consistent criticism that an alternate colour scheme might be a slight improvement. As such we have agreed to experiment with those improvements. In addition to changing the colour scheme the layout of the website was also changed to provide a more modern approach to how the websites information was displayed.

Testing also revealed that a JavaScript application to allow continuous requests to the HTTP server as well as utilize a mobile platform would have greater applicability to the Guelph team. This improvement would come alongside a function on the website to display statistics. This would help insure greater availability. The ability to track the database live would be both useful to the testing team as well as an entertaining aspect for the product demo.

The last major improvement provided was an improved JavaScript plugin for drawing the skeleton on top of an image. Testing revealed that there was flicker with the drawn skeletons, and as such, could lead to false negatives, where a skeleton that appears inaccurate is actually just limited by our current skeleton drawing methods.