Manuel Rebol, MSc.

Generating Non-Verbal Communication From Speech

Master's Thesis
to achieve the university degree of
Master of Science
Master's degree programme: Computer Science

submitted to

Graz University of Technology

Supervisor
Christian Gütl, Assoc.Prof. Dipl.-Ing. Dr.techn.

External Co-supervisors
Krzysztof Pietroszek, Ass.Prof. M.Sc. Ph.D.

Institute of Interactive Systems and Data Science
Head: Stefanie Lindstaedt, Univ.-Prof. Dipl.-Inf. Dr.

Graz, November 2020

Manuel Rebol, MSc.

Synthetisierung von nonverbaler Kommunikation

Master Arbeit
zur Erlangung des akademischen Grades eines
Diplom-Ingenieur
Masterstudium: Computer Science

eingereicht an der

Technische Universität Graz

Betreuer
Christian Gütl, Assoc.Prof. Dipl.-Ing. Dr.techn.

Externer Mitbetreuer
Krzysztof Pietroszek, Ass.Prof. M.Sc. Ph.D.

Institute of Interactive Systems and Data Science
Vorstand: Stefanie Lindstaedt, Univ.-Prof. Dipl.-Inf. Dr.

Graz, November 2020

Affidavit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master‘s thesis.

Date

Signature

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Das in TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Dissertation identisch.

Datum

Unterschrift

Abstract

The communication between human and virtual agents is becoming more important because of the increased usage of virtual environments in everyday life. Applications such as virtual learning platforms, online meetings and virtual assistance are gaining popularity. However, communication with virtual agents is often unintuitive and does not feel similar to talking with a person. Therefore, we focus on improving natural non-verbal human-agent communication in our work. We develop a speaker-specific model that predicts hand and arm gestures given input speech.

We model the non-deterministic relationship between speech and gestures with a Generative Adversarial Network (GAN). Inside the GAN, a motion discriminator forces the generator to predict only gestures that have human-like motion. To provide data for our model, we extract the gestures from in-the-wild videos using 3D human pose estimation algorithms. This allows us to automatically create a large speaker-specific dataset despite the lack of motion capture data.

We train our gesture model on speakers from show business and academia using publicly available video data. Once generated by our GAN, we animate the gestures on a virtual character. We evaluate the generated gestures by conducting a human user study. In the study, we compare our predictions with the original gestures and predictions from uncorrelated speech in two different tasks.

The results show that our generated gestures are indistinguishable from the original gestures when animated on a virtual character. In 53 % of the cases, participants found our generated gestures to be more natural compared to the original gestures. When compared with gestures from an uncorrelated speech, participants selected our gestures to be more correlated 65 % of times. Moreover, we show that our model performs well on speakers from both show business and academia.

Kurzfassung

Durch den vermehrten Einsatz von virtuellen Umgebungen in Alltagssituationen wird auch die Verständigung zwischen Mensch und virtuellem Kommunikationspartner immer wichtiger. Das gilt insbesondere für virtuelle Lernplattformen, online Meetings oder auch für die Verständigung mit digitalen Assistenten. Aktuell erfolgt die Kommunikation mit virtuellen Gesprächspartnern aber noch nicht intuitiv und sie fühlt sich nicht nach einer Unterhaltung mit einer Person an. Um dies zu verbessern, beschäftigen wir uns in dieser Arbeit mit nonverbaler Kommunikation zwischen Mensch und virtuellem Gesprächspartner. Wir erzeugen ein personenspezifisches Modell, das Körpersprache in Form von Hand und Armgesten passend zum Gesprochenem automatisch synthetisiert.

Wir modellieren den komplizierten Zusammenhang zwischen Sprache und Gestik mittels eines Generative Adversarial Network (GAN). Im GAN implementieren wir einen Diskriminator, dessen Aufgabe darin besteht, den Generator dazu zu bewegen, möglichst menschenähnliche Bewegungen zu generieren. Um unser Modell mit Daten zu versorgen, extrahieren wir die Gesten von unterschiedlichsten Videos mithilfe von 3-D-Erkennung von Personen. Diese Technik erlaubt uns, große Datenmengen zur Verbesserung unseres Modells automatisch zu generieren.

Wir trainieren unser Modell zur Gestenerzeugung mit Videodaten von Universitätsvorlesungen und Fernsehshows. Nachdem wir die Gesten mithilfe unseres GANs generieren, animieren wir sie an einem virtuellen Avatar. Wir evaluieren die Qualität der erzeugten Körpersprache anhand einer Nutzerstudie. In dieser Studie vergleichen wir die generierten Gesten mit den Originalgesten und mit Gesten, die nicht zum Gesprochenen passen.

Die Ergebnisse zeigen, dass die synthetisierten Gesten nicht von den originalen Gesten unterscheidbar sind. In 53 % der Fälle glaubten die Teilnehmer der Studie, dass die synthetisierten Gesten besser zum Gesprochenen passen. Beim Vergleich zwischen Gesten, die zum Gesprochenen passen und unpassenden Gesten, entschieden sich 65 % der Teilnehmer für die Gesten, die mithilfe der dazugehörigen Sprache erzeugt wurden. Somit konnten wir zeigen, dass unser Modell natürliche Gesten für Fernsehmoderatoren und Professoren generieren kann.

Contents

1. Introduction 1

1.1. Use Case . . . 2

1.2. Basic Idea . . . 4

1.3. Structure of the Work . . . 5

2. Background and Related Work 7

2.1. Human Body Language . . . 8

2.1.1. Types of Gestures . . . 8

2.1.2. Nonverbal Communication . . . 9

2.2. Mathematical Notation and Conventions . . . 10

2.3. Machine Learning . . . 12

2.3.1. Discriminative vs Generative Models . . . 12

2.3.2. Supervised vs Unsupervised Learning . . . 12

2.3.3. Classification vs Regression Tasks . . . 13

2.4. Convolutional Neural Network . . . 14

2.4.1. Neural Network . . . 14

2.4.2. Neural Network Architecture . . . 15

2.5. Generative Adversarial Network . . . 18

2.5.1. Generator . . . 18

2.5.2. Discriminator . . . 19

2.5.3. Training . . . 19

2.6. Body Language Synthesizing . . . 19

2.6.1. Motion Capture . . . 20

2.6.2. 3D Reconstruction . . . 20

2.6.3. Speech Gesture Relationship . . . 21

2.6.4. Our Work Compared to the State-Of-The-Art . . . 23

2.7. Encoding Human Body Language and Speech . . . 24

2.7.1. Speech Processing . . . 24

2.7.2. Transferring 2D Video into 3D Virtual Reality . . . 25

2.8. Related Work Summary . . . 32

3. Structural Approach and Development 33

3.1. Conceptual Architecture . . . 33

3.2. Structure . . . 34

3.3. Hardware and Setup . . . 35

3.4. Data Collection . . . 36
3.4.1. TV Show Speakers . . . 36
3.4.2. Academic Speakers . . . 36
3.4.3. Data Preparation . . . 37
3.5. Human Pose Estimation . . . 40
3.5.1. 2D Pose Estimation . . . 40
3.5.2. 3D Body Pose Estimation . . . 41
3.5.3. 3D Hand Pose Estimation . . . 43
3.6. 3D Skeletal Gesture Generation . . . 47
3.6.1. Gesture GAN . . . 47
3.6.2. UNet Generator . . . 48
3.6.3. Motion Discriminator . . . 49
3.6.4. Training the Parameters of our Model . . . 50
3.6.5. Quantitative Evaluation on Human Pose . . . 50
3.7. Virtual Human Animation . . . 53
3.7.1. Skeleton to Virtual Human Animation . . . 53
3.7.2. Designing the Environment . . . 54
3.8. Development Summary . . . 55

4. User Study 57
4.1. Study Design . . . 57
4.1.1. Research Questions . . . 57
4.1.2. Methodology . . . 58
4.2. Setting and Instruments . . . 59
4.2.1. Environmental Setting . . . 60
4.2.2. Instrument Design . . . 60
4.3. Procedure . . . 61
4.4. Study Participants . . . 64
4.5. Results and Discussion . . . 64
4.5.1. Results . . . 64
4.5.2. Discussion . . . 66
4.6. User Study Summary . . . 68

5. Lessons Learned 71
5.1. Background and Related Work . . . 71
5.2. Development . . . 72
5.3. User Study . . . 72

6. Conclusion and Future Work 75
6.1. Conclusion . . . 75
6.2. Future Work . . . 76

Bibliography 79

A. List Of Acronyms **87**

List of Figures

1.1. Sophia Robot 3
1.2. Gesture Generation 4
  a. Professor Jonathan Gruber 4
  b. TV Host John Oliver 4

2.1. The “OK” Sign 8
2.2. The UNet Architecture 17
2.3. GAN Framework 18
2.4. Motion Capture 21
2.5. Mel Spectrogram 25
2.6. Human Pose and Shape Estimation VIBE 26
2.7. Human Pose and Shape Estimation MTC 27
2.8. OpenPose Body Keypoints 28
  a. Keypoints Pose 25 28
  b. Keypoints Pose 18 28
2.9. OpenPose Hand and Face Keypoints 29
  a. Keypoints Hand 29
  b. Keypoints Face 29
2.10. Hand Model Estimation 31

3.1. Gesture Pipeline 35
3.2. Valid vs Invalid Frames 39
  a. Valid Frame 39
  b. Invalid Frame 39
3.3. Detectron 2D Human Pose 41
3.4. 3D Human Pose Estimation 43
  a. Front View 43
  b. Top View 43
3.5. Gaussian Kernel 45
3.6. 3D Hand Pose Estimation Input 45
3.7. 3D Hand Pose Estimation 46
  a. 2D Hand Keypoints 46
  b. 2D Keypoint Heatmap 46
  c. 3D Hand Pose 46
3.8. Gesture GAN 48
3.9. Qualitative Results 52

4.1. Virtual Scene in Unity 59
4.2. Virtual Setup 61
4.3. User Study Example Comparison 63
4.4. Quantitative Results 65

List of Tables

2.1. Mathematical Notation 11
2.2. 3D Human Body Keypoints 30

3.1. Speaker Sequence Detection Constraints 38
3.2. Keypoints of Interest 40
3.3. Quantitative Evaluation of Gestures 51

4.1. Scientific Study Procedure 61
4.2. User Study Results 65

1. Introduction

Nowadays, the interaction between humans and virtual agents is becoming more important because virtual environments gradually become part of our everyday lives. Applications that are already widely adopted include virtual learning platforms, online communication systems, remote diagnosis, virtual assistants and many more. However, an existing problem is that the communication with virtual agents is ineffective. Despite ongoing research, interacting with a virtual human does not feel like talking to a person. In this work, we tackle this problem by investigating non-verbal human communication. We aim for synthesizing non-verbal communication for computer-animated avatars and humanoid robots. We mainly focus on hand and arm gestures as a form of non-verbal communication.

Our objective is to improve the interaction between humans and machines. Whenever people communicate, body language plays an important role. Hand and arm gestures are used to emphasize different parts of the communication. Gestures are also highly correlated to speech. The correlation between speech and gestures ensures the authenticity of a person. The visual picture that is painted using gestures is of great importance because images are what will be remembered by the human brain. Besides gestures, the general posture shows the attitude and feelings of a speaker. Another powerful non-verbal communication channel are facial expressions. They can convey agreement, disbelieve and emotions such as excitement, anger, sadness and happiness.

In management positions and politics, non-verbal communication plays such a great role that people are trained to consciously utilize body language to motivate and convince listeners. This underpins the importance of body language in human communication.

In contrast, it is very difficult to hide non-verbal communication signs, because people are used to expressing themselves to others. Professional poker players are trained to cover their emotions to neither let opponents know about their current feelings nor their hand (Schneider, Hempel, & Lynch, 2013). Another great example that shows the power of body language is the Clever Hans Effect discovered by Pfungst (1911). Clever Hans was a horse that was believed to be able to solve arithmetic problems. The trainer of Clever Hans gave the horse a simple computation task such as three plus five. Then, Clever Hans answered by tapping his hoof eight times. However, instead of performing the computation, what Clever Hans did was he observed the body language of his trainer and other people around. He knew when to stop tapping his hoof by understanding human body language. More interestingly, the trainer of the horse was entirely unaware that he was providing cues. This example not only shows that human non-verbal communication is also understood by animals, it also shows that animals in some cases can even be better in understanding body language.

However, when considering human-machine communication, in most cases non-verbal communication is non-existent. When an employee is doing office work on a personal computer the missing non-verbal communication between user and computer is not an issue because gestures do not provide any benefits in this case. Nevertheless, as machines become smarter, robots and virtual communication are more broadly used which results in non-verbal human-machine communication becoming more important. We show applications of non-verbal human-machine communication in the next section.

One of the problems of state-of-the-art robots and virtual assistants is that, when animated, they do not act human-like. A person recognizes quickly if their communication partner is a human or a machine. This results in robots not being considered a human-alike conversation partner. Another problem related to animation on virtual characters is the Uncanny Valley Effect (Seyama & Nagayama, 2007). This effect demonstrates that whenever a virtual character mostly looks and acts like a real human but in some cases shows unnatural behavior, the switch between reality and unnatural behavior is perceived as very disturbing by humans. The problem is that people get distracted and their attention shifts towards detecting strange behavior instead of listening to the character. This results in less human-like virtual characters performing better at communication than nearly human-like virtual characters.

An example of a robot that imitates human face expression is Sophia (Greshko, 2020) shown in Figure 1.1. Although Sophia and other humanoid robots already show great results when performing certain tasks, there is still room for improvement towards a robot with human non-verbal communication skills. The objective of this work is to generate natural body language in the form of gestures for robots and virtual characters.

1.1. Use Case

The synthesizing of body language is needed in many different domains. The application domains can be grouped into two categories: the animation of virtual characters and robotics.

Examples of virtual character animation exist in the movie and game industry. The animation of characters for movies or video games is mainly produced by the expensive motion capture technique. Actors need to perform the scenes while being recorded by a capturing system such that the virtual characters can be animated. This process takes a lot of effort and is also expensive. It

Figure 1.1.: Sophia Robot. Sophia is a robot with natural human face expressions (Hanson-Robotics, 2020).

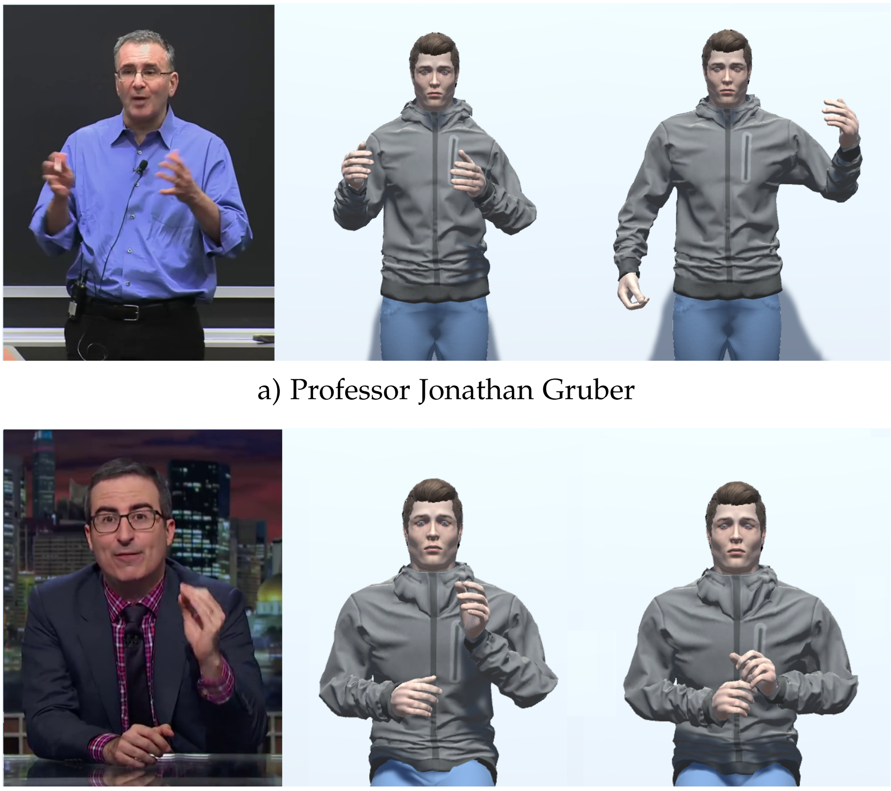
is a time-consuming process that inherits the typical problems of film making. Alternatively, a system that automatically generates gestures from input speech would save time and money. Although it could not be used for specific motion generation, it would be very useful for automating the animation of conversations by utilizing learned person-specific body language.

Another application are online learning environments. A recent trend shows a steep increase in online learning. However, online learning environments are often not well designed, especially for young children. A solution to this problem would be to include virtual teachers which are animated using a gesture generation algorithm. More engaging and intuitive environments allow children to learn effectively. Teachers would have the possibility to create and explain content using their own avatar. This would also embrace a smoother transition between traditional teaching and virtual online learning.

Animated characters can also be used in augmented or virtual reality. Virtual assistants can be animated in a 3D environment such that the user gets a better experience when communicating with them. Furthermore, online voice chats and messaging could be extended with personalized avatar animations.

Besides virtual gesture animations, body language can also be applied to humanoid robots. A use-case includes healthcare, where human-like behavior is important such that patients feel taken care of. Therefore, empathy needs to be expressed authentically by body language.Similarly, service robots at a restaurant should not only do their job but also be able to entertain customers, for example by telling jokes.

More generally, human-machine interaction is getting more relevant in soci-



a) Professor Jonathan Gruber

b) TV Host John Oliver

Figure 1.2.: Gesture Generation. We compare the generated gestures of speaker Jonathan Gruber (top) and John Oliver (bottom). The first column shows the original video frame. In the second and third column we compare the original gesture and the predicted gesture, respectively, animated on a virtual character.

ety. An autonomously driving car needs to be able to convey gestures which otherwise the driver would show to other people. An example would be a hand sign signaling a pedestrian that he can safely cross the street. The latest solutions to this problem involve having displays with smileys (Semcon, 2020) or text at the front of an autonomous car. A survey of different techniques is presented by Rasouli and Tsotsos (2019).

1.2. Basic Idea

We develop an algorithm that is capable of transferring speech to gestures. Once hand and arm gestures are predicted by our algorithm, they are animated in a virtual environment. To learn the relationship between speech and gesture, we implement a data-driven approach. We utilize the large amount of video data to train a model such that it learns the relationship by observing samples of speakers. We develop a speaker-specific model because body language is individual. We capture the individual gesturing style of speakers from different genres. We show examples of two speakers and their original and generated gestures animated on a virtual character in Figure 1.2.

To extract the gesture from our large video dataset we use state-of-the-art human pose estimation algorithms. We adopt them such that they estimate 3D human gestures. We encode the speech using Mel spectrograms and pass them together with gestures to our Generative Adversarial Network (GAN) (Goodfellow et al., 2014) model. After training the parameters of our GAN model, we animate and evaluate the generated gestures on virtual humans.

1.3. Structure of the Work

In this chapter, we introduced our work and showed possible use cases. We also outlined the basic idea which will be explained more in detail throughout the following chapters.

In Chapter 2, we give background information and discuss related work. We present an overview of different approaches towards gesture generation. We compare forms of non-verbal communication and types of gestures. Furthermore, we discuss the basic principles behind data-driven models, including machine learning techniques and neural networks. Once we introduced the fundamentals, we compare past and state-of-the-art methods for gesture generation. After analyzing the advantages and disadvantages of different approaches, we explain our method.

In Chapter 3, we explain our method step-by-step from video selection until gesture animation. We filter suitable video material by estimating the human pose of the speaker for each individual frame. By grouping consecutive valid frames we obtain suitable sequences. We use the sequences to train a GAN. We compute the Mel spectrograms from the speech to pass an encoding in which prosody is detected by the UNet architecture. We extract the gestures in the 3D human pose format from the frames. We train the GAN with the UNet gesture generator and a motion discriminator. Once the gestures are generated we animated them on a virtual avatar.

In Chapter 4, we evaluate the quality of the generated gestures on the virtual avatar by conducting a user study. We recruit over 100 participants and confront them with two different tasks. In the first task, they have to compare our generated gestures against the original gestures. In the second task, they compare gestures which are correlated to speech with uncorrelated gestures. We find that participants cannot distinguish our generated gestures from the original ones. In the second task, the speech correlated gestures are found to be more natural.

Furthermore, we reflect and present lessons we have learned by composing this thesis in Chapter 5. We highlight the most important aspects that have to be kept in mind when conducting similar research.

Finally, we give an outlook on future work in Chapter 6. Ideas include the improvement of the generation of hand gestures and the prediction of full body language including facial expressions. We also summarize the most important findings of this work in Chapter 6.

2. Background and Related Work

We cover four segments in which we give an overview of non-verbal communication generation. First, we analyze different forms of non-verbal communication to encourage a general understanding of the problem domain. Gestures are used for different purposes throughout a speech.

Second, we give an introduction to the basics behind state-of-the-art methods to generate non-verbal communication in the form of gestures. Since most recent algorithms learn the relationship between speech and gestures using previously collected data, we focus on machine learning and neural networks. Especially, in image and speech processing Convolutional Neural Networks (CNNs) are the essential backbone. Since the invention of Generative Neural Networks (GANs), the networks are capable of modeling generative processes. We use these underlying concepts to generate our gestures from speech.

After discussing the basic principles of state-of-the-art work, we compare different approaches towards gesture generation. An accurate, but also expensive method is to use actors in a motion capture or 3D reconstruction environment. In this method, the motion of the actors is tracked and virtual characters are animated accordingly. This technique is used mainly by film and game studios with sufficient resources to fund this expensive method. More interestingly, current research has shown that the relation between speech and gestures can also be learned by a model after training it using recorded data. In previous work, this was achieved using discriminative models that map input utterances to specific output gestures. More recently, generative models predict one of many possible output gestures, given an input speech sequence. The generative approach produces better results because it is more closely related to human communication behavior.

