SOFTWARE REQUIREMENTS SPECIFICATION

for

CS 4ZP6 Capstone Project

Version 0.0

Prepared by Brendan Duke, Andrew Kohnen, Udip Patel, David Pitkanen, Jordan Viveiros

Using Volere Template Edition 13

McMaster Text to Motion Database

October 12, 2016

Contents

Revision History

Name	Date	Reason For Changes	Version
Brendan Duke	Oct. 7th, 2016	Initial Version	0.0

1 Project Drivers

1.1 The Purpose of the Project

1.1.1 The User Business or Background of the Project Effort

There is an existing project at the University of Guelph that aims to create a system for "computational storytelling". The goal of the computational storytelling project is to create a system that takes as input a basic story composed of five sentences, and outputs an animated movie based on the story, which is produced in collaboration between an AI and human director.

As an initial step in the computational storytelling project, the University of Guelph group requires a database of "human motion" that is stored with rich text annotations. Such a database is required as a source of training data for the computational storytelling project to use in their methods to convert text to animated motion.

No satisfactory database of human motion data that is stored with associated text descriptions exists currently. However, there are existing databases of videos of people doing various actions with accompanying text describing those actions, for example the Charades database or MSR-VTT.

Our McMaster group has been approached to assist in this initial step in the computational storytelling project in two ways. Firstly, we are to develop software, based on existing research, that is able to process video and derive human motion data (e.g. joint positions over time) from the video. Secondly, we are to utilize data from an existing database that already has text annotations, such as Charades, using our video-to-motion processing software to generate a new database that contains both rich text annotations and motion data.

1.1.2 Goals of the Project

The goal of this project is to create a database, web-interface to said database, and a deployable software bundle providing access to already-established human pose estimation methods. Creating this database, website and software suite will allow the larger text-to-motion project to use the relationships between motion data and text annotations developed through the pose estimation software in order to provide a pose and word pairing, which can be used for animation.

1.2 The Client, the Customer, and Other Stakeholders

1.2.1 The Client

The current clients for this project are Dr. Taylor and his graduate student Thor Jonsson. Dr. Taylor is the primary driver to develop a website and database where annotated motion information can be generated and pulled from as a growth point into the larger text to motion project. They will be using the database to train Recurrent Neural Networks (RNNs) that will pair actions and their pose found within the database to words or combinations found in the input story.

1.2.2 The Customer

The customers are included within the clients since building this database and website combination will be utilized by Dr. Taylors research team and their external partners. In addition to Dr. Taylor and his research team this project would appeal to anyone that needed a pairing of actions and pose estimations as the website would be readily available to others.

In general, customers of the product will be researchers in the machine learning community who are interested in multi-modal learning, and specifically in systems that link text to human motion. Said customers will have a high degree of knowledge related to machine learning theory. However, they cannot be assumed to have a high degree of skill in any programming language with a steep skill curve, such as C++ or Haskell. Also, the customer is unlikely to be willing to invest a large amount of time in learning how to use the software produced by the McMaster Text-to-Motion project.

1.2.3 Other Stakeholders

Other stakeholders affected by the project include Dr. He, our group's internal supervisor and teacher of the CS 4ZP6 Capstone Project course, and the team members of our group.

Dr. He is a professor at McMaster who may not have the same specialized research knowledge as members of Dr. Taylor's group. Dr. He requires an explanation of all aspects of the project, as she will be responsible for assigning a grade to the entire group. Dr. He will require updates on the progress of the group in the form of deliverables that are part of the CS 4ZP6 syllabus.

The members of our CS 4ZP6 capstone group, namely Brendan Duke, Andrew Kohnen, Udip Patel, David Pitkanen and Jordan Viveiros, are also stakeholders affected by the project. For the most part, our group members did not have any specialized knowledge related to the project before commencing, although that knowledge is being acquired as the project progresses. The group members will require full involvement in all aspects of the project, as well as supporting knowledge and direction from Thor Jonsson and Dr. Taylor.

1.3 Users of the Product

1.3.1 The Hands-on Users of the Product

User Category	User Role	Subject Matter Experience	Technological Experience
Dr. Taylor's Group	Using the database to train an RNN to create animations from text.	Master	Master
Other machine learning researchers	Using the product for any multi-modal machine learning use-case involving text and human motion.	Master	Journeyman. This user category cannot be assumed to have a high degree of skill in complex programming languages such as C++.
Amateur machine learning enthusiasts	Using the McMaster Text-to-Motion database and software suite to learn about multi-modal machine learning and human pose estimation.	Journeyman	Journeyman

1.3.2 Priorities Assigned to Users

Our **key users** are members of Dr. Taylor's research group. **Secondary users** are other members of the machine learning community. Amateur machine learning enthusiasts are **unimportant users**.

1.3.3 User Participation

Thor Jonsson and Dr. Taylor will be expected to assist in supporting our group with their domain knowledge of deep learning methods. They will also be expected to participate in shaping the interfaces to the product (both the web interface and the programming interface to the database) by using the prototypes of those interfaces and providing feedback.

The minimum amount of participation from Dr. Taylor and Thor would be participation in a meeting with our group members on a bi-weekly to monthly basis, as well as participating in weekly correspondence electronically (e.g. by e-mail).

1.3.4 Maintenance Users and Service Technicians

Maintenance users would certainly be members of Dr. Taylor's research group, as they will be using the software produced by the project after its completion and may need to add changes to the product.

