Using Natural Language Processing for Automatic Generation of Animated Dialogue Scenes in Video Games

Mikolaj Panasiuk

A dissertation submitted in partial fulfilment
of the requirements for the degree of
Bachelor of Science
of the
University of Aberdeen.

Department of Computing Science

2018

Declaration

No portion of the work contained in this document has been submitted in support of an application for a degree or qualification of this or any other university or other institution of learning. All verbatim extracts have been distinguished by quotation marks, and all sources of information have been specifically acknowledged.

Signed:

Date: 2018

Abstract

Modern video games (especially those of the RPG genre) can feature over a hundred hours of dialogue. During a dialogue scene the characters are expected to behave naturally and use body language in a way that is 'normal' and corresponds to what they are saying. Due to the sheer amount of dialogue animation needed it is impossible to animate all the scenes manually. Many solutions to automatic generation of such scenes (most of which are not available publicly and produce animations that still require a fair amount of manual work) have been tried with mixed results.

This project aimed to develop a tool capable of generating 3D animated dialogue scenes with virtually no manual labour required. This is achieved by using natural language processing to extract the mood of characters directly from script and using a database of pre-recorded motion capture animations to assemble a dialogue scene. Additionally, the tool was aimed to be cross-platform, easy and cost-efficient to use, as such a tool would be most useful for small teams and amateur developers.

The tool was evaluated by a real audience whose task was to compare scenes from actual games with scenes generated by the developed software. The tool succeeded in creating rather convincing animation that was often described as more enjoyable or engaging than the animations from games; Unfortunately the generated scenes lacked realism and the NLP approach was not accurate enough when dealing with sarcasm or ambiguity. Because of those issues the project did not achieve a full success.

The software itself would need some serious improvements before becoming commercially viable. It is however developed enough to prove the usefulness of NLP in game animation. This approach has a potential to be incorporated into a larger animation generation framework (where other approaches besides NLP collaborate to create the most lifelike animations) and even to be extended into other areas of computing science such as robotics.

Acknowledgements

First and foremost I would like to thank Professor Ehud Reiter for his invaluable guidance, supervision and patience.

Secondly, I would like to thank the Blender Foundation for providing free, open-source, modern 3D content creation tools for over 15 years and the Max Planck Institute for Biological Cybernetics for making the Emotional Body Motion Database available publicly.

I would also like to thank the BioWare Montreal studio for developing *Mass Effect: Andromeda*. The game famously suffered from multiple issues in many areas of game development. Its weird and unnatural dialogue animation constantly broke the player's immersion and turned serious moments into comedic scenes. The following blank-verse poem commemorates the achievements, the failure and the disbandment of the BioWare Montreal studio:

BioWare Montreal

It seems like we hardly knew you,
probably because you only existed for nine years and were relatively small
The more surprising is the fact that
you have been put in charge of the most significant of BioWare's franchises

Pity,
because you looked like a pretty cool place to work at
Even though Glassdoor reviewers complained that there are no windows at the office
and that they all feel like nocturnal creatures.
Montreal also seems like a good place to live I guess
despite all the snow in the winter and the roadwork in the summer
But how would I know, I have never been to Canada anyway

You should have went gently into that good night
But you didn't
And now your game is a laughing stock for the entire gaming community
Press F to pay respects
F

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The game industry now is bigger than ever before and still growing. Along with technological advancements as well as a rise in popularity games themselves have become bigger and more polished. With increasing size and quality the number of man-hours required to build a game rises drastically. A lot of work is being put into creating tools that enable faster creation of content. However, there is still a lot left to be optimized and automated.

One domain of game development that drains lots of man-hours into monotonous processes that could potentially be automated is animation. Most studios will rarely animate everything by hand as there is simply too much content to cover. While some animated content needs to be very polished (cutscenes - action sequences, parts of game that greatly influence the plot development), some animation might be cruder (dialogue sequences). Games like the RPGs[1] will feature a lot of dialogue - during the dialogue the characters cannot just stand still as it would negatively impact player's immersion in the game world. The characters must move and perform gestures that underline their speech in a natural manner. These animations cannot be all done by hand because of their sheer cumulative length. For instance:

- Mass Effect Andromeda and Fallout: New Vegas feature 65,000 lines of dialogue[2][3].
- The Witcher 3 features roughly 35 hours of dialogue scenes[4].

The main challenge is to make the dialogue scenes (and other automatically generated animation) look indistinguishable from the cutscenes (usually manually created and well-polished). In many games a player will experience watching well-polished animation that immediately switches to poor, clunky and unrealistic animation. The lesser in quality dialogue scenes break the immersion of the player and negatively impact the overall experience. Easier, faster and better quality methods of generating dialogue scenes would be a great asset to the gaming industry.

1.1 Motivation

The purpose of this project is to develop a tool that helps generate animated dialogue scenes and minimizes the amount of manual work by using natural language processing. Generating the scenes directly from script would pose several benefits:

[1] RPG - Role-playing game.
[2] https://www.pcgamer.com/mass-effect-andromeda-has-over-1200-speaking-characters/
[3] https://www.pcinvasion.com/fallout-new-vegas-will-have-65000-lines-of-dialogue
[4] https://www.pcgamer.com/most-of-the-witcher-3s-dialogue-scenes-was-animated-by-an-algorithm/

- A script is often written to design the plot of a game. The same script could be fed into a program to generate the animations.
- The script is semi-structured natural language. Using natural language would help shorten the gap between artists, writers, animators and technicians.
- Such tool can be used to prototype scenes quickly and easily.
- Such tool program can be used by people who know nothing about animation.

The program I propose would create prototype animation with almost no amount of manual work required. Requiring less manual labour than other approaches would make this tool a very cost efficient way to create realistic dialogue scenes.

1.2 Objectives

The Projects Objectives are as follows:

Develop a tool able to interpret a natural language script

The tool must be able to read a semi-structured script and recognize dialogue lines and emotions of the characters.

Develop a tool able to blend a final dialogue scene

The tool must be able to output a fully editable dialogue scene. The scene is assembled using pre-made motion capture clips.

The scenes created by the software will be very crude and unpolished. Scenes generated by the tool will need to be polished manually - the amount of polish required can be decided by assessing the importance of a given scene. However there is a chance that the scene quality will be much inferior to scenes generated by similar tools that use different approaches. Therefore an important question this project tries to answer is whether taking the NLP approach to animation is feasible in the games industry given current technology.

Chapter 2

Background and Related Work

2.1 Background

Manually crafting every animation in the game is unrealistic due to cost and time requirements. Many games have employed various approaches to computer generated animation in order to generate hours of realistic content. No game however has succeeded in making the dialogue animation indistinguishable from manually animated cutscenes.

2.2 Existing Systems

There has been a variety of approaches featured in games. Many of them ended up generating dialogue scenes that are highly repetitive, not very realistic and in general not matching the sentiment and emotion of the speech with character movement. The only system that did not seek to find cheap workarounds around the issue and instead embraced the full complexity of the problem is the dialogue scene generator used in The Witcher 3. The system developed by CD Projekt Red made computer generated dialogue sequences in many cases barely distinguishable from those made by an artist, allowing less important scenes to be left completely untouched by a human animator. [6]

The tool created by CD Projekt Red takes information on initial state of involved characters (position, stance, emotions, etc.) and audio recording of the dialogue lines. The tool chooses matching pre-made animated clips and outputs a fully editable animated scene of characters conversing with one another. The tool uses audio recordings to aid the animated clips (analysing the audio waves may help decide when characters accent or put stress on some information). This tool is the current state-of-the-art and has produced the best effects in terms of amount of work to quality of animation ration. However, there are some serious drawbacks to this project. It still requires a fair amount of work as for every scene the initial state must be specified manually. Moreover, the system requires audio to be recorded first. The tool was never released to be used commercially outside CD Projekt. [15]

The tool I propose would hardly be able to compete with that of CD Projekt Red, however it would have some significant advantages. It would make generating the scenes even faster (requiring less manual work and preparation) and would in general be more appealing to people who are not experts on animation.

2.3 Related Work

The main focus of this project is the usage of NLP for generating animated sequences. While this project puts particular emphasis on generating scenes of dialogue, there exist a multitude of projects that explore the usage of NLP in animation in a variety of ways.

A very early research (1991) explores usage of NLP for creating animations that would help engineers demonstrate tasks in an easy and safe way (demonstrating tasks personally might be unsafe, reading manuals might be insufficient to understand the task in full) [9]. The system would take as input a set of natural language *directives* or *commands* (e.g. 'move cup to table'). The system would interpret such an instruction into a series of steps (tasks) that are carried out in a given order. Based upon that sequence an animation would be generated.

The project however seems to have a few significant problems. Most importantly, the end resulr was not editable. In my research I believe that the end animation will not be immediately satisfactory without any manual improvements and I believe that the outputted scenes should be fully editable. The other issue with this project is that the end result is not realistic or immersive (this was not a priority of that research, but is important for me). The animations were automatically generated in full, which I do not believe to be a viable approach for my project. To improve realism of the scenes, the animation should be created using motion capture clips.

A lot more research has been done in this area with certain degree of success. Another interesting paper explores a topic more similar to the focus of this report. 'Generating Animation from Natural Language Texts and Framework of Motion Database' proposes using a database of pre-recorder motion data instead of fully generating the movements [13]. The paper argues that motion synthesis requires too much manual setup (which a normal user might find too hard). The paper also proposes an architecture for a database - a way to store motion capture clips and their associated meta-data. The system described in this paper is however essentially different from what I propose in this report, as the described report focuses on action-driven animation, while my tool would focus on dialogue-driven animation.

While many other research focused on animation in general, some research focused on usage on NLP generated animation for use in games specifically. One aimed to create a parser that would transform natural language text to a set of instructions for the animation layer. The end goal was real-time character control by natural language commands. The animations were not intended to increase realism, but rather to accomplish goals in a strategic manner. The research was generally successful showing some potential usefulness of NLP in game development. [3]

So far the main focus of similar research has been generating animated scenes from text with focus on actions, often by employing motion synthesis. Some research focused on generating animation in real time rather than preparing a huge amount of animation in bulk.

2.4 Emotion Analysis

The task of emotion analysis is a subset of natural language processing. The task of analysing natural language text in search of subjective information such as sentiment or emotion is known as sentiment analysis.

2.4.1 Sentiment Analysis

Sentiment Analysis can be broadly defined as a computational approach to discovering opinions and attitudes expressed in text by opinion holders. In its most basic form it focuses on binary classification of the sentiment of the opinion holder (positive or negative) [14], but can be extended to mine for more complicated opinions such as emotions or detecting sarcasm. The most popular uses of sentiment analysis are predicting stock market behaviour and getting immediate feedback on products, political campaigns and decisions by monitoring social media [5].

Approach to sentiment analysis might differ depending on whether the focus is on document-level analysis or sentence level analysis - where document-level analysis assumes the entire document to have one clear area of focus, while sentence level analysis analyses each sentence individually. Apart from those, there are more types of sentiment analysis such as aspect-based analysis (which is used when the document or sentence does not focus on a single entity), or comparative analysis (when direct opinion is not desired, but it is needed to contrast opinions with each other) [5]. For the purposes of this project, it seems that the sentence-level approach is the most suitable, as dialogues comprise of mostly single-sentence lines (rarely more than three sentences per dialogue line) and because the emotional payload may change as the dialogue progresses.

Traditionally, sentiment analysis is performed by various classification methods (usually supervised learning). Naive Bayes classifier and support vector machines were proven to yield pretty accurate results (over 80% accuracy in general) [14]. The major constraint of those methods is that their quality is tightly linked with the quality of the lexicon used and the size and quality of training datasets.

One state-of-the-art solution to emotion analysis problem has recently became publicly available. Created by IBM, Watson Tone Analyzer is a tool capable of accurate emotion labelling (fear, joy, anger, etc.) as well as tone labelling (analytical, confident, tentative, etc.). The system is based on the *Big Five personality traits* model [4]. Thanks to this approach, Watson is capable of much more detailed analysis of natural language than most other tools.

2.4.2 Emotions

To fully understand how to approach the problem of animation generation from emotional natural language it is important to look at how can emotions be understood and represented in computational terms. Emotions are complex and subjective in nature, so it is important to deconstruct emotions into something that can be worked with. A traditional approach to this problem is using the six basic emotion model. Specified by Paul Ekman and Wallace V. Friesen, who by studying native Papua-new Guinean tribes have discovered six universally recognized and understood emotions: [2]

- Joy
- Anger
- Surprise
- Disgust
- Fear

- Sadness

Although various research proposes changes to the six basic emotions model (such as reducing it to just four basic emotions [8]), this model is widely accepted and used in literature and software (such as IBM Watson tone analyzer, or various lexicons such as the NRC Word-Emotion Association Lexicon [1]).

Psychologists theorize that the basic emotions can be used as building blocks to achieve more complex emotions. The specific emotion can be described as a mixture of basic emotions and their intensities. Robert Plutchik, a supporter of that theory, has represented emotions on a wheel-like diagram 2.1 that depicts how two basic emotions combine to create a more complex emotion. [2]

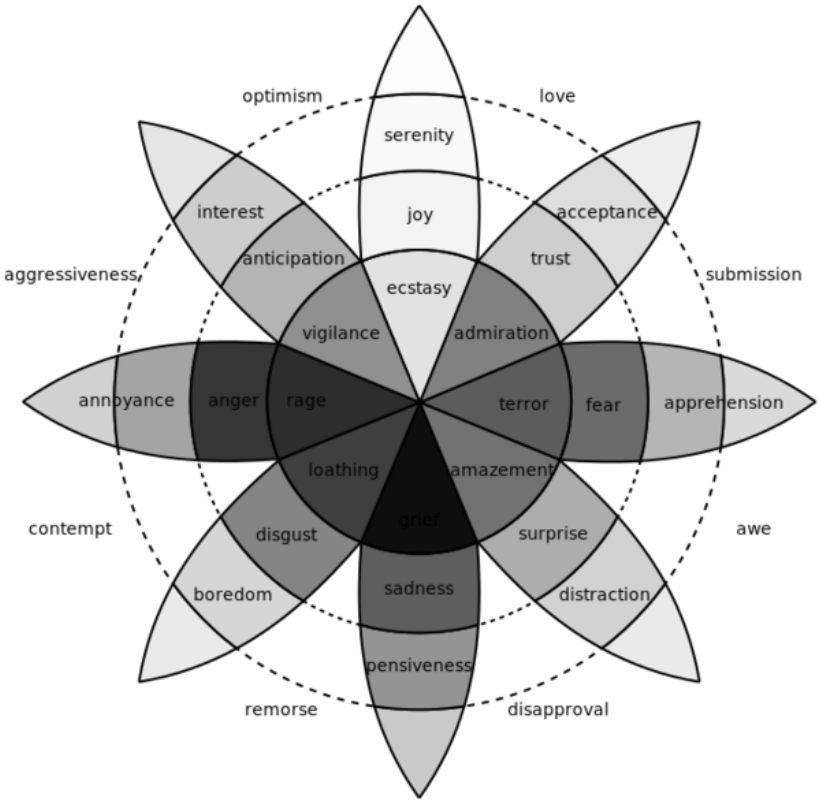

Figure 2.1: Plutchik's wheel

It seems that it is enough to extract the very basic emotions from the text, as more complex emotions can be understood as combinations of those basic ones.