Finally, we address the speech processing and computer vision related problems to data-driven gesture generation. Speech processing ensures that the correct audio features are extracted. It allows us to predict gestures that correspond to speech. Computer vision algorithms extract human pose information from in-the-wild video data of speakers from different genres such as show business and education. By combining techniques from the two disciplines, we can prepare the data and design a model that allows us to better understand and generate realistic gestures for virtual agents.

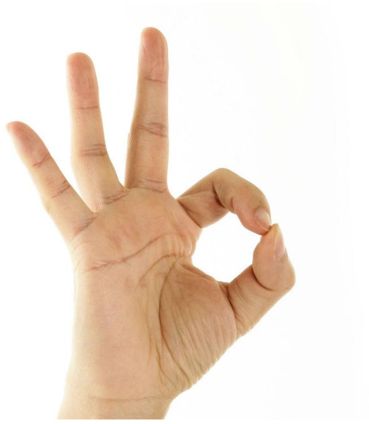


Figure 2.1.: The "OK" Sign. The "OK" sign is globally known but can have different meanings depending on the culture (Getty Images).

2.1. Human Body Language

To better understand human body language and gestures, we have to inspect different forms of it. The most structured type of gestures is sign language. Analogous to spoken language, depending on the region of the world different sign languages are "spoken". Sign languages are mainly used and understood by people who are deaf or dumb. However, a few signs are recognized by most people. One example is the "OK" sign show in Figure 2.1. It can have different meanings depending on the culture. Although there exist successful attempts to recognize sign language (T. Starner & Pentland, 1997; T. Starner, Weaver, & Pentland, 1998), we do not focus on this type of communication in this work.

2.1.1. Types of Gestures

Instead, we focus on gestures that appear together with spoken language. The role of these gestures is to support the understanding and communication. Gestures that underpin spoken language are grouped into four different categories (Huang & Mutlu, 2013):

- Iconic Gestures
- Deictic Or Pointing Gestures
- Metaphoric Gestures
- Beat Gestures

Iconic Gestures Iconic gestures are used when describing the size of objects, spatial relationships and specific actions. When a speaker talks about a past situation in which somebody was standing right next to him, he could use an iconic gesture to indicate at which position the other person was located relative to his position. Another example would be a speaker talking about a tall building. The typical corresponding iconic gesture would be to raise one arm above the head.

Deictic or Pointing Gestures Deictic gestures which are also referred to as pointing gestures are used to direct the visual attention of the listener to a specific object. When presenting in front of a screen or blackboard, pointing gestures are used to highlight a specific location on the surface. Another example would be a person pointing to a clock when mentioning the current time. Typically, pointing gestures are used by teachers in a school to select a student who should answer a question asked.

Metaphoric Gestures Metaphoric gestures aim to represent abstract concepts. They are sometimes related to sign language. An example includes putting the fingers into the shape of a heart to represent the concept of love. Another example is stretching the pointer and middle finger to represent a "V" shape signaling victory or win.

Beat Gestures Beat gestures are rhythmic gestures that do not transmit any semantic content. However, they are very important as they are used to highlight certain parts of a speech. An example would be a speaker hitting the podium repetitively. Furthermore, beat gestures have to power to transmit feelings such as anger and excitement.

The main focus in our work is on beat gestures, as they represent the largest portion of gestures during a speech. Compared to the other three types, beat gestures do not transmit semantic content which means that no semantic understanding of the speech is necessary to generate them. Instead, audible features such as emphasis and the pitch of the voice are of greater importance.

2.1.2. Nonverbal Communication

According to Schmitz (2012), gestures are only a subgroup of several nonverbal communication channels. Gestures are a small but important part of nonverbal communication. They distinguish between the following nonverbal communication categories:

- Kinesics
- Haptics
- Vocalics
- Proxemics
- Chronemics
- Personal presentation and environment

Kinesics The category Kinesics includes the movement of different body parts during communication. The most important subcategories are hand and arm movements which are referred to as gestures. Besides gestures, head movements, facial expressions and posture are also important nonverbal communication channels. Kinesics also includes eye contact with other people.

Haptics Haptics is another nonverbal communication category. It includes all scenarios in which people touch each other. The most important touching interaction is the handshake. It can be used to transfer different emotions.

Vocalics The category Vocalics deals with vocalized but not verbal aspects. These aspects include the volume, pitch and emphasis of speech. Professional speakers are able to use these aspects to highlight different parts of their speech to convey their message more effectively.

Proxemics Proxemics focuses on space and the distance between parties during a communication. Different communication distances are grouped into zones reaching from the public zone, where parties are far away from each other, to the intimate zone, in which distance is small.

Chronemics Chronemics refers to the temporal aspect of nonverbal communication. On the one hand, a party can communicate very fast such that it seems that they are stressed out. On the other hand, a more relaxed style can convey calmness.

Personal Presentation and Environment The environment and surroundings also affect communication behavior. Examples include the seating positions and the room layout in general. On some occasions, informal meetings or speeches are also held outdoors to create a more relaxed atmosphere.

In this work, we mainly focus on the Kinesics of nonverbal communication. Besides the animation of body movement and gestures, we also set up a simple indoor presentation environment. We have no possibility to include Haptics because we animate a virtual avatar. In our case, the communication parties are physical and virtual. We do not influence Vocalics and Chronemics, but we analyze them when generating speech-related gestures.

2.2. Mathematical Notation and Conventions

We define concepts using mathematical expressions. In order to support readability and understanding of these expressions, we maintain consistent notation.

Entity	Notation
Scalar	x, x_i, M
Vector space N	$\mathbb{R}^N$
Vector in $\mathbb{R}^N$	$\mathbf{x} = (x_1, \dots, x_N)^{\mathrm{T}}$
Matrix space $M \times N$	$\mathbb{R}^{M \times N}$
Matrix of dimension $\mathbb{R}^{M \times N}$	$\mathbf{M} = \begin{bmatrix} m_{1,1} & \dots & m_{1,N} \\ \vdots & \ddots & \vdots \\ m_{M,1} & \dots & m_{M,N} \end{bmatrix}$
ℓ_p norm of vector	$\|\|\mathbf{w}\|\|_p$
Cardinality of set $\mathbb{S}$	$\|\mathbb{S}\|$
Operation	**Notation**
Matrix concatenation	$\|$
Specific Naming	**Notation**
Discrete probability distribution on random variable X	$p(X), q(X)$
Conditional probability of Y given X	$p(Y\|X)$
Joint probability of X and Y	$p(X, Y)$
Loss function	$\mathcal{L}$
Learning rate	η
Hyper parameters	λ
Parameter of a model	Θ

Table 2.1.: Mathematical Notation. We define the mathematical notation which is used consistently throughout this thesis.

A summary of this notation is illustrated in Table 2.1. Whenever we use scalar values, we use the italic font, *e.g.* a, B. Vectors are written in bold using lower case, *e.g.* $\mathbf{c}$. Matrices and tensors of higher dimension are also written in bold, but use caps. An example matrix would be $\mathbf{D}$. Whenever we define a set of numbers, we use blackboard bold typeface. Hence, the set of real numbers is defined as $\mathbb{R}$. We introduce these conventions to ensure good readability of our work.

2.3. Machine Learning

We apply the knowledge from communication theory to generate nonverbal communication in the form of gestures. The relation between speech and gestures is not bijective. Instead, the generation of gestures is very complex. We use machine learning to approach the task.

By using machine learning techniques we can implement a data-driven approach. We make use of the large amount of video data publicly available and train a model that learns the relation between speech and gestures. Machine learning techniques recognize patterns and relationships in data. The disciplines of machine learning can be distinguished by the following properties:

2.3.1. Discriminative vs Generative Models

One possibility to classify machine learning algorithms is by separating them into discriminative and generative models (Jebara & Meila, 2006). Discriminative models learn to predict an output y given an input x. From a probability theoretic standpoint, it models the conditional probability $p(Y|X)$. For example, discriminative models in image classification output a probability value for the output class dog, given an input image $p(Y = \text{Dog}|X)$. If there exists a dog in the input image, the model is trained to output a value close to one, $p(Y = \text{Dog}|X = \text{Image with Dog}) \rightarrow 1$, and zero otherwise.

Generative models, in contrast, are a statistical model of the joint probability distribution $p(X, Y)$. In the previous example, this would mean that the probability of a specific input image x is also taken into account when predicting the output y.

2.3.2. Supervised vs Unsupervised Learning

Machine learning algorithms are also grouped into supervised and unsupervised learning models. In supervised learning, the models are trained to predict an output $\hat{y}$ which should be close to a humanly annotated target output y. For example, in object detection, a task would be to identify a car inside an image and to draw a bounding box around the car. To train the algorithm, example images are shown to the network. These example images contain a humanly annotated ground truth, where the bounding boxes of the cars are drawn by humans. Example methods include logistic regression, support vector machines and artificial neural networks.

In unsupervised learning, the target is not given by human annotators. Instead, the objective is to identify structures in data. An application would be an algorithm that learns to predict future video frames given the current and past frames. Typical unsupervised learning algorithms include k-means clustering and the principal component analysis.

There exists another group of algorithms which is a mixture of supervised and unsupervised learning called semi-supervised learning. In semi-supervised learning, a small number of input data samples x contain a target label y. In between, the model performs unsupervised learning.

Another method of how machine learning algorithms are trained is reinforcement learning. It is also referred to as try and error based learning. In reinforcement learning, a reward is assigned to the algorithm whenever the desired output is created. The algorithm learns to find a strategy by maximizing the reward.

2.3.3. Classification vs Regression Tasks

Another method to distinguish learning algorithms is by the task the algorithm is learning. The two different tasks are classification and regression. A classification algorithm predicts a discrete output label drawn from a finite set of possible output labels. For a pet image classification task, the output labels could be $Y = \{\text{dog}, \text{cat}, \text{bird}\}$. The task of the classification algorithm in this case is to predict one of the finite labels $\hat{y} \in Y$.

In contrast, regression tasks predict a continuous output value. The predicted continuous output $\hat{y}$ could be a value that does not exist in the training data y. An example of a regression task would be to estimate housing prices given the location and the size of a property.

We include a discriminative and generative component in our gesture generation model. The discriminative model learns to predict gestures given input speech. The prediction of gestures is a regression task. The position of the different body parts is predicted using continuous coordinate values. To train such a discriminative model, a supervised approach is used where the target labels are extracted from the input video.

The generative component within our gesture generation model focuses on the motion of gestures. It is trained using unsupervised learning comparing real motion and generated motion. The generative component performs a binary classification task deciding whether the motion is real or generated.

This two-component approach unites the strengths of several machine learning techniques. Such an algorithm can be implemented utilizing the artificial neural network framework.

2.4. Convolutional Neural Network

The design of artificial neural networks (ANNs) is inspired by the human brain. According to neuroscientific estimations (Herculano-Houzel, 2009), the human brain consists of about 100 billion neurons and thousands of synopses for each neuron. Such high complexity cannot be modeled by ANNs. Especially, the high number of connections between neurons is currently completely infeasible to model using ANNs. The type of ANNs which are most closely related to the biological brain are Spiking Neural Networks introduced by Maass (1997).

In 2012, AlexNet developed by Krizhevsky, Sutskever, and Hinton (2012) was the first neural network (NN) to surpass the performance of traditional computer vision algorithms in the ImageNet challenge (Russakovsky et al., 2015). From that moment on, NNs gained popularity and are used in many different disciplines such as natural language understanding, translation, image understanding and image generation.

2.4.1. Neural Network

The basic component of an artificial neural network, referred to as neural network in this work, is the neuron. A neuron takes input $x_1, ..., x_n \in \mathbb{R}$ where n equals the number of input connections to this neuron. Then, each input is multiplied with the individual parameters $\theta_1, ..., \theta_n \in \mathbb{R}$. Finally, the sum of the individual values and the bias b are passed to an activation function $\phi(\cdot)$ to produce the scalar output y. The computation of a neuron is given by

$$y = \phi(x_1 \cdot \theta_1 + x_2 \cdot \theta_2 + ... + x_n \cdot \theta_n + b). \tag{2.1}$$

The activation function decides whether the neuron gets activated or not. In the most simple case, the activation function is the threshold function with threshold τ,

$$\phi_{\text{Threshold}}(z) = \begin{cases} 0 & \text{for } z < \tau \\ 1 & \text{for } z \geq \tau \end{cases} \tag{2.2}$$

where z represents the pre-activated output of the neuron. In this case the neuron gets activated if a certain threshold is reached. Otherwise, the output is zero.

In practice, more complex activation functions are used such as the Rectified Linear Unit (ReLU) by Hahnloser, Sarpeshkar, Mahowald, and Douglas (2000) and the Sigmoid function. An activation function has to be differentiable in order to train the parameters of the network.

Neural Network Structure A neural network is built by combining many neurons. Within a neural network, neurons are grouped into layers, where each layer consists of a certain amount of neurons. The input layer is the first layer of neurons which processes the raw input to the network. The output layer is the last layer. The activated values of the output layer represent the result of the network computation. In between the input and output layer, many hidden layers exist within the network. The hidden layers are needed to recognize and understand patterns in complex data. If the number of hidden layers is high, which means about larger than ten, the neural network is also referred to as "Deep Neural Network" and the process of training the parameters is called "Deep Learning". To visualize the architecture of neural networks, a directed graph is used in which the nodes represent the neurons and the directed edges represent the information flow between neurons from one layer to the next. The graph is structured layer by layer in a grid layout.

Learning The most important part is training the parameters Θ of a neural network model. During training, the model is provided with input which is passed through the network to predict an output. The predicted output is then compared to the desired output and the deviation is measured by a loss function. The error between predicted and desired output is used to improve the value of the parameters Θ to prevent this error from happening in future predictions. The total error is backpropagated to each individual parameter θ in the neural network model using differential calculus. For each layer in the neural network architecture, the chain rule needs to be applied to determine the update for the parameters at that layer. The backpropagation computation starts at the output layer and ends at the input layer after applying several iterations of the chain rule.

2.4.2. Neural Network Architecture

The basic neural network architecture is not efficient for speech to gesture transformation. In our data-driven approach towards gesture prediction, we process video data. This means that there is a high amount of audio and visual input data. Consequently, a fully connected neural network introduces too much complexity.

Convolutional Neural Networks In audio and image processing, Convolutional Neural Networks (CNNs) are a more efficient architecture choice (Krizhevsky et al., 2012; Long, Shelhamer, & Darrell, 2015; Mueller et al., 2018). In a CNN, a neuron does not have a connection to the whole input. Instead, neurons are only connected to a small fraction of input nodes. However, the neurons are moved over the whole input similar to a sliding window. In 2D image processing, a CNN filters local 2D neighborhoods for image features. A commonly used convolution filter size is 3 × 3. A CNN layer which is located close to the input filters for low-level image features such as edges and texture. CNN layers close to the output filter for high-level image features such as complex shapes or faces.

Receptive Field The receptive field of CNN neurons close to the input is very low. Input layer neurons only compute their activation value based on very local information in the input data. In contrast, neurons located in deeper layers of the network see a more global context. This is due to the hierarchical structure of the network. For example, if the first layer of a CNN contains neurons with filter size 3 × 3, the receptive field of this layer also equals 3 × 3. If the second layer again contains a 3 × 3 filter, then the receptive field of the second layer is already 5 × 5. If the third layer has the same filter size, the receptive field equals 7 × 7. This shows that the size of the receptive field for neurons increases with depth even though the filter size stays constant.

In speech processing, the shallow network layers detect changes in pitch and distinguish different sounds. The deeper layers filter for more complex concepts such as words. In gesture recognition, CNNs filter for edges and textures in layers close to the input. Deeper layers detect, for example, the hand or the face of a person.

UNet Architecture The UNet architecture developed by Ronneberger, Fischer, and Brox (2015) is a CNN encoder-decoder architecture. We show a diagram of the UNet architecture in Figure 2.2. It is also referred to as hourglass architecture, because of its encoder-decoder design. Inside the encoder, the spatial dimensions decrease gradually layer after layer. At the same time, the channel size which refers to the number of neurons of a layer, increases. At the bottleneck between encoder and decoder, a compact feature representation is created. The compact representation encodes the most important input information for a given task, such as object detection or human pose information. The decoder then performs the inverse operation by means of transposed convolutions. It increases the spatial dimensions while decreasing the number of channels. The result is a dense prediction that can be used for semantic segmentation as suggested in Figure 2.2.

In our work, we use neural networks to perform gesture generation. CNNs are the state-of-the-art method for human pose extraction. They can be used to detect humans in an image and determine the location of different body parts. We extract the human pose from video input data to track the motion and the gestures of a speaker. Additionally, we use the UNet architecture to transfer speech into gesture information. In this scenario, the encoder received audio input and creates a compact representation in the UNet bottleneck.

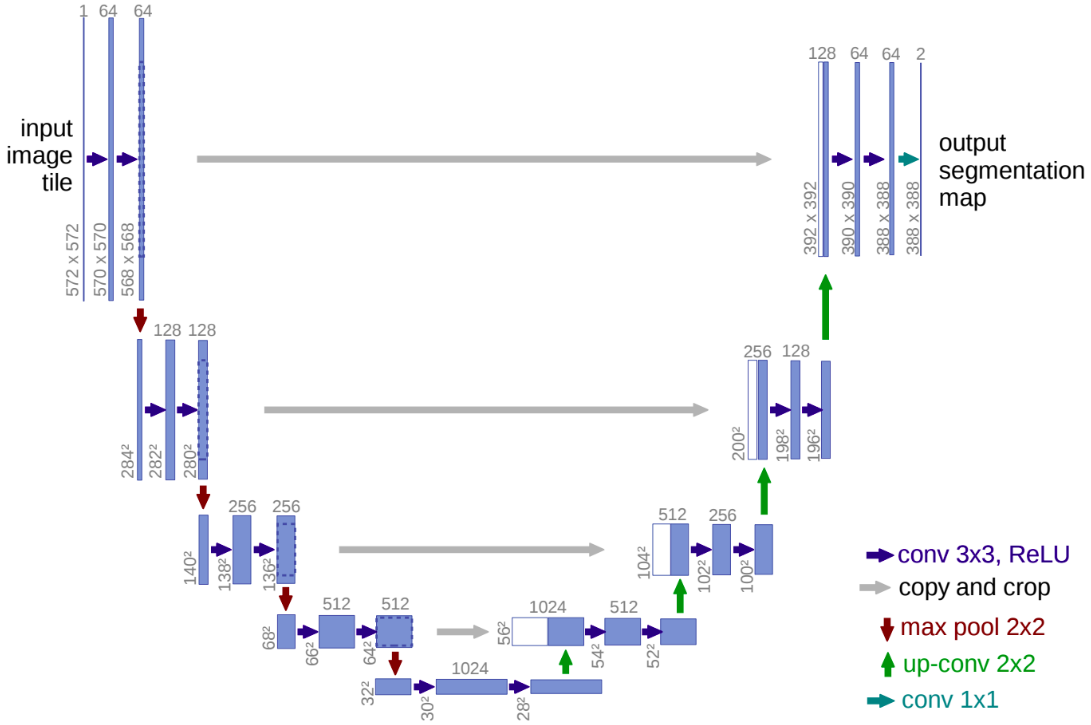


Figure 2.2.: The UNet Architecture. The CNN architecture was developed by Ronneberger, Fischer, and Brox (2015). It consists of an encoder (left) and a decoder (right). The UNet was originally developed for image segmentation, but the architecture is also applied to different domains. It consists of convolution layers with a 3×3 filter followed by a ReLU activation function indicated by the blue arrow pointing to the right. In the encoder, 2×2 max pooling layers marked with the red arrows pointing downwards reduce the spatial dimensions by half. Simultaneously, the number of channels is increased by a factor of two. The gray arrows depict the skip connections from the encoder to the decoder. In the decoder, the green upward arrows indicate transposed convolutions with a 2×2 filter. The last convolution with a 1×1 filter performs learned feature compression and is represented using a turquoise arrow pointing to the right.

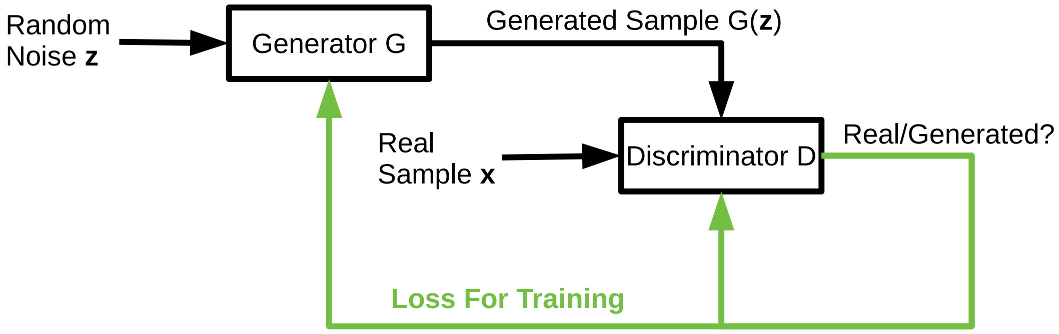


Figure 2.3.: GAN Framework. The two components of the GAN framework are the generator G and the discriminator D. The generator creates a data sample given input noise $\mathbf{z}$. The task of the discriminator is to decide whether a given sample is real $\mathbf{x}$ or generated $G(\mathbf{z})$. The GAN loss is backpropagated to generator and discriminator to compute their parameter updates. Therefore, the performance of both improves during training.