Once the product is open-sourced into the community, maintenance users could range from machine learning researchers to amateur machine learning enthusiasts. These users could be expected to fix bugs or add new features that were not in the initial scope of the project.

2 Project Constraints

Note that constraints also use the requirement shell, and therefore also use the requirement numbering.

Customer satisfaction and dissatisfaction have been evaluated in the "Priority" entry.

Also there are no conflicts between requirements, or supporting materials, so these entries have been dropped.

2.1 Mandated Constraints

2.1.1 Solution Constraints

Req. #: 1 **Req. Type**: 2.1.1 Solution Constraint **Use Case #**: 3

Description: The human pose estimation component should use deep learning methods.

Rationale: This constraint is to allow Dr. Taylor's group to integrate the software into their existing text-to-motion pipeline.

Originator: Dr. Graham Taylor

Fit Criterion: Dr. Taylor should confirm that the deep learning methods used in the human pose estimator are satisfactory.

Priority: High **History**: Created September 26th, 2016.

Req. #: 2	**Req. Type**: 2.1.1 Solution Constraint	**Use Case #**: 4

Description: Use a standard format such as LMDB or HDF5 for storing text-motion data.

Rationale: Having the data in a standard format will enable users to re-use existing code to manipulate that data.

Originator: Thor Jonsson

Fit Criterion: Run a set of existing tests to manipulate the standard data format (e.g. LMDB) and assert that those tests must pass.

Priority: High **History**: Created October 3rd, 2016.

2.1.2 Implementation Environment of the Current System

Req. #: 3	**Req. Type**: 2.1.2 Implementation Environment	**Use Case #**: 3

Description: The Text-to-Motion Software Suite must run under Linux.

Rationale: Linux is the operating system used by the Guelph Machine Learning research lab, and also the most commonly used operating system in the research community.

Originator: Dr. Graham Taylor

Fit Criterion: Automated builds and testing should pass on popular Linux distributions: Ubuntu, Fedora and RHEL.

Priority: High **History**: Created September 26th, 2016.

Req. #: 4	**Req. Type**: 2.1.2 Implementation Environment	**Use Case #**: 3

Description: Major APIs to the Text-to-Motion database must be accessible from the Python programming language.

Rationale: Python is the language used by the rest of Dr. Taylor's text-to-motion pipeline. Python is a popular, easy-to-use, and quick-to-prototype language, and is therefore one of the most favoured programming languages among the Machine Learning research community.

Originator: Dr. Graham Taylor

Fit Criterion: There must be hooks to all major interfaces written in Python, and there must be tests that are directly testing the Python interfaces.

Priority: High **History**: Created September 26th, 2016.

2.1.3 Partner or Collaborative Applications

The McMaster Text-to-Motion database software will collaborate with the rest of the University of Guelph group's text-to-motion pipeline. While their application is still in a research stage, we can expect that their application will be written in Python.

Furthermore, the animation component of Dr. Taylor's group's project will be using the proprietary software application Muvizu.

2.1.4 Off-the-Shelf Software

There are a number of off-the-shelf libraries that can and should be used by this project in order to implement the requirements. In particular, deep learning libraries should be used for the training of neural networks and for doing numerical computations.

Two notable deep learning frameworks are listed below.

Caffe, originally developed by the Berkeley Vision and Learning Center (BVLC), is a deep learning framework written in C++ with Python and Matlab wrappers.

Caffe has a strong ConvNet implementation and is popular amongst the computer vision community. However, Caffe's RNN implementation and support for language models is lacking compared with other libraries.

Caffe is released under the BSD 2-Clause license.

TensorFlow, originally developed by researchers at Google, is a deep learning and numerical computation framework written in C++ and Python.

TensorFlow has a clean, modular architecture and since it provides both Python and C++ interfaces, code can be prototyped in a rich high-level interpreted language before

deployment in a high-performance, scalable environment.

TensorFlow is open sourced under the Apache 2.0 open source license.

It will also be necessary to use a library for accessing video codecs. This is in order to convert video streams to and from sets of images, which will be input to the human pose estimation software.

FFmpeg is a leading multimedia framework that is able to decode and encode video, in addition to having many other features. FFmpeg can be utilized for any video encoding and decoding purposes in the project.

To implement search-by-text functionality into the McMaster Text-to-Motion Database, it will be helpful to leverage an existing software package that implements full-text search.

Sphinx is such a software package, as it implements full-text search. Sphinx also integrates well with SQL databases.

Sphinx is licensed under GPL version 2.

2.1.5 Anticipated Workplace Environment

As the McMaster Text-to-Motion Database is a software product, there are no requirements considerations specifically pertaining to the anticipated workplace environment of users of the product. Any considerations of environment, such as operating system environment, have already been covered under the "Implementation Environment" section.

2.1.6 Schedule Constraints

Req. #: 5	**Req. Type**: 2.1.6 Schedule Constraint	**Use Case #**: 3

Description: A demonstration of the product must be completed by February 13th, 2017.

Rationale: This is a deadline that is part of the CS 4ZP6 course.

Originator: Dr. Wen Bo He

Fit Criterion: By February 13th, 2017, fully functional versions of the web interface, database interface and human pose estimation software must all be working as verified by Dr. He.

Priority: High **History**: Created September 26th, 2016.

Req. #: 6	**Req. Type**: 2.1.6 Schedule Constraint	**Use Case #**: 3

Description: The project must be completed by April 5th, 2017.

Rationale: The project is part of the CS 4ZP6 Capstone Project course.