2.5 Animation Software

The output scene must be created in some software able to play and use the scene. Developing a tool for this would be too time consuming - such software is not a small undertaking and with the time constraints this would result in a very rudimentary framework that would be severely limiting. Therefore a choice must be made between existing frameworks.

The animation software must satisfy the following requirements:

[1] `http://www.saifmohammad.com/WebPages/NRC-Emotion-Lexicon.htm`

- Flexibility and editability - As the scenes created by the generator will not be perfect, the software must provide powerful animation editing features.
- Exporting - It is desirable for the animation to be able to be exported into a variety of different formats that can be used by other frameworks useful within game development.
- Availability - As the proposed tool is developed with small developers in mind, it is most desirable for the tool to be available without any unnecessary fees or licensing. The proposed tool should also provide help for people unfamiliar with animating, who would be unwilling to pay additional fees for something they are not familiar with.
- Familiarity - Although most animation software is based on similar concepts, due to the time constraints my previous experience with the software plays a role in the final choice.

2.5.1 Maya

Maya is a 3D computer animation software developed by Autodesk. It supports modelling, rendering, simulation, texturing, and animation. This software is an industry standard and has been used for such projects as the Halo franchise. It supports all the animation editing and exporting features needed for this project, however it comes with a pretty harsh price tag of $180 per month [2]; a price which would make the potential reach of the proposed tool much smaller. Moreover, although I am familiar with animation concepts, I am not familiar with Maya.

2.5.2 Blender

Blender is an open source animation software. Similarly to Maya, it supports modelling, rendering, simulation, texturing and animation. Due to its open source nature the software may be less usable or stable at times however it is still very powerful and recognized within the industry. Blender's rendering engines are not as sophisticated as those of Maya. It supports exporting the animations to Collada, Alembic, 3D Studio, FBX, Motion Capture (.bvh), Wavefront, X3D and Stl file formats. This means that the final animation could be exported and used by pretty much any other tool. Blender is completely free to use and available to anyone. I am familiar with the tool

2.5.3 Unreal Engine

Unreal Engine is by far the most popular and most powerful publicly accessible game engine. Since the proposed tool would find most use in games it would make sense to create the scenes directly in a game engine. This approach however has some disadvantages - animation editing features in game engines are much more limited than those of a software dedicated to creating animation. Moreover, any game engine will not support the same exporting features (however, since the animation would already be in the engine, the need for exporting is arguable). Unreal engine is free to use, however Epic Games will seize a portion of income generated by a product developed with Unreal Engine. [3]

2.5.4 Unity 3D

Unity 3D is the most popular freely available engine after Unreal. It suffers from the same drawbacks regarding animation editing features and is in general less stable and sophisticated. The

[2] https://www.autodesk.com/products/maya/subscribe/
[3] https://www.unrealengine.com/en-US/faq/

only reason why Unity would be more suitable than Unreal for this project is my familiarity with the Unity 3D framework.

2.5.5 Final Choice

Upon taking a closer look on the available software, I conclude that Blender is the most suitable tool for the task as it satisfies the requirements best. It provides all the necessary exporting and editing features, is free and easily available and I already have experience with using it.

2.6 Motion Database

The dialogue scenes in this project are going to be assembled from existing short motion capture clips. While a potential user of the tool proposed by this paper (a game studio) would most likely possess resources to create their own motion capture clips and be in possession of legacy motion capture data recorded for past projects, I must resort to use other resources. There exists a multitude of motion capture data freely available online[4]. Among them the most famous is the Carneigie Mellon University's CMU Graphics Motion Capture Database[5]. It contains thousands of recordings of people walking, dancing, running, performing everyday activities, playing sports and performing gestures.

Only a small portion of that database would be useful to this project. There is, however, another database that focuses on matters more aligned with this project - the Emotional Body Motion Database [6]. The database was created by the Max Planck Institute for Biological Cybernetics and features solely recordings of people performing gestures associated with some emotion. The recordings feature all the basic emotions listed earlier as well as other emotions such as pride, relief, shame and more. There are three main types of motion capture recordings in this database:

- Narration - Actors were recorded while reading a story - their gestures were captured while narrating a part of the story that emphasizes some emotion.
- Non-verbal - Actors were recorded performing a gesture associated with some emotion in a non-verbal setting.
- Sentence - Actor were recorded while performing gestures when talking.

The database contains over 1400 clips, each labelled with what emotion it is supposed to represent, and what emotion would an observer associate the recording with [16] [17]. An example entry in the EBMD can be seen in figure 2.2.

ID	Download as	Intended emotion	Intended polarity	Perceived category	Perceived polarity	Accurate category	Accurate polarity	Duration	Peaks	Speed	Span	Acting task
		anger	---	anger	---	---	---					Narration
1354	bvh mvnx	anger	negative	anger	negative	1	1	3.125	3.667	0.09641	0.1206	Narration

Acting subtask	Actor	Gender	Age	Handedness	Native tongue	Responses	Consistency	Text
---	---	---		---	English	search		search
tale_six_swans	SiGl	f	21	right	English	anger, anger, anger, sadness, neutral, sadness, anger, neutral, anger, neutral, anger	0.545	NA

Figure 2.2: An example emdb entry

[4] http://jeroenvanboxtel.com/MocapDatabases.html
[5] http://mocap.cs.cmu.edu/
[6] http://ebmdb.tuebingen.mpg.de/index.php

2.7 Uncanny Valley

When dealing with animation, modelling, sculpting, etc. it is important to remember about the phenomenon of the uncanny valley. Not taking this phenomenon into account might cause unexpected negative results.

2.7.1 What is the uncanny valley

The uncanny valley is a phenomenon regarding aesthetics. It states that when a humanoid object (a character model, a robot, etc.) behaves in a more realistic, human-like fashion, it might not be necessarily be perceived as more human-like. The *valley* suggests that there exists a certain *dip* in human observer's affinity for a human replica [10].

The concept was first identified by robotics professor Masahiro Mori in 1970. In his essay on the uncanny valley he explains the relationship between a humanoid object's human likeness and an observer's affinity towards the object. He has noted, that as the object becomes more human like, the observer's affinity increases. However, when the human likeness of the object reaches a certain point, the human affinity for it decreases drastically. Then again, when the human likeness becomes very high, the affinity drastically rises again. The observers might perceive humanoid objects that are not quite human-like as funny, cute, in need of repair, etc. The observer perceive very human-like objects are almost indistinguishable from real humans - and while they may not be perfect, there is nothing alarming about them. However, the observers have described *fairly* human-like humanoid replicas as *creepy* or *eerie*. [10] [11]

The concept is easily depicted on a graph. Figure 2.3 draws the relationship between human likeness and observer's affinity. According to this representation, when a replica's human-likeness reaches about 80% the observer's affinity quickly becomes negative. Figures 2.4, 2.5 and 2.6 show a few examples of this concept by showing three different robots and their relation to the uncanny valley principle.

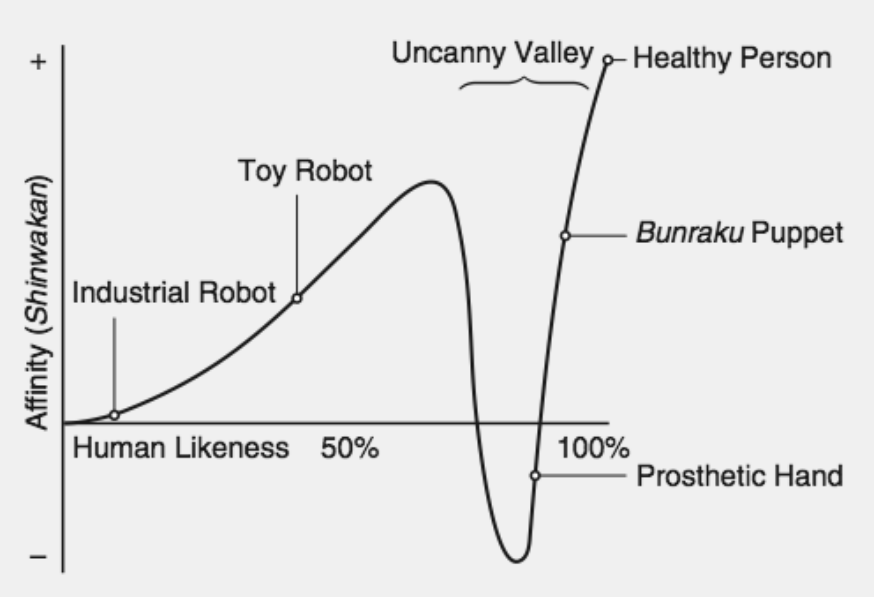

Figure 2.3: The uncanny valley

Figure 2.4: This robot does not fall into the uncanny valley category. It is a humanoid, but it is not realistic - it is usually perceived as funny or cute.

Figure 2.5: A robot that falls into the uncanny valley category - It is realistic enough to be human-like, but it is not human-like enough to be perceived positively and described as realistic. It provokes a feeling of eeriness or being alarmed in the observers.

Figure 2.6: This robot does not fall into the uncanny valley as it is realistic enough to feel familiar and normal.

2.7.2 Why uncanny valley matters in games and animation

Since animation and modelling deals with humanoid objects, the uncanny valley principle applies to it as well. Many game studios will prefer to use comic-like graphics and other solutions in order to purposefully make the characters less realistic. Oftentimes a less realistic game will be received better than a game that is quite realistic but not exactly so. Characters that fall into the uncanny valley category are often no longer perceived as humans, they are not relatable and laughable. [1]

This problem is one of the reasons that caused the spectacular failure of Mass Effect Andromeda. BioWare's process of creating large amounts of dialogue animation has produced animations which are quite realistic, but do not feel human. The players quickly lost interest in the characters as serious scenes seemed out of place, were unpredictable, and did not highlight emotions that the characters were trying to convey through speech. The case study of this game shows not only why a good system of generating animation is important, but also that a generation system that is generally realistic might be too little and too much at the same time. [7]

This project is also subject to uncanny valley. This principle might cause a set of pretty decent computer generated animations to be perceived very negatively by the audience. I will evaluate on that in section 3.3.

Chapter 3

Analysis

The methodology used to develop the project, technologies involved in the development and a breakdown of risks that concern the project are described in this chapter..

3.1 Methodology

The activities required in order to develop the project are listed below:

- Develop a project plan.
- Review literature and existing software, learn about other approaches, analyze related research.
- Review technology. Find out about software may be useful to this project and learn how to use it.
- Analyze risks. Specify what might be the potential pitfalls that may deem this product unsuccessful.
- Prototype a module that uses processes the script and outputs a structured file that can be used to generate the animated scene.
- Obtain a sufficient amount of animated clips. Create a custom downloader/importer if necessary.
- Tag and classify animation clips. Categorize clips by emotion, length, etc.
- Prototype a Blender extension that uses the created structured files to output an animated scene.
- Iterate on the existing software adding new features.
- Obtain more animated clips, classify and store them.
- Evaluate the prototype with real audience.

The minimum working prototype will be developed first. When all three modules are ready and in place, more features will be added iteratively. The software will work better if many animation clips are available, therefore as many clips as possible will be obtained and classified.

The generated animation must be evaluated with help of a real audience. How animation is perceived is subjective and cannot be decided by one person only. The evaluators will be asked to watch animated dialogue clips from various games and then watch this dialogue recreated using the proposed software. The evaluators will be asked to fill an evaluation survey. This way it will be possible to determined the usefulness and successfulness of the proposed software.

3.2 Technologies and Resources

1. **Emotion analysis** - The first module of the project focuses on NLP. This module should be extract emotions and actions from the script. The most important tool used will be IBM Watson Tone Analyser. If that tool turns out to be for any reason ineffective or imperfect, a customary tool (naive Bayes classifier or a keyword classifier) can be built for that task.

2. **Motion capture** - The EBMD (emotional body motion database) will be used to source the animations of body movement and gestures. The database provides the recordings as BVH files which is convenient as most animation software (such as Blender or Maya) can import BVH files. The emotional meta-data about the animated clips will be stored in an SQLite database. Storing the data using this method will allow the data to be easily and quickly searchable while less complicated than using a full fledged database software (such as MySQL od PostgreSQL) (very few tables are needed, there is no need to use advanced DB software).

3. **Animation software** - As aforementioned in section 2.5, the animation software that satisfies all the specified requirements is Blender. Blender supports all the necessary modelling and animation features. It also supports creating extensions allowing the animation to be created and assembled by code.

4. **Programming** - Python 3 will be used to implement the software. Python is the only language supported for add-on development for Blender. For other modules that do not rely on Blender, Python was chosen due to its development speed and in order to keep consistency among modules.

3.3 Risk Analysis

This subsection evaluates on ways in which this project can fail. Since reception of animation is subjective it may be hard to pinpoint what exactly went wrong with the software. It is possible that the audience will decide that the animations are not good enough but they will not be able to describe why. Because of that it is important to understand those outline those issues before conducting any tests.

3.3.1 Failure due to emotion analysis

The first way in which the software may fail is due to the emotion analysis component. If the final animation does not reflect by body language the emotions of the characters it may mean that the dialogue were not analysed properly, meaning that the results provided by emotion analysis are simply not accurate enough.

In this project I will be using IBM Watson which is a pretty good benchmark regarding emotion analysis. It is safe to say that is Watson is unable to perform the task well enough there is no software that would be. If a failure is identified to be caused by the emotion analysis, this might mean that:

- The NLP methods are not yet advanced enough to perform this task. This project will remain infeasible until we see an improvement in emotion analysis techniques.
- NLP alone is not enough to solve this problem. Other approaches should be used alongside NLP emotion analysis, such as analysis of the tone of the recorded voice or emotional analysis of facial expressions of the voice actors.

Since there is no significant research done regarding combining emotion analysis and animation, there is no way to know for certain whether this approach will work. Because of that the risk of such a failure is moderately high.

3.3.2 Failure due to the motion capture data quality

The software proposed by this paper can only ever be as good as the motion capture data provided. While the software will be designed to work with custom motion capture databases, the EBMD is used for this project. This means that if the clips provided by EBMD are too ambiguous - that the recorded body language do not convey the emotion clearly enough, or that the recorded movements are unnatural, too subtle or too over emphasised, the output animation will not be perceived as successful. In this case this might not be the problem of the software or the approach at all - the failure is generated only and specifically by the motion capture recordings.

The EBMD provides quite some meta-data regarding each clip. Among the meta-data provided there is information about what emotion the movement tries to convey and what emotion was perceived by the audience when watching the clip. Because of that the risk of failure due to motion capture data quality is fairly low as the animation should in general unambiguously convey the intended emotion. There is however still a risk that the motions will be perceived as conveying correct emotion, but being too unnatural and unrealistic.