Subsequently, the decoder takes the compressed speech representation and predicts gestures in the human pose format.

2.5. Generative Adversarial Network

Generative Adversarial Networks (GANs) are an innovative neural network design introduced by Goodfellow et al. (2014). The GAN framework allows the development of generative models using neural networks. They are applied in numerous tasks as shown by the work of Wang et al. (2018), Isola, Zhu, Zhou, and Efros (2017) and Karras, Aila, Laine, and Lehtinen (2017).

The GAN consists of two main components, the generator $G(\cdot)$ and the discriminator $D(\cdot)$. Both of them have individual objectives but are trained together. We illustrate the GAN framework in Figure 2.3.

2.5.1. Generator

The objective of the generator is to generate data samples from a given domain which are as realistic as real data samples $\mathbf{x}$ from that domain. For example, the objective could be to generate natural images of dogs. In this example, the data set $\mathbf{x}$ consists of real dog images and the objective of the generator is to create similar images. The input for the generator is a random vector $\mathbf{z}$ in latent space. Latent space vectors contain compressed information of some input domain. The generator uses a decoder network to decode the latent vector and generate a data sample $G(\mathbf{z})$. However, the generator does not evaluate how close this generated sample $G(\mathbf{z})$ is, compared to the real samples $\mathbf{x}$. Therefore, the discriminator is invented.

2.5.2. Discriminator

The objective of the discriminator is to decide whether a given data sample is drawn from the real data set $\mathbf{x}$ or if it was generated by the generator $G(\mathbf{z})$. In the example mentioned above, the task would be to decide whether the dog image was taken by a camera or if it was generated. The architecture of the discriminator is similar to an encoder architecture. Instead of a latent vector, the discriminator only outputs a single value. This value represents the probability that the input was drawn from the real distribution. Consequently, it is close to one if the input to the discriminator was drawn from the real distribution and close to zero if the input was passed from the generator.

2.5.3. Training

The objective of the discriminator is described as

$$\max_D \Big(\mathbb{E}_{(\mathbf{x})}[\log D(\mathbf{x})] + \mathbb{E}_{(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))] \Big) \ . \tag{2.3}$$

The first term ensures that the output for real samples is close to one. Likewise, the second term forces the probability of fakes to be close to zero.

The objective of the generator becomes to fool the discriminator by generating realistic output. Therefore, the objective function is given by

$$\min_G \Big(\mathbb{E}_{(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))] \Big) \ . \tag{2.4}$$

A minimax two-player games is created, because the second term of the discriminator equals the objective of the generator, but in the opposite direction. The final GAN objective function is defined as

$$\min_G \max_D \ \mathrm{GAN}(G, D) = \mathbb{E}_{(\mathbf{x})}[\log D(\mathbf{x})] + \mathbb{E}_{(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))] \ . \tag{2.5}$$

We implement the GAN framework to generate gesture sequences. Our generator predicts gesture sequences and the discriminator evaluates if the predicted motion of the gestures is realistic. The generator takes the latent vector space of encoded speech as an input and implements the UNet to generate gesture sequences in human pose format. The discriminator decides if the motion of the generated gesture sequence is realistic by comparing it to the motion of gestures from real video data.

2.6. Body Language Synthesizing

After presenting the basic concepts behind data-driven body language synthesis, we want to examine various approaches towards body animation more in detail.

We outline the advantages and disadvantages of synthesizing techniques and algorithms.

2.6.1. Motion Capture

One approach towards body language animation is motion capture. Motion capture algorithms track the movement of the human body. An overview of different motion capture algorithms is given by Moeslund and Granum (2001) and Moeslund, Hilton, and Krüger (2006). The people inside a motion capture system wear markers on different body parts such that they are tracked. Depending on the application, the markers are located on the whole body or on special regions, such as the face to capture facial expressions. The markers are either active or passive. Passive markers reflect the light which is generated by a light source close to the capturing camera. Active markers emit their own light. In both techniques, cameras determine the position of the marker by filtering for the tracking color. To generate 3D information, a camera system of at least two cameras is needed. Due to occlusions and accuracy improvements, more cameras are used (Aurand, Dufour, & Marras, 2017). Once the cameras are calibrated and the 3D positions of the markers are tracked, a human skeleton is animated (Kirk, O'Brien, & Forsyth, 2005). An example is presented in Figure 2.4.

Motion capture is widely used in the movie and game industry because it is an accurate method to animate virtual characters. However, the technique is also expensive and can therefore only be used by studios for large projects. The room and camera setup is costly. Additionally, actors which perform the scenes that are animated need to be hired.

2.6.2. 3D Reconstruction

A very similar technique to motion capture is 3D reconstruction from multiple images. Instead of tracking markers on a human body, the objective is to compute the 3D surface of the whole scene. This increases the computational complexity because correspondences are computed for every pixel in the image. The number of cameras needed depends on the application and the method used. Geiger, Ziegler, and Stiller (2011) perform 3D reconstruction of humans using a stereo camera setup. This only allows for 3D reconstruction from a certain perspective. Izadi et al. (2011) use a moving depth camera to reconstruct 3D scenes. The matching of images from a moving camera produces good results on a static scene but is not suitable for dynamic scenes. This also applies to algorithms that utilize RGB videos to reconstruct 3D scenes as shown by Pollefeys et al. (2008).

Similar to motion capture, 3D reconstruction also requires a camera setup and actors to animate virtual characters. Compared to motion capture, it is

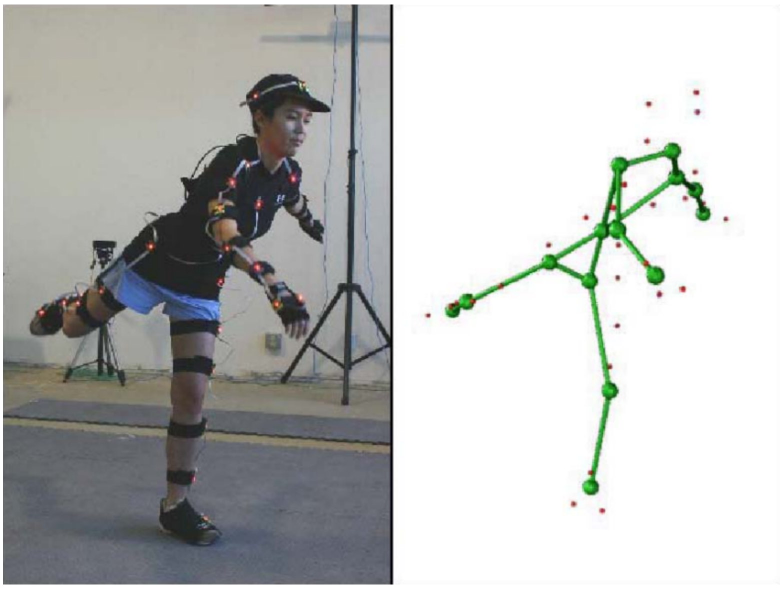


Figure 2.4.: Motion Capture. Motion capture using active markers (left) and the corresponding human pose estimation (right) by Kirk, O'Brien, and Forsyth (2005).

more suitable for static human body shape capturing as shown by Remondino (2004).

2.6.3. Speech Gesture Relationship

The drawback of capturing and reconstruction techniques when used for animation is that an actor is required for every scene that needs to be recorded. To overcome this problem, data-driven approaches are used to model the relationship between speech and gestures. Once a model has learned about speech and gestures by observing communications, it is used to predict plausible gestures from speech.

Semantic Meaning Bergmann and Kopp (2009) model the generation of iconic gestures using a Bayesian network. Iconic gestures are speaker-specific. To generate iconic gestures, a transcript has to be extracted from the input speech. Similarly, Yan (2000) also generates a framework that maps semantic to gestures. The disadvantage of only considering the semantic meaning when predicting gestures is that mainly iconic and metaphoric gestures can be predicted which only account for a small portion of gestures during a speech.

Prosody of Speech Chiu and Marsella (2011) generate gestures based on the prosody of speech rather than utterances. They consider the prosodic features pitch and intensity as well as correlation audio features. The advantage of this approach is that a model trained on prosody also generates gestures for new utterances. Chiu and Marsella (2011) show that their generated gestures are perceived similarly to real human gestures when evaluated with a human study. Furthermore, the generated gestures are judged as substantially better when compared to human gestures from uncorrelated utterances.

Instead of modeling a direct relationship between speech and gesture, Chiu and Marsella (2014) introduce an additional layer that contains gesture embeddings. This intermediate representation is interpolated within a sequence such that coherent motion is created. In the first step, gesture positions are created from audio. In the second step, the gesture positions are interpolated to create realistic motion.

Variable Sequence Length More recent approaches also utilize the prosody of speech to generate realistic gestures. Hasegawa, Kaneko, Shirakawa, Sakuta, and Sumi (2018) extract features from speech using Mel-Frequency Cepstral Coefficients (MFCC) which holds the prosodic features. A recurrent neural network (RNN) allows processing sequences of speech in variable lengths. More specifically, they implemented the bidirectional Long Short-Term Memory (LSTM) architecture introduced by Hochreiter and Schmidhuber (1997), which allows them to process the speech forward and backward when predicting the gestures in 3D human pose format. They also apply a smoothing layer as a post-processing step to enforce temporal coherence. A drawback of the work by Hasegawa et al. (2018) is, that they only infer a dataset of a single speaker which is recorded using a motion capture system. Since gestures are speaker-specific, their model is not broadly applicable.

Similarly, Ferstl and McDonnell (2018) also implement an RNN for gesture prediction. They insert the Gated Recurrent Unit (GRU) cell developed by Cho et al. (2014) between the encoder and the decoder. The encoder transforms the speech audio input into latent space. Once the GRU has added temporal information, the decoder computes the gesture predictions. Ferstl and McDonnell (2018) found that the Mel-scaled spectrogram produces better results than the MFCC encoding used by Hasegawa et al. (2018). Following up on the results of the work, we decided to also use Mel spectrograms to extract the prosody of speech in our work. Ferstl and McDonnell (2018) also created their own four hour single speaker data set for their experiments. However, we want to utilize existing video data from different speakers. An interesting finding of Ferstl and McDonnell (2018) is, that there is no benefit of using transfer learning from a pretrained motion model. Therefore, we use a GAN to enforce realistic motion in our approach.

Fixed Sequence Length Kucherenko, Hasegawa, Henter, Kaneko, and Kjellström (2019) present a encoder-decoder approach towards 3D gesture generation from speech. An autoencoder neural network is trained to find a compressed motion representation of the gestures in human pose format. Then, the decoder is used in combination with a speech encoder to predict gestures from speech. Compared to methods using an RNN architecture, fixed-length sequences are processed. Similar to previous work, they only created a speaker model from the Japanese conversation data set by Takeuchi, Kubota, Suzuki, Hasegawa, and Sakuta (2017). Also, only a causal relationship between speech and gestures is modeled which does not comply with communication theory.

Generative Gestures Most recent approaches focus on the non-deterministic relationship between speech and gestures. To overcome the problem of predicting the mean gesture, Alexanderson, Henter, Kucherenko, and Beskow (2020) introduce a probabilistic approach. They adapt the MoGlow framework (Henter, Alexanderson, & Beskow, 2019) which implements LSTM cells to generate gestures with realistic motion. Ferstl, Neff, and McDonnell (2019) use a GAN model in their non-deterministic approach. They train the adversary to generate realistic sub-features such as plausible dynamics and smooth motion. The annotated dataset used consists of 3.75 hours of gestures. In contrast, the work of Ginosar et al. (2019) implements a single adversary which ensures realistic motion. Compared to the other approaches, they use in-the-wild YouTube video data to train their speaker models. It allows them to generate models for speakers from different genres, *i.e.* show business, education and religion. Furthermore, their dataset is much larger than previous datasets. For some speakers, they process more than 20 hours of data. This allows the models to learn from a variety of different input gestures. One disadvantage of the work of Ginosar et al. (2019) compared to others is that they process 2D input and their model therefore predicts 2D gestures. Although they evaluate their results comprehensively, the human study is only conducted on gestures in the skeletal human pose format.

2.6.4. Our Work Compared to the State-Of-The-Art

In our work, we use many of the concepts presented by Ginosar et al. (2019). We also implement a generative model, because of the nature of gestures. We also design a GAN with a motion discriminator. In contrast to the 2D output of Ginosar et al. (2019), our model is trained to generate 3D gestures. Moreover, one drawback of the evaluations by Ginosar et al. (2019), Ferstl et al. (2019) and Alexanderson et al. (2020) is that they are performed on a human pose skeleton. We evaluate the gestures generated by our model on a virtual human. In contrast to previous work, we animate the predicted human pose on a virtual human.

Our user study consists of two different tasks as proposed by Hasegawa et al. (2018). In the first task, we compare our generated gestures against original gestures. In the second task, we compare them against gestures generated from different utterances.

We create models for speakers from different genres as suggested by Ginosar et al. (2019). In order to have enough data we also need to utilize recorded online video data such as videos on YouTube. One of our challenges is to extract 3D gestures from 2D video data.

Besides the gesture format, speech audio encoding also varies across different work. We find a very comprehensive comparison of the Mel spectrogram and MFCC encoding in the work of Ferstl and McDonnell (2018). Following the advice of previous research, we encode our speech input using Mel spectrograms.

2.7. Encoding Human Body Language and Speech

It is important to implement an efficient representation of gestures when synthesizing human body language. One option would be to use the complete raw video data similar to Deepfake approaches (Nguyen, Nguyen, Nguyen, Nguyen, & Nahavandi, 2019). However, the problem with this representation is that it not efficient. Video data contains a lot of unnecessary information such as the background or the clothes a speaker is wearing. This information does not only increase the data size, it also allows the model to learn irrelevant information for gesture generation. Additionally, Deepfake approaches do not always generate realistic images (Rössler et al., 2019) which is a disturbing factor in human-agent communication.

More efficient gesture representations include the human pose skeleton or 3D human shape. Previous work used 2D and 3D human pose information as shown in Section 2.6. The advantage of 3D gesture prediction is that it can be animated in virtual reality and on humanoid robots. Likewise, the audio also needs to be processed such that background noise is removed and the prosodic features of the speech are encoded efficiently.

2.7.1. Speech Processing

As found by Ferstl and McDonnell (2018), Mel spectrograms are an efficient encoding for speech to gesture modeling. To generate a spectrogram, the raw audio waveform is transferred into the frequency-amplitude domain using the fast Fourier transform. By obtaining multiple Fourier transforms on different windowed segments of the audio input signal, a spectrogram is generated. The frequency axis of the spectrogram is then transferred into the Mel scale. The Mel scale ensures that the difference between signals at the low and high frequency spectrum is encoded as perceived by humans. An example of a Mel spectrogram is shown in Figure 2.5.

By analyzing the Mel spectrogram it is possible to identify phonemes. It can be directly observed how the voice of the speaker changes over time. The Mel spectrogram visualizes changes in prosody such as volume and pitch. This

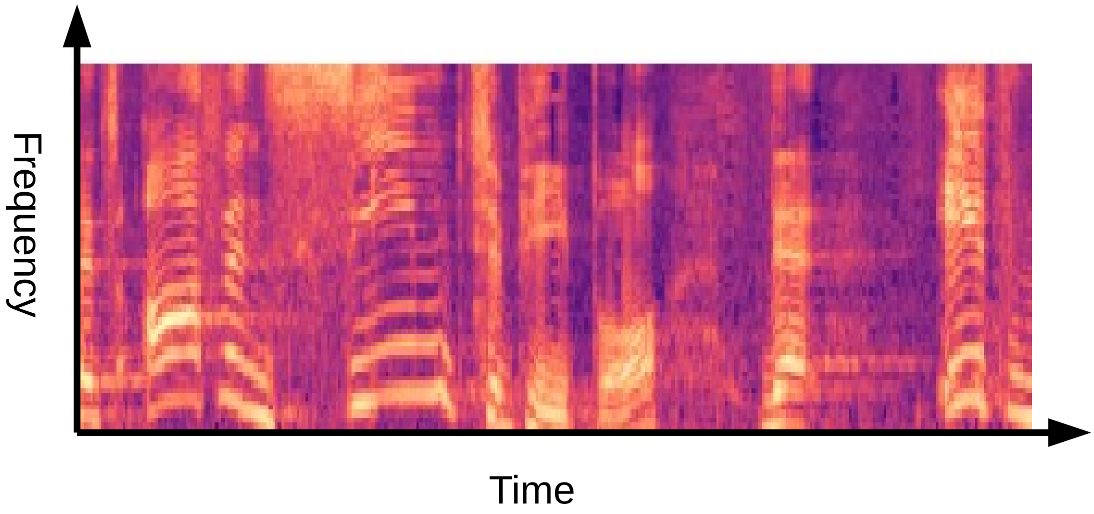


Figure 2.5.: Mel Spectrogram. In the Mel spectrogram, the audio is visualized on a time-frequency graph. The color refers the amplitude in decibels. Dark color represents low amplitude and orange and yellow areas represent high amplitude.

encoding allows the use of image processing techniques to filter for prosodic features.

Additional features can be extracted from the input audio using speech recognition algorithms such as Povey et al. (2011) and Graves, Mohamed, and Hinton (2013). Such features allow the model to also consider the semantic meaning of utterances. This is used when modeling iconic and metaphoric gestures as shown by Bergmann and Kopp (2009) and Yan (2000).

2.7.2. Transferring 2D Video into 3D Virtual Reality

The advantage of a data-driven approach is that a large amount of 2D speaker video data can be accessed using YouTube and other online platforms. However, 3D video data is scarce and not available for education and TV show speakers. Thus, we need to detect the speaker in the 2D videos and estimate a 3D human model.

Human Model A 3D human pose and shape estimation from a 2D video is presented by Kocabas, Athanasiou, and Black (2020). They introduce adversarial training to generate plausible motion data despite the lack of 3D ground truth. We show an example frame of the 3D estimation in Figure 2.6.

When tested empirically on our speaker dataset, we find the results to be temporally incoherent. Additionally, the hands of the speaker are not estimated accurately. The accuracy is not sufficient such that typical gestures of speakers cannot be identified after applying the model. One of the problems is that the

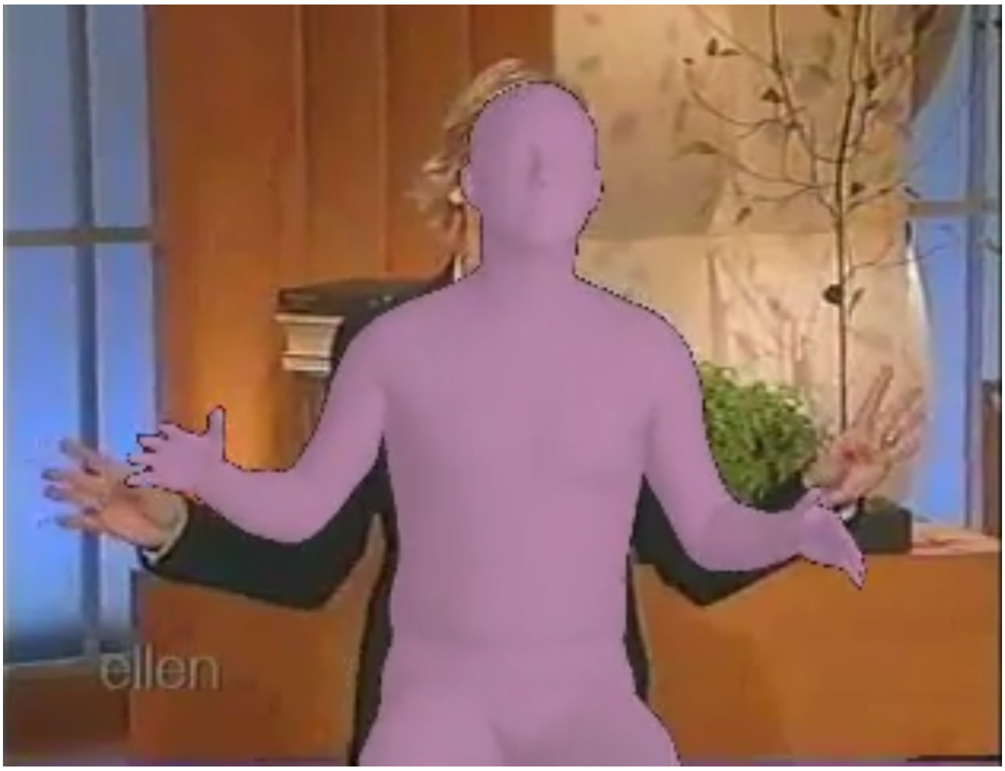


Figure 2.6.: Human Pose and Shape Estimation VIBE. The human body pose and shape estimation by Kocabas, Athanasiou, and Black (2020) is represented by the purple shape. The algorithm shows inaccuracies when predicting the hand pose of speaker Ellen DeGeneres.