Originator: Dr. Wen Bo He

Fit Criterion: All documentation, testing and implementation must be completed and checked in to GitHub by April 5th, 2017.

Priority: High **History**: Created September 21th, 2016.

2.1.7 Budget Constraints

There is no budget allocated for this project. As such, hardware requirements such as GPUs to carry out the computations required for the project must be provided by the client, Dr. Taylor, or borrowed through other means such as seeking help from McMaster professors.

2.2 Naming Conventions and Definitions

2.2.1 Definitions of All Terms, Including Acronyms, Used in the Project

The Project when used, is referring to the McMaster Text to Motion Database project. The project aims to generate a database of human pose estimation model information that is linked to videos of human motion containing rich text annotations.

Human Pose Estimation is the process of estimating the configuration, or pose, of the body based on a single still image or a sequence of images that comprise a video. Human pose estimation may find the chin, radius, humerus, and other bone and joint positions.

Charades is a dataset composed of approximately 10K videos of daily indoor activities, complete with associated action-describing sentences, collected through Amazon Mechanical Turk[1].

MSR-VTT, standing for "Microsoft Research Video to Text", is a large-scale video benchmark for the task of translating video to text. MSR-VTT provides 10K video clips spanning 41.2 hours and containing 200K clip-sentence pairs in total[2].

Feedforward Neural Networks are artifical neural networks where connections between the units do *not* form a cycle). They are the simplest type of neural network, because information moves in only one direction.

ConvNets or **Convolutional Neural Networks** are a type of feed-forward artificial neural network. ConvNets are inspired by the visual cortex and are commonly used in

visual recognition applications.

RNNs or **Recurrent Neural Networks** are a class of artificial neural networks where units form a directed cycle, in contrast with feed-forward neural networks.

Deep Belief Networks are a type of deep neural network composed of multiple layers of "hidden units" (variables that are not observable), with connections between layers but not between units of a given layer.

Multi-modal neural language models are models of natural language that can be conditioned on other modalities, e.g. high-level image features[3].

Muvizu is an interactive 3D animation package designed for quick storytelling.

An **autoencoder** is an artificial neural network used to learn a representation of a set of data. The simplest form of an autoencoder is a feed-forward neural net with an input layer, an output layer, and one or more hidden layers. Instead of being trained to predict an output Y from input X, an autoencoder is trained to reconstruct its own input X' from X.

Ground truth refers to the expected classification, i.e. the expected output in a machine learning classification problem, based on a given input.

CMU Graphics Lab Motion Capture Database is a dataset of human motions that were collected at the Motion Capture Lab of Carnegie Mellon University. The motion capture data was collected by recording the movements of humans wearing a body suit with 41 markers taped on. The 3D motion data in this dataset is available in two formats (.vsk/.v or .asf/.amc), which describe a *skeleton and its joints* and *movement data*, respectively.

2.2.2 Data Dictionary for any Included Models

Here we define the different information flows displayed in Figure 3.1, the work context diagram.

A **skeleton overlay** refers to a set of human joint and body-part estimation data that has been associated with a given video stream. A skeleton overlay may also encompass an actual graphical stick-figure, displayed on top of the video stream, with circles indicating the position of each joint.

Text annotated video refers to a video that has a time-stamped natural-language description associated with it. Similarly, **text annotated motion data** refers to human pose data that is time-stamped and coupled with a natural-language description of the actions comprised by those human pose data.

2.3 Relevant Facts and Assumptions

2.3.1 Facts

- The training of deep neural networks can take on the order of days to weeks. For example, training two of the different neural network architectures used in [4] on four NVIDIA GTX Titan GPUs took 3 and 7 days, respectively.

- ConvNets take a fixed size input and generate a fixed size output, whereas RNNs can handle arbitrary input/output lengths. This is one reason why ConvNets are more popular in vision applications, while RNNs are used for language applications. However, neural network architectures can be used that combine both ConvNets and RNNs.
- The uncompressed size of the Charades dataset is 55GB at full resolution, and 16GB at 480p.
- The goals of this project, specifically generating a database of text-annotated human motion data, constitute new research and there are no existing solutions available.

2.3.2 Assumptions

- It has been assumed that it is possible to re-use existing software components in order to achieve the human pose estimation use-case, as it would take a full eight months just to implement an algorithm such as that of [4] from scratch.
- It has been assumed that this project will not encompass generating a dataset of videos with text annotations, and that we will be able to rely on an existing dataset such as the Charades dataset.
- It is assumed that a storage format for human pose estimation data already exists, and will be available for this project to leverage.

3 Functional Requirements

3.1 The Scope of the Work

3.1.1 The Current Situation

There is a large amount of existing research into human pose estimation, which this project will leverage. Based on Constraint 1, we focus on existing solutions that use deep learning methods.

[4] present a ConvNet architecture for human pose estimation from videos, which is able to benefit from temporal context across multiple frames using optical flow. This work is focused on upper-body human pose estimation only.

[5] propose a ConvNet model for predicting 2D human body poses in an image. This model is able to achieve state-of-the-art results using a simple architecture, and draws on the work done in [4].

[6] introduces *Convolutional Pose Machines (CPMs)* for pose estimation in images. CPMs consist of a sequence of ConvNets that iteratively produce 2D belief maps. Source code from this paper is available, and makes use of the Caffe deep learning library.

In [7], the authors estimate 3D full-body human poses from a monocular image sequence. They train a ConvNet to predict the locations of joints in 2D, then estimate human poses in 3D via an Expectation-Maximization algorithm over the entire video sequence.