3.3.3 Failure due to the uncanny valley principle

The uncanny valley principle poses a high threat to the project. The final animation satisfying the uncanny valley principle is possibly the worst scenario in which the project may fail. This failure is easiest determined by comparing the final animation with other games. Two types of game animation will be shown to the audience - animation is advanced and generally considered good, and animation that is basic, unrealistic, seemingly random and monotonous. If the audience judges the animations generated by this software to be worse than both of these animation types then it probably means that the animations comply with the uncanny valley principle. It means that while the animation generally is realistic and matches the intended emotion of the speech, the animation still feels unnatural and it might have been better to go with a much simpler approach, such as keeping the characters mostly static.

If the uncanny valley is achieved by the output animation, this might mean a few things. If the uncanny valley effect was created by the nature of the EBMD recordings, this means that replacing

the database with a different might just solve the problem. However, it might mean that the NLP emotion analysis approach to generating animation is inherently flawed. It might just prove that text data alone is never enough to produce convincing animation and while this method might be useful in combination with other methods, emotion analysis approach will always produce unconvincing animation.

Chapter 4

Requirements Specification

This section describes functional and non-functional requirements of the proposed software. The software must be developed accordingly to the requirements and must satisfy all of them in order to be truly successful.

4.1 Functional Requirements

1. **Analysis of a semi-structured script** - The software takes a semi structured script as input. The structure of a script must resemble a structure of a film script. The script provides 2 kinds of information - characters involved and lines of dialogue spoken by them. The software must be able to extract emotions from the dialogue lines.

2. **Store motion capture data with regards to emotions** - The software must store and tag the motion capture clips with relevant meta-data about what kind of emotions they represent. The database must be easily and quickly searchable.

3. **Find relevant animation clips** - The software must take information about character's emotions and actions and choose animation clip that best represents character's behaviour. The chosen clips must reflect character's emotions, but also need to be of correct length to match the speed of the speech, as well as not repeat too often. The system must be compatible with Emotional Body Motion Database published by the Max Planck Institute for Biological Cybernetics.

4. **Assemble the final scene** - The software must be able to output the final animated dialogue scene. The outputted scene must be fully editable, enabling various adjustments before rendering. The final scene must also be exportable to other formats so it can be used with game engines or other editing software.

4.2 Non-Functional Requirements

- The user should be able to use the software with a custom motion-capture database (Usability).

- The user should be able to assign different character models to different characters (Usability).

- The user should be able to fully customize and edit the final scene (Usability).

- The software is designed to automatically create big amounts of animated scenes. The time of generating a scene is not a high priority, but it must be reasonable (Performance).
- The software must support common animation file formats (fbx, bvh) (Portability).
- The software must be modular enough so that different parts of it can be replaced with ease .It must be possible to replace emotion analysis software and 3D animation editing software with other software (Portability).
- The output animation must be perceived well by the audience and judged to be acceptable for use in a game given minor adjustments (Quality).

Chapter 5

Design And Architecture

This chapter describes the design of the system. This includes both the underlying decisions of the system as well as the user interface design. The chapter also presents on a more detailed view of the system's architecture.

5.1 System Design

The system has to work in two major steps. The first step is to take semi-structured natural language script and analyse it. This means that the system must analyze each dialogue line and infer some emotional values from them. Using those emotional values and the length of the dialogue line, the system must find best matching animation clips from the motion capture database. The results of this step is a file that specifies which characters say which lines of dialogue and it also specifies which animated clips accompany those dialogue lines.

After this step is completed, the file must be interpreted into a final scene. The importer must be able to read the file, import required models and animations and generate the final scene. As every animation software and game engine is a little different, each of them would need a custom importer. Those would be very similar in principle, but differ slightly because of the implementation of given software. For my project I have created the importer for Blender. This is because Blender is open source and available to anyone, as well as I am the most familiar with this software.

5.2 Emotions

For my project I have decided to only use the following emotions: joy, fear, disgust, anger and sadness. Traditionally, surprise is also part of the six basic emotions model by Paul Ekman and Wallace V. Friesen. However, IBM Watson Tone Analyzer, that I am using for sentiment analysis, is unable to detect surprise in the text (more on that in section 5.7) - therefore I had to abstain from using this emotion in this project. The emotions are expressed as decimal numbers between 0 and 1 - this identifies whether a given emotion is present in a specific motion clip or text and how intense that emotion is. It also allows to create a mixture of emotions in order to represent more complex emotions. The EBMD describes the animations in terms of their emotion category, but provides no value (the EMDB will not differentiate between an angry gesture, and a *very* angry gesture). Therefore each animation clip has to be manually reviewed in order to assess it's emotional properties.

5.3 Architecture

The architecture of the system is essentially a pipeline. The modules process resources and pass them onto the next module. They can be completely unaware of each other. This allows for a lot of flexibility and helps achieve some of the requirements. The emotion analysis API can be replaced with a different one, the user can use a custom motion capture database, and a different importer can be created if the user wishes to use software other than blender.

There are three main modules:

- Text Analysis
- Animation Clip Matcher
- Animation Generator

The Text Analysis module parses the text and analyses emotion using some API. It takes a script as an input and passes the parsed and analysed text to the Animation Clip Matcher.

The Animation Clip Matcher uses the motion capture database to find the best animation clips to accompany the speech. The Animation Clip Matcher must take into account the emotion analysis of the text and find animations with similar emotional score. Another important constraint is the time of the animation. The animated clips have different lengths and a chosen clip must not be significantly longer than the spoken/read text. In case the animated clip is shorter, the Matcher must choose more than one matching clip. The Matcher outputs a JSON file which specifies all dialogue lines; it states which characters perform which emotional gesture actions and where is the animation file located.

The Animation Generator reads the JSON file and imports all needed animations. The user is now able to assign character model for each character and run the generator. The generator will assemble the animations in correct order, focus the camera on a currently speaking character and add subtitles. The output is an animated scene that can be either manually edited, exported to a file or rendered.

The full diagram of the system can be seen in figure 5.1.

5.4 Character Armature and Model

As the system is designed to be used in games, one of the main requirements was for the system to handle various character models. Because in a game each character is represented by a different model, this is a necessity. The model itself is relatively irrelevant - it must however support the same or similar enough armature to the armature the system was designed around. The armature is shown in detail in figure 5.2. For an armature to be supported, the bones might have corresponding names. Any extra bones or bones with unmatching names will be left not animated.

5.5 EBMD Importer and Showcase Generator

The following are small tools created to help automate / bootstrap the process of acquisition and categorization of the animations.

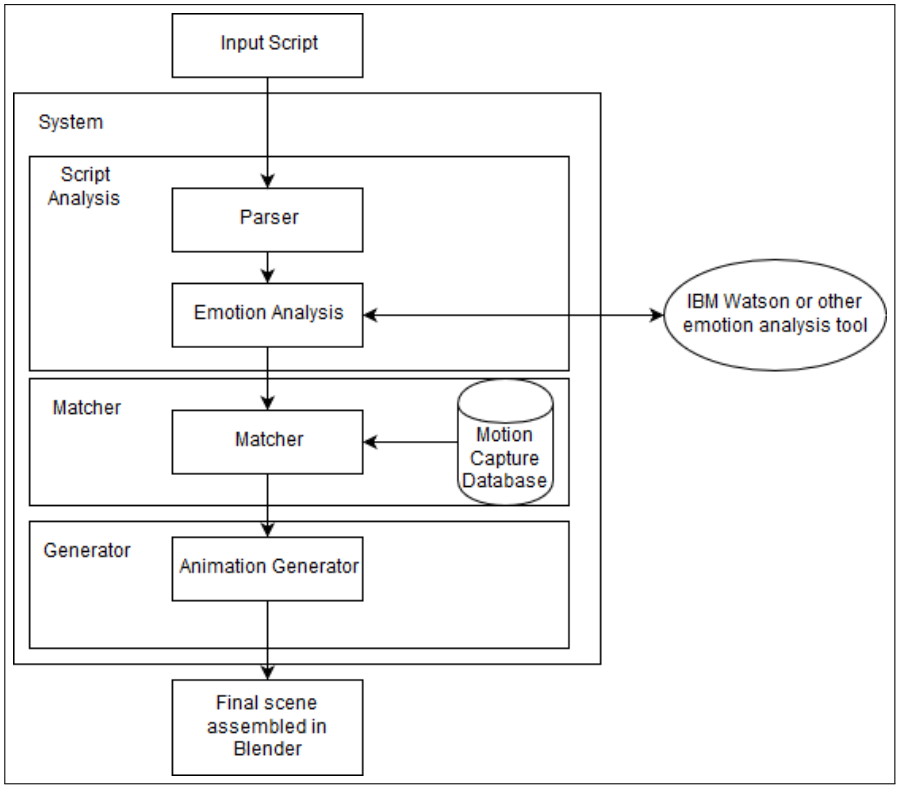

Figure 5.1: The pipeline architecture of the system

5.5.1 EDMB Importer

The EBMD importer's purpose is to help prepare the animations obtained from EBMD to work with the system. The EBMD animations have a few issues making them incompatible with the system - The armatures are a little different with unmatching bone names, many animations were recorder while actors were seating [1] and are saved in BVH files [2].

To import an animation, the importer must follow through the following steps:

1. Download animation from link to file (receive http link as input) and import the animation from the downloaded BVH file
2. Remove animation data from the legs
3. Rename the bones that the animation data applies to so that they match those of the target architecture
4. Import the target armature
5. Apply the action from imported from EMDB to the target architecture
6. Export the animation data to FBX file

The EMDB importer help automate the process of acquiring animations by allowing to download and process a huge bulk of animations, saving them in folders that correspond to their emotion category and showing their length in frames in the filename.

[1] Leg animation is irrelevant for this system.
[2] BVH files are suitable for motion capture, but can be more problematic for generic animation than FBX.

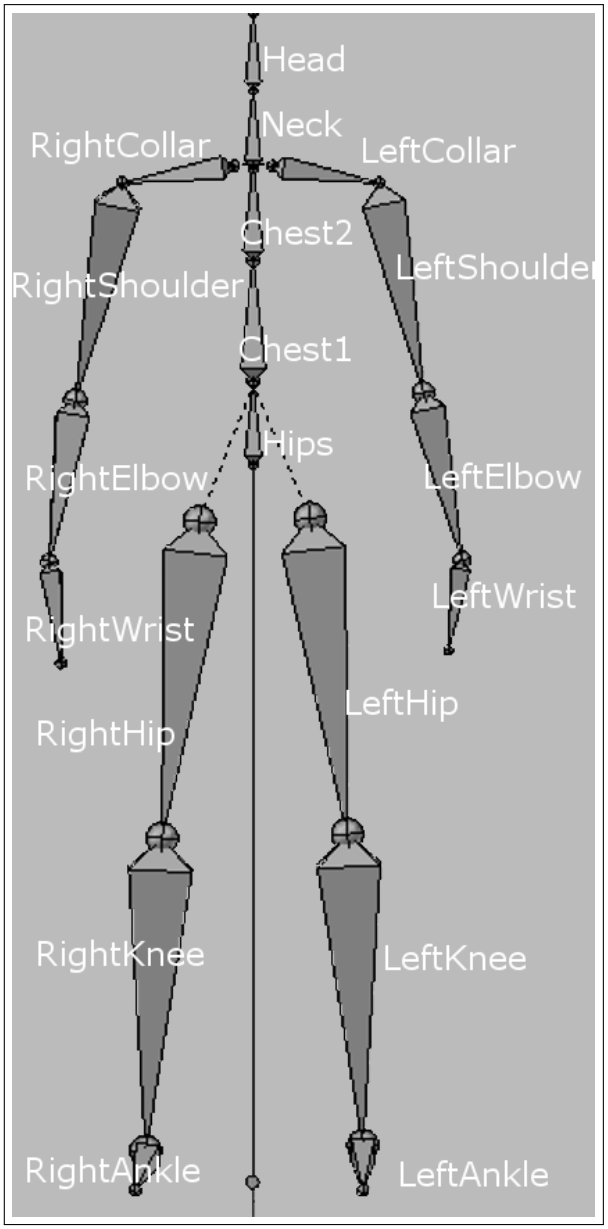

Figure 5.2: The armature supported by the system

5.5.2 Showcase Generator

The purpose of the showcase generator is to help categorize and store the animations in the database (section 5.8). The showcase generator outputs instructions for the animation generator (section 5.10) which allow the animation generator to create an animation which shows all the animations in a given directory one after another. Since the animations have to be analyzed manually (in order to assess their emotional value), a clip showing all animations allows for a quick and easy analysis.

5.6 Input

The input of the system is as follows:

The input is a file that represents a dialogue. Each character that participates in the scene must be clearly stated. Each dialogue line must have a dialogue line must have a character clearly

associated with it. Character names are specified after five tabs. Dialogue Lines are specified after three tabs. Each file must end with "ENDSCRIPT" with no indentation.

I used this format as this format is often used to represent movie scripts. One can find hundreds of scripts saved in this format on The Internet Movie Script Database (`www.imsdb.com`). The example input can be seen in figure 5.3.

```
                                SANDRA
                    Sonny?  Sonny, who is it?
                    What is it?

                                SONNY
                    They shot the old man.

                                SANDRA
                    Oh God...

                                SONNY
                    Honey...don't worry.  Nothing else
                    is going to happen.

ENDSCRIPT
```

Figure 5.3: An example of system's input

5.7 Emotion Analysis

As aforementioned, there are many ways to perform emotion analysis of the text. I have chosen to use IBM Watson Tone analyser for this task as it offers a state-of-the-art service accessible for prototyping (a lot of tools offering high quality emotional analysis are built for commercial purposes and not available without paying high fees).

The module takes each dialogue line and sends to IBM Watson Tone Analyzer for analysis. Watson's REST API is used to accomplish that. Watson returns all the emotions with values between 0 and 1 found in the text. Each dialogue line now has emotional values assigned to it.

5.8 Database

The motion capture database consists of two main parts: the files that contain the animated clips and a database that holds the meta-data about the animations and character models.

The animation data is stored in FBX files - one animated action per file. The folder structure can be customized - for the purposes of this project the animations are stored in a way that describes the emotions they convey [3]. The folder structure is irrelevant as long as it stays consistent with the entries in the SQL database.

The database is implemented using SQLite. It is the perfect tool for this task as the database needs to be simple, relatively small and easily searchable. The animation metadata is held in a table that look like this FIGURE. The emotional values of each animated clip need to be manually

[3] For example: animations > anger > subtle > disbelief1.fbx

adjusted. When all values are set to zero, it means that the clip carries no emotional impact (neutral). An example of a few database records can be seen in figure 5.4.

	id	name	file	Frames	anger	joy	fear	disgust	sadness
		Filter	Filter	F...				F...	Filter
1	1	angry_shrug	anger/anger1...	40	0.5	0.0	0.0	0.0	0.0
2	2	shrug_hand_t...	anger/anger2...	48	0.3	0.0	0.0	0.0	0.0
3	3	shrug_hand_t...	anger/anger3...	52	0.4	0.0	0.0	0.0	0.0
4	4	cower	fear/fear1.fbx	40	0.0	0.0	0.9	0.0	0.0
5	5	cover_head	fear/fear2.fbx	60	0.0	0.0	0.7	0.0	0.0
6	6	disbelief	anger/anger4...	62	0.65	0.0	0.0	0.0	0.0

Figure 5.4: Database of animation clips

The other purpose of the database is to store the information about the models. The important information about a model is its name, file location, camera offset and rotation. Because models may be of different shapes and sizes, camera positioning during dialogue will not be the same. The database allows to specify a desirable camera position relative to the character that will allow to fully capture the character at a good angle. Example of a few model database records can be seen in figure 5.5.