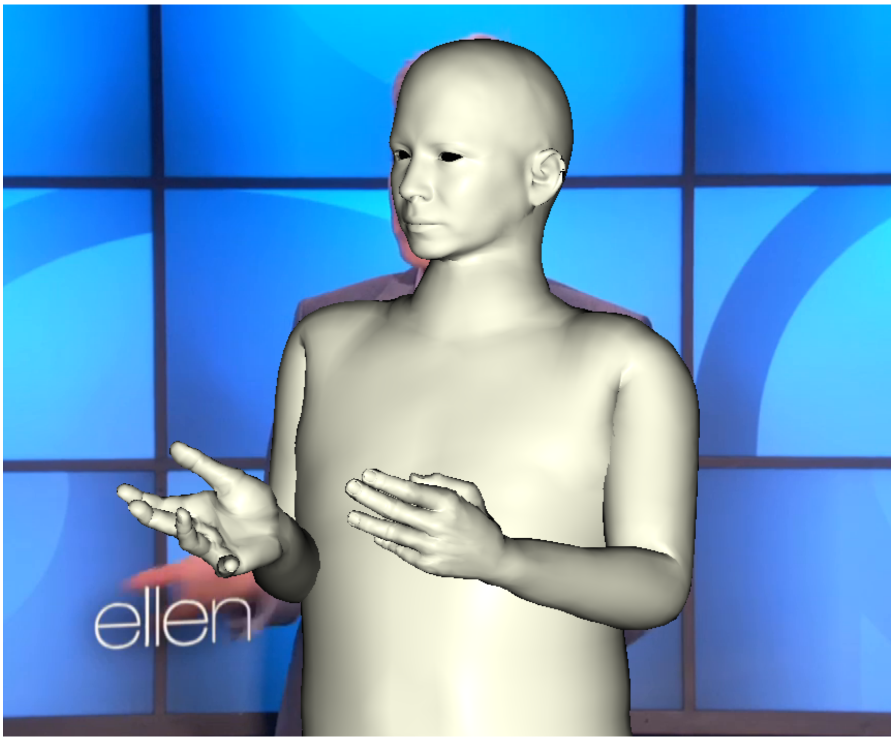


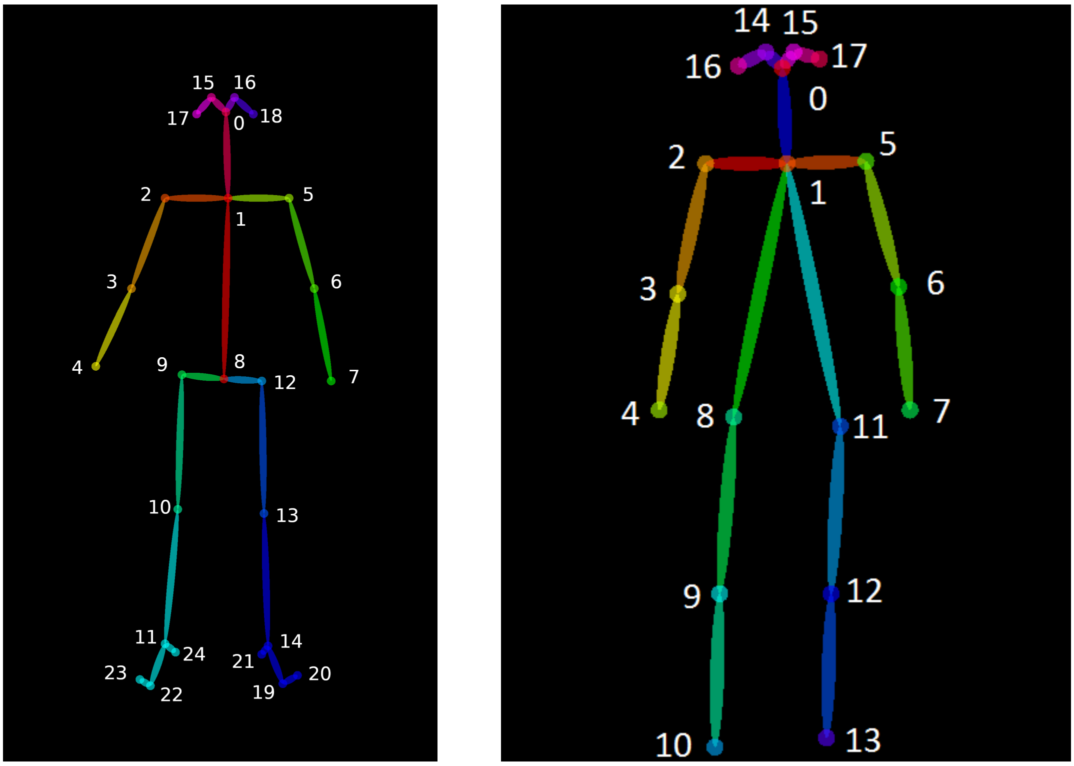
Figure 2.7.: Human Pose and Shape Estimation MTC. The model proposed by Xiang, Joo, and Sheikh (2019) shows superior accuracy compared to previous approaches on the data of Ellen DeGeneres. However, the algorithm shows still inaccuracies when estimating the 3D hand pose.

proposed model is trained on videos in which the human is visible as a whole. Our speaker dataset however does only record speakers from the hips upwards.

Similarly, Xiang, Joo, and Sheikh (2019) use the model developed by Joo, Simon, and Sheikh (2018) which also captures human pose and shape information. Compared to Kocabas et al. (2020), they train the model on upper body part data which allows them to be used with our speaker data set. The deform-able human model Adam is creating after extracting 3D Part Orientation Fields for different body parts. We show an example of fitting the model to our dataset in Figure 2.7.

Although the human pose model fitting by Xiang et al. (2019) produces better qualitative results than previous methods, we still detect inaccuracies with the hand estimation. Furthermore, the temporal coherence is low which would cause problems when training a model for plausible motion generation.

Human Pose Estimation As already suggested by previous work in the field of speech to gesture transformation, we also use the human pose format without additional shape information. We argue that body shape information is not



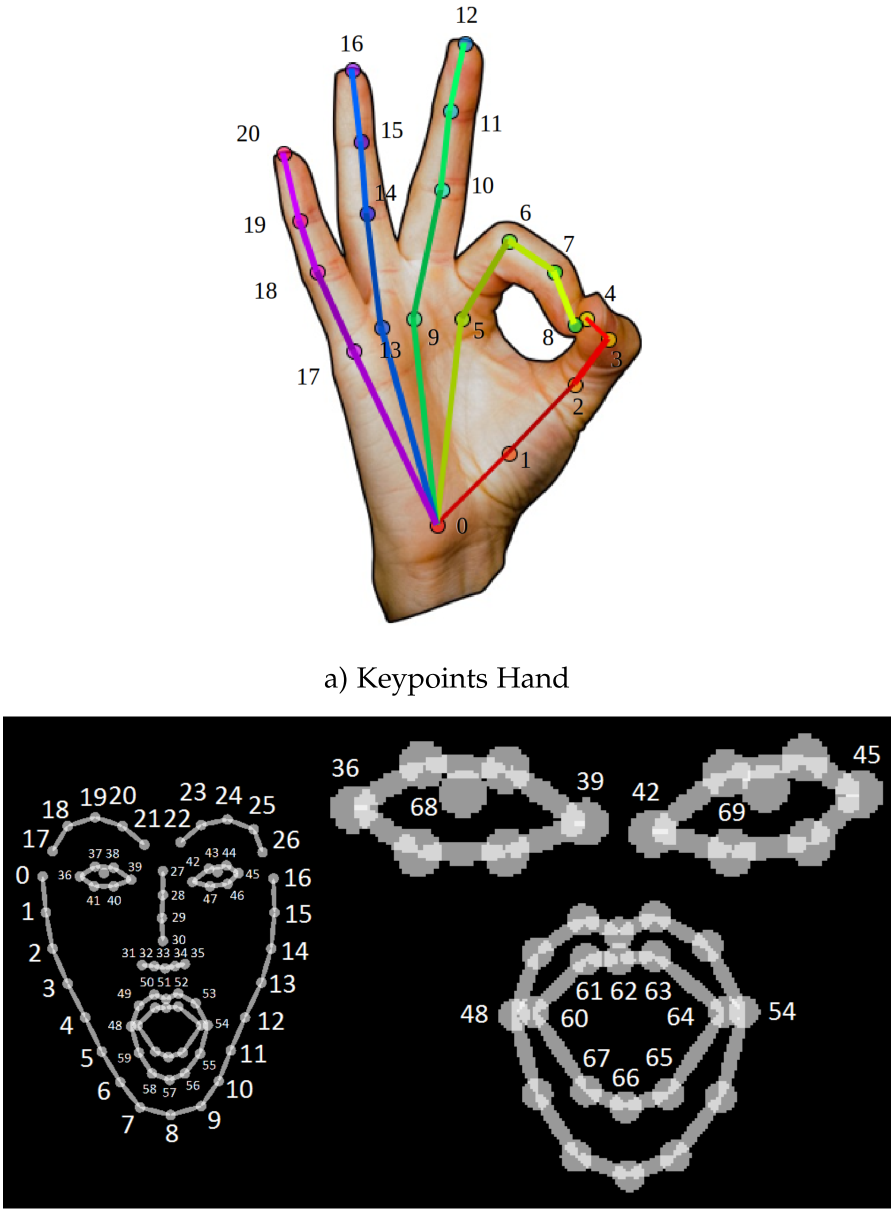
a) Keypoints Pose 25

b) Keypoints Pose 18

Figure 2.8.: OpenPose Keypoints. We illustrate two different representations of the human body pose suggested by Z. Cao, Hidalgo Martinez, Simon, Wei, and Sheikh (2019).

needed for gesture prediction and if needed it can be computed in a separate step. The most popular 2D human pose extraction framework is OpenPose which combines work by Z. Cao, Hidalgo Martinez, Simon, Wei, and Sheikh (2019), Simon, Joo, Matthews, and Sheikh (2017), Z. Cao, Simon, Wei, and Sheikh (2017) and Wei, Ramakrishna, Kanade, and Sheikh (2016). The framework predicts the human body pose using either 18 or 25 keypoints as illustrated in Figure 2.8. The OpenPose model predicts the human pose in real-time which allows us to estimate the pose of our large speaker dataset. Besides the body pose, OpenPose also predicts hand and face keypoints as shown in Figure 2.9. To generate the human skeleton, OpenPose uses Part Affinity Fields which locate body parts that move as a whole in a certain direction.

An OpenPose comparable 2D human pose estimator is Detectron2 by Wu, Kirillov, Massa, Lo, and Girshick (2019). However, the Detectron2 human pose estimation is slower than the OpenPose detection which is problematic considering our large speaker dataset. Other state-of-the-art methods are presented in the work of Sun, Xiao, Liu, and Wang (2019) and Chen et al. (2018).



a) Keypoints Hand

b) Keypoints Face

Figure 2.9.: OpenPose Hand and Face Keypoints. We figure illustrates the representation of the hand pose and face expressions as introduced by Z. Cao, Hidalgo Martinez, Simon, Wei, and Sheikh (2019).

Keypoint #	Description	Keypoint #	Description
0	Center Pelvis	9	Nose
1	Right Pelvis	10	Forehead
2	Right Knee	11	Left Shoulder
3	Right Ankle	12	Left Elbow
4	Left Pelvis	13	Left Wrist
5	Left Knee	14	Right Shoulder
6	Left Ankle	15	Right Elbow
7	Spine	16	Right Wrist
8	Neck		

Table 2.2.: 3D Human Body Keypoints. This table shows the positions of the keypoints on the human body as predicted by Pavllo, Feichtenhofer, Grangier, and Auli (2019).

3D Body Pose Besides estimating the 2D keypoints, they need to be lifted into 3D space such that gestures can be animated on a virtual character. Traditional algorithms achieve this by utilizing the fact that the human bone length is constant. The nearest neighbor approach for 3D pose estimation is to store 2D to 3D mappings in a dictionary. At inference time, the 3D pose of the 2D pose from the dictionary which is closest to the 2D pose observed is selected. The problem of this approach is that a very large number of possible human poses exist. Furthermore, body parts are sometimes occluded which means that the mapping is not identical. The high number of ambiguities makes this approach inferior to current methods.

Pavllo, Feichtenhofer, Grangier, and Auli (2019) present a time coherent approach to lifting 2D body keypoints into 3D space. They train their model on the Human3.6M (Ionescu, Papava, Olaru, & Sminchisescu, 2014) and the HumanEva-I (Sigal, Balan, & Black, 2010) dataset. Furthermore, unsupervised training using back-projection of the 3D estimations into 2D space is used to improve the performance even with a limited amount of training data. They estimate the 3D pose in real-time by implementing dynamic programming inside the CNN. The keypoint output format is shown in Table 2.2. We use the model of Pavllo et al. (2019) in our framework because it achieves high accuracy and temporal coherence using OpenPose 2D body pose predictions as input.

3D Hand Pose Compared to 3D body pose estimation, hand pose estimation is more difficult, because the hand bones are smaller which causes the keypoints to be located more closely to each other. This means that the keypoints are located within a very small region of the image and therefore, the resolution is low. Another problem is that almost all the time when recording a person from

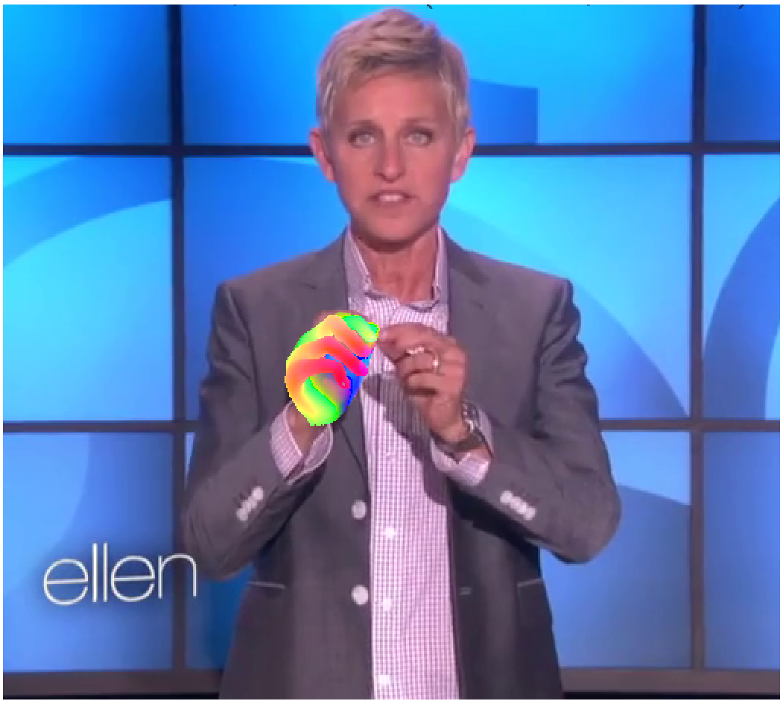


Figure 2.10.: Hand Model Estimation. We show the hand pose estimation of Panteleris, Oikonomidis, and Argyros (2018) on a frame with speaker Ellen DeGeneres.

a specific position, a certain part of the hand is occluded. This causes the 3D positions of the occluded part to be completely estimated by only taking the visible parts and learned data into account.

Panteleris, Oikonomidis, and Argyros (2018) approach this problem by fitting a 3D hand model. Although they optimized their algorithm for video, we find empirically that it is not robust to fast motion and hand pose changes. An example of 3D hand model estimation for speaker Ellen DeGeneres is shown in Figure 2.10.

We run into similar issues when evaluating the work of Ge et al. (2019). Their models are highly optimized for the training dataset but the performance on in-the-wild video data is low. One of the problems for 3D hand pose estimation models is the lack of a 3D ground truth dataset. To tackle this problem, Mueller et al. (2018) enhanced the existing data with synthesized hand pose image data generated by a GAN model. Their large-scale dataset consists of 330,000 color images with hands include the 21 keypoint 3D ground truth positions.

Similarly, Zimmermann and Brox (2017) extend the existing dataset with synthesized hand models. They split up the 3D estimation into two prediction steps. In the first step, the algorithm predicts a 2D confidence map for the position of each keypoint. In the second step, a 3D hand model is fitted to the confidence map. This allows for compatibility with other 2D hand keypoint estimation algorithms such as OpenPose. Therefore, we use this algorithm together with our 2D OpenPose keypoint predictions, to estimate the 3D hand pose of our speakers.

2.8. Related Work Summary

In this chapter, we discuss background information and related work in the field of non-verbal communication. The non-verbal communication category which we focus on is Kinesics. Kinesics include the movement of hands and arms during communication. Within this category, rhythmic beat gestures are the ones used most often during communication. Therefore, we design our gesture model to be capable of predicting beat gestures. When compared against other approaches, we find that Bergmann and Kopp (2009) and Yan (2000) focus on the semantic meaning of speech and consequently predict iconic and metaphoric gestures. Instead, we extract prosody from speech. The work of Ferstl and McDonnell (2018) shows that our Mel spectrogram encoded audio input provides better performance than MFCC. Most recent work in the field of gesture prediction utilizes generative models (Ginosar et al., 2019; Alexanderson et al., 2020; Ferstl et al., 2019) to describe the relationship between speech and gestures. The GAN model which we implement is similar to the architecture introduced by Ginosar et al. (2019). However, we modify the GAN such that input and output gestures are 3D instead of 2D. Furthermore, we also change the evaluation procedure. Instead of performing a human study on the skeletal format, we animate our gestures on virtual characters. The advantage of our method is that our animation is visually more appealing than abstract representations.

3. Structural Approach and Development

In this chapter, we explain the path we choose to generate and animate gestures on virtual humans. Besides the general method, we also explain what experiments we perform to validate our method. For each of the experiments, we show the results and what impact they made. Our final and most important evaluation of our method, the user study, is presented in Chapter 4.

In Chapter 2, we found that most successful work in gesture prediction implement generative models. Thus, we model the relationship between speech and gestures by a Generative Adversarial Network (GAN). Besides modeling the transfer from speech to gestures, we also need to ensure plausible motion. This was accomplished in the past by a motion discriminator, which we also implement in our GAN architecture.

Related work shows that it takes a lot of effort to create motion capture data for body language modeling. However, recent work utilizes publicly available video data to train speaker-specific gesture models. We also use in-the-wild video data to train our GAN. An important factor is the quality in which we extract gestures from the video data. Therefore, we implement state-of-the-art human pose estimation algorithms. The human pose representation is the golden standard in current research related to gesture synthesizing.

3.1. Conceptual Architecture

Based on what we have learned from related work, the Gesture GAN is our core component for gesture generation. It learns to model beat gestures by filtering for prosodic features given input speech. The input speech is transferred into a Mel spectrogram to support the detection of prosody. The Gesture GAN implements a motion discriminator which ensures that the predicted gestures have realistic motion. To ensure realistic motion, the discriminator needs a real motion input which leads us to our second building block.

Our second building block is the extraction of motion from in-the-wild video data. This is possible by using state-of-the-art computer vision algorithms, which extract the human pose of frames in which the speaker is present. Besides extracting the human pose, we can also project the human pose information into 3D space. The 3D information allows us to animate gestures in virtual environments. By extracting the human pose from video we guide our Gesture GAN into two directions. First, the human pose is used to learn gestures. Second, it is also used to learn motion.

The third building block we implement differentiates our work from many other approaches. Instead of evaluating the gestures in the human pose format, we animate the gestures on virtual avatars. We implement inverse kinematics, temporal smoothing and kinematic constraints to convert the human pose format to an animation.

3.2. Structure

Our data-driven approach towards the generation and animation of gestures on a virtual human consists of three main steps. First, we collect our input video data and extract the 3D Human Pose of the speaker for each of the frames in the video. Second, based on the 3D Human Pose information, we train a model to predict gestures from speech input. The model is guided by the 3D Human Pose information extracted from the videos. Third, we animate the generated gestures in 3D Human Pose format on virtual humans. To recover lost information about rotation angles, we use inverse kinematics. An overview of our method is shown in Figure 3.1.

3D Human Pose Estimation from Video We have to use an efficient representation of gestures, otherwise our model becomes complex and does not learn the task of gesture generation. Therefore, we represent the gestures in the 3D Human Pose Estimation format. However, this format cannot be extracted directly from video. The reason is that there exist many ambiguities and occlusions, which make the task difficult. We approach the problem with three different models. The first model extracts the 2D Human Pose from the raw video. The second and third model then project the 2D Pose into 3D space for the large body parts and hands, respectively.

Gesture Generation Once we have computed the efficient 3D Human Pose representation, we use this representation to train our Gesture Generative Adversarial Network (Gesture GAN) model to learn the relation between speech and gestures. Similarly to frame processing, we also convert the audio into a different format, namely the Mel spectrogram. Inside the Gesture GAN, the generator predicts gestures from input speech. The discriminator forces the generator to predict gestures which are similar, in terms of motion, to real human gestures.

Gesture Animation At the end of our gesture generation and animation pipeline, we transform the gestures represented in the efficient 3D Human

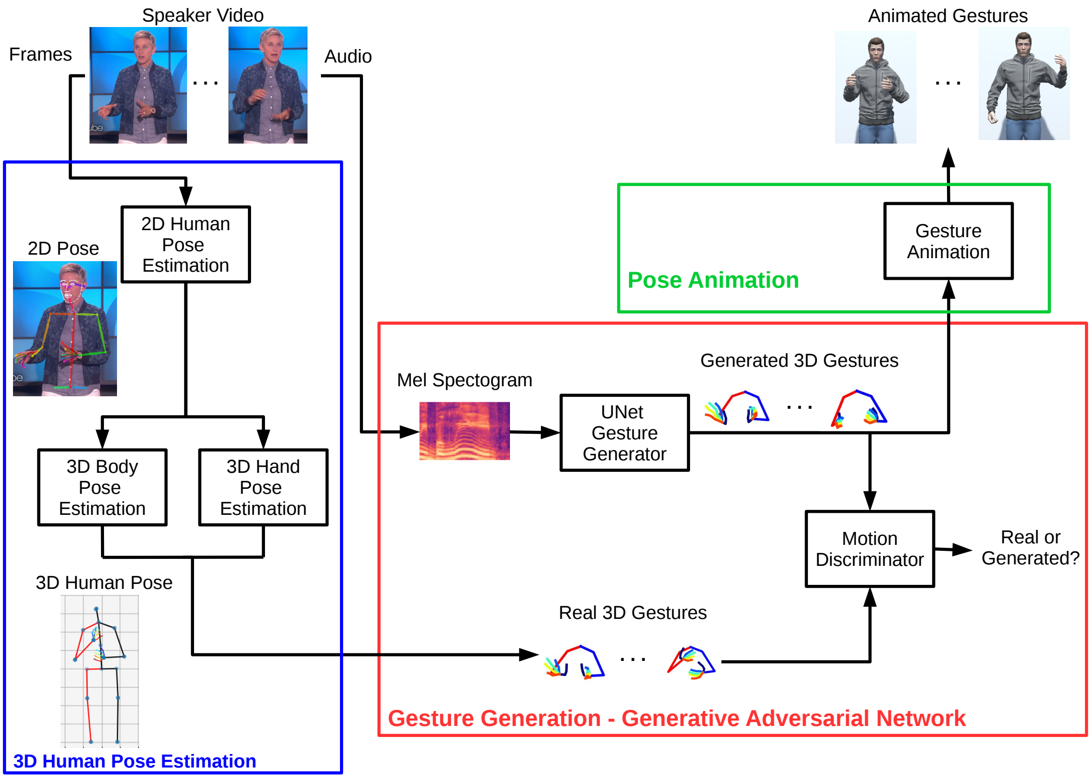


Figure 3.1.: Gesture Pipeline. Our gesture generation pipeline takes videos of speakers as an input (top left) and produces animated gestures (top right). In between, three major components exist: 3D Human Pose Estimation (blue), Gesture Generation using a Generative Adversarial Network (red) and Pose Animation (green).

Pose format to animations on virtual humans. Therefore, we use the rotation angles between bones in the human skeleton. Since not all rotation angles can be recovered, we use inverse kinematic computations to reveal the missing angles. In addition, we apply motion constraints such that only anatomical plausible gestures are animated. We surround our virtual human with a simple virtual environment in which we conduct the user study.

3.3. Hardware and Setup

We run different parts of our experiments on the following Hardware and Software setup. We perform computer vision tasks such as pose extraction, 3D estimation and gesture generation on a desktop PC with an Intel Core i7-8700 CPU and 32 GB memory. Whenever possible, we parallelize the computation on the two high-end consumer graphics cards Geforce RTX 2080 TI and GeForce RTX 2070.

For the computation of computer vision and machine learning algorithms on the GPUs, we use the CUDA 11 developer toolkit. Our image processing algorithms are implemented in Python 2 and 3. We run our deep learning models on the two most popular machine learning frameworks PyTorch and Tensorflow.

Our virtual human gestures are animated using the Unity engine with UMA2 and LipSync Pro assets. After animating the characters in Unity, we set up an Apache2 web server for our user study. We install a WordPress front end and a MySQL database on our web server. To run the user study, we select a domain from dyndns.org and add the Quiz And Survey Master WordPress plugin to create our user study. The videos presented in the user study are stored on a non-listed YouTube server.

3.4. Data Collection

The first important step towards our data-driven approach is the collection of video data for different speakers. We select public speakers from show business, politics and academia because we find a large amount of suitable data for speakers in these fields. Although there exists a lot of video material for *e.g.* political speakers, we are not able to find enough clean data which can be used for our experiments. The main problem is that video data from politicians does often not include their hand gestures because the videos are recorded from a position close to the speaker. When recorded from a larger distance, their hands are often hidden behind a podium which also leads to problems. We are not able to collect data of speakers from politics and similar domains, because the total amount of video data that needs to be stored and processed to get sufficient clean data is too large.

3.4.1. TV Show Speakers

Instead, we collect data from comedy TV show hosts and academic speakers. When considering TV show hosts, most video data is suitable, because they usually have monologues in which they face the camera and hands as well as arms are visible in the video. They are also trained to use body language and gestures effectively, which makes them the perfect data source. We select Ellen DeGeneres and Jon Oliver for our experiments. We use some of the existing video links from Ginosar et al. (2019) to download video data.

3.4.2. Academic Speakers

When considering speakers from academia, the task of finding suitable speakers is more difficult. The problem is that academic speakers show a less active communication style and their hands are often hidden behind a podium. Furthermore, their speech is often interrupted by questions from the audience, slide shows or writing on a blackboard. Nevertheless, a large number of different lectures from different speakers is online thanks to programs such as MIT OpenCourseWare, Open Yale Courses and Stanford Online which makes the search easier.

We select Jonathan Gruber and Shelly Kagan because they are both active speakers and they usually do not use digital presentations. Jonathan Gruber is an MIT professor in economics and Shelly Kagan is a philosophy professor at Yale University.

3.4.3. Data Preparation

Once we have selected the speakers, we have to find suitable video material. Therefore, we manually search for videos of each speaker and check a few positions in the video to see whether the quality is sufficient and the speaker is visible most of the time. After the coarse manual investigation, we download the video material. We then perform a more detailed frame by frame investigation to decide which sequences are suitable for gesture generation. We automatically select valid sequences using the 2D human pose estimation framework OpenPose (Z. Cao et al., 2019). To detect valid frames and sequences, we use the following constraints:

- Exactly one person must be detected by OpenPose. This constraint ensures that only the speaker is visible and no person from the audience is detected. Furthermore, it ensures that the speaker is visible and that the video is not showing a presentation slide or other object.

- The speaker which is detected needs to be large enough in size on the video. On a 16 : 9 video of height 720 pixels, the neck to hip distance must be at least 150 pixels. When the video height is 360 pixels, the neck to hip distance minimum is 90 pixels. We need this constraint to ensure that body pose estimation will be accurate. The factor is proportionally higher on the video with 360 pixel height, because the resolution is worse.

- The number of keypoints detected must exceed a threshold. Out of the 70 face keypoints, more than 60 must be detected and out of the 21 hand keypoints, at least 11 must be detected on each hand. We do not apply a constraint on the body keypoints, because the face and hand keypoint constraints are harder to meet for OpenPose. This constraint ensures that the number of keypoints that are interpolated does not become too large and that the quality can be preserved.

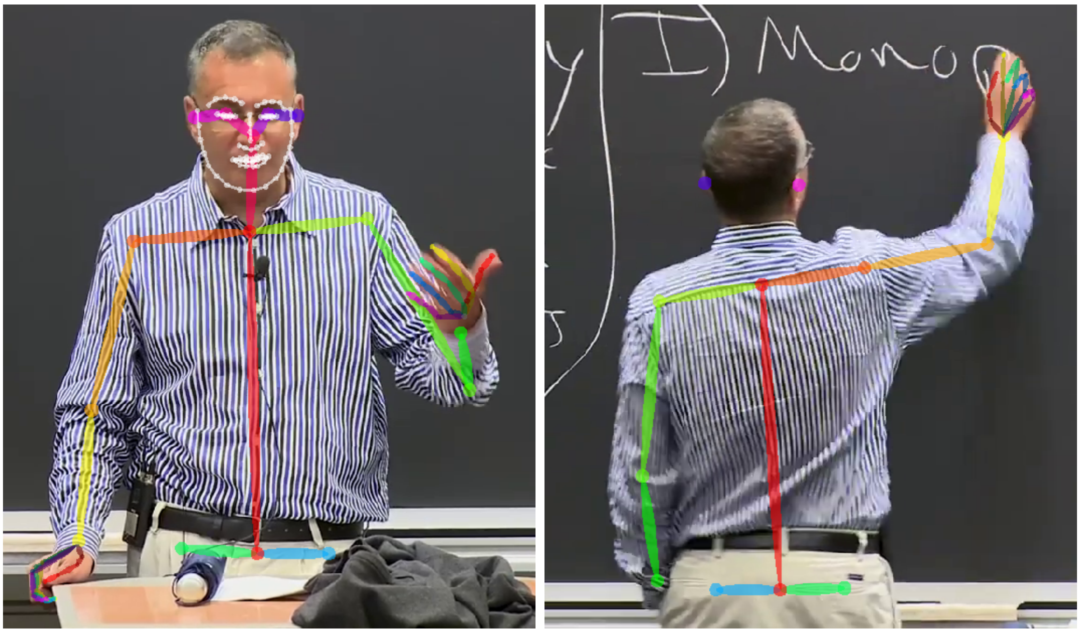
Constraint	**Short Description**
Number of people == 1	Ensure that speaker is visible
Neck-hip distance > 150 or 90	Speaker visible in good quality
Number of valid face keypoints > 60	Face is detected accurately
Number of valid hand keypoints > 10	Hand is detected accurately
Shoulder position x: left > right	Speaker is facing camera
Invalid frame interval < 0.2 sec	Large parts missing must not be interpolated
Sequence length > 5 sec	Enough temporal context

Table 3.1.: Speaker Sequence Detection Constraints. We summarize the constraints we apply for the generation of valid speaker intervals. The upper part shows the constraints applied on each frame whereas the lower part contains constraints applied on a sequence. The constraints are enforced using the keypoints detected by the OpenPose framework.

- The speaker needs to face the camera. This is ensured by the constraint that takes into account the shoulder positions. The right shoulder position needs to be left of the left shoulder position in the frame.
- The sequence can contain intervals of invalid frames of up to 0.2 seconds. This constraint ensures that sequences can still be valid even if a few intermediate frames are invalid. The lost information can still be approximated by interpolation.
- The total length of a sequence has to be at least 5 seconds. This constraint ensures that enough temporal context of speech and gestures is available.

We summarize the constraint is Table 3.1. For illustration purposes, we also show an example of a valid next to an invalid frame in Figure 3.2.

We apply all our constraints on visual properties and do not apply constraints on speech. We found that in our datasets, the speaker from which we retrieve data is speaking all the time, except for very few intervals.



a) Valid Frame

b) Invalid Frame

Figure 3.2.: Valid vs Invalid Frames. We apply constraints to determine whether frames and sequences are valid for gesture learning. On the left we show a valid frame of speaker Jonathan Gruber where all detected keypoints are within the constraint boundaries. On the right we show an invalid frame where three constraints are not met. The left hand of the speaker is not visible, the shoulder positions indicated that the speaker is not facing the camera and face keypoints are not detected.

Keypoint #	Description	Keypoint #	Description
0	Nose	8	Mid Hip
1	Neck	9	Right Hip
2	Right Shoulder	10	Left Hip
3	Right Ellbow	15	Right Eye
4	Right Wrist	16	Left Eye
5	Left Shoulder	17	Right Ear
6	Left Ellbow	18	Left Ear
7	Left Wrist		

Table 3.2.: Keypoints of Interest. Out of the 25 OpenPose body keypoints, we estimate the 18 keypoints illustrated in this table from videos in our speaker dataset.

3.5. Human Pose Estimation

Once we have selected our speakers and downloaded the video data, we apply human pose estimation to extract the relevant information for training our gesture model. We apply human pose estimation in two steps. First, we estimate the 2D human pose from the video input. Second, we take the estimated 2D human pose and project it into 3D space. We perform the 3D projection for the hands separately from the other body parts. We also apply temporal smoothing to ensure temporal coherent motion prediction.

3.5.1. 2D Pose Estimation

We use the raw video material as an input for the OpenPose human pose predictor. We select the Body 25 output format shown in Figure 2.8. Since the input videos only show the speaker from the hip upwards, we can only extract the body keypoints shown in Table 3.2. For 3D projection, we later estimate the missing keypoints. An example of all extracted body keypoints is shown in Figure 3.2a.

In addition to the body keypoints, we also extract hand and face keypoints. We extract all of the 21 hand and 70 face keypoints shown in Figure 2.9, because the hands and face of our speakers are always visible in the input videos.

When running the OpenPose keypoint estimation, we have to make a speed-accuracy trade-off, because our large amount of input video data forces us to lower the quality of hand pose estimation. When maximizing the hand pose estimation quality by using six different input scales for the detection model, the estimator only processes three frames per second (FPS). However, this speed is too slow to process several hours of input video data. Therefore, we choose

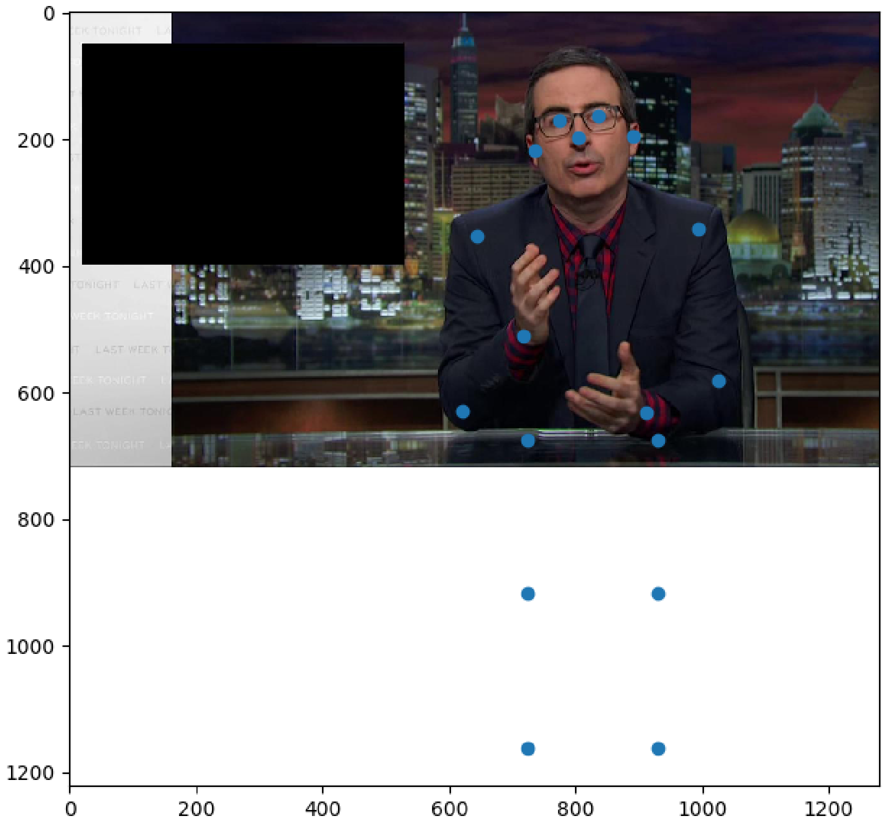


Figure 3.3.: Detectron 2D Human Pose. The image shows an example of the Detectron 2D human pose format on speaker John Oliver before the human pose is projected into 3D (Figure 3.4). Each blue point represents one of the 17 keypoints. We coarsely approximate the position of the four leg keypoints using the pose information obtained from the OpenPose detection. The missing color information in the lower part of the image does not influence the 3D projection, because for this procedure only the keypoint positions are relevant.

a setting that produces slightly worse outputs but instead achieves 30 FPS in computation time. The output of the 2D keypoint estimation consists of x and y-coordinates and a confidence value for each keypoint $\mathbf{K}_{2D} \in (\mathbb{N}^{127\times 2}|\mathbb{R}^{127\times 1})$. We store the output in JSON format for each frame.

Once we have identified valid and invalid frames using the 2D human pose estimation output, we group consecutive valid frames into sequences. These sequences are then taken as input for the 3D projection algorithm.

3.5.2. 3D Body Pose Estimation

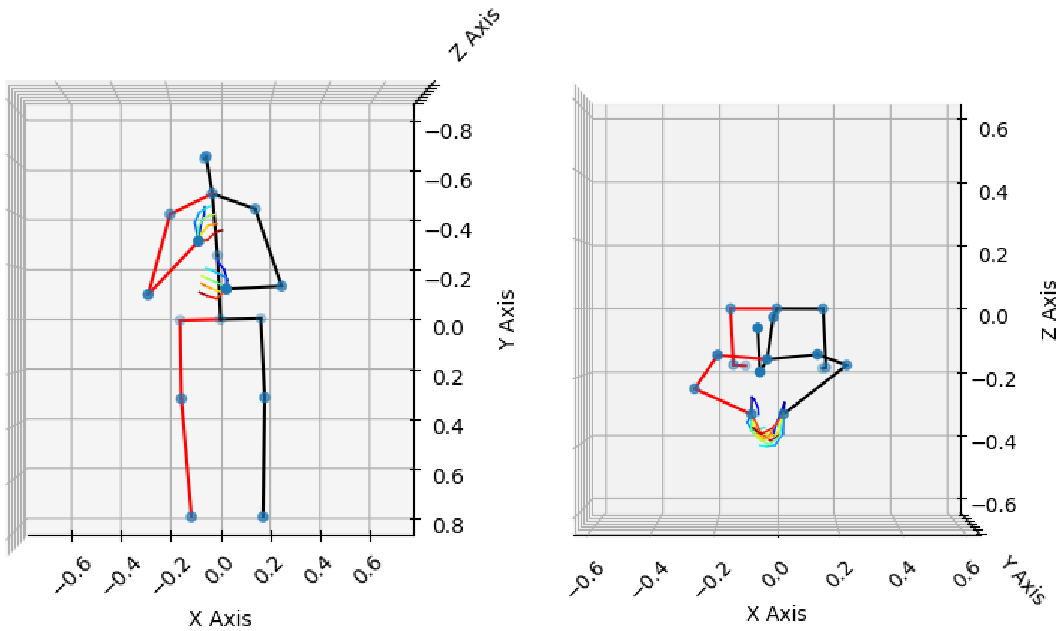
The 2D human pose provides a basis for our gesture prediction. Yet, we want to animate virtual humans in 3D. Therefore, we lift the 2D keypoints into 3D. We handle hand and body projection into 3D space separately.

Data Format We use the model introduced by Pavllo et al. (2019) for 3D human body projection. This model projects 2D human pose information into 3D while also enforcing temporal coherence. We cannot directly input the OpenPose keypoints but have to pre-process the data. From the 2D Keypoints $\mathbf{K}_{2D}$ we select 13 keypoints that correspond to the Detectron (Girshick, Radosavovic, Gkioxari, Dollár, & He, 2018) human pose encoding. Thus, we neglect the keypoints neck and mid hip. An example of the Detectron keypoints on speaker John Oliver is illustrated in Figure 3.3. Furthermore, we do not use the confidence values computed by OpenPose. We found that this information is not needed for 3D body pose prediction. We end up with the matrix $\mathbf{K}_{2dDetectron} \in \mathbb{N}^{17\times2}$.

Data Modification In addition to changing the data format, we also need to handle missing information. The human pose data from OpenPose has two problems: missing keypoints and missing lower body information. In order to run the 2D to 3D projection model, all 13 2D keypoints need to be available for all frames within a sequence. Therefore, we apply linear interpolation to approximate the position of the missing keypoints. Note that we have ensured that within a sequence, the maximum time of invalid frames with many missing keypoints is 0.2 seconds. We also have ensured that valid frames have a high percentage of successfully estimated keypoints.

For successful 3D human pose estimation, we need 2D information about the whole body which also includes leg keypoints. Unfortunately, our source video data does not include the lower body of the speakers. Consequently, we approximate the missing data using information about the human anatomy. The human anatomy states that the distance between eyes and hips is about the same as the distance between the hips and ankle. For the computation of the eye position, we consider both eye, both ear and the nose keypoints. Similarly, we use the left hip, mid hip and right hip keypoints for estimating the hip position. The distance from hips to knees is about the same as the distance from knees to ankles. Using this information allows us to approximate the missing information with high enough precision to successfully run the 3D projection. The inaccuracies introduced by this estimation are not problematic because we only use the lower body parts for the 3D human projection and do not need them for the gesture generation. An example of the leg keypoint approximation is shown in Figure 3.3.

3D Projection After passing the input in Detectron format to the trained model from Pavllo et al. (2019), we receive the 3D human pose estimation consisting of 17 keypoints $\mathbf{K}_{3D} \in \mathbb{R}^{17\times3}$. The matrix $\mathbf{K}_{3D}$ contains X, Y and Z-values of each keypoint in meters. The positions are relative to the Center Pelvis keypoint which is located at $(0,0,0)$. The keypoints are predicted at



a) Front View

b) Top View

Figure 3.4.: 3D Human Pose Estimation. The two skeletons illustrate an example of the 3D human pose estimation of speaker John Oliver. The front and top view are presented on the left and right, respectively. The corresponding 2D frame is shown in Figure 3.3. The blue points correspond to the 17 different keypoint positions predicted. The color intensity of these points indicate the distance to the viewer, where high intensity indicates proximity to the viewer. The red lines connect keypoints on the right side of body, whereas the black lines connect central and left keypoints. The five lines within the color range from red to blue represent the estimated 3D pose of each hand. The unit for all axes in both plots is meters.

different locations of the human body compared to the input. The location of the output keypoints is depicted in Table 2.2.

We show an example of the connected output keypoints, referred to as skeleton, in Figure 3.4. Note that in Figure 3.4 we also include the hand skeleton, which is not predicted by the body pose prediction model. The corresponding input is depicted in Figure 3.3.

3.5.3. 3D Hand Pose Estimation

Besides estimating the larger body parts, we also need to exact the hand positions of each speaker to train our gesture generation model. For 3D hand pose estimation, we use a model based on the work of Zimmermann and Brox (2017). The basis which we use for our prediction is a 3D hand pose estimation model which takes a heatmap $\mathbf{H} \in \mathbb{R}^{32\times32\times21}$ as input and predicts the 3D keypoints $\mathbf{K}_{3dHand} \in \mathbb{R}^{21\times3}$ for each hand. We only use this part of the work presented by Zimmermann and Brox (2017).

Keypoint Heatmap The heatmap volume $\mathbf{H}$ consists of 2D heatmaps of size 32×32 for each hand keypoint which are stacked in the third dimension. The value at each position indicates the likelihood that the actual 2D keypoint is located at that position. Values close to one indicate high probability and values close to zero indicate low probability. The heatmap is generated such that it fits the size of the hand. Therefore, the positions on the heatmap are relative and scaled compared to the absolute positions in a video frame. One example of how the 2D heatmap for a single keypoint looks like is similar to a Gaussian kernel depicted in Figure 3.5. In this example, the position of the keypoint will most likely be in the center of the 32×32 image.