On the animation side of the larger computational storytelling problem our project is assisting in solving, [8] use deep learning to synthesize character movements based on high-level parameters such as a curve that the character should follow. The learned motion is represented by the hidden units of a convolutional autoencoder.

3.1.2 The Context of the Work

The context diagram, displaying adjacent systems to this project, is shown in Figure 3.1.

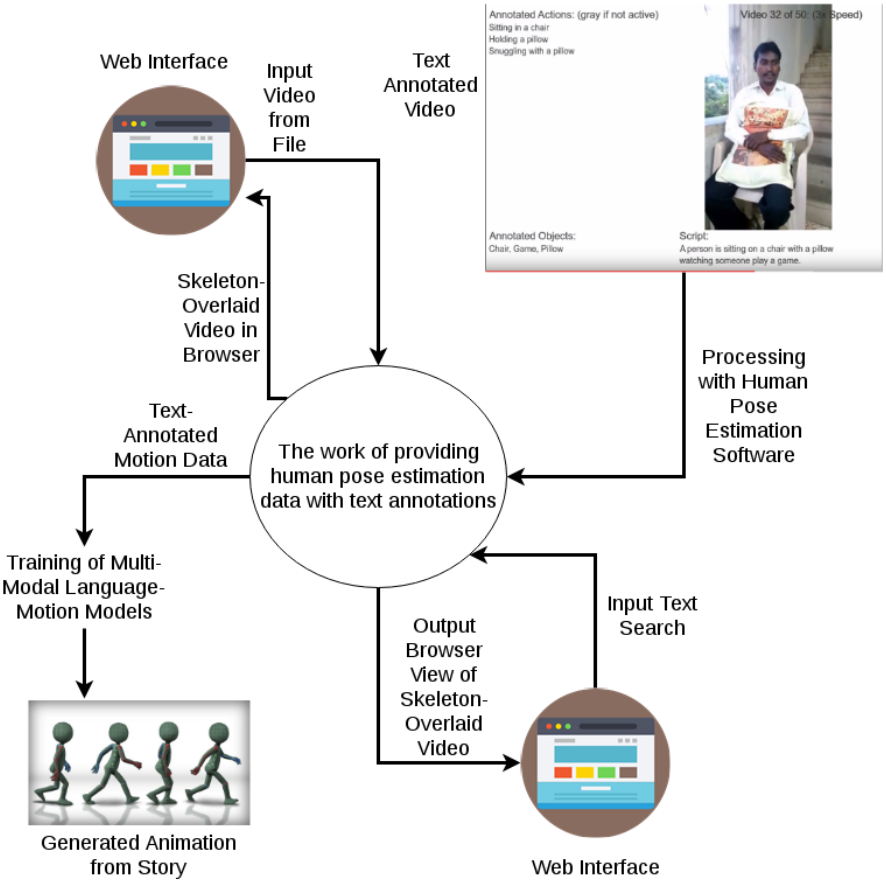
3.1.3 Work Partitioning

The events/use-cases for the project are laid out in Table 3.1.

Table 3.1: Business Event List

Event #	Event Name	Input and Output	Summary
1.	Web Interface Skeleton Overlay	**IN**: An image or video with humans in it. **OUT**: The same image or video, with a skeleton overlaid on top of all humans indicating their bone and joint positions.	Allow users to observe the human pose estimation component in real time through a web interface.
2.	Web Interface Text-to-Motion	**IN**: Word or phrase describing a human pose or action. **OUT**: Rich-text-annotated video corresponding to the input word/phrase, complete with overlaid skeleton.	Allow users to see the output of searches on the database using pose and/or action keywords, such as "run" or "kneeling".
3.	Software Interface Skeleton Overlay	**IN**: A stream of video with humans depicted. **OUT**: A set of human pose estimations corresponding to the video, in a standard data format.	Users should be able to use the human pose estimation solution to generate their own motion data set.
4.	Database Interface Text-to-Motion	**IN**: Word or phrase describing a human pose or action. **OUT**: Video in common encoding (e.g. MP4), associated rich-text-annotations, and human pose estimations in a standardized format.	Provide users direct access to the raw motion-estimation data format based on action-keyword database lookup.

Figure 3.1: McMaster Text-to-Motion Database Work Context Diagram



3.2 The Scope of the Product

3.2.1 Product Boundary

We do not include a product boundary diagram, and instead refer to Figure 3.1.

All interfaces to the McMaster Text-to-Motion Database will be software interfaces except the web interfaces, which will have a human component. There will be a search box requiring user input to do text-to-motion lookups, and there will be a graphical web interface for users to upload videos to use the skeleton-overlay functionality.

3.2.2 Product Use-case List

The product use cases correspond exactly to the business events list, and therefore to see product use-cases one should refer to Table 3.1.

3.2.3 Individual Product Use Cases

Use cases 1 and 2 would be used in a scenario where the user is looking to do experiments with immediate feedback, in order to gain intuition about the product's functionality. E.g. a researcher may want to use the Web Interface Skeleton Overlay feature (use case 1) to visually gauge the human pose estimation software's accuracy before spending the time to process a large amount of video data.

Use cases 3 and 4 could be used in a more large scale experiment. E.g. human motion data could be attached to a large amount of video data using use case 3, or a large amount of text-annotated motion data could be retrieved from the database programmatically using use case 4.

3.3 Functional and Data Requirements

3.3.1 Functional Requirements

Req. #: 7	**Req. Type**: 3.3.1 Functional Requirement	**Use Case #**: 3

Description: The text-to-motion software suite will provide an API to read individual frames in RGB format from a video stream. At least MP4, MP2 and AAC must be supported.

Rationale: Researchers may wish to do their own processing on RGB frames before feeding those frames into the human pose estimation module.

Originator: Brendan Duke

Fit Criterion: For a given set of test video streams, the frame-capture API must produce RGB frames identical to known reference frames.

Priority: Moderate **History**: Created October 5th, 2016.

Req. #: 8 **Req. Type**: 3.3.1 **Use Case #**: 3

Functional Requirement

Description: The skeleton overlay software must output a set of human pose data that is time-stamped and accurate to the ground truth joint positions in the frames corresponding to those time stamps.

Rationale: If the software does not have accurate output, then the output cannot be used to generate human motion.

Originator: Brendan Duke

Fit Criterion: A set of double-blind tests should be designed that allows test subjects to compare the human pose data output from the McMaster Text-to-Motion software suite again human motion skeletons rendered from the CMU Graphics Lab Motion Capture Database. The fit criterion would be that the human motion data produced by the McMaster software is statistically indistinguishable from CMU's MoCap data from sensors on actual humans.

Priority: High **History**: Created September 26th, 2016.

Req. #: 9 **Req. Type**: 3.3.1 **Use Case #**: 4

Functional Requirement

Description: The domain of inputs to the text-to-motion database search should output a range that covers the entire set of videos in the database.

Rationale: There is text associated with each video, therefore for every video there should be some set of matching text that can retrieve it.

Originator: Brendan Duke

Fit Criterion: A test application could be designed to intentionally pull text data from the description of each video and assert that the video from which that text data was obtained appears in the output of the search.

Priority: High **History**: Created October 10th, 2016.

Req. #: 10	**Req. Type**: 3.3.1 Functional Requirement	**Use Case #**: 4

Description: The videos retrieved from a text-to-motion search should not contain any results with text descriptions not containing the search terms.

Rationale: False positives would make the text-to-motion retrieval difficult to use, since returned videos would not reliably match the search terms.

Originator: Brendan Duke

Fit Criterion: A test application could be designed to, for each video, search a random set of terms not in that video's text description and assert that the given video is not returned in the search results.

Priority: High **History**: Created October 10th, 2016.

Req. #: 11	**Req. Type**: 3.3.1 Functional Requirement	**Use Case #**: 4

Description: The videos retrieved from a text-to-motion search should be ordered by relevance.

Rationale: Users will want to use the motion data that most closely fits the text they have searched for.

Originator: Brendan Duke

Fit Criterion: A smaller set of test data can be created by hand and sorted by a human, and the results can be compared against an expected output, also generated by hand. The output from a larger set of test data could be compared against the output from a known full-text search engine, such as Sphinx, or if we are using Sphinx, then Lucene.

Priority: High **History**: Created October 10th, 2016.

3.3.2 Data Requirements

The McMaster Text-to-Motion project is not strictly adhering to an object-oriented design, and therefore there is no class model that must be followed as part of the requirements.

Interactions between classes, or modules, will be diagrammed as part of the module

hierarchy during the design iterations of the project development.

4 Nonfunctional Requirements

4.1 Look and Feel Requirements

4.1.1 Appearance Requirements

Req. #: 12	**Req. Type**: 4.1.1 Appearance Requirement	**Use Case #**: 1

Description: The web interface shall have a color-scheme that is focused on contrast.

Rationale: A color shceme with bright and dark colors clashing is bound to captivate users of the website.

Originator: Udip Patel

Fit Criterion: At least 6 out of every 10 users should be left with a positive impression of the color scheme.

Priority: Low **History**: Created October 8th, 2016.

4.1.2 Style Requirements

Req. #: 13	**Req. Type**: 4.1.2 Style Requirement	**Use Case #**: 1

Description: The web interface shall be designed to look minimalistic (in terms of text) and visually informative through graphs and other visual aides.

Rationale: To be able to convey the complex processes of a deep learning model in a simple way that can be understood by any end user, regardless of their level of knowledge. The amount of text displayed should also be minimized to avoid overwhelming a first-time user with too much information.

Originator: Udip Patel

Fit Criterion: The web interface shall be able to display a graph that tracks how the deep learning model analyzes an input image. This is similar to an open-source website called TensorFlow Playground.

Priority: Low **History**: Created October 8th, 2016.

4.2 Usability and Humanity Requirements

4.2.1 Ease of Use Requirements

Req. #: 14	**Req. Type**: 4.2.1 Ease of Use Requirement	**Use Case #**: 1

Description: The functionalites of the web interface shall be easy to use for any casual internet user above the age of 12 who understands how to upload and download files to a website.

Rationale: Having a simpler interface can open up this web app to a lot of potential users.

Originator: Udip Patel

Fit Criterion: Any end user at least 12 years of age shall be able know how to upload an image to the web interface within 15 seconds of visiting the website.

Priority: Low **History**: Created October 10th, 2016.

Req. #: 15	**Req. Type**: 4.2.1 Ease of Use Requirement	**Use Case #**: 1

Description: The user shall not be expected to remember anything to be able to interact with the Web Interface.

Rationale: This is just to reduce the amount of work that the user has to do in order to be able to get the desired result.

Originator: Udip Patel

Fit Criterion: The Web Interface shall only ask for images, videos or text from the user, and without prompting the user again, it will return the given image or video (with or without text annotations).

Priority: Low **History**: Created October 10th, 2016.

Req. #: 16	**Req. Type**: 4.2.1 Ease of Use Requirement	**Use Case #**: 2

Description: The user shall be able to type in a descriptive word or phrase into a text box that will be easy to find from the web interface.

Rationale: This way, the functionality provided by the web application will be made obvious and accessible to the user.

Originator: Udip Patel

Fit Criterion: The end user shall be able to input one word or phrase into the text box (written in HTML/CSS) using their keyboard, and be able to click on a 'Submit' button to view the functionality of thew web app.

Priority: Medium **History**: Created October 10th, 2016.

4.2.2 Personalization and Internationalization Requirements

As of Revision 0, there are no requirements for Language support. Since the web Interface and the Software Interface would rely on Natural Language Processing for English, adding language support for multiple languages would be too large of a feature to implement for now.

4.2.3 Learning Requirements

Req. #: 17 **Req. Type**: 4.2.3 Learning Requirement **Use Case #**: 1

Description: Any end-user shall be able to use the web application and generate a skeleton-overlay without having any prior training.

Rationale: This is because all the user needs to be able to do is upload and download a file to/from the website.

Originator: Udip Patel

Fit Criterion: Any user that visits the website shall be able to understand how to use the skeleton-overlay functionality within 1 minute of visiting the website.

Priority: High **History**: Created October 10th, 2016.

Req. #: 18 **Req. Type**: 4.2.3 Learning Requirement **Use Case #**: 2

Description: Any end-user shall be able to use the web application to instantiate a 'text-to-motion' event without having any prior training.

Rationale: This is because all the user needs to do is type in a descriptive word or phrase into a text box.

Originator: Udip Patel

Fit Criterion: Any user that visits the website shall be able to understand how to use the 'text-to-motion' functionality within 30 seconds of visiting the website.

Priority: High **History**: Created October 10th, 2016.

Req. #: 19	**Req. Type**: 4.2.3 Learning Requirement	**Use Case #**: 3

Description: The functionalities of the software interface shall be easily understood by programmers who have experience with APIs.

Rationale: This is because using a command line API still takes some base knowledge of software to set up, and we cannot assume that every user will be able to set up our API without any problems even if they follow all of the detailed steps.

Originator: Udip Patel

Fit Criterion: Any user should be able to set up the Software Interface and be able to use its functionalities (described in events 3 and 4) through the command line interface within 15 minutes.

Priority: Medium **History**: Created October 10th, 2016.

4.2.4 Understandability and Politeness Requirements

Req. #: 20	**Req. Type**: 4.2.4 Understandability and Politeness Requirement	**Use Case #**: 1

Description: The end user of the web interface shall not be able to see the inner workings of the deep learning model that is used to estimate the human pose or action from an image/video

Rationale: This is because the user should be obscured from having to deal with the complexities of deep learning. Furthermore, it would be computationally expensive to show the users such low-level detail in a way that they could understand it.

Originator: Udip Patel

Fit Criterion: The end user of the web interface shall be able to get a valid output from the skeleton-overlay function without seeing any of the deep learning code or processes.

Priority: High **History**: Created October 10th, 2016.

Req. #: 21	**Req. Type**: 4.2.4 Understandability and Politeness Requirement	**Use Case #**: 4

Description: The end user of the software interface shall not know how the program is able to map a word or phrase to a human pose or action

Rationale: Using the software interface should be simple, and if we show these highly complex data mappings to the user, we might risk overwhelming the user by giving them a large amount of information that is of very little use.

Originator: Udip Patel

Fit Criterion: The end user of the web interface shall be able to get a valid output from the software interface without being exposed to the database mappings that are used to determine the valid output.

Priority: High **History**: Created October 11th, 2016.

4.2.5 Accessibility Requirements

As of Revision 9, there are no accessbility requirements for this application. Since both interfaces only contain a virtual/visual component and no physical mechanisms, there is no need to provide any accessibility requirements for now

4.3 Performance Requirements

4.3.1 Speed and Latency Requirements

Req. #: 22	**Req. Type**: 4.3.1 Speed and Latency Requirement	**Use Case #**: 2

Description: The web interface shall be able to connect to an external database with no problems and be able to store items and query for them quickly

Rationale: The speed is necessary here because generating the actual output itself is going to be computationally intensive so the program should not be spend a lot of time waiting on responses from the database.

Originator: Udip Patel

Fit Criterion: The web interface should be able to establish an initial connection to the database within a span of 5 seconds. The web interface should be able to transfer a file of size 500MB to the database in around 50 seconds, assuming an upload speed of 10Mbps. The web interface should be able to transfer a file of size 500MB from the database to the user in around 25 seconds, assuming a download speed of 20Mbps.

Priority: High **History**: Created October 11th, 2016.

5 Project Issues

5.1 Open Issues

Here we discuss issues not addressed in the functional or non-functional requirements.

One open issue common to most machine learning problems is the quality of the fit, as this can always be improved upon.

We are using ready-made algorithms, and there are many such options. So another open problem is to choose amongst these pose estimation algorithms. A possible solution is, since there are multiple options that there could be an option where the user can select their own algorithm to analyze the data. However due to time constraints we will need to choose one without first testing. So one problem is simply choosing an algorithm that will work without having tested it first.

A second problem is the hardware that we intend to use for the problem. We are analyzing and performing pose estimations on an enormous database of video. This is a computationally intensive task so we need a fast computer hardware system to perform our computations. One problem is choosing the hardware solution to this problem. We will also be creating a web application so it may be in our best interest to host the application using a server of our creation. In this case we would also have to create and maintain this system.

5.2 Off-the-Shelf Solutions

5.2.1 Ready-Made Products

In choosing an algorithm for the pose estimation problem there is already software that is available online that implements these algorithms. We intend to use these resources in our project.

5.2.2 Reusable Components

We are creating an intermediate software for pre-processing data and placing this data in a database that can be easily queried through a user interface. The pre-processing and sorting/searching of data is a common task in machine learning development. For this reason it is hopeful that the product we develop is not specialized to the pre-processing algorithms we use or the database that contains the database we analyze.

Ideally we will create a multi-tiered system that will have many components that could be re-used for future data analysis and viewing tasks.

5.2.3 Products That Can Be Copied

The product that we aim to create will implement machine learning algorithms on a database of media in order to produce a new database of processed media.

Some popular machine learning algorithms are the Short Long Term Memory (LSTM) neural networks. Due to their popularity many web applications have been created which allow users to visualize the way that these networks learn and they allow them to see the output of the networks for given input. Two such web applications are the TensorFlow playground produced by Google and Visual Analysis of Recurrent Neural Networks offered by Harvard. We will aim to reproduce many of the features that are present in these products.

The machine learning algorithms typically have many parameters in them which can be adjusted to optimize their performance. The previously mentioned products let the user choose these parameters through the user interface. We will aim to implement these features as well.

5.3 New Problems

There are several problems which we wish to avoid when creating our system. Our project is a small part of a larger project. Each of the sub-projects have been broken down as a modular hierarchy. We hope that our system will be able to work with the other modules without corrupting their performance and correctness.

Our project will house a large database of media data. The database will need to be queried and the data manipulated. However through our manipulation we will not want the data stored in the database to be corrupted so that the performance of our system degrades the more it is used. We will also aim to make it so that our program does not cause side-effects on the larger projects that make use of it.

5.3.1 Effects on the Current Environment

Currently if someone wishes to analyze the video data offered online they will have to analyze raw video data. Our application will offer a new interface in which this data has been processed in a manner that is useful if the user should wish to analyze the human motion in the video.

The alternative web interface we offer will be different from the previously made available systems. However the format of the video will be preserved in our transformations.

The finished product will exist as a standalone web-application hosted on a server. The server will allow queries to be made. We hope that the system will thereby be able to exist independently of the larger project of which it is a part of, so that our product will not have to be installed on every user's computer.

5.3.2 Effects on the Installed Systems

The transformations that we will apply to the video data should be very useful for

machine learning researchers. Having transformed data can speed up the learning rate in many machine learning algorithms and we expect the product will benefit users.

5.3.3 Potential User Problems

Users of our software should not suffer any adverse or ill effects due to said usage, so this section is not applicable.

5.3.4 Limitations in the Anticipated Implementation Environment That May Inhibit the New Product

We refer here to constraints 3 and 4. While the Linux constraint should not inhibit the new product, the limitation of using a Python API may present issues with performance, scalability, and maintainability in the long term.

5.3.5 Follow-Up Problems

As we have mentioned this project is a sub-component of a larger project that is meant to facilitate the change in representations from text to motion. In this current project we are only analyzing the problem of pre-processing the data. So, another problem that will need to be considered in the future is how the data we generate can be used to train a deep neural network to make the previously mentioned transformation from text to motion.

5.4 Tasks

5.4.1 Project Planning

To develop our software product we will make use of the iterative and incremental software development process shown in Figure 5.4.1.

5.4.2 Planning of the Development Phases

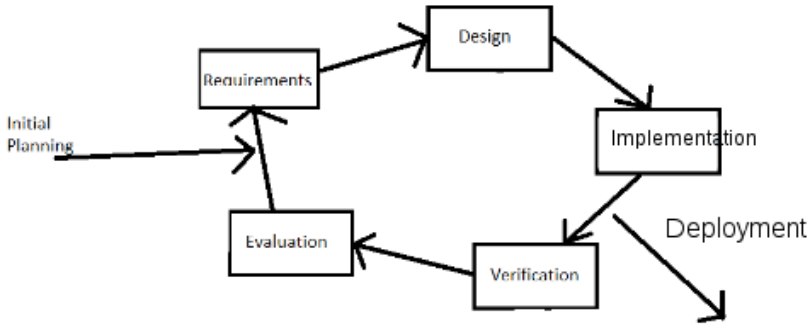
We have passed through the initial planning step. In this step we decided on a project to pursue, and explored whether or not the idea was achievable by talking to researchers and exploring published papers in this area.

The next step of the project is the requirements stage where we formally state the objectives of our project.

An important feature of this stage is that the process: Requirements, Design, Implementations, Verification and Evaluation can be run in parallel for each module of the product, followed by systems level testing at the end. This method of unit testing in parallel with development ensures that each module in the system will run without producing errors with the other parts.

Our system involves several separate components. It is a web application and needs to store a lot of data in a database. So the program will consist of a data layer, a layer that

Figure 5.1: Iterative and Incremental Software Development Process



performs the machine learning algorithm on the data and a layer for the user interface. Each of these separate layers can be developed in parallel in our system.

5.5 Migration to the New Product

5.5.1 Requirements for Migration of the New Product

As we have mentioned in Reusable Components (Section 5.2.2) that we plan to use the implemented versions of the machine learning algorithms that are already available online by the authors of the papers. However much of this code is written in Matlab and since the program is to be Python based some of the code will need to be rewritten in this language. However since we are re-writing the code we will be able to test our output by comparing it to the output of the older code we are rewriting.

5.5.2 Data That Has to Be Modified or Translated for the New System

If we choose to migrate existing software from one deep learning framework to another, e.g. from Caffe to Tensorflow, it may be necessary to regenerate the model corresponding to the neural network architecture on one framework into a model that is compatible with the new framework.

5.6 Risks

We are taking algorithms that are meant to analyze images and applying them to video. The transition from image to video is a step up in computational tractability. There is a risk that the increase in volume of data will cause performance issues for our program.

Another worry is the possibly that algorithms will not generalize well from images to video. They have not been tested on these tasks so it is possible that they will create very noisy data in their output and will not generalise well.

5.7 Costs

For our program we will be using freely available and open source programs. The list of programs we will make use of was mentioned in Off-the-Shelf Solutions (Section 5.2).

However, for our hardware we may wish to run our own server to host the application. In this case there will be maintenance and hardware costs. The maintenance cost will be in terms of depreciation of the equipment, the cost of connecting to a network and streaming data over the internet and power consumption. The power consumption, however, should be low.

The hardware we would need to purchase would include an NVIDIA GTX 1070 and an Intel i5 ($1600).

The reason for the NVIDIA GPU is to maintain compatibility with deep learning libraries, which are relying on NVIDIA's proprietary CUDA hardware abstraction layer at the lowest level.

The reason for an Intel i5 is so that the CPU will not bottleneck the GPU's performance.

Description	Cost
GTX 1070 Intel i5	$1600
Data Plan (approximate use?)	$10/month
Power	<$5
Maintenance of equipment	Unknown at this point

The second cost that we have to consider is the amount of time that the project will require in order to implement.

Our project has four use-cases discussed in Section 3.1.3 of our report. We estimate the amount of time required for one person to complete each use-case, assuming they spend 20 hours per week.

Due to the research nature of the project, these estimates are subject to change based on the success or failure of our prototyping efforts.

Functional Requirement	Time Necessary
Web Interface Skeleton Overlay	Two weeks implementation, one month testing and debugging.
Web Interface Text-to-Motion	One month implementation, two months of testing and debugging.
Software Interface Skeleton Overlay	One month research, one month implementation, two months of testing and debugging.

Database Interface text to motion	One month implementation, two months of testing and debugging.

5.8 User Documentation and Training

5.8.1 User Documentation Requirements

As one component of our application will be a standalone web application, we will have a help feature.

1. Technical Specifications document - specifies which algorithms our program uses and their expected accuracy
2. User Manual or help feature - instructs the user on navigating the user interface, provides information on the contents of the database as well as a list of contacts for additional help.

5.8.2 Training Requirements

The intended users of our program will be machine learning experts who have plenty of experience working with software interfaces. So the training these users will need to use our program will be minimal. The software will only have the four use-cases we have mentioned in our functional requirements.

5.9 Waiting Room

As we have mentioned we are using off the shelf version for the machine learning algorithms used in this project. A next step in our project would be to improve on these algorithms. The procedures are designed for working on images, but it would be interesting to optimize or change the algorithms so they take advantage of the fact that the data we are using is video and not strictly images.

We are at the moment focusing on human motion and labelling body parts. The motion of other objects such as cars and animals is also interesting. A next interesting step in this project would be to consider the motion of a more diverse range of objects.

5.10 Ideas for Solutions

The ultimate solution of the product is the generation of animated motion from text. This is the step that is to be taken after our project is complete.

Bibliography

[1] Charades dataset. Allen Institute for Artificial Intelligence. [Online]. Available: http://allenai.org/plato/charades/

[2] J. Xu, T. Mei, T. Yao, and Y. Rui, "Msr-vtt: A large video description dataset for bridging video and language." IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), June 2016. [Online]. Available: https://www.microsoft.com/en-us/research/publication/msr-vtt-a-large-video-description-dataset-for-bridging-video-and-language/

[3] R. Kiros, R. Salakhutdinov, and R. S. Zemel, "Unifying visual-semantic embeddings with multimodal neural language models," *CoRR*, vol. abs/1411.2539, 2014. [Online]. Available: http://arxiv.org/abs/1411.2539

[4] T. Pfister, J. Charles, and A. Zisserman, "Flowing convnets for human pose estimation in videos," *CoRR*, vol. abs/1506.02897, 2015. [Online]. Available: http://arxiv.org/abs/1506.02897

[5] V. Belagiannis and A. Zisserman, "Recurrent human pose estimation," *CoRR*, vol. abs/1605.02914, 2016. [Online]. Available: http://arxiv.org/abs/1605.02914

[6] S. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," *CoRR*, vol. abs/1602.00134, 2016. [Online]. Available: http://arxiv.org/abs/1602.00134

[7] X. Zhou, M. Zhu, K. Derpanis, and K. Daniilidis, "Sparseness meets deepness: 3d human pose estimation from monocular video," in *CVPR*, 2016.

[8] D. Holden, J. Saito, and T. Komura, "A deep learning framework for character motion synthesis and editing," *ACM Trans. Graph.*, vol. 35, no. 4, pp. 138:1–138:11, Jul. 2016. [Online]. Available: http://doi.acm.org/10.1145/2897824.2925975