	ID	name	file	cam_offset_x	cam_offset_y	cam_offset_z	cam_rot_x	cam_rot_y	cam_rot_z
		F...	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	4	Blue	blue.fbx	-0.5	-1.8	1.65	0.0	0.0	1.3
2	3	Green	green.fbx	-0.5	-1.0	2.0	-0.3	0.0	1.5
3	2	Test	untitled.fbx	-0.5	-1.8	1.65	0.2	0.0	3.14
4	1	Basic	untitled.fbx	-0.5	-1.8	1.65	0.0	0.0	1.3

Figure 5.5: Database of character models

5.9 Matcher

The matcher module depends on the emotion analysis and database modules. It takes the emotion data and searches the database for a suitable animated clip. The matcher's input is a dictionary where the keys are emotions (anger, fear, etc.) and values are e 0 to 1 decimal.

5.9.1 Score matching

Firstly, the matcher will decide whether the text carries enough emotions. If no emotion surpasses the constant relevancy threshold (in this project the threshold is set to 0.2) the animation is deemed as neutral. If there is only one emotion, the module will look for an animation matching that emotional value. If more than one emotion exceed the threshold, only the two most important (highest value) will be considered. That is because certain emotions often go in pairs (such as anger and sadness), but combinations of more than two emotions are rare, confusing and hard to represent with an animation. The searches the database for animations that may be fitting and compares the text emotional values and animations emotional values. The pseudocode behind these calculations can be found in figure 5.6.

5.9.2 Length Matching

The matcher must account that even the most emotionally fitting animations may not be the most suitable. It is very important to take into account the speed of speech - gestures performed in an animation must match the length of character's dialogue line. Otherwise there will be situations where a short animation is chosen for a few sentences worth of dialogue, or a very long animation is chosen for just a few words of dialogue.

To account for this, constant frames per second (to define animation length in real world terms) and words per second (to calculate the length of a dialogue line in seconds) must be defined. The matcher uses those to calculate the ratio between an animated clip length and speech length. This ratio is used to adjust the score calculated using emotional values (section 5.9.1). How the two matching operations work together can be seen in the pseudocode in figure 5.6.

Additionally, the matcher will ensure that the animation is not too monotonous. The matcher will not choose twice the same animation in one dialogue - it will prioritize a similarly matching animation that has not yet been used. The animations will be reused if no other matching animations are available.

5.10 Animation Generator (Importer)

As aforementioned, in this project I have used Blender to support the animation generator model. The generator is a Blender add-on. It can either be used as a script, or be installed as an add-on to be fully and permanently available within Blender.

The generator first reads the JSON file prepared by previous modules and loads required animation clips from files. The user needs assign a 3D character model to each character participating in the dialogue. The required 3D character models are animated and the camera is positioned accordingly to the model's meta-data stored in the database. The animated characters are positioned to directly face each other (the program currently supports only up to two characters in one scene). Each animated action is assigned to a corresponding character model and then pushed on a separate NLA strip (figure 5.7). The program keeps track of how many frames are being used so that newly added actions begin just when the previous actions have ended.

To setup some basic background and feel, a floor (plane) is added beneath the characters and two directional lights cast light down from above the characters.

The camera is added to the scene and positioned using the meta-data retrieved from the database. At the start of each character's actions the camera is positioned and rotated to focus at the character performing an action.

Subtitles are added as Video Editor Sequence Strips spanning between frames that correspond to the animation (figure 5.8).

```
constants: RELEVANCY_THRESHOLD
           WORDS_PER_SECOND
           FRAMES_PER_SECONDS

function get_length_match(text, animation)
  animation_length = animation['frames'] / FRAMES_PER_SECONDS
  text_length = count_words(text) / WORDS_PER_SECOND
  score = min(animation_length, text_length) / max(animation_length, text_length)

  return score
end_function

function get_emotion_match(emotions, animation)
  differences = []
  for emotion in emotions
    value1 = emotion.value1
    value2 = animation[emotion].value
    differences.append(abs(value1 - value2))
  end_for

  average = sum(differences) / length(differences)
  score = 1.0 - average

  return score
end_function

function get_matching_gestures(text, emotions)
  # Sort emotions so that the most relevant
  # is at the start of the array
  emotions = sort(emotions)
  emotions = reverse(emotions)

  matching_animations = set() # Don't allow for duplicates
  neutral = False
  if emotions[0].value < 0.2 #neutral
    neutral = True
    matching_animations = get_emotions_from_database('neutral')
  else
    matching_animations = get_emotions_from_database(emotions[0].emotion)
    if emotions[1].value >= 0.2
      matching_animations.append(get_emotions_from_database(emotions[1].emotion))
    end_if
  end_if

  emotion_scores = {} # a dictionary {animation: score}
  # Calculate scores
  for each animation in matching_animations
    score = 1
    if not neutral # All neutral animations get the same score
      score = get_emotion_match(emotions, animation)
    end_if
    emotion_scores[animation] = score
  end_for

  # Adjust for length
  for each animation in matching_animations
    # Multiply to adjust accordingly
    emotion_scores[animation] *= get_length_match(text, animation)
  end_for

  # Sort so that most matching animation is first
  emotion_scores = reverse(sort(emotion_scores))
  return emotion_scores # Return list of matching animations
end_function

```

Figure 5.6: Pseudocode explaining how the matcher calculates the scores

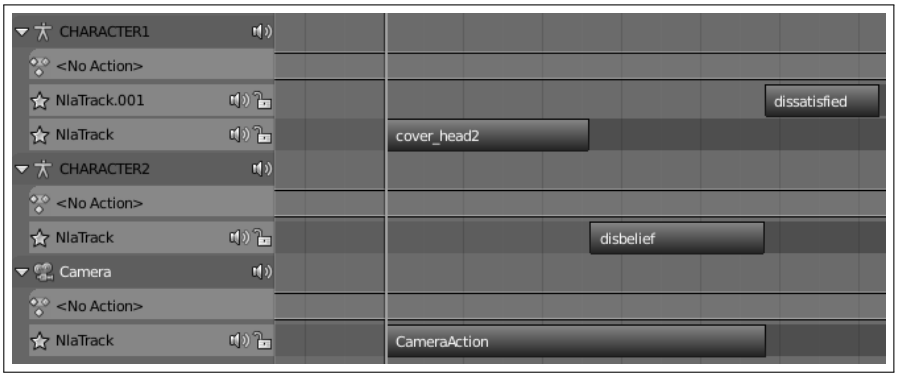

Figure 5.7: NLA strips of the final animation

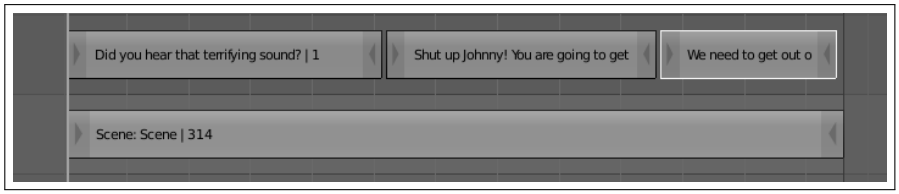

Figure 5.8: Video strips featuring subtitles

The final animation can be edited and adjusted in Blender (figure 5.9), rendered (figure 5.10) or exported to file.

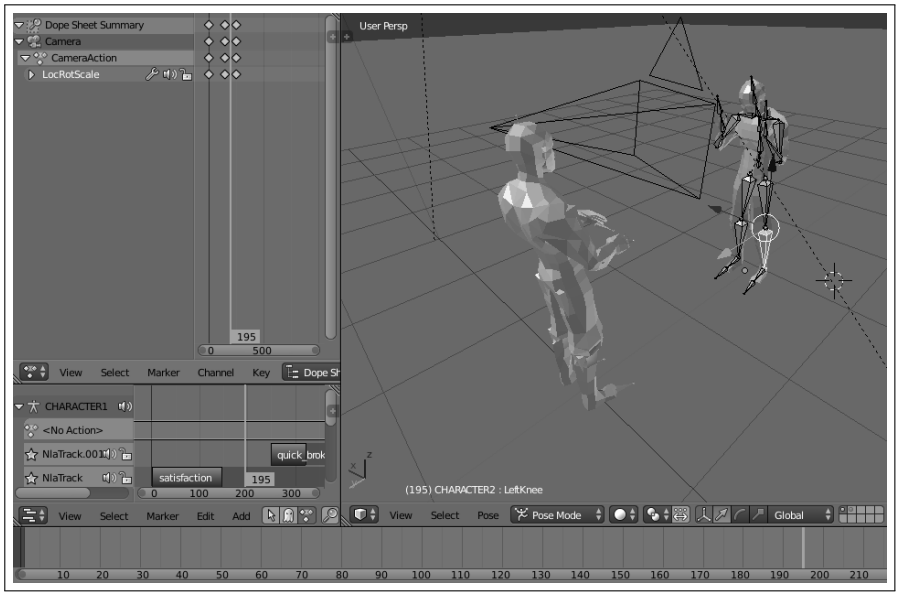

Figure 5.9: Editing the final animation in Blender

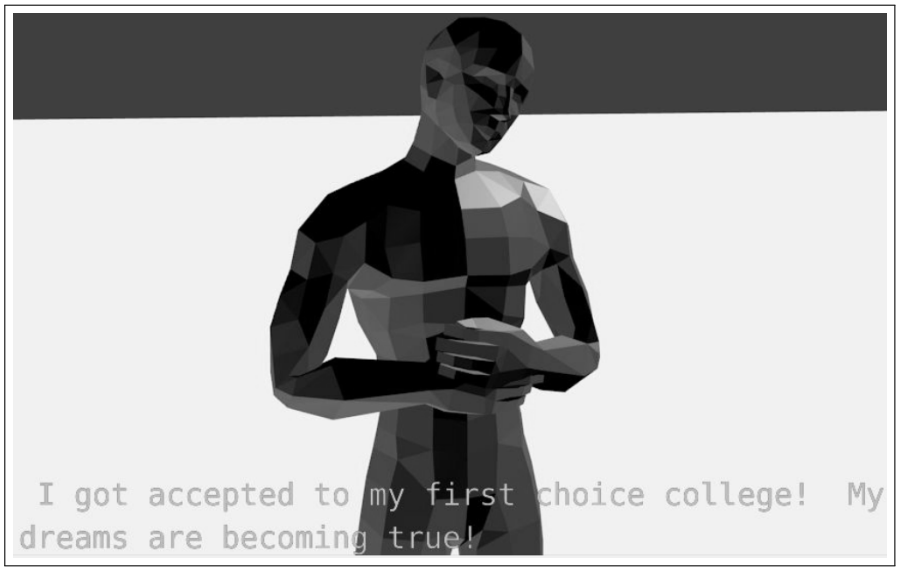

Figure 5.10: Rendered final animation

5.11 User Interface

The Animation Generator module is the only module that features a Graphical UI. The module is embedded into Blender and uses extends Blender's interface.

The UI is minimalistic and simple to use. Upon installation and activation a new tab is added to the menu on the left hand side. Figure 5.11. Rectangle 1 represents the left hand side menu. Number 2 shows the tabs where the bottommost one belongs to the extension. Number 3 shows the actual UI of the extension.

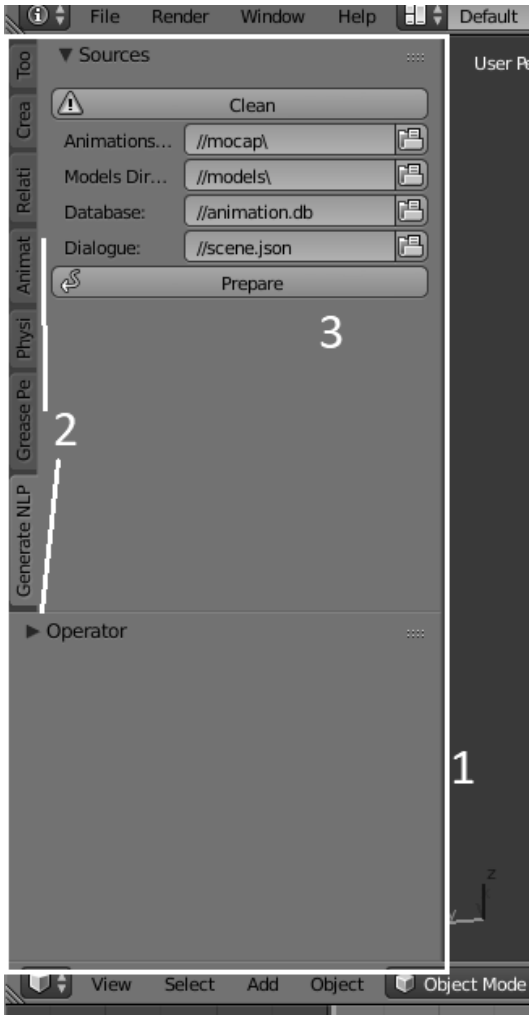

Figure 5.11: The addon UI

From here the user is able to either clean the scene (might be useful especially if something unexpected happens or the work of the extension is interrupted) or prepare the scene. To prepare the scene the user must first initialize the four variables. The user needs to point the program to where the animations and models are stored as well as the database file and the scene file (JSON file generated by previous modules).

Upon pressing 'Prepare' the program will prepare data for generating a scene. Required animations will be imported and the scene file will be parsed. There is one more step left to finalize the animation. When preparation is finished, a new menu will pop up below the currently existing one. The menu can be seen in figure 5.12. In the menu the user is required to assign a

character model to each character existing in the scene. Upon pressing 'Finalize' the full scene will be generated complete with lighting, camera movement and subtitles.

Figure 5.12: The finalization step (new menu highlighted by white rectangle)

They key aspect of the interface is its simplicity as it allows users to generate relatively complicated animation sequences by essentially using two buttons, with no knowledge about animation and little Blender expertise required. The UI also provides flexibility as the user can decide upon which character models to use.

Chapter 6

Testing and Performance

This section describes testing and potential issues of the developed software.

6.1 Components

The testing of the components was an ongoing activity during development - the creation of the software alternated between phases of development and testing. The components were fed typical and erroneous inputs and their outputs were verified for correctness. The testing phase continued until all discovered issues were taken care of and the development could continue.

6.2 Analyzer

There are a few issues that the user should focus on before providing input to the first module. Firstly, the input must be in English as the emotion analysis is set to only use the English language. The user should make sure that the text they intend to analyse is in English and does not contain non-English characters.

Secondly, the user must make sure that the character names and their dialogue lines are correctly indented. The module was designed to be able to analyse scripts copied directly from the Internet Movie Script DataBase. Those scripts often have other information (non-relevant to this software) specified at different indentation levels. The software simply ignores that information - therefore if any of the important information is specified at wrong indentation level it will be ignored.

6.3 Matcher

The testing of the matcher aimed to make sure that the matcher outputs animations of correct emotional sentiment and length. There is no other way to do that than to test it empirically. The module was fed with emotional values and text lengths. Whether the output animation matched the emotions provided and its length felt natural (just enough time to read the subtitles, as one would while watching a film) was assessed manually. The module (along with the database) was fine-tuned until it gave satisfying results.

6.4 Animation Generator

There are some serious limitations to testing of this module. It is hard for a computer to decide whether the animation looks natural. Some animations imported from EMDB experience import problems which results in an animation with broken and unnatural movements. While manually analysing the emotional values of the animations, it was important to check each animation for

movements that may have been incorrectly imported. Such animations cannot be used with the software.

The animation generator is able to handle only up to two characters in the scene (although the module was designed with extensibility in mind). If a scene JSON file contains information on more than two characters (or no characters at all) the scene will not be assembled and an error message will be displayed instead. In case the scene file is corrupted and cannot be parsed the user will also be presented with an error message. The JSON file can become corrupted if a user does manual changes to it or if the input of the script analysis module is in a wrong format.

6.5 Scalability

The system was tested with inputs of at least fifty lines of dialogue. The system was able to generate the scenes successfully and in reasonable time (more on time of execution in section 6.7). A user would rarely need to generate a single scene so long and would rather produce a multitude of shorter ones. However

6.6 Samples

Sample inputs and outputs of the system can be accessed using this link: `http://mpanasiuk.me/bscthesis/samples.zip`

6.7 Performance

One of the important requirements of the software was the speed of execution. For creating an animation consisting of 6 dialogue lines (an average dialogue length used while testing the software) using the first module takes up to 18 seconds. The matcher module is so quick that it's execution does not influence the total execution time. The generator module takes another four seconds on average. This results in an execution speed of about 3.6 seconds per dialogue line. This fulfils the requirement as the time required to assemble an animation is incomparably less than doing that manually and is not a limitation for an animator working on the scene.

Chapter 7

Evaluation

This chapter describes how the system was evaluated and explains why such a method of evaluation was chosen. The evaluation aims to determine whether the created system was successful or not and due to what reasons.

7.1 Problems with evaluation

Evaluation of animation is quite problematic in nature. The main issue is that how an animation is perceived is very subjective - it is near impossible to computationally determine whether an animation looks natural, whether an animation goes in pair with the speech and if the whole 'feel' of the dialogue scene is good. It is then important for the animation to be judged by people. I myself however am not fit to decide that - I have spent far too much time creating the animated scenes and watching both failed and successful results. Because of that I am obviously seriously biased and cannot judge the animations from an objective standpoint.

7.2 Survey

In order to account for my bias, I have decided to use interview people about the generated animations. People who have not had a part in the development of the software can provide valuable opinions.

7.2.1 Methodology

People may find it hard to provide opinion about quality of an animation if they have nothing to base their answer upon (especially if they are not animators or developers). There needs to be some sort of a benchmark or a control sample to enable people to judge the animations more efficiently.

To account for that I have chosen the following approach: I would first find a dialogue scene in a relatively popular, successful game and recreate it using the developed software. The participants would be shown both the original and recreated scene and describe how they compare - they would be asked to decide whether using the software has improved or decreased the quality of the scene.

Preparation

I have chosen three games from which I extracted the dialogue scenes. I did not want to extract a few dialogue scenes from one game in order to provide variety, but I also did not want to use more than three games as it would make the questionnaire too long and make the participant loose interest or become confused.

The three games are all of the RPG genre and differ mostly by the year of release - I have used a relatively old game, relatively new game and a state-of-the-art game - for if the software is deemed to produce animation of lower quality than the state-of-the-art game, comparing it to an old game may help decide whether the software simply needs improvements or if it the approach itself is wrong.

Not simply any game could have been chosen. In my project I have to rely on EBMD to generate the animations - and the motion capture data provided by the database is not suitable for any game. For example, if a game features a lot of soldier to soldier military interactions it is expected that the characters will behave in a more serious, less expressive manner (I evaluate on that in section 9.2). The EBMD does not supply such animations that would be useful in this situation and using a wrong game would cause unnecessary confusion of the participants (for such games a custom motion database can be used with the software). The chosen games must feature some of more relaxed, natural conversations. This certainly tightens the pool of games usable for the evaluation, but I believe that the chosen games are a rather representative sample of the RPG genre.

The chosen games are:

- Fallout 3 (released in 2008)
- Fallout 4 (released in 2015)
- Horizon: Zero Dawn (released in 2017)

The dialogues from the games were chosen randomly, however there were some constraints that limited the choice:

- The scene cannot feature more than two characters (The software was designed with extendibility in mind, however at the moment it does not support more than two characters).
- The dialogue has to convey some emotions (dialogues which result in only neutral gestures are not truly representative of the software).
- The dialogue cannot be exposition dialogue (Why that is the case and what exactly is *exposition dialogue* I explain in sections 7.3 and 9.2).

Before conducting an interview it is important to know whether the interviewee has experience with games and animations and development. The answers of people knowledgeable in this area might differ from those provided by people new to this area and it might be helpful to be able to distinguish between them.

The questionnaire

The questionnaire consists of four sections. The first section provides a little detail about what the software is trying to achieve and what is expected of the participant. This section also asks whether the participant considers themselves to be knowledgeable about games or animations. This is the only question in that might be viewed as personal.

The other sections all ask the same set of questions about different pairs of videos[1]. At the start of each section the participant will be shown two videos - an original and a recreated dialogue scene. After watching both of them they will be asked a couple of questions that help determine whether they find the videos realistic, whether the animations correspond to the conveyed gestures and whether they think that using the software improves the scenes. The participants are able to evaluate in more detail on the answers if they wish so. The questions that are designed to be answered on a linear scale resemble the Likert scale[2].

The original dialogue scenes were modified in two ways. Firstly, apart from the in-game subtitles another set of subtitles was added for accessibility purposes. The added subtitles were the same as the original subtitles but bigger and easier to read. Secondly, the videos were muted. Audio is completely irrelevant to this project. I could not procure audio for the recreated scenes[3] therefore the recreated scenes are mute. People watching first a scene with audio and then a scene without audio would focus too much on that difference instead on the difference in animation. Therefore it is best to remove any sound from all the videos.

The surveys were conducted as an one-on-one interview if possible. If the participants were unavailable or preferred not to do this personally an online version of the questionnaire was provided. The participation in the questionnaire was fully voluntary. The participants were presented with a consent form before answering the questions and were provided with the details of the software and how the study is going to be conducted. No aspect of the software or the study was hidden from the participants and they were able to both request more details or quit any time they wished to do so.

As mentioned before, the preferred approach was a supervised survey as it is important that the participants correctly understand what the project is about and what it is trying to achieve. The survey was targeted mostly at my colleagues (Bachelor computing science students), as certain level of computing science and application development expertise is desirable. That is because people who are unacquainted with prototyping and development might focus too much on how unpolished the animation is (since it is only a prototype and the animations are rather crude - there is no background in the scene, no audio and no texturing/graphics) than on the gestures and dialogue itself.

The full questionnaire can be found in the appendices in section C.2.

Objectives

The survey has following objectives:

To test whether using the NLP approach for animation generation has created scenes that are more realistic and natural than the original scenes

To test whether the animations generated using the NLP method are preferable to those

[1]The animations and videos can be accessed at `http://mpanasiuk.me/bscthesis/samples.zip`. A sample dialogue transcript with still images of the scene can be found in appendix section C.3.

[2]The Likert scale is a psychometric scale created by a psychologist Rensis Likert. It is commonly used in questionnaires in order to determine whether the participants agree or disagree with given statements.

[3]Simply overlaying the audio from the original scene over the recreated scene is not a good solution either. The audio was not created with the recreated scene in mind and the recreated scene did not take audio into account. Because of that, the characters would perform body motion that puts stress on one part of a sentence while audio would put stress on a different part of a sentence. This makes the audio unusable with a scene as it would make the scene feel completely unrealistic.

created by traditional methods

Determine how much manual adjustment is required before the animations generated by the software are satisfying

7.3 Other findings

Apart from the findings of the questionnaire I also wish to describe my own realisations that I devised during the development of the software. I know that I cannot use them to decide the successfulness of the software because of my bias towards it, however I think there are a few aspects worth mentioning that will not be found by the questionnaire participants.

As I was testing the software many times and I tried to recreate many scenes from RPG games I have noticed that the software does not perform as well if certain conditions are not satisfied. For example: the general 'feel' of the game must match the gestures provided by the motion database, the software does not perform with scenes that contain exposition or sarcasm and more. I will explain those issues in more detail in section 9.2.

Chapter 8

Results

8.1 Survey

The survey was completed by 12 people in total (5 interviewed in person, 7 interviewed online). Half of the participants have described themselves as well-acquainted with the area of games or animation. One person was not sure and the rest was not acquainted with that topic. However it appears that this is never strongly correlated (using Pearson correlation coefficient) to any of the other answers. It seems that having experience in games or animation does not significantly influence how the participants perceived the animations.

The participants were asked to decide whether the animations presented to them were realistic. The question was asked about each of the six scenes. The t-test was performed taking into account the answers about all the original scenes and the recreated scenes; The t-test shows that these results are not statistically significant. The realism of the scenes was described on a one to five scale[1], where the average score for both original and recreated scenes is about 3.1. These results show that animations generated by this project are no more realistic than the originals and that both original and recreated scenes are hardly realistic at all.

A similar set of questions was asked about the effectiveness in conveying emotions. The participants were asked to decide whether the animations correctly convey the feelings and emotions of the characters in the scene. The participants' answers suggest that the animations reproduced by the software are slightly better at conveying feelings and emotions (the mean score of the recreated animations was by 0.3 (out of 5) better than the originals). These results show a very slight advantage of the recreated animations over the original ones. This however was not enough for the results to be statistically significant.

The figure 8.1 shows a breakdown of these results for each game. The graph shows that the software is capable of creating more realistic animation than animation featured in an old game, but struggles to keep up with newer titles. The recreated animations are on average considered more realistic than animation from a 2008 game. The software produces animations of similar realism as animations featured in a 2015 game and is slightly outperformed by a 2017 game. One thing that is worth noticing is that the data suggests that the recreated scenes are capable of conveying the emotions and feelings correctly, however this is not enough for them to be seen as 'realistic'.

[1]The participants were asked to say whether they agree with the statement 'video X is realistic'. The answer was taken on a one to five scale where one means 'strongly agree' and five means 'strongly disagree'

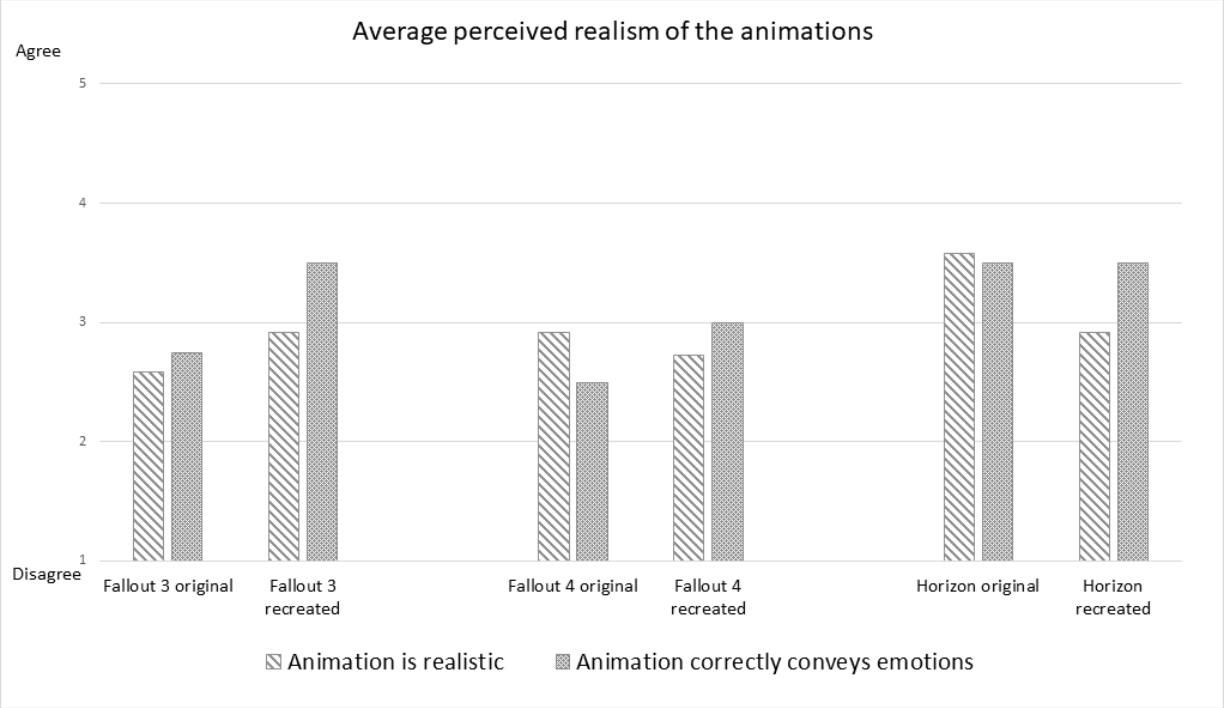

Figure 8.1: Average perceived realism of the animations

Despite the not so positive results above, the participants actually seem to prefer the recreated videos. When ad if the recreated animation makes the scene more enjoyable almost 70% of participants agreed (figure 8.2). Similarly, over 60% of participants (figure 8.3) pointed to the recreated animation when asked which animation do they subjectively prefer. The interviewees justified their choice by statements such as 'Body language matches with what is being said' or 'More accurate expression of feelings'. Adversely, the supporters of the original animations described the recreated animation as 'over the top' and with 'weird hand movements'. The answers do not differ greatly game to game.

The last important piece of information gathered from the participants was whether they though that the animations generated by the software need adjustments before being used in a finished game. The answers lean slightly into the 'needs serious adjustments' side, however the average suggests that the animations need a moderate amount of adjustments.

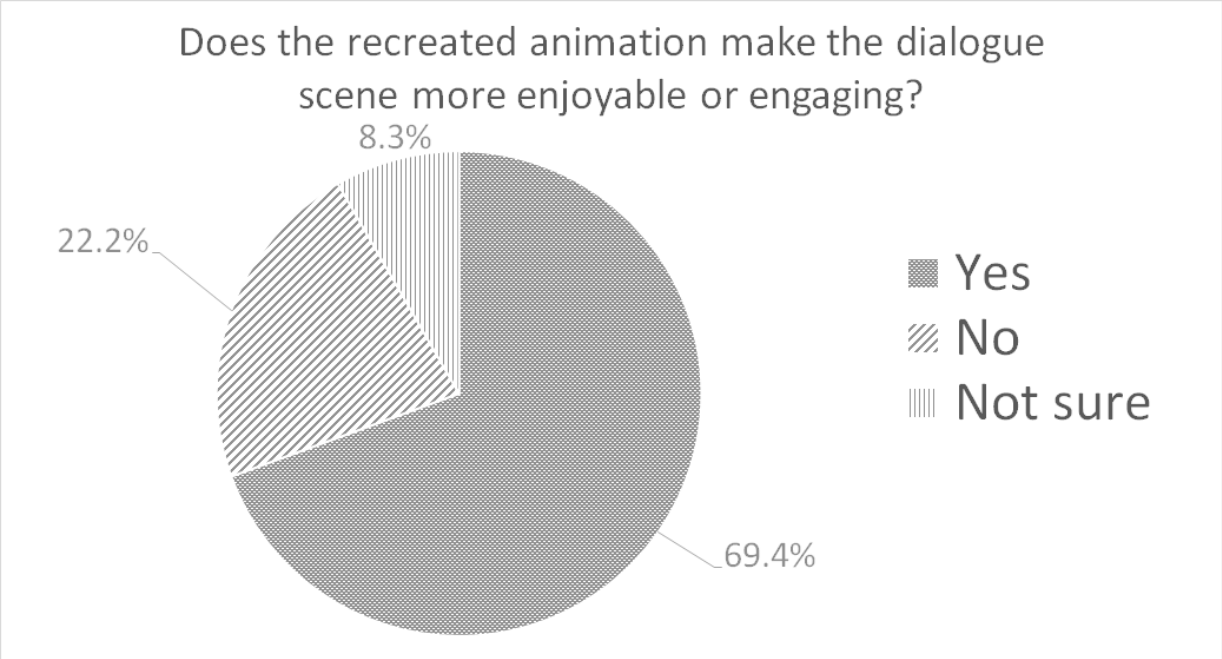

Figure 8.2: Does the recreated animation make the dialogue scene more enjoyable or engaging?

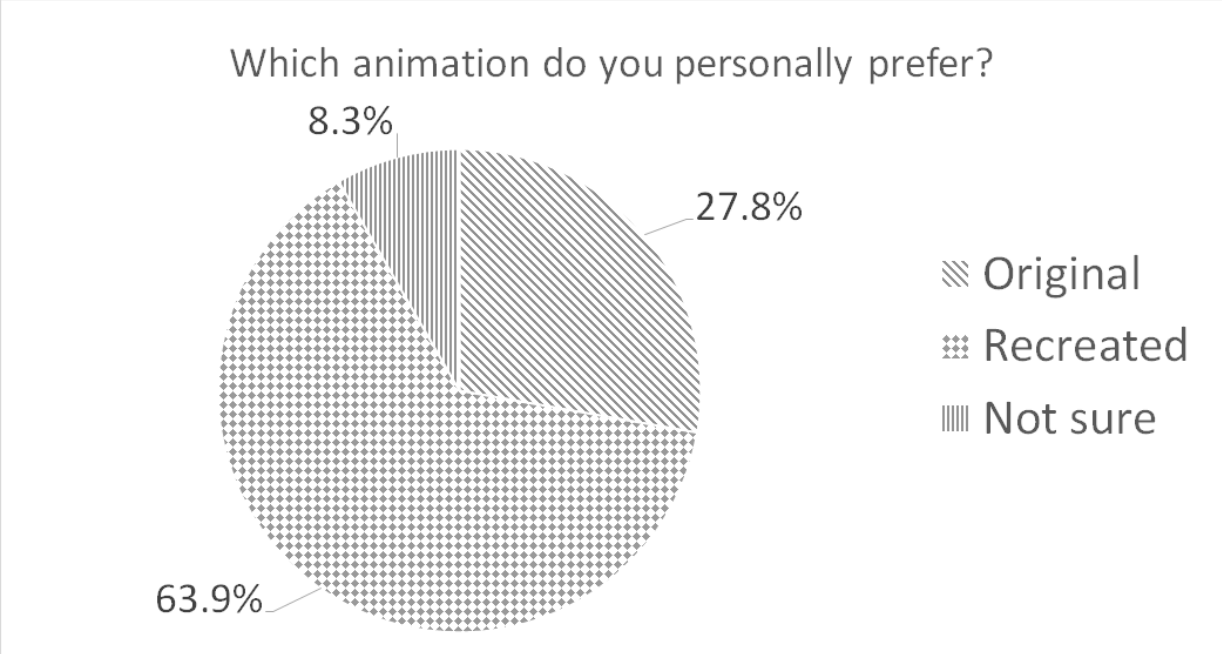

Figure 8.3: Which animation do you personally prefer?

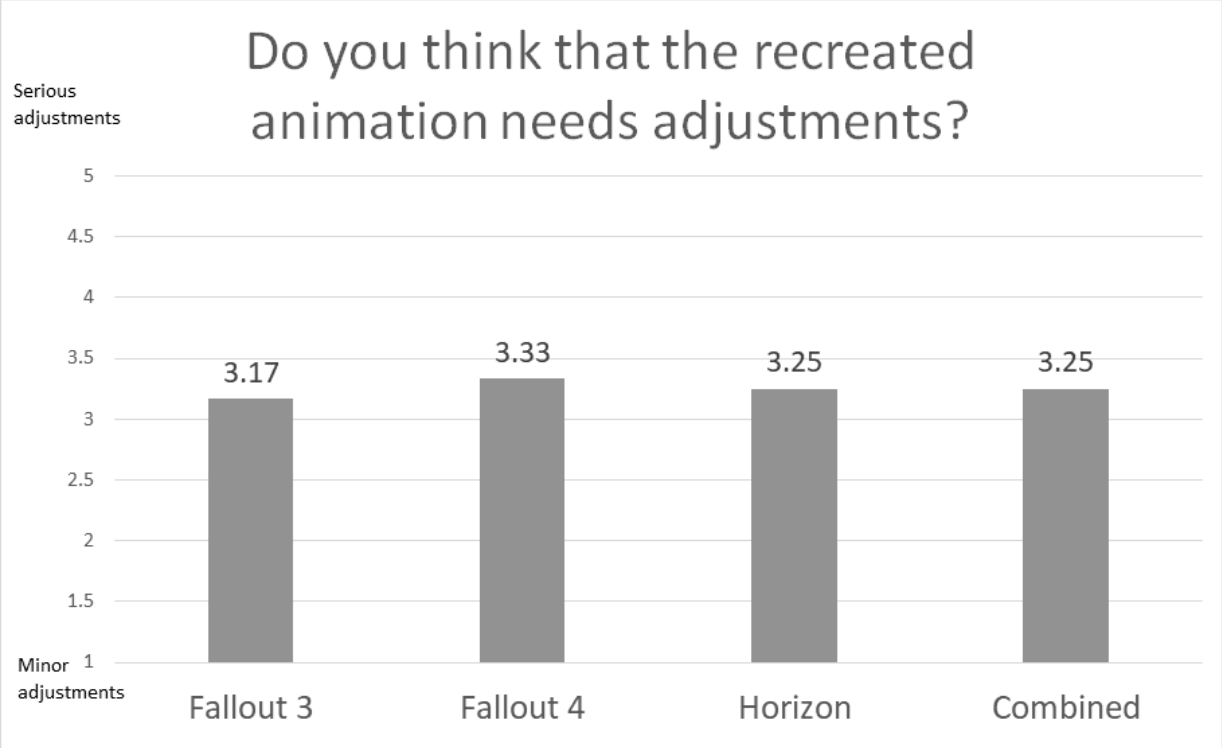

Figure 8.4: Do you think that the recreated animation needs adjustments (before being used in a finished game)?

Chapter 9

Discussion

This section discusses and evaluates the findings accumulated during the development and evaluation of the software.

9.1 Survey results

9.1.1 Realism

According to the results of the survey the realism of the generated animations is rather disappointing. The animations managed however to be relatively as realistic as an animation from a quite recent game, meaning they might be good enough to be used in a game (given how little it takes to generate them). The very important aspect is that the generated animation were not seen as realistic, but were seen as correct (matching speech with emotions and body language). People dissatisfied with the recreated animations said that the movements of characters are 'weird' and that the software is too sensitive with characters overreacting to the actual situation.

The survey successfully carried out its first objective, showing that the NLP approach did not produce animations more (or much less) realistic than the originals. There are a few explanations to that. The possible causes have been described in section 3.3.

The lack of realism was probably not caused by the emotion analysis. The interviewees rather agreed that the emotions in the speech match the body language. The results suggest that the problems lies within the movements itself.

Both the motion capture data quality and the uncanny valley principle are possible causes of those results. It is impossible to say now which one is more important. A similar study which uses a different motion database could be helpful in determining that.

It is also possible that the results are skewed because of the crudeness of the entire scene. The models are rather bulky and poorly detailed, untextured, there is no background or audio. That is because the software focuses solely on the animation. However, the interviewees are not used to watching animation being developed and might instinctively see such a scene as less realistic and be unable to look at the animation fully objectively. Given enough time and resources these issues could be addressed by adding detail to the scene by professional artists and the system should be then evaluated by a truly random audience (not dominated by CS students).

The generated scenes reach an acceptable level of realism, however they are not realistic enough to deem the NLP method of generating animation to be better than other, traditional ones.

9.1.2 Preference

Surprisingly, although the generated animations were not highly realistic, they were often preferred to the original ones. It seems that the recreated animation may have been over the top, while in the original scenes there was not much going on (the gestures being very subtle and ambiguous). It seems that the participants found the slightly overdone body language more enjoyable and less boring than the blandness and genericness of the originals.

Realism itself is often not a key aspect to a game's success. It might be a great advantage for a game to feature scenes that are similarly or slightly less realistic than the norm when those animations are more engaging. This might be particularly true when a given game is not trying to be realistic (it might be over the top and cartoony on purpose).

9.1.3 Adjustments

The results regarding adjustments of the animation were not surprising. The software was never intended to produce perfect animation which needs minimal adjustments before shipping (there is hardly any software capable of that). Moderate adjustments of the animation are perfectly acceptable. If this software was ever to become fully commercially viable, it would be a good idea to further develop the animation generation. Better camera work, smooth blending of the movements, etc. can be done automatically and would further minimize the need for manual adjustments.

9.2 Other findings

In previous sections I mentioned that the scenes chosen for the questionnaire are not *fully* representative of all the dialogue scenes in the game. When a scene satisfies some conditions, the software clearly under-performs. Below are some of such cases that I discovered.

9.2.1 Mismatch between animation database and the 'feel' of the game

The software produces undesirable results when there is a clear mismatch between the provided gestures and the 'feel' of the game. By 'feel' I mean the general topic and overtone of the game. For instance, a game such as any in the Mass Effect series has a very strict and militaristic feel. While characters still converse in a more casual environment a certain level of professionalism is always required of them. A game as Yooka-Laylee also contains a number of dialogue scenes, but the general feel of the game is childish and cartoon-like. The conversation happen in a much more quirky and exaggerated way.

I was not able to use the EBMD with the Mass Effect games. The EBMD provides very casual, relaxed gestures while almost every character in the Mass Effect game is military. While without context the scenes looked good, when keeping in mind that the scene happens in a militaristic environment one can immediately see that while the characters move and gesticulate in a natural manner, this is not appropriate in a current situation.

This however is not an inherent problem with the software as it allows for using a custom animation database. It could potentially be even extended to support a different set of animations for each important character to underline their personality even more. However because I used only EBMD for this project I am unable to test how if the software would be successful with scenes that have different contexts.

9.2.2 Exposition dialogue

In media such as books, films and games exposition is pretty common. Exposition is an aspect narration that focuses on introducing character's backstories, prior plot events, context and other background information. Because typically games feature less narration than books or films most of exposition is handled through dialogue.

The exposition dialogue is already often unnatural and seems out of place. The software generating those scenes made them seem even less natural. During exposition dialogue the character's should remain rather neutral as often they will be describing events that happened in the past or that they have no direct connection with. The software however will find emotions in this dialogue and make characters overjoyed or fearing while there is nothing happening that would provoke such emotion. In many such dialogue scenes the software was simply over-sensitive.

This problem could potentially be solved if the exposition dialogue was tagged in the script by the writers. The emotions extracted from the text could be decreased by some factor to make them less intense. This of course assumes an increase in manual work required as the exposition dialogue would have to be manually tagged.

9.2.3 Sarcasm and ambiguity

NLP has struggled with sarcasm for a long time. IBM Watson Tone Analyzer seems to be no closer to solving this problem. The dialogue that contain sarcasm are simply misinterpreted and the final scenes look terribly out of place. For instance a character saying 'Oh, I'm so scared!' can be either scared, or indifferent and boasting about their courage. This might depend on the context and the tone of their voice. The software will however always associate this with fear and the resulting scene might use animations that convey opposite emotions to those that were meant to be conveyed.

By looking at many dialogue scenes it became apparent that there is no way to overcome this problem using NLP only as there are too many variables outside of the text that define it (context, setting, personality of the character, recent interactions between characters involved in the scene). The tone of voice might provide more information that would help solve this problem.

Similarly to the exposition problem, the sarcasm problem can also be solved by tagging it in the script. When using sarcasm the writers could indicate that and provide the intended emotion. That assumes a slight increase in the amount of work required and that all sarcasm present in the script is recognized and tagged.

9.3 Summary

The following is a breakdown of what the software managed and failed to achieve.
Successes:

- The software is able to generate scenes quickly with minimal amount of manual work required.
- The animations do not need extensive manual polish before being used in a game. The software needs to provide smoother movements and blending between gestures.
- The software is flexible and capable of using custom character models and animation data.

- The software is cross-platform and the animations can be exported to a multitude of file formats.
- The generated scenes are often more enjoyable than the actual scenes in games.

Failures:

- Generated scenes are not any more realistic than those featured in modern games.
- The software under-performs when dealing with sarcasm, exposition or ambiguity.

Other key points:

- The lack of realism may be caused by the data provided by EBMD.
- The animation generation process can be improved further using methods already known to the industry.
- The NLP approach might in the future be combines with other methods (such as emotion analysis of the audio) to produce even better results.

Chapter 10

Conclusion and Future work

This chapter concludes the project and provides suggestions as to how the project can be potentially improved upon in the future.

10.1 Conclusion

This project has developed software capable of creating natural-looking animated dialogue scenes using nothing but a script (resembling a film script) as input. While the gestures and motions are often 'correct' they are often overdone reducing the overall realism o the scene. Despite that, the generated scenes were often found more enjoyable than those of modern games by the audience.

The main problems that the software faces are sarcasm and ambiguity. There exists a possibility that these issues cannot be solved by natural language processing alone and other methods should be employed to solve these issues.

The developed software requires far less manual preparation and work than other existing systems. The trade-off between realism (and quality) and the amount of required work makes the system viable for: studios which can't afford to develop their own animation generation system or hire of additional animators, amateur and indie developers who either lack skill or time to produce large quantities of animation and prototyping - the quick execution of the program allows for a quick assembly of a scene which can be then looked at and improved upon.

The project, especially upon further development, has potential to bring cheap and relatively good animation to games that were unable to do so. However, big studios will would probably remain uninterested in this approach as they already have developed their in-house solutions to the problem of animation generation and have far more resources available.

10.2 Potential use outside of games

The software was developed primarily for use in games, however there are also other areas that would benefit from a similar system. One of such areas is robotics, which suffers from a very similar problems to games. Robots, when interacting with people, must behave in a human-like manner with proper gestures and body language. Matching emotions with the motion capture database is something that could be a potential solution - only instead of creating an animation, the movements could be re-enacted by a robot. It has been established that robots which clearly indicate their intentions with body language are perceived as much more pleasant and easy to interact. [12]

10.3 Future work

While the software managed to deliver on its most important requirements, there is a lot of room for improvements. Most importantly, this project focused mostly on how the animations are assembled from the database with use of emotion analysis. The generation itself was not itself a priority as there exist a lot of system that do this part, so there was nothing to be learned here. However, for this project to become commercially viable, the generation of animations could be improved. Potential improvements include: more complex camera movement, inclusion of more characters in the scene, blending the motions so that switching from one gesture to another looks more natural, creating the scene immediately with background objects (possibly also described with the script), etc.

The hardest problem this software has yet to overcome is sarcasm and ambiguity. The emotions and body language of a person are not only and fully dependent and what they say - it is also important how they say it and when, what is the context and setting of the scene and more. I believe that this software could become a part of a bigger emotion-to-dialogue-animation tool, which features other approaches to solve the problem. Potential extensions to the software:

- Usage of machine learning to analyse emotions from audio (if audio is available before scene generation).
- Usage of machine learning to analyse emotions from facial movements (the voice actors' faces could potentially be recorded while reciting the script).
- Using NLP to also analyse past interactions between characters (and build a profile for each character and their relations with others) and taking into account not only the emotions of the person currently speaking but of all characters within the scene.

Bibliography

[1] Brodkin, J. (2014). Game developers crossing the uncanny valley. https://insights.dice.com/2014/02/14/game-developers-crossing-uncanny-valley/.

[2] Burton, N. (2016). What are basic emotions? https://www.psychologytoday.com/blog/hide-and-seek/201601/what-are-basic-emotions.

[3] Cavazza, M. and I., P. (1999). Natural language control of interactive 3d animation and computer games. Technical report, University of Bradford.

[4] Chang, L. (2015). Ibm's watson may now be able to tell how snarky your email is with its new tone analzyer. https://www.digitaltrends.com/cool-tech/watsons-new-tone-analyzer-can-tell-if-your-email-is-too-snarky/.

[5] Feldman, R. (2013). Techniques and applications for sentiment analysis. *Communications of the ACM*, 56:82–89.

[6] Fenlon, W. (2016). Most of the witcher 3's dialogue scenes were animated by an algorithm.

[7] Hernandez, P. (2017). People are trashing mass effect: Andromeda's animation. https://kotaku.com/people-are-ripping-apart-mass-effect-andromedas-animati-1793334047.

[8] Jack, R. E., Garrod, O. G., and Schyns, P. G. (2014). Dynamic facial expressions of emotion transmit an evolving hierarchy of signals over time. Technical report, University of Glasgow.

[9] Levison, L. (1991). Action composition for the animation of natural language instructions. Technical report, Universirt of Pennsylvania.

[10] MacDorman, K. F. and Chattopadhyay, D. (2014). Reducing consistency in human realism increases the uncanny valley effect; increasing category uncertainty does note. Technical report, Indiana University School of Informatics and Computing.

[11] Mori, M. (1970). The uncanny valley. https://spectrum.ieee.org/automaton/robotics/humanoids/the-uncanny-valley.

[12] Mutlu, B., Shiwa, T., Kanda, T., Ishiguro, H., and Hagita, N. (2009). Footing in human-robot conversations: How robots might shape participant roles using gaze cues. In *Proceedings of the 4th ACM/IEEE International Conference on Human Robot Interaction*, HRI '09, pages 61–68, New York, NY, USA. ACM.

[13] Oshita, M. (2009). Generating animation from natural language texts and framework of motion database. Technical report, Kyushu Institute of Technology.

[14] Rohit, W. and Jagdale, R. (2018). An approach to sentiment analysis. Technical report, Department of CS & IT, Dr. BAMU, Aurangabad, Maharashtra, India.

[15] Tominski, P. (2016). Behind the scenes of cinematic dialogues in 'the witcher 3: Wild hunt. http://www.gdcvault.com/play/1022988/Behind-the-Scenes-of-Cinematic.

[16] Volkova, E., Mohler, B., de la Rossa, J., and Bulthoff, S. (2014a). The mpi database of emo- tional body expressions common for narrative scenarios. Technical report, Max-Planck Institute for Biological Cybernetics in Tuebingen, Germany.

[17] Volkova, E., Mohler, B., Tesch, T., and Bulthoff, S. (2014b). Emotion categorisation of body expressions in narrative scenarios frontiers in psychology. Technical report, Max-Planck Institute for Biological Cybernetics in Tuebingen, Germany.

Appendix A

User Manual

The system is designed to be cross platform, however it is advised to use Windows as it is the only system on which the project was thoroughly tested.

A.1 Requirements and installation

Before using the software, several requirements must be fulfilled.

A.1.1 IBM Watson Tone Analyzer Setup

Before continuing with any of the sections you must setup the Tone Analyzer service. To do that please visit the following link: `https://www.ibm.com/watson/services/tone-analyzer/`. You will need to create an IBM account and log in. after logging in add a new Tone Analyzer service and create a new set of API credentials that can access it. Please refer to the official guide for more details: `https://console.bluemix.net/docs/services/tone-analyzer/getting-started.html#getting-started-tutorial`.

When you setup your Tone Analyzer service, open file 'analyzer/tone.py'. Please find the declaration of the 'tone_analyzer' variable in the beginning of the file. You should see a snippet of code resembling the one that can be seen in figure A.1. Replace the username and password with your credentials (Changing the API version might cause potential problems and unexpected behaviour) and save the file.

```
8  # Specify API version and credentials
9  tone_analyzer = ToneAnalyzerV3(
10   version='2016-05-19', # Please don't change the API cersion
11   username='67peb3d5fd-57ef-4133-a7c0-b44424221b94', # Replace with your username
12   password='d7qPAsswE7TT' # Replace with your password
13 )
```

Figure A.1: Tone Analyzer credentials.

A.1.2 Software and Libraries

Required software

- Python 2.7
- Pip
- Blender version 2.79 or newer[1]

[1] Blender can be obtained from the official Blender website: `http://www.blender.org/`

Libraries

Only for Linux systems: Download and install library python-dev (or python-devel) before following the instructions below

Following python libraries must be installed (using pip, easy-install, or any other python packet manager):

- setuptools
- watson-developer-cloud
- wget

The packages can be installed in bulk by running:

‘path_to_python/Scripts/pip install -r requirements‘

in the main directory. As Blender usually comes with its own Python environment, you will need to run the command:

‘path_to_blender/version/python/Scripts/pip install -r requirements‘

To install the libraries for Blender.

A.1.3 Install Generator as Blender Addon

The following steps are optional. Installing the module as addon will enable the module to be permanently incorporated into Blender.

To install the addon:

1. Open Blender
2. Open user settings (File > User preferences)
3. Open Add-ons tab
4. On the bottom panel press ‘install add-on from file‘
5. Navigate to ‘project folder > generator‘ and double click on ‘addon.py‘
6. In the search box on the top right corner type ‘NLPanim‘ - a greyed-out item called ‘Object: NLPAnim‘ should appear
7. Check the box next to the item
8. Press ‘Save user settings‘ in the bottom left corner
9. Exit user preferences

The module is now installed as an add-on. In the 3D-view (default view) on the bottom of the left-hand side menu there should appear a tab called ‘Generate NLP anim‘ (figure A.2).

A.2 Generating animations

Now that all the requirements are met, the animation can be generated. There are two main steps to generating. Firstly, use the analyzer module to create a scene.json file[2]. Secondly, use the generator to assemble the animation.

[2]scene.json is a JSON file which contains instructions for the generator module that describe how to assemble the animation.

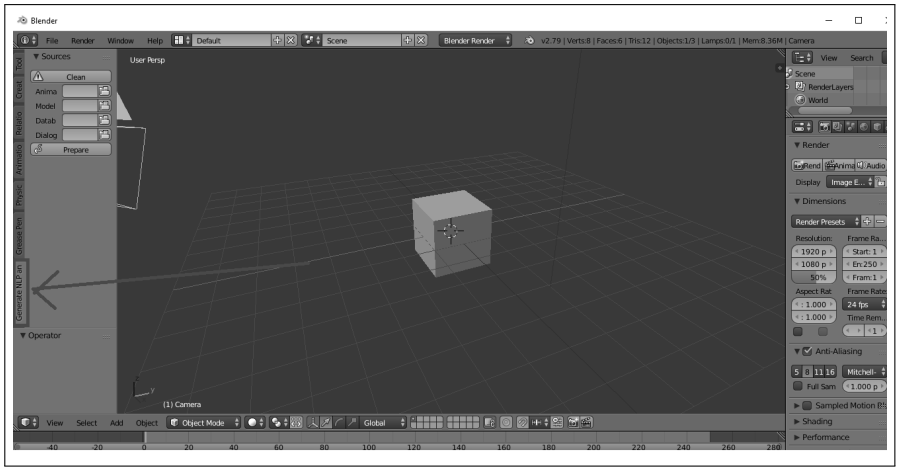

Figure A.2: The generator addon menu

Analyze script

The input script must resemble the script in figure A.3

```
                              SANDRA
                    Sonny?  Sonny, who is it?
                    What is it?

                              SONNY
                    They shot the old man.

                              SANDRA
                    Oh God...

                              SONNY
                    Honey...don't worry.  Nothing else
                    is going to happen.

ENDSCRIPT
```

Figure A.3: An example of system's input

The characters must be specified after 5 tabs. The dialogue text is specified after 3 tabs. The file must end with an 'ENDSCRIPT' with no indentation. For reference please refer to either:

- Example script files in folder 'movies'
- Movie scripts hosted on `http://www.imsdb.com`

The script now can be process using the following command:

'python analyzer/script_analyze.py path_to_dialogue_script db/animation.db'

The program will output a file called 'scene.json'. This file is used to generate the animation.

Generate Animation

If you did not install the module as a Blender add-on (if you skipped section A.1.3), please follow these steps:

1. Open the file ‘rendered.blend’ with Blender
2. Click ‘Run Script‘ on the bottom of the scripting view (figure A.4)
3. The tab called ‘Generate NLP anim’ should appear on the right hand side menu of the 3D view
4. Click on the ‘Generate NLP anim’ tab

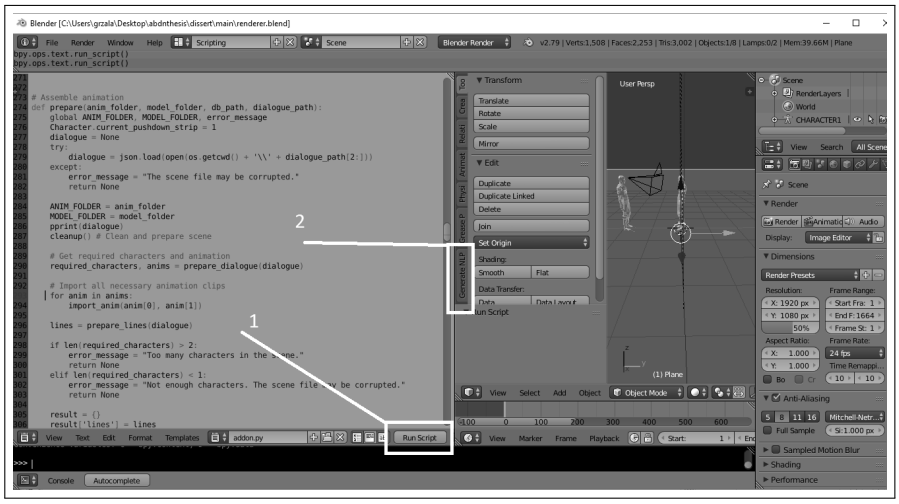

Figure A.4: Running the generator without installing it as an add-on

Please follow these steps only if you installed the generator as an add-on:

- Open Blender
- Press ‘File > Save’ and save the file in a preferred location
- **Important:** close Blender and reopen the saved file
- Open ‘Generate NLP anim’ tab

After finishing the above instructions, do the following to finalize the animation:

1. Press ‘Clean’ at the top of the menu
2. Choose the directory ‘mocap’ for ‘Animations Directory’
3. Choose the directory ‘models’ for ‘Models Directory’
4. Choose the file ‘db/animation.db’ for ‘Database’
5. For ‘Dialogue’ choose the ‘scene.json’ file you created when following the section A.2
6. Press ‘Prepare’ and wait until a ‘Finalize’ menu appears below the ‘Prepare’ button
7. Assign models for the characters from the drop-down lists
8. Press ‘Assemble’ and wait until done

The animation is now finalized. It can now be viewed, edited or rendered. Please refer to Blender official documentation and tutorials for more information on this.

A.3 Using Custom Animation and Models

This section describes how to extend or replace the animation database.

To insert a new animation file, ensure that the file is saved as .fbx file and contains only one animation (action). Please make sure that the armature[3] is compatible with the one found in 'EMBD importer/armature.fbx' file.

To continue you will need to install 'DB Browser for Sqlite'[4]. Use that program to open the file 'db/animation.db'. Browse the table 'animations' and add a new record. Provide a name for the animation file (please note the names must be unique) and a path to the animation file. You must also fill the emotion values for the animation (set to 0.0 if the emotion is not carried by the animation).

To insert a new model, browse the 'models' table. Again, provide a unique name and a file path for the .fbx model. You will also need to provide information on how to position and rotate the camera to focus it on the characters face (relative to the origin of the character model[5]). You will need to provide six values in total.

- Offset on axis X, Y and Z (positive Z means upwards, negative Y is forwards and positive X is to the right) in Blender units (distance).
- Rotation on axis X, Y and Z in degrees.

You can use your own database file as long as the database keeps the following structure:

- 'animations' table:
  - id (Integer, unique, primary key)
  - name (Text, unique)
  - file (Text)
  - Frames (Integer)
  - anger (Numeric)
  - joy (Numeric)
  - sadness (Numeric)
  - fear (Numeric)
  - disgust (Numeric)
  - surprise (Numeric)
- 'models' table:
  - ID (Integer, unique, primary key)
  - name (Text, unique)

[3] Armature is a skeleton; A set of bones that can be animated. If you are unsure of what an armature is, you should avoid using custom animations and models.

[4] The DB browser for Sqlite can be obtained from the sqlitebrowser official website: `http://www.sqlitebrowser.org`

[5] The origin of a character model is usually a point on the floor between the character's feet (directly below the spine).

- file (Text)
- cam_offset_x (Numeric)
- cam_offset_y (Numeric)
- cam_offset_z (Numeric)
- cam_rot_x (Numeric)
- cam_rot_y (Numeric)
- cam_rot_z (Numeric)

Keep the emotion values from 0.0 to 1.0.

A.4 Importing More Animations From EBMD

If you wish to import more animations from EBMD you can use the EBMD importer tool. It will allow you to quickly process huge amounts of EBMD animations and export them to .fbx files compatible with the project.
To use the importer tool

1. Open file 'EMBD importer/importer.blend' with Blender
2. In the text editor on the left scroll to the bottom
3. There you will find arrays with links to EMBD animations
4. Visit `http://ebmdb.tuebingen.mpg.de/index.php`
5. Browse the EMBD website in search of desired animations
6. Copy the links to BVH files and paste them in the arrays in the script
7. Press 'Run Script' on the bottom panel
8. **Important:** The process might take a long time depending on the amount of required animations - please do not interrupt or close Blender even when the system labels the process as unresponsive
9. The imported files can be found in the folder 'imported'

A.5 Generate a Showcase

Since all animations must be reviewed manually in order to assign their emotion value the showcase generator tool was created to help with this task. The process of generating a showcase will take a number of animations and display them all in one animated clip while showing their filenames as subtitles
To generate a showcase:

1. Run the command 'python analyzer/generate_showcase.py path_to_directory_with_animations'
2. The program will create a number of json files (showcase0.json, showcase1.json, etc) in the 'showcases' folder

3. Each showcase file contains instructions for the generate to create an animation consisting of up to 10 animation (splitting the scene into 10 clips each makes it easier to render and use)

4. Use the showcase files as scene files with the generator module (section A.2)

5. Render the animation

Appendix B

Maintenance Manual

B.1 Installation and requirements

For information on installation and dependencies please refer to sections A.1 and A.1.3.

B.2 Space and Memory requirements

The project needs enough memory to install Blender and Python as well as another 100MB to fit all the animation data. Please keep in mind that rendering the animations drastically increases the memory needed as one short clip may take up to 50MB. The project comes with a few example rendered videos which take another 150MB of memory

B.3 Temporary Files

When using the EBMD importer the program will store the downloaded EBMD BVH files in the 'EMBD importer/tmp' directory. These files can take a considerable amount of space and can be deleted at the convenience of a user.

B.4 Files and Directories

The files and directories are described in tables B.2, B.3, B.4, B.5 and B.1.

Main project directory	
Item	Description
analyzer/	The directory that contains the first and second module source code (emotion analysis and matcher)
db/	The directory that contains the sqlite database file which holds information on the animations and models
EMBD importer/	The directory that contains the EMBD importer tool
generator/	The directory that contains the Blender add-on
mocap/	The directory that contains the .fbx animations
models/	The directory that contains the .fbx character models
inputs/	The directory that contains sample dialogue scripts
showcases/	The directory that contains sample rendered scenes
renderer.blend	Blender project prepared for use with the system
requirements	File containing python dependencies

Table B.1: Contents of the main directory

'analyzer' directory	
Item	Description
db.py	The source code of the matcher module
generate_showcase.py	The source code of the showcase generator (more information in sections 5.5.2 and A.5)
parse.py	Script used to parse the dialogue script
script_analyze.py	The main file of the analyzer program. It handles the resource pipeline.
tone.py	The script that uses the IBM Watson API to perform emotion analysis

Table B.2: Contents of the 'analyzer' directory

'db' directory	
Item	Description
animation.db	The sqlite file that holds the information on the animations and character models

Table B.3: Contents of the 'db' directory

'EMBD importer' directory	
Item	Description
imported/	The directory that holds the imported animations
tmp/	The directory that caches downloaded EMBD animations
armature.fbx	Character armature used for reference during importing. This armature is used for all the animations in the project
importer.blend	Blender project prepared to handle importing animations from EMBD
script.py	Source code of the importer. This script is meant to be run from inside of Blender

Table B.4: Contents of the 'EMBD importer' directory

'generator' directory	
Item	Description
addon.py	The source code of the generator module

Table B.5: Contents of the 'generator' directory

B.5 Known Issues

Please make sure that the dialogue script is correctly formatted before running the emotion analysis. Wrong indentation levels may cause unpredictable results.

The generator module will provide you with an error message if the input (scene.json) is malformed. However if the file parses correctly but it's data is corrupted it may lead to unpredictable results. This situation will never take place is the scene.json file was generated by the script_analyze.py program.

Appendix C

Other

C.1 Useful Links

Sample generated animations and animations used in the questionnaire: `http://mpanasiuk.me/bscthesis/samples.zip`
The Emotional Body Motion Database: `http://ebmdb.tuebingen.mpg.de/`
IBM Watson Tone Analyzer: `https://www.ibm.com/watson/services/tone-analyzer/`
Tone Analyzer Guide: `https://console.bluemix.net/docs/services/tone-analyzer/`
Blender: `http://www.blender.org/`
International Movie Script DataBase: `http://www.imsdb.com`
DB browser for Sqlite: `ttp://www.sqlitebrowser.org`

C.2 The questionnaire

Please see figures C.1 to C.6.

C.3 Sample generated scene used in the questionnaire

The following scene comes from the game *Horizon: Zero Dawn*. The screenshots depicting the scene can be found in figures C.7 to C.10. Please note that the top images represent the original scene while the bottom ones show the recreated scene.

C.4 Interviewees' comments about the recreated animations

The following is a full list of comments that the interviewees made when asked to justify some of their choices. Those comments were not required of the participants and some refused to give more detailed feedback. Please note that Animation/Option/Video 1 refers to the original animation, while the Animation/Option/Video 2 refers to the recreated one.
Comments in favour of the recreated animations:

- 'The second animation is corresponding better to the situation in the game.'
- 'Active body language refers to what is being said.'
- 'Better expression of emotions but sometimes they were exaggerated.'
- 'Both animations present natural movements of the body.'
- 'No body language [in animation 1].'

- 'More accurate expression of feelings.'
- 'Option 2 because character has more natural behavior.'
- 'Body language matches with what is being said.'
- 'Natural body motions [in animation 2].'

Comments in favour of the original animations:

- 'Weird hands movement in the 2nd one.'
- 'Option 1 because in option 2 author moved too far.'
- 'Weird hand movement in the 2nd one.'
- '[Animation 1] Looks better.'

Other comments:

- 'Both about the same, animation 1 didnt represent any emotion, animation 2 was little over the top'
- 'Both animation represented the scene about the same, didnt really have a preference.'
- 'I think both represented the emotion about the same.'
- 'Option 1 is not bad, option 2 is overly exaggerated.'

Natural language processing for animated dialogue scene generation

Please answer the questions and follow the instructions below
Please keep in mind that the project focuses solely on animation (body movements). Please try to ignore the backgrounds, graphics and facial features as much as possible.

Please note that all the videos have been muted on purpose. In some videos you will notice that there are two sets of subtitles. If that is the case it means that an enlarged version of the subtitles was added next to in-game subtitles to improve readability.

*Required

1. **Do you consider yourself to be well-acquainted with games and/or animation in general?**
   Mark only one oval.

   - ◯ Yes
   - ◯ No
   - ◯ Not sure

Part 1

Please watch the following videos:

Video 1: https://vimeo.com/263161821
Video 2: https://vimeo.com/263161000

Now please answer the following questions:

2. **The animation in video 1 is realistic** *
   Mark only one oval.

	1	2	3	4	5	
Strongly Agree	◯	◯	◯	◯	◯	Strongly Disagree

3. **The animation in video 2 is realistic** *
   Mark only one oval.

	1	2	3	4	5	
Strongly Agree	◯	◯	◯	◯	◯	Strongly Disagree

Figure C.1: The questionnaire - page 1

4. **If you chose option 1 or 2 in the question above, please explain why**

5. **The animation in video 1 correctly conveys the feelings and emotions of the characters** *
Mark only one oval.

	1	2	3	4	5	
Strongly Agree	◯	◯	◯	◯	◯	Strongly Disagree

6. **The animation in video 2 correctly conveys the feelings and emotions of the characters** *
Mark only one oval.

	1	2	3	4	5	
Strongly Agree	◯	◯	◯	◯	◯	Strongly Disagree

7. **Do you think that animation in video 2 makes the dialogue scene more enjoyable or engaging than the animation in video 1?** *
Mark only one oval.

- ◯ Yes
- ◯ No
- ◯ Not sure

8. **Animation in video 2 needs only minor adjustments before being used in a finished game** *
Mark only one oval.

	1	2	3	4	5	
Strongly Agree	◯	◯	◯	◯	◯	Strongly Disagree

9. **Which animation did you personally prefer** *
Mark only one oval.

- ◯ Video 1
- ◯ Video 2
- ◯ Not sure

Figure C.2: The questionnaire - page 2

10. **Explain what motivated your choice in the question above (optional)**

Part 2

Please watch the following videos:

Video 1: https://vimeo.com/263163099
Video 2: https://vimeo.com/263163430

Now please answer the following questions:

11. **The animation in video 1 is realistic** *
Mark only one oval.

	1	2	3	4	5	
Strongly Agree	◯	◯	◯	◯	◯	Strongly Disagree

12. **The animation in video 2 is realistic** *
Mark only one oval.

	1	2	3	4	5	
Strongly Agree	◯	◯	◯	◯	◯	Strongly Disagree

13. **If you chose option 1 or 2 in the question above, please explain why**

14. **The animation in video 1 correctly conveys the feelings and emotions of the characters** *
Mark only one oval.

	1	2	3	4	5	
Strongly Agree	◯	◯	◯	◯	◯	Strongly Disagree

Figure C.3: The questionnaire - page 3

15. **The animation in video 2 correctly conveys the feelings and emotions of the characters** *
Mark only one oval.

	1	2	3	4	5	
Strongly Agree	◯	◯	◯	◯	◯	Strongly Disagree

16. **Do you think that animation in video 2 makes the dialogue scene more enjoyable or engaging than the animation in video 1?** *
Mark only one oval.

- ◯ Yes
- ◯ No
- ◯ Not sure

17. **Animation in video 2 needs only minor adjustments before being used in a finished game** *
Mark only one oval.

	1	2	3	4	5	
Strongly Agree	◯	◯	◯	◯	◯	Strongly Disagree

18. **Which animation did you personally prefer** *
Mark only one oval.

- ◯ Video 1
- ◯ Video 2
- ◯ Not sure

19. **Explain what motivated your choice in the question above (optional)**

Part 3

Please watch the following videos:

Video 1: https://vimeo.com/263208705
Video 2: https://vimeo.com/263208708

Now please answer the following questions:

Figure C.4: The questionnaire - page 4

20. **The animation in video 1 is realistic** *
Mark only one oval.

	1	2	3	4	5	
Strongly Agree	◯	◯	◯	◯	◯	Strongly Disagree

21. **The animation in video 2 is realistic** *
Mark only one oval.

	1	2	3	4	5	
Strongly Agree	◯	◯	◯	◯	◯	Strongly Disagree

22. **If you chose option 1 or 2 in the question above, please explain why**

23. **The animation in video 1 correctly conveys the feelings and emotions of the characters** *
Mark only one oval.

	1	2	3	4	5	
Strongly Agree	◯	◯	◯	◯	◯	Strongly Disagree

24. **The animation in video 2 correctly conveys the feelings and emotions of the characters** *
Mark only one oval.

	1	2	3	4	5	
Strongly Agree	◯	◯	◯	◯	◯	Strongly Disagree

25. **Do you think that animation in video 2 makes the dialogue scene more enjoyable or engaging than the animation in video 1?** *
Mark only one oval.

- ◯ Yes
- ◯ No
- ◯ Not sure

Figure C.5: The questionnaire - page 5

26. **Animation in video 2 needs only minor adjustments before being used in a finished game** *
Mark only one oval.

	1	2	3	4	5	
Strongly Agree	◯	◯	◯	◯	◯	Strongly Disagree

27. **Which animation did you personally prefer** *
Mark only one oval.

- ◯ Video 1
- ◯ Video 2
- ◯ Not sure

28. **Explain what motivated your choice in the question above (optional)**

Powered by Google Forms

Figure C.6: The questionnaire - page 6

Figure C.7: Dialogue line 1.
Transcript: 'Aloy: Are you sure you're okay?'

Figure C.8: Dialogue line 2.
Transcript: 'Erend: I'm sober enough, all right? I don't need another lecture!'

Figure C.9: Dialogue line 3.
Transcript: 'Aloy: That's not what I meant, I was talking about what happened outside with the crowd.'

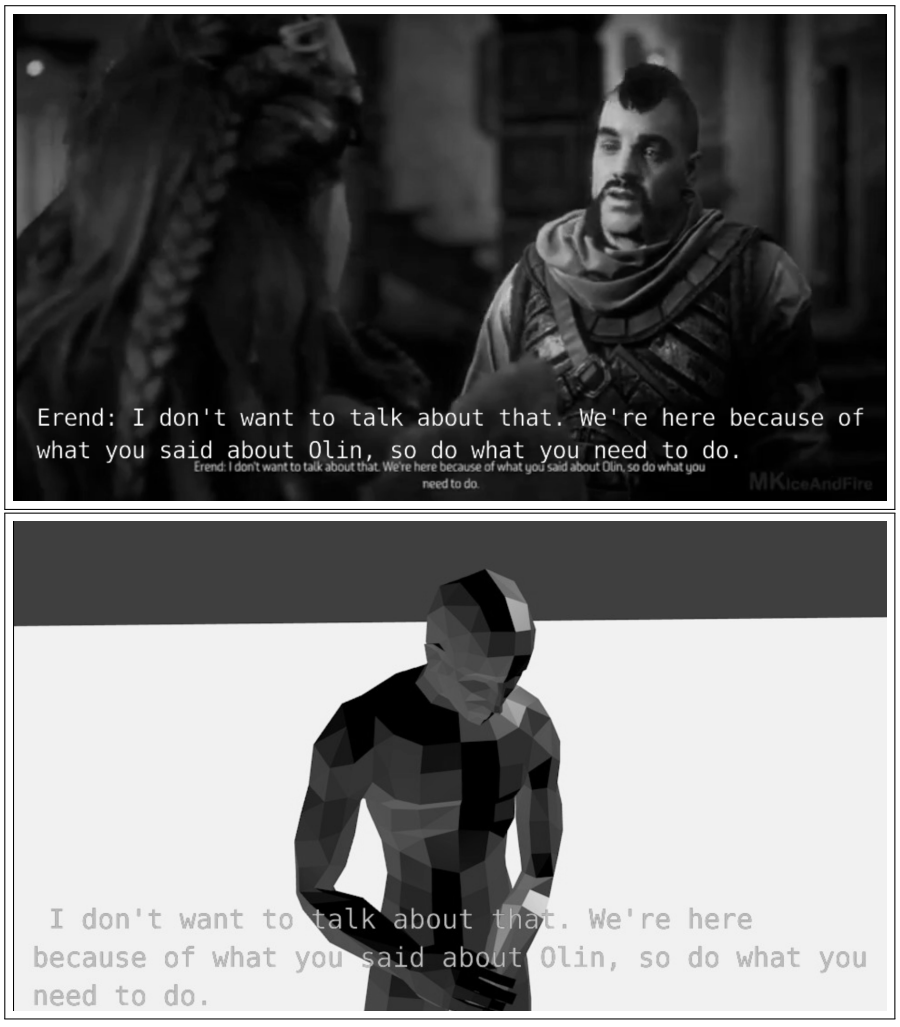

Figure C.10: Dialogue line 4.
Transcript: 'Erend: I don't want to talk about that. We're here because of what you said about Olin, so do what you need to do.'