2D Hand Pose to Heatmap To create the heatmap, we take the 21 right and 21 left hand keypoints $\mathbf{K}_{2D}$ predicted by OpenPose. For each hand keypoint, we use the confidence value $p \in [0, 1]$ provided by OpenPose and apply a 2D Gaussian kernel centered at the estimated 2D keypoint position shown in Figure 3.5. An example of an input frame of speaker Shelly Kagan including the right hand pose detection by OpenPose is shown in Figure 3.6. We apply an image padding to ensure that the heatmap creation will work when the hand keypoints lie on the image border. We set the pixel dimensions of the 2D Gaussian kernel to $0.2 \cdot h \cdot \frac{200}{720}$, where h represents the height of the input image. We find empirically that using the standard deviation $\sigma = 8$ for the 2D Gaussian kernel produces heatmaps which result in accurate 3D hand pose estimations. To fill the heatmap with the probabilities we multiply the kernel with the OpenPose confidence and center the output at the predicted OpenPose position. Following this procedure, the 2D heatmap for each keypoint is created.

Once we have created the heatmap on the full input image resolution, we crop and scale the heatmap to the size the 3D estimator was trained on. Therefore, we take the 2D positions of each hand and crop the heatmap using the smallest possible square that contains all hand keypoints. We also apply a margin of 25 % such that no keypoint is located directly at the border. The margin also ensures that the Gaussian kernel of keypoints located at the border is not cut in half and remains fully visible in the heatmap. After cropping the 2D heatmap such that it only contains the pixels needed for 3D hand pose estimation, we downscale the heatmap of arbitrary size to the fixed resolution 32×32. An example of the 2D heatmap is shown in Figure 3.7b. The corresponding input frame is shown in Figure 3.6.

Temporal Smoothing We ensure that the heatmaps which we create from the 2D OpenPose prediction are temporally coherent. To enforce temporal coherence, we iteratively create the 2D hand keypoint heatmap using the

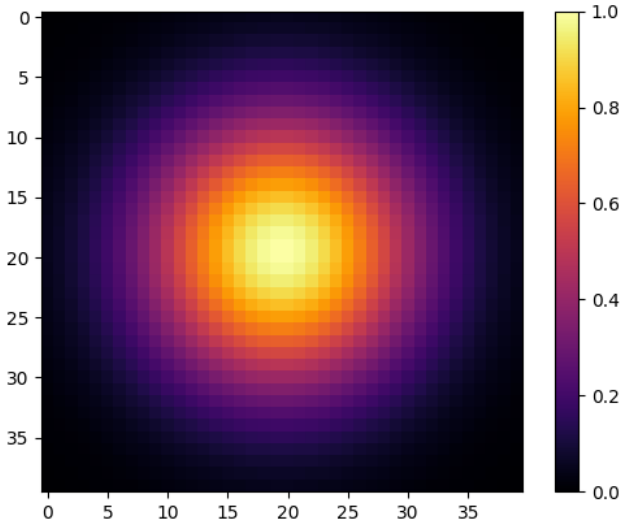


Figure 3.5.: Gaussian Kernel. We use a Gaussian kernel to transfer the 2D keypoint prediction into a heatmap of possible 2D positions. In this example we show a kernel with standard deviation $\sigma = 8$ on a 40×40 image.

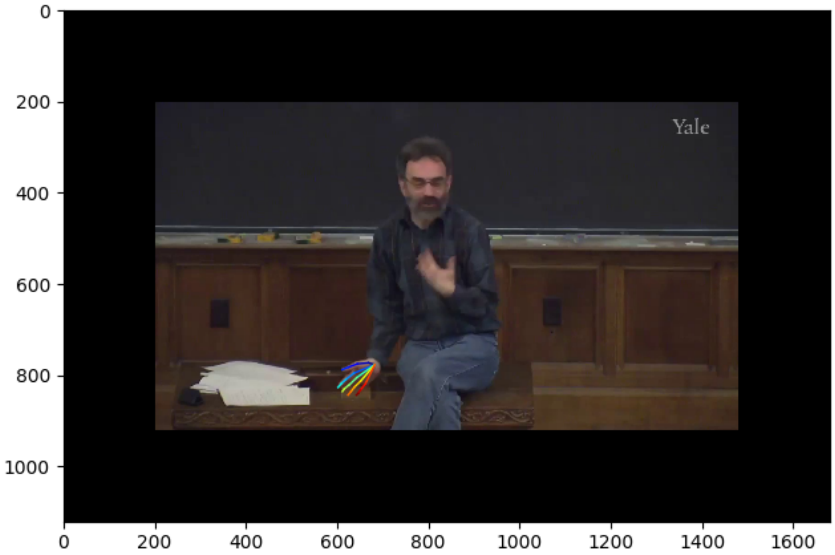
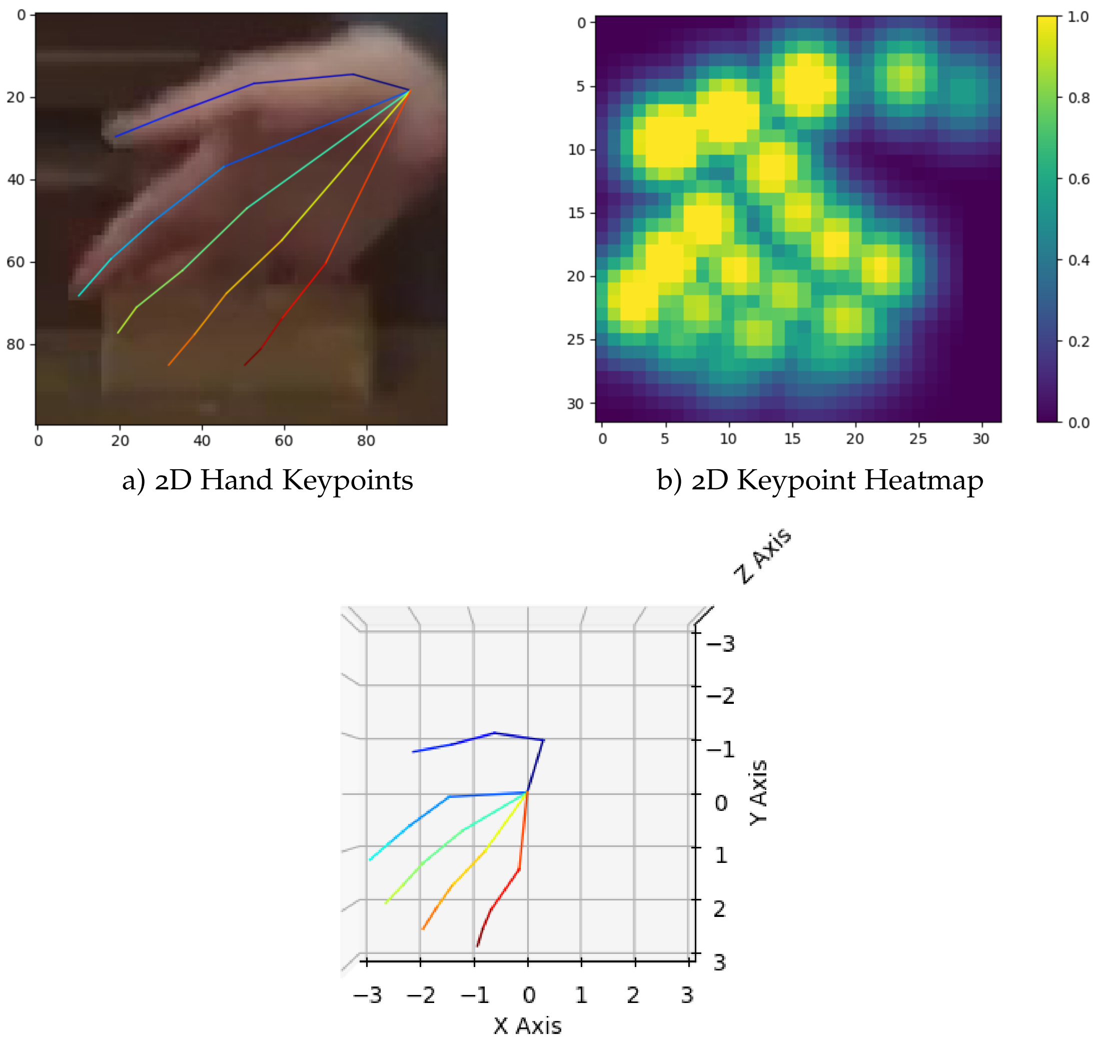


Figure 3.6.: 3D Hand Pose Estimation Input. The input for our 3D hand pose estimation are the 2D hand keypoint locations and their 2D estimation confidence value. This figure shows speaker Shelly Kagan and the 2D right hand pose estimation by OpenPose. The colored lines represent the connections between the keypoints for each finger from the little finger (red) to the thumb (blue). We pad the input frame, using black pixels to support hand keypoints that lie on the image border. This is needed because a keypoint that lies on the image border produces a keypoint heatmap in which some pixels lie outside of the original image.



a) 2D Hand Keypoints

b) 2D Keypoint Heatmap

c) 3D Hand Pose

Figure 3.7.: 3D Hand Pose Estimation. We present an example of the 3D hand pose estimation for the right hand of speaker Shelly. The corresponding input frame is shown in Figure 3.6. The input for our 3D hand pose estimation model are the 2D hand keypoints predicted by OpenPose. We connect and visualize them with the colors blue to red representing the fingers thumb to little finger. The 2D keypoint heatmap illustrates the confidence value for each keypoint at a certain location. Areas in yellow indicate high confidence. Keypoints which are occluded or visually similar to the background such as the little finger in this example have lower confidence values. The graph in the second row shows the estimated 3D Hand Pose.

heatmap from the previous frame

$$\mathbf{H}^{t}_{smooth} = \mathbf{H}^{t} + \lambda_{smooth} \cdot \mathbf{H}^{t-1}_{smooth} , \tag{3.1}$$

where $\mathbf{H}^t$ represents the heatmap of input keypoints $\mathbf{K}^t_{2dHand}$ at time step t and λ_{smooth} is the smoothing factor. We qualitatively compare $\lambda_{smooth} = [0.3, 0.5, 0.7]$ and find that $\lambda_{smooth} = 0.5$ produces the best results.

3D Prediction Given the keypoint heatmap, we predict the 3D keypoints using the model introduced by Zimmermann and Brox (2017). An example of the 3D hand pose prediction is shown in Figure 3.7c. We scale the predicted 3D hand pose such that the distances between the keypoints are approximately given is meters. Note that the scaling will not be taken into account when animating the hand on a virtual human.

After predicting the 3D body pose and the 3D hand pose, we connect the wrist keypoint of the human pose with the wrist keypoint of the hand pose to get the complete 3D human pose estimation. An example of the complete 3D human pose estimated from the input frame illustrated in Figure 3.3 is shown in Figure 3.4.

3.6. 3D Skeletal Gesture Generation

Given the 3D human pose data, we train a model to predict gestures from audio input. During training, we input speech and predict gestures encoded using the human pose representation. When designing the model we have to take into account that the relation between speech and gestures is not bijective. A specific utterance can lead to different gestures. Similarly, many possible utterances could appear given a certain gesture. Consequently, a simple discriminative model $p(y|x)$ will not describe the relation between the speech and gestures correctly. It will regress to the mean of many possible gestures for given speech. Instead, we design a generative model that can describe the joint probability distribution $p(x, y)$ between speech and gestures.

3.6.1. Gesture GAN

We use the Generative Adversarial Network (GAN) introduced by Goodfellow et al. (2014) for gesture generation. The GAN allows us to perform a data-driven approach on a generative model. An overview of our GAN design is shown in Figure 3.8. In our GAN architecture, we implement a UNet generator with the task of predicting gestures from input speech. The objective of the generator is twofold. First, the generator is trained to predict gestures close to the ground truth gestures extracted from the input video. Second, the main objective of the

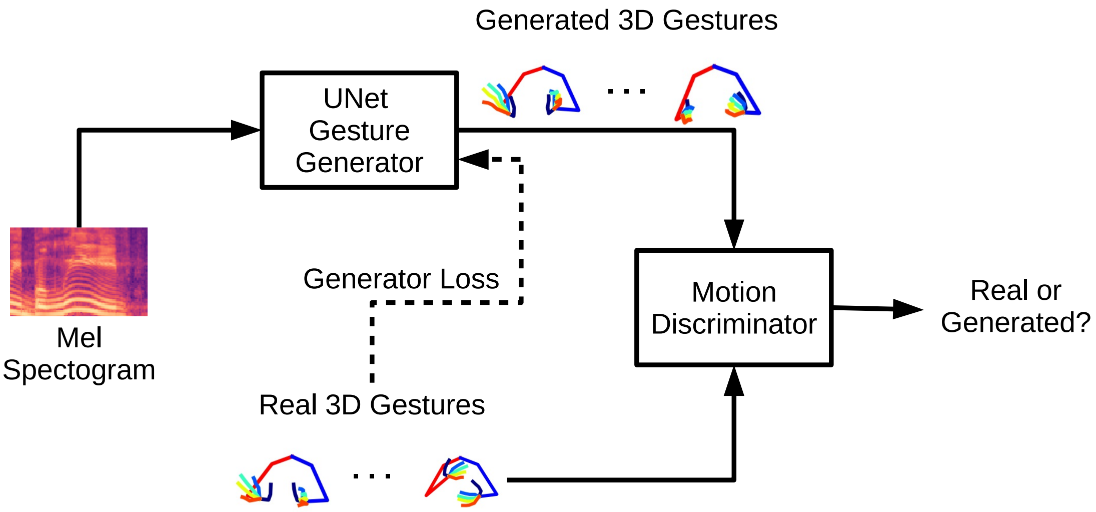


Figure 3.8.: Gesture GAN. An overview of the architecture of our Gesture GAN. The inputs of the GAN are the Mel Spectrogram and the Real 3D Gestures. Our Gesture Generator implements the UNet architecture and predicts 3D Gestures from the Mel Spectrogram. The Gesture Generator is trained by a loss function that penalizes deviations from the Real 3D Gestures. The Generated 3D Gestures and the Real 3D Gestures are then forwarded to the Motion Discriminator which has to determine the origin of each of the two gesture sequences by analyzing the motion.

generator inside the GAN framework is to fool the discriminator. Consequently, the predicted gestures should have realistic motion. The generator receives a Log Mel Spectrogram sampled at a rate of 16,000 Hz from the raw audio as an input. The output gestures are encoded in the Human Pose format.

The discriminator receives two different gesture sequences as input. One gesture sequence is predicted by the generator, whereas the other gesture sequence is directly obtained from the input video. Instead of providing the gestures directly, we compute the difference between consecutive pairs of gestures and provide the gesture motion as an input. Therefore, the discriminator is referred to as motion discriminator in our GAN architecture. The objective of the discriminator is to detect which gestures are real and which are generated. Consequently, we have created a two-player game in which the generator tries to predict gestures which are close to the real gestures in terms of the motion. The discriminator tries to detect the gestures which are predicted by the generator even as they get more realistic.

3.6.2. UNet Generator

We implement the UNet architecture developed by Ronneberger et al. (2015) to generate gestures encoded in the human pose format from speech. This architecture acts as the generator in our GAN framework. We implement the UNet architecture because the bottleneck and the skip connections are beneficial for our task. The skip connections forward low-level prosodic features directly filtered from the input audio. These features are necessary to predict smaller beat gestures. The bottleneck extracts high-level features that contain information about long input sequences. This is useful to predict the posture of the speaker.

The input for the generator is a Log Mel Spectrogram of audio data directly extracted from the training videos of the speaker. We sample the audio data at a rate of 16,000 samples per second. In order to provide the generator with the temporal context, the input interval is four seconds long. The task of the generator is to predict a natural 3D human pose. This will be enforced by a regression loss on the prediction given the pseudo ground truth gestures estimated. The loss function for the generator is given by

$$\mathcal{L}_{\text{Gen}}(G) = \mathbb{E}_{(\mathbf{s},\mathbf{p})}[||\mathbf{p} - G(\mathbf{s})||_1] + \lambda_{\text{bone}}\, \mathbb{E}_{(\mathbf{s})}[||B(G(\mathbf{s}_t)) - B(G(\mathbf{s}_{t-1}))||_1], \tag{3.2}$$

where vector $\mathbf{s}$ refers to the Log Mel Spectrogram input and vector $\mathbf{p}_t$ refers to the pseudo ground truth body keypoints. The function G represents the generator model and B computes the bone length which is computed by the euclidean distance between pairs of keypoints at consecutive time steps t and $t - 1$. Therefore, the second term in Equation (3.2) ensures that the bone length stays constant over time. The first term ensures that the predicted output matches the ground truth gestures extracted from the video. The hyperparameter $\lambda_{\text{bone}} \in (0, 1)$ is used to weight the importance of constant bone length in the prediction.

3.6.3. Motion Discriminator

Communication theory shows that the same utterance can to different gestures. Hence, the regression loss between generated gestures and the pseudo ground truth does not model the relation between speech and gestures as a whole. The problem is the regression towards the mean. For example, if the input data includes the same utterance twice where one time the gesture moves into the direct opposite direction of the other time, the model learns to predict no motion. When considering the whole training set, this will result in predictions with less motion compared to the training data set. To avoid this phenomenon, our discriminator inside the Gesture GAN architecture ensures that the motion of the generated gestures is similar to the motion extracted from video.

We add a discriminator to our gesture generation model which decides if a given gesture is generated or directly extracted from video. The input for the discriminator is the motion of keypoints throughout a sequence. The motion between two time steps is computed by $\mathbf{m}_t = \mathbf{p}_t - \mathbf{p}_{t-1}$. The discriminator should prevent the generator from regressing to the mean of multiple possible movements. Therefore, the generated gestures are more natural and dynamic.

The complete GAN loss function including the discriminator D is defined as

$$\mathcal{L}_{\text{GAN}}(G, D) = \mathbb{E}_{(\mathbf{m})}[\log D(\mathbf{m})] + \mathbb{E}_{(\mathbf{s})}[\log(1 - D(G(\mathbf{s})))] \;. \tag{3.3}$$

The objective of the discriminator is to maximize this function. Consequently, the term loss is only true with respect to the generator. The discriminator is trained to output $D(\cdot) \to 1$ if input motion is real and $D(\cdot) \to 0$ if the input motion is generated.

3.6.4. Training the Parameters of our Model

For training the parameters of our GAN model, we use the training data set consisting of valid video sequences for a specific speaker as explained in Section 3.4. To cut our four second training intervals from the arbitrary length sequences, we use a sliding window with a shift of five frames which is equal to 0.33 seconds.

We train the parameters of our model by combining the loss functions shown in Equation (3.2) and Equation (3.3). The final objective function is defined as

$$\min_{G} \max_{D} \mathcal{L}_{\text{GAN}}(G, D) + \mathcal{L}_{\text{Gen}}(G). \tag{3.4}$$

The generator G has the objective to minimize this function whereas the discriminator D aims to maximize $\mathcal{L}_{\text{GAN}}$.

We optimize the parameters of both the generator and the discriminator with Adam introduced by Kingma and Ba (2015). The learning rate is set to $\eta = 10^{-4}$. We train the parameters of the GAN for each speaker for a total of 300,000 iterations. At each iteration, we input the training data set using four second long intervals of speech-gestures pairs stacked in batches of size 32.

3.6.5. Quantitative Evaluation on Human Pose

For a quantitative comparison of the four different speakers, we use the Probability of Correct Keypoint (PCK) metric introduced by Yang and Ramanan (2013). This metric computes the percentage of predicted keypoints that fall into a region close to the ground truth keypoint. The radius of the sphere representing the region is computed by $\alpha \cdot \max(h, w)$, where h and w refer to the height and width of a bounding box of the speaker. For our experiments we set $\alpha = 0.2$.

In addition to the PCK metric, we quantitatively compare the results using our GAN loss defined in Equation (3.4). It consists of both the discriminator loss and the generator regression loss. We illustrate our quantitative results in Table 3.3.

TV Show Speakers We compare the four speakers Ellen, Oliver, Shelly and Gruber. The quantitative results on Ellen are worse compared to Oliver because the size and the quality of the training data set is lower. Furthermore, Ellen's upper body movement is more dynamic, because she is standing during her

Category	Speaker	Loss ↓	PCK ↑
TV Show	Ellen DeGeneres	1.53	31.0
	John Oliver	1.09	**60.3**
Education	Shelly Kagan	**0.98**	40.6
	Jonathan Gruber	1.11	23.6

Table 3.3.: Quantitative Evaluation of Gestures. We evaluate the four speakers from the two categories TV Show and Education on our validation data sets. For comparison we use the Percent of Correct Keypoint (PCK) metric and the evaluation loss defined in Equation (3.4). The arrows ↓ or ↑ indicate whether a lower or higher value is better. We highlight the overall best results in **bold**.

speech. In contrast, Oliver is always sitting which causes shoulder motion to be very low.

Educational Speakers We see similar results when comparing the educational speakers. Shelly is most of the time sitting when presenting in front of the class. Therefore, the shoulder and neck positions are more static and the PCK metric is higher than the one of Gruber. Gruber is a very dynamic speaker who shows a variety of different gestures and pose positions. Consequently, the model has more problems estimating the correct keypoint positions.

TV Show vs Educational When comparing the quantitative results of the two categories we find that the PCK of the TV show speakers is slightly higher while the regression and discriminator loss is also slightly higher. We conclude that the lower PCK of the educational speaker is due to the fact that they show more variability in their presentation style. The main finding is that across categories, the results highly depend on the individual presentation and communication style of the speaker and not about the category itself.

Overall Best Our overall best results with 60.3 PCK and 1.09 Loss are achieved by the model of speaker John Oliver. We conclude that the main reasons for the high prediction accuracy are the high quantity and quality of input data and the style of the speaker. We show the qualitative results of the animated prediction for speaker John Oliver in Figure 3.9.

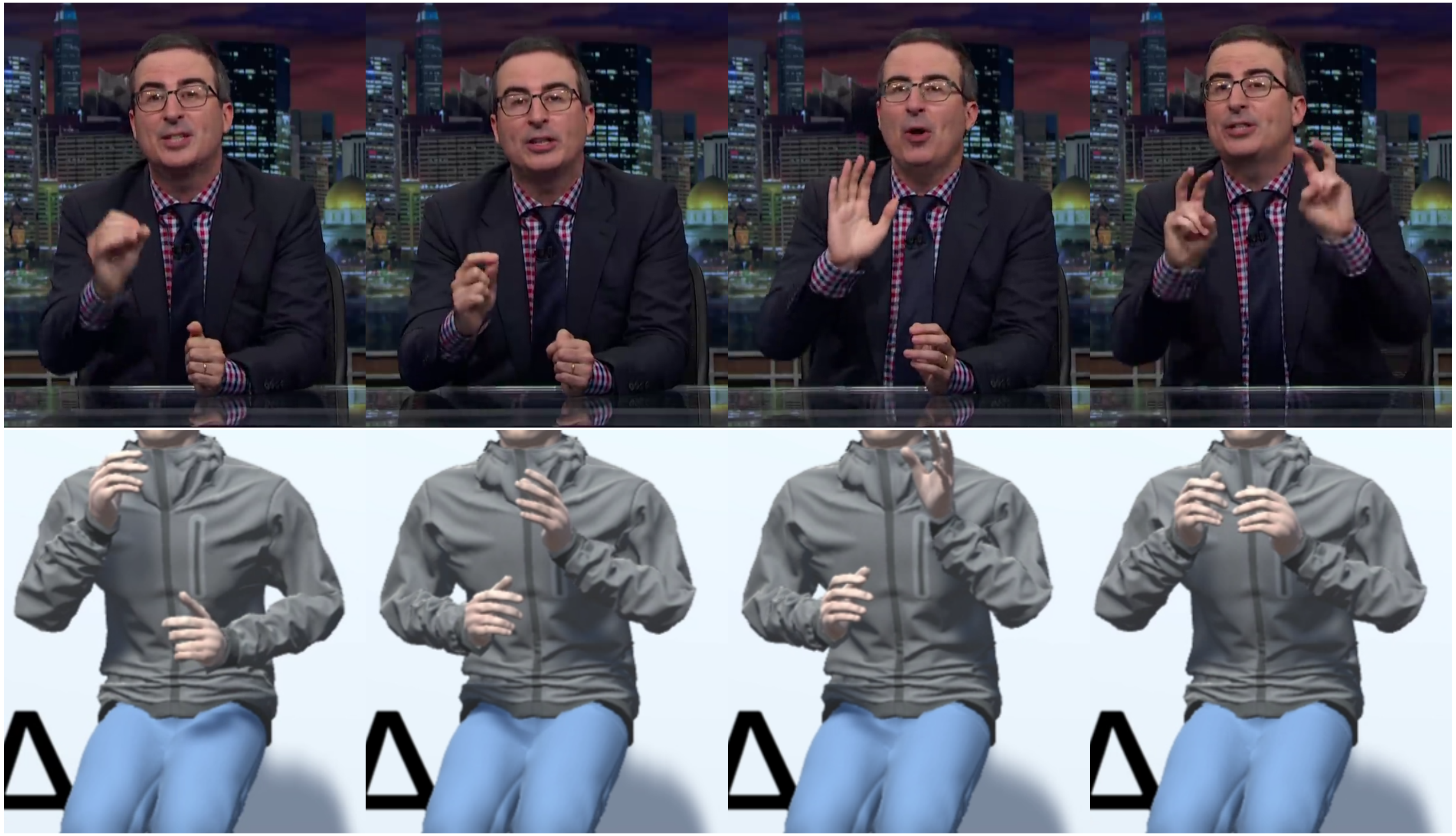


Figure 3.9.: Qualitative Results. We compare the original video (top) and the predicted gestures animated on a virtual character (bottom) of speaker John Oliver. Overall, the motion and the beat gestures are very similar. Whenever John Oliver lifts his hand, the same motion is predicted by our model. In column two and three we show that sometimes the other hand is lifted in the prediction. In column four, we show that metaphoric gestures such as "quotes" are not learned by our model.

3.7. Virtual Human Animation

Our Gesture GAN model produces 3D Human Pose sequences which we animate on virtual humans using rotation angles between bones and inverse kinematic computation. By connecting the keypoints predicted by our Gesture GAN, we create a skeletal representation of the human pose. We use this skeletal representation of the gestures to animate a virtual human in 3D space. Besides gesture animation, we also synchronize the motion with the speech and experiment with face animations to make the output look more realistic.

3.7.1. Skeleton to Virtual Human Animation

One challenge when transferring the predicted 3D Human Pose into a 3D animation is the missing information about anatomical details and ambiguities. However, we develop an approach that generates realistic 3D animations. Since we animate the gestures on a virtual human of different size and shape compared to the original speaker, we omit the information about the bone length of the generated skeleton. Instead, we only consider the rotation between the bones.

When considering the angles between two bones in the Euler angle representation, we obtain the pitch and yaw angles from the skeleton. However, we do not have any information about the roll angle. One example is the rotation of the upper arm. From the input skeleton, we can exactly determine the pitch and the yaw angle. However, we cannot determine the rotation of the bone around the axis going through the shoulder and elbow. Consequently, it is not possible to know whether the biceps muscular is oriented towards the chest, forward-looking or even stretched away from the body. To deal with this problem, we have to approximate the roll rotation of different body parts using inverse kinematics.

Inverse Kinematics To estimate the roll angle of the hand palm and of the lower arm we use the position of the knuckle keypoints. We fit a line through the knuckles and intersect it with a plane representing the hand in a neutral position. This plane represents the palm of the hand. The angle θ between vector and plane in 3D is computed by

$$\theta = \arcsin\left(\frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\|\|\mathbf{b}\|}\right), \tag{3.5}$$

where vector **a** represents the line and vector **b** the plane normal or vice versa. Since it is only possible to cover 180° of the roll angle by this intersection, we intersect the knuckle line with a second plane. The second plane lies perpendicular to the hand palm plane such that it is parallel to the middle finger when stretched. With the results of both intersections, we can cover the whole 360° roll angle of the hand palm and the lower arm.

Similarly, we approximate the roll angle of the upper arm using the direction in which the lower arm is pointing. We span a plane connecting the wrist, elbow and shoulder keypoint. The roll angle of the upper arm is approximated by the orientation of this plane with respect to the upper arm bone vector. We do not animate the roll angle of fingers, because this motion is not significant in reality.

Plausible Motion In addition to recovering the missing information about the roll angle, we also have to deal with the second problem introduced when converting the human pose representation to a virtual human animation, which is implausible motion. Although our Gesture GAN is trained to predict motion which is similar to real human motion, there exists no constraint which particularly enforces anatomically plausible motion. Hence, the predicted motion in some cases appears to be very artificial, especially when animated on a virtual human avatar. This problem is very distracting for the viewer of the animation because anatomically implausible motion is quickly noticed. To overcome this issue, we introduce motion constraints on the fingers. The motion constraints enforce that the fingers can only be bent in anatomically possible angles and directions. In addition to the motion constraints, we also introduce motion smoothing. Because the output human pose gestures are generated at 15 FPS, we interpolate the time in between. Each angle is rotated by 10 % into the direction of the target rotation at every frame update.

3.7.2. Designing the Environment

We implement our visualization on the Unity platform. We animate our gestures on the Unity Multipurpose Avatar (UMA) 2. Compared to other humanoid avatars this specific type provides many possibilities for extension and variation. The UMA avatar is also compatible with the LipSync Pro module. This module allows us to automatically generate lip animation which corresponds to speech. This is achieved using the Montreal Forced Aligner introduced by McAuliffe, Socolof, Mihuc, Wagner, and Sonderegger (2017). Besides, the LipSync Pro module also animates the eyes of the avatar.

We light the environment with a single light source placed in front of the avatar. The environment around the avatar is simple and static. It consists of a floor on which the virtual human is standing and a wall in the back. Both are implemented using a plane with a solid color surface. To synchronize the speech with the animation, we query the current audio position inside the rendering method and load the new gesture position every $\frac{1}{15}$ seconds.

3.8. Development Summary

We develop a fully automated pipeline that extracts 3D human pose information from in-the-wild videos, uses this pose information to generate gestures from speech and animates the generated gestures on virtual avatars. In the first step of gesture extraction, data cleaning plays an important role. Whenever in-the-wild video data is used, a large number of unusable sequences exist because body parts such as the hands of the speaker are hidden in the video. Once we filtered for clean data, we implement 3D human pose estimation to extract gestures from the videos. We split the process of 3D estimation into two steps. In the first step, we only extract the 2D human pose. In the second step, we project the 2D pose into 3D space. We use the estimated gestures together with the speech encoded as Mel spectrogram to train our Gesture GAN. We train our Gesture GAN to produce natural gestures for four different speakers from the genres show business and academia. After predicting the gesture, we animate them in Unity on a UMA character. We select the popular UMA character model to ensure compatibility with other plugins.

4. User Study

In this chapter, we focus on the user study we conducted to evaluate the quality of the animated gestures generated by our method introduced in Chapter 3. In the previous chapter we have obtained quantitative results by measuring metrics, *i.e.* the PCK metric and the regression loss, which compare the generated gestures to the real gestures. We measure the metrics in the human skeleton representation. However, the goal of this work is to not only generate gestures in a skeleton representation but to animate them on virtual humans. We conduct a user study to evaluate the quality of our animated gestures and to measure if they appear to be natural and realistic to humans.

4.1. Study Design

The focus of our research is to synthesize natural human body language which is coherent with speech. To measure how realistic our generated human body language is perceived, we design a study that answers our research questions.

4.1.1. Research Questions

The main research question we want to answer by designing a suitable user study is:

> *Is it possible to train a model using in-the-wild video data that generates person-specific 3D gestures of speakers that seem plausible and natural when animated on a virtual human?*

We break down our main research question into three more detailed research questions. These detailed questions function as a basis for constructing our research methodology.

1. Are we able to generate arm and hand gestures given speech, which are indistinguishable from the original gestures when animated on a character using state-of-the-art detection algorithms?

2. Are we able to generate arm and hand gestures, which are coherent with the corresponding speech?

3. How is gesture generation from audio perceived on different speakers from different genres, *i.e*. TV show speaker and academic speaker?

The third question suggests that we have to include both a TV show speaker and an academic speaker in the study. From the first two questions, we conclude that we have to design two subtasks. One subtask focuses on how realistic the generated gestures are and the second subtask determines the coherence with speech.

4.1.2. Methodology

To answer the above questions, we designed two experimental tasks. Both tasks are comparison tasks, in which the user compares two different gesture animations. In the first, we compare real and generated gestures, whereas in the second task we compare speech correlated and non-correlated generated gestures.

Generated vs Real In the first task, we present participants a series of side by side videos of two identical avatars performing different gestures for the same, 12-second long speech fragment. The participants listen to the speech while watching the side-by-side comparison. One side shows the ground truth gestures extracted from the video of the speaker and applied to the avatar. The other side shows the gestures synthesized by our model. The left-right position of the videos is randomized. The participant is asked to decide which sequence of gestures is correlated to the speech fragment.

Speech Coherence In the second task, we also present the participant with a series of videos of two identical avatars performing different gestures for a 12-second long speech fragment. In this task, gesture sequences for both avatars are synthesized. However, one of the gestures sequence, either right or left, is synthesized from the speech fragment heard by the participant, while the other avatar is performing gestures synthesized for a different, randomly selected speech fragment. The participant is asked to decide which sequence of gestures is the speech correlated gesture for the presented speech fragment.

Hypothesis For the first task, the hypothesis is that the proposed gesture synthesis is perceptually indistinguishable from the ground truth. For the second task, the hypothesis is that the speech fragment is correlated with the generated gesture.

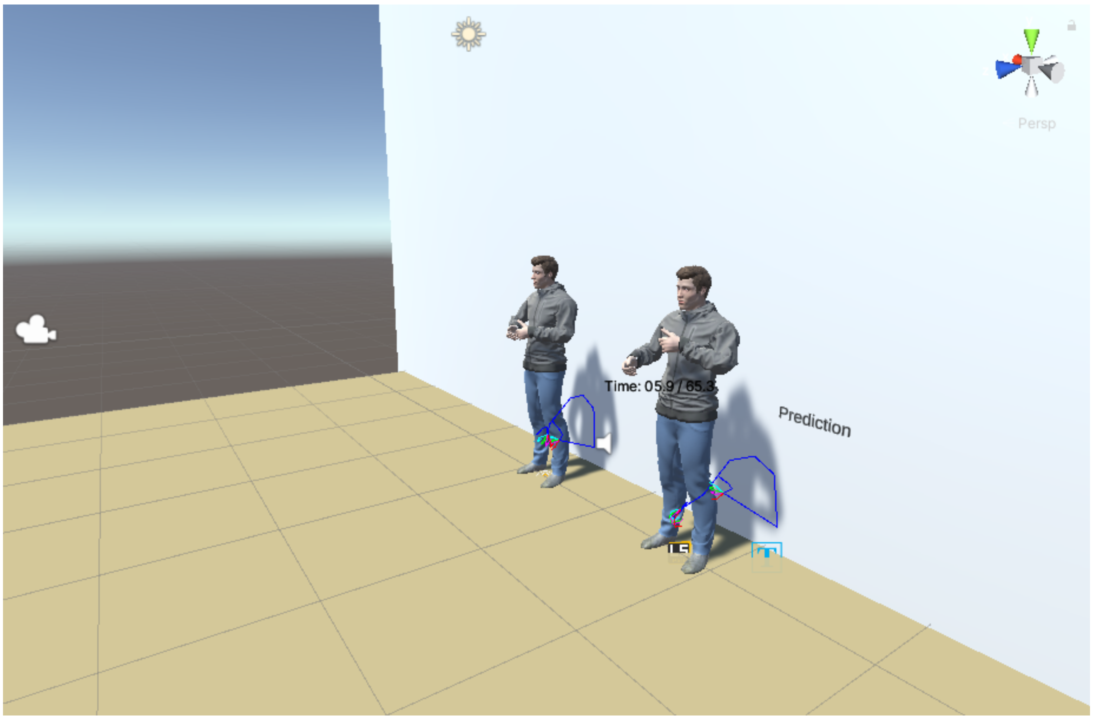


Figure 4.1.: Virtual Scene in Unity. The scene shows the two virtual characters next to each other. The light source is positioned above their heads and the camera is located in front of them. For debugging purposes, we display the source gestures in the human pose format next to each speaker.

4.2. Setting and Instruments

We decide to conduct the user study online. This is especially reasonable because the current COVID-19 pandemic makes studies where participants are physically present difficult. A general advantage of an online study is scalability in terms of the number of participants. Therefore, we create an online study and upload it to our private web server, where we make it accessible through a link. By using our private web server we make sure that data remains confidential. In addition to the online study, we also conduct a small expert study where we ask participants open questions about the quality of our animations.

We set up our web server with WordPress version 5.4 and implement our study using the Quiz And Survey Master plugin version 7.0. Since we use a private web server, the upload speed is not high enough to support that multiple participants can stream videos. Therefore, we upload the videos to a non-listed YouTube channel.

4.2.1. Environmental Setting

For the visualization of our generated gestures, we animate the UMA 2 Multipurpose Avatars using the Unity 3D game engine version 2019.3. Both animations of the side-by-side comparison run on two identical avatars. The avatars both wear blue pants and a gray jacket. The only difference between the avatars are the gestures which they perform.

Scene Setup The avatars are placed against a neutral background, with a light source in front of them. The scene is designed such that it seems that both avatars are standing inside a room with a bright floor and walls. The avatars are positioned at a distance of 1.5 meters next to each other, such that they do not interfere with each other while gesturing. Both avatars face the camera at the same 180° angle. We place two labels, A and B, besides the avatars. The labels indicate that the avatar on the left is referred to as speaker A. Likewise, we label the right avatar speaker B. The camera is positioned three meters in front of the avatars. The camera angle and shot size are adjusted such that it records the upper body of both avatars without showing the face. In addition, we place an audio source close to the heads of the virtual humans. During the gesture animation, the speech is played using the audio source. We ensure that gesture animation and speech are synchronized. The scene setup is shown in Figure 4.1.

Camera Setup While the avatars are performing the gestures in 3D space, the virtual camera records the animation in 2D at the resolution of 1000×360 and 30 frames per second. We record the avatars from the hip upwards to only capture the information relevant for the gesture evaluation and to eliminate the facial expression as a confounding factor. In addition to the frame recording, we also record the speech played through the audio source. An example of our camera setting is shown in Figure 4.2.

In both tasks, the user can hear the speech and see the avatar performing the gestures, but the face of the gesturing avatar is hidden. We consider face expression and lips synchronization a confounding factor in our study, resulting from the uncanny valley effect.

4.2.2. Instrument Design

Our experimental design is similar to the original Turing test (Turing, 2004). In the Turing test, the judges ask interlocutors A and B, a series of question. The goal of each judge is to decide which answers, A or B, are synthesized by a computer program rather than a human interlocutor. Each judge can ask a limited number of questions. The computer program passes the Turing test if the judges are unable to distinguish between the answers generated by a

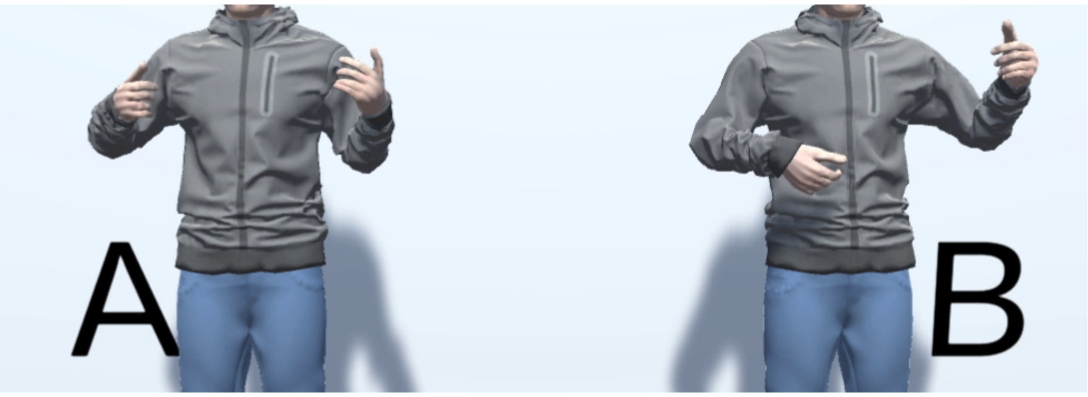


Figure 4.2.: Virtual Setup. The gestures are animated on two identically looking avatars, referred to as A and B.

Phase	Task	Speaker	# Seq	Time
Preparation	Generated vs Real	Oliver	4	48 sec
	Generated vs Real	Gruber	4	48 sec
Study	Generated vs Real	Oliver	20	4 min
	Generated vs Random	Oliver	10	2 min
	Generated vs Real	Gruber	20	4 min
	Generated vs Random	Gruber	10	2 min

Table 4.1.: Scientific Study Procedure. This table gives an overview about the structure of our user study. We separate the user study in a preparation and the actual study phase. During the study phase, two different tasks for the speakers John Oliver and Jonathan Gruber are performed.

computer and the answers generated by a human interlocutor. A system that passed the Turing test is therefore a system for which the judges were unable to establish a statistically significant difference between the synthesized answers and ground truth answers. Similarly, our system passed the non-verbal version of the Turing test, if after a limited number of attempts, the human judges are unable to establish a statistically significant difference between the sequence of gestures generated by our system and the ground truth gestural sequence.

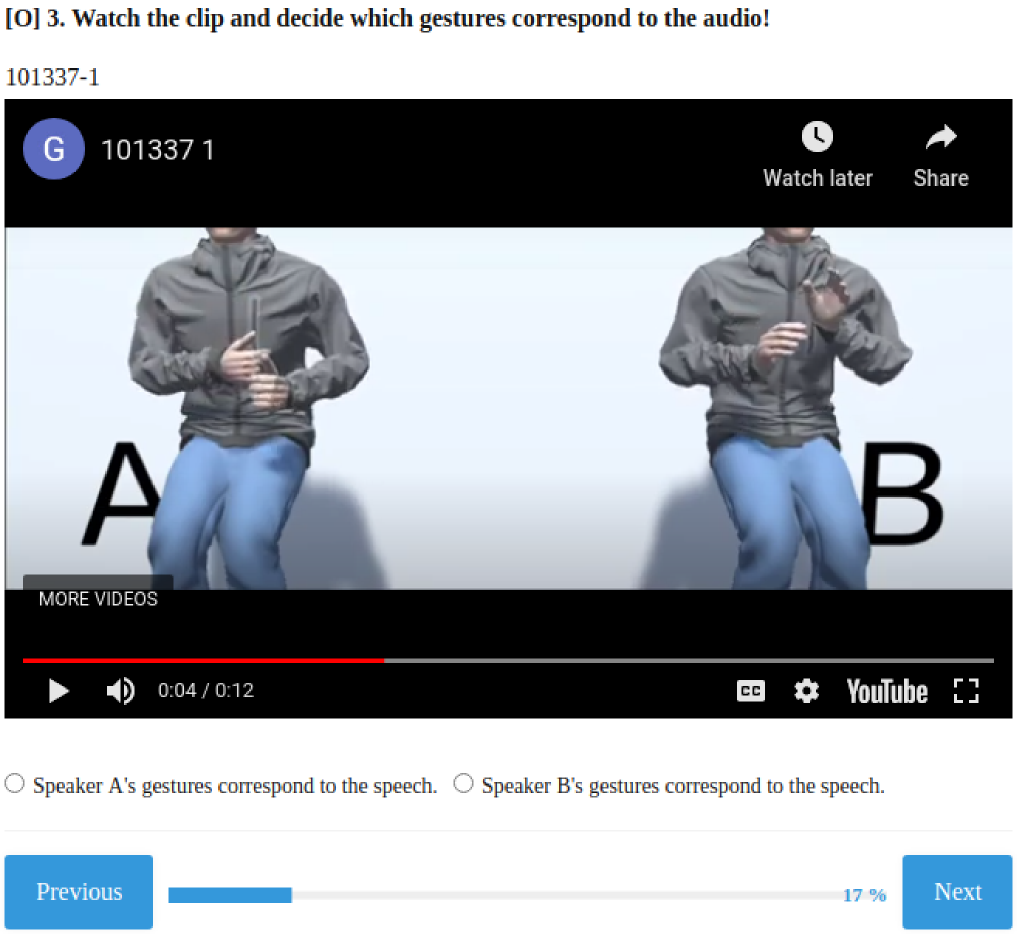
4.3. Procedure

For our user study, we select the speakers John Oliver and Jonathan Gruber to cover one TV show speaker and one educational speaker. Through our study, the participants are not informed that they have to compare the sequences of two different speakers. However, they will notice from the speech they are listening to, that the animations are generated for two different speakers.

Sequence Sampling To generate the pairs of ground truth and prediction, we evaluate our model on the test data set. The test data is not used during training and validation. Therefore, the test audio sequences are completely new to our models. We randomly shuffle the test sequences and choose the first 24 intervals of each speaker for the prediction vs ground truth comparison. Similarly, we randomly sample 10 pairs of generated speech corresponding and generated speech non-corresponding sequences.

Study Design We choose the number of sequences we show to the participants of the user study such that the task does not become exhausting. Our study is designed such that it takes 17 minutes to complete it when assuming a minimal decision time of 3 seconds after watching the 12-second sequences. This leads to an estimated average completion time of about 25 minutes. We structure the study into two main parts: a short preparation of 8 sequences and the actual study consisting of 60 sequences. An overview of the structure is shown in Table 4.1. At the beginning of the study, the participants are presented with the description of the study and asked to confirm their willingness to participate. Then, the preparation phase consisting of 4 sequences for each speaker follows. During the preparation phase, the participants only work on the Generated Vs Real task. After completing the preparation, the main study follows. The main study starts with 30 sequences of speaker John Oliver and ends with 30 sequences of speaker Jonathan Gruber. The first 20 sequences of each speaker correspond to the Generated Vs Real task and the last 10 sequences of each speaker correspond to the Generated Vs Generated Random task. However, participants are not told which task they are currently working on. Instead, they are only told to select the avatar which performs the most realistic gestures. After completing all sequences the results are submitted and a conclusion page is shown.

Comparing Sequences For each sequence, participants are faced with a two-option single forced choice. The participants are asked to watch the sequence and decide which gestures correspond to the speech. An example comparison sequence is shown in Figure 4.3. The participants have the chance to watch the sequences multiple times and they cannot continue to the next sequence without selecting one of the two options. However, we do not force them to watch the full video before selecting an option. At the bottom of each page, we show the participants the progress they made.



[O] 3. Watch the clip and decide which gestures correspond to the audio!

101337-1

○ Speaker A's gestures correspond to the speech. ○ Speaker B's gestures correspond to the speech.

Figure 4.3.: User Study Example Comparison. The participants in our study are asked to decide which animated gestures they perceived as more natural. The gestures are animated on two identically looking avatars, referred to as A and B. The comparison video is embedded in our user study application. At the bottom we show the progress for each participant.

4.4. Study Participants

For our study, we recruited 113 participants on Amazon Mechanical Turk (MTurk) platform. Each participant received compensation of 3 $. Out of the 113 responses, 101 are valid. We consider a response to be valid if the time to complete a survey is above the minimum time of 17 minutes and below 35 minutes.

Hiring the Participants We implemented the work assignment iteratively. First, we only created a work batch for a single MTurk work. After receiving his feedback, we improved the task description and created another batch for 10 workers. Although we improved the study throughout the process, the main task always remains the same to ensure comparability. The changes were design improvements, to improve the quality of the study. We switched between creating batches for 10 and 30 workers until we reached a total of over 100 participants. We only hired MTurk workers with the qualification type masters to ensure that the responses are of high quality. We set the assignment duration to 40 minutes and provided a link to access the user study on our webserver.

Demography From the responses received from the MTurk Masters workers, we conclude that the participants are fluent in English. This was also necessary to complete our study. MTurk workers come from different regions of the world and the age groups are diverse. We did not collect any demographic data or any identifiable data or information about participants' prior computer-related experience. All collected data was anonymous at all times.

4.5. Results and Discussion

Once all participants have completed the user study, we evaluate the results and draw our conclusions. We present quantitative results consisting of percentages of how often participants selected the gesture animation from a certain source.

4.5.1. Results

We show the results of the two quantitative experiments and also one qualitative experiment in which we performed an unstructured interview. The quantitative results are summarized in Table 4.2. A bar chart illustrates the results in Figure 4.4.

Generated vs Real In this task we were determining which gestures are generated and which are ground truth. The participants mistook synthesized gestures as ground truth 52.5 % of the time on average, with variance $\sigma^2 = 1.0\,\%$.

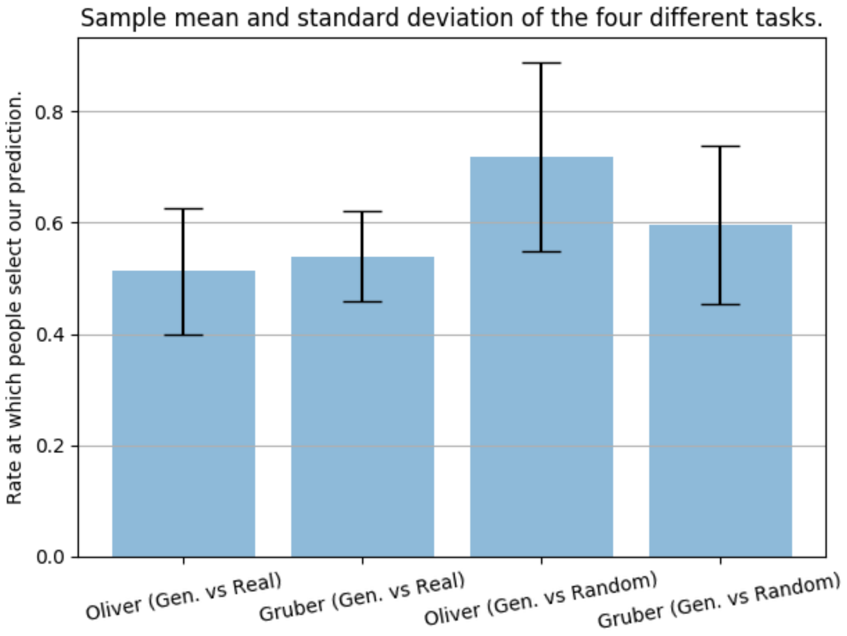


Figure 4.4.: Quantitative Results. This bar chart illustrates the quantitative results of our user study. The four bars refer to the four different tasks the participants completed during the user study for the speakers John Oliver and Jonathan Gruber. The height of the bar represents the rate at which the participants selected the gestures predicted by our model. The vertical intervals at the top of each bar indicate the standard deviation across the participants.

	Result	**Oliver**	**Jonathan**
Task 1	Participants selected the prediction of our model	51.2 ± 1.4	53.8 ± 0.6
Task 2	Participants selected coherent prediction-audio	71.2 ± 3.1	58.8 ± 2.1

Table 4.2.: User Study Results. We compare gesture generation for two speakers – John Oliver and Jonathan Gruber. In the first row, we show the mean and the variance percentage of how often participants decided that our model generates more realistic gestures than the ground truth. The second row illustrates the mean and the variance percentage of how often participants chose the gesture animation generated for the presented speech fragment over the gesture animation generated for a random speech fragment.

The difference between selecting ground truth and synthesized gestures were not significant. The results for individual speakers were not significant either, with participants selecting synthesized gesture over the ground truth, in average, 51.2 % times for John Oliver, with variance $\sigma^2 = 1.4\,\%$, and 53.8 % times for Professor Jonathan Gruber, with variance $\sigma^2 = 0.6\,\%$.

Generated vs Random In this task we were determining which gesture was generated for a given speech fragment. The participants chose the gesture sequence that was indeed generated for the given speech fragment on average 65 % of times, with variance $\sigma^2 = 3.55\,\%$. Interestingly, the participants were significantly more likely to point at a correct gesture sequence in case of gestures generated for John Oliver, selecting it correctly 71.2 % of times, while correctly identifying gesture sequence, on average, in 58.8 % of times for gestures generated for professor Jonathan Gruber. In both cases considered individually, the experiment showed a significant difference between the gestures generated for the heard fragment of speech and gestures generated for a randomly selected fragment of speech.

Qualitative Evaluation We received qualitative results from conducting unstructured interviews. The main findings are listed below.

- Overall, the generated gestures are only distinguishable from the original gestures, if the speaker performs iconic or metaphoric gestures.
- The animation quality still needs to be improved, because it sometimes seems unnatural.
- The hand position is most of the time in a neutral position. To be perceived natural, more stretching and bending needs to be animated.
- Larger body motion such as arm movement is perceived good. Smaller motion such as finger movements are perceived worse.
- Gestures performed by John Oliver seem to be more exaggerated and better correlated with his speech than the gestures of professor Jonathan Gruber.

4.5.2. Discussion

Based on the quantitative results, we conclude that our system passed the non-verbal Turing test since the participants were unable to determine whether the gesture animations were generated or captured. It is important to note that our study does not attempt to establish that our gesture synthesis creates gesture sequences that are perceptually superior to the ground truth.

Null Hypothesis We describe our null hypothesis:

> The quality of the generated and the captured gesture are of the same quality and therefore one is not able to select the captured one which is corresponding to the speech.

Consequently, we define our null hypothesis:

$$\mathcal{H}_0 : \mu = 0.5, \tag{4.1}$$

where μ represents the rate at which a person is fooled by our generated sequences.

Despite a relatively large and diverse sample, the participants are unable to reject the null hypothesis for task Generated Vs Real, while at the same time being able to reject the null hypothesis for task Generated Vs Random. In other words, the closer our system is to ground truth, the more difficult it is to reject the null hypothesis for the task Generated Vs Real. Rejecting the null hypothesis in task Generated Vs Real would mean that our system generates gestures that are perceptually inferior as compared to the ground truth. However, we cannot conclude the opposite from not rejecting the null hypothesis. Nevertheless, we have shown that the difference between the synthesized gestures and ground truth gestures must be small.

Inability to reject the null hypothesis for task Generated Vs Real could be a result of either the difference between the two groups being too small for our sample size to be significant or it could be a result of the participants not paying attention to the study and selecting right or left videos at random. However, rejection of the null hypothesis for task Generated Vs Random serves as a validation for the claim that our system passed the non-verbal Turing test, because finding a significant difference in task Generated Vs Random suggests, although does not guarantee, that the participants were indeed paying attention to the presented videos and were not selecting answers A or B at random.

Types of Gestures We also took a closer look at each specific user's decision in judging the generated vs ground truth gesture animations. The analysis revealed that participants were fooled by our gesture prediction in those gesture sequences, in which the ground truth does not include an iconic gesture. Whenever the speaker performs an iconic gesture, subjects were able to identify the ground truth. This is an interesting result since it suggests that the fact that our system does not model the speech semantics and therefore does not explicitly generate iconic gestures might provide our users with a winning strategy: searching for iconic gestures in the speech and identifying those. It remains an open question whether or not a sufficiently large training dataset would result in the system encoding the correlation between the semantics of speech and iconic gestures.

Qualitative Results Another qualitative observation is the difference in the correct recognition of gestures generated for a given speech fragment between John Oliver and Jonathan Gruber. We observe that ground truth gestures of John Oliver, a comedian, are more pronounced and exaggerated. Possibly, the gestures performed by John Oliver, an experienced actor, are better correlated with his speech than the gestures of professor Jonathan Gruber. If that is true, it may be possible to create a model that generates more exaggerated gestures for a given speaker than the speaker's own gestures.

Our Results vs Related Work A very similar user study was conducted by Ginosar et al. (2019). They hired 300 participants on MTurk and also created tasks in which they compare two alternatives. Furthermore, they also forced the participants to choose one of the two options. In contrast to our virtual avatar animation, they only showed the skeletal gesture videos to the participants. Instead of comparing a TV show and an academic speaker, they compared two TV show speakers against each other. When comparing the predicted gestures against the original gestures of John Oliver 27.8 ± 3.9 % selected the predicted gestures. For the same task, 33.1 ± 4.2 % selected the predicted gestures of Seth Meyers. Compared to our results, this percentage is very low. In our study, over 50 % selected our generated gestures for both speakers which indicates that the quality of our model is better. In the second task, they compared speech uncorrelated generated gestures with the original gestures. In this experiment, 29.1 ± 3.7 % selected the uncorrelated gestures of John Oliver and 34.3 ± 4.4 % selected the uncorrelated gestures of Seth Meyers. The results of this experiment are similar to what we observe in our study. However, the results show that Ginosar et al. (2019) were not able to show a statistically significant difference between uncorrelated and correlated gesture prediction. Our model can produce significantly different results for uncorrelated speech and our predicted gestures are perceived as natural as the original gestures.

4.6. User Study Summary

To conclude this chapter, we summarize the answers to our research questions stated in Section 4.1.1. By conducting the user study, we have shown that our generated arm and hand gestures are indistinguishable from captured original gestures. In our study, participants chose our generated gestures to be more realistic at a rate of about 50 %. Furthermore, about 65 % of times the participants of the study decided that the generated gestures have a higher correlation to speech that randomly selected gestures. This states that the generated gestures are natural and correspond to speech. When considering the different genres, *i.e*. TV show and academia, we find that our generated gestures are plausible in both genres. Besides, we find that the correlation between speech and gestures is easier to detect for TV speaker John Oliver. We assume the reason is that John Oliver is trained in performing vivid and meaningful gestures.

5. Lessons Learned

In this chapter, we focus on the lessons we learned within the scope of this thesis. We distinguish between the phases of the literature review, the development of our method and conducting the user study. In the end, we highlight the five most important lessons within each phase.

5.1. Background and Related Work

The background research allowed us to get a comprehensive overview of human non-verbal communication and human gesture categories. It also embraces understanding the fundamentals of the methods which are currently used in other work. It is important to know the basic machine learning techniques and how they are applied to gesture generation and other applications.

Once the basics behind generative speech-gesture modeling are understood, it is important to get a more detailed view of state-of-the-art research and know how the problem is approached. Besides current work, also previous work needs to be considered in order to get the whole picture of the research in non-verbal communication. This allows to for example understand why in the past deterministic speech-gesture models were developed. Now generative models are used, because of the rise in data availability and deep learning algorithms.

After analyzing the state-of-the-art, different research can be compared by pointing out the advantages and disadvantages of certain approaches. This makes it possible to take the best out of different work and enhances success. If something has already worked in the past, chances are higher that it also works in a different setting.

Another important aspect when comparing different related work is to also implement different approaches and see how they work on a specific problem set. In our research survey, we found that certain hand pose estimation algorithms performed completely different compared to what was expected from comparing benchmark results. The reason is that especially in machine learning results vary between dataset and domains. We found that some algorithms only work a certain domain given a specific setting. Therefore, it is important to implement methods discovered during the literature survey.

5.2. Development

Whenever developing a machine learning algorithm it is essential to start with a very simple example and a small dataset. In the first step, the model can be overfitted. The goal is to get an idea about if the model could work as early as possible. Once the algorithm performs as expected on a small task, it is always possible to add complexity to the task. When training our GAN model we found that we can already detect if the implementation could work by running it on a single speaker with ten minutes of training data.

A less complex model and simple example also support the testing of different hyper-parameters. When selecting a complex setting the computation time is high, the feedback takes long and the effect from changing the model might not be visible. We found that it is best to create a lightweight setting for hyper-parameter and loss function comparisons. When improving 3D hand estimation, we visualize the current prediction and check after 50 images how the model is performing, compared to other settings. This allows us to adjust the model quickly. We never process the whole 20 hours of video data or compute complex post-processing steps. Similarly, we train our GAN model with a different loss function for a few iterations and check the results early. We perform many validation rounds with several validation sequences to get fast and comprehensive feedback.

We find that it is important to focus on creating a running prototype as early as possible, even though it might not be perfect. Otherwise, we spend too much time and focus on unimportant details. We found out that some implementations are not needed. For example, we design the GAN to work with hand and arm keypoints only, instead of supporting a high number of different body keypoints including feet. Supporting all possible human pose keypoints is not necessary because some keypoints do not exist in the dataset.

Furthermore, we learned not to stick to one approach for a long period of time if it is not working. It is recommended to test an alternative approach as early as possible. First, we tried to post-process the estimated 2D gestures and lift them into 3D space. Later we found that it is better to extract 3D gestures from video in the first place and also run the gesture generation in 3D. Spending time on making the first approach work, in hindsight, was a waste of time.

5.3. User Study

When designing the user study, we found that it is important to remove confounding factors in the virtual environment such as artifacts. Otherwise, the participants will get distracted and judge the whole performance of the gesture prediction based on small visualization issues.

When hiring participants on Amazon MTurk, it is important to select highly qualified workers only, despite the additional cost. This ensures that the percentage of people who will do the work seriously is much higher. Despite hiring workers with a good track record, it is also important to create a method in the study to check if the participants are taking the task seriously. In our case, we included gesture sequences with iconic gestures where the correct answer was easy to find. Additionally, we measure the time each participant needed to complete the user study.

By evaluating the responses from our user study, we found that our user study took too long and the tasks were repetitive. Many participants complained after finishing the work. According to the feedback, the study should only take about half the time which would be ten minutes in our case. To receive the same number of responses, we would only need to double the number of participants from 100 to 200 in our case.

Whenever available, use standardized methods and existing tools for implementing the user study. We first stored the video sequences on our own server infrastructure. Then we found out that the upload speed of our server connection is too low to support multiple participants at the same time. Consequently, we uploaded the videos to YouTube and only embedded them in our study environment. For the design of our study, we used an existing WordPress survey plugin. This allowed us to inherit the basic functionality and a good design. Furthermore, typical user study functions are already implemented including session management and logging to the database.

Finally, it is essential to have the design of the user study and the research questions early in mind. This should be known during implementation, such that the right output formats and appropriate visualization is developed.

We highlight the five most important lessons learned for each phase:

Related Work

- Know the state-of-the-art before developing an own approach.
- Understand the principles behind state-of-the-art methods.
- Identify the advantages and disadvantages of different approaches to better classify related work.
- Implement proposed methods to see how they work in a different setting.
- Select methods which already had some success on the same or other application in the past.

Development

- Start with a very simple and small example when developing a machine learning algorithm.
- Minimize computation time when comparing different hyperparameters.

- Create working prototypes as early as possible.
- If a certain approach does not seem to work, switch to another path as soon as possible.
- Do not always think about details and future changes during development.

User Study

- Select participants with a proven track record in taking the study seriously.
- Create a method to check if participants take the task seriously.
- Make the user study as short and less repetitive as possible.
- Use standardized methods and existing tools whenever available.
- Have the design of the user study and the research questions early in mind.

6. Conclusion and Future Work

In this chapter, we summarize the most important findings of this thesis. We also explain future ideas and applications of our work. Our focus is on how the predicted gestures can be improved further.

6.1. Conclusion

The communication between virtual characters and humans is getting more important in recent days. Reasons include the broad usage of online learning environments, virtual assistance and communication tools. However, the creation of body language for a virtual character is still an open research topic that faces many challenges.

In this work, we improved state-of-the-art speech to gesture translation models. In contrast to existing video approaches, we predict the gestures in 3D which allows us to animate them on virtual characters in a 3D environment. Furthermore, we implemented a novel 3D human pose estimation pipeline, which allows us to train speaker-specific gesture generation models from in-the-wild video data. Compared to previous work, we implement the non-deterministic relationship between speech and gestures using a generative model.

We found that it is possible to animate a 3D virtual character by using 2D video sequences to train a GAN model. By implementing 3D human pose estimation we managed to reconstruct the gestures of the speaker in 3D space. This creates the opportunity to train models using large-scale video datasets which allows the model to learn from a variety of different gestures.

Besides, we completely automate the extraction of 3D gestures in form of human pose information from video. Hence, we can use unsupervised learning to train our model. We save the time needed to manually create annotations which are costly to compute using motion capture. This is a breakthrough compared to previous work which relies on motion capture data. Hence, previous work could only predict gestures for a single speaker and the variety of different gestures was low.

By implementing a motion discriminator, we can generate realistic gestures. Furthermore, the motion discriminator forces the model to predict speaker-specific gestures as observed in the original input videos. When the model is applied to a different speaker, the generated gestures are still similar to the ones of the original speaker. This shows that we can capture personal styles of body language.

By conducting the user study, we found that the gestures predicted by our model are natural and correspond to speech. 51.2 % and 53.8 % of participants selected that our generated gestures seem more natural than the original gestures for speakers John Oliver and Jonathan Gruber, respectively. Furthermore, when comparing speech correlated and uncorrelated gestures, participants identified the speech correlated gestured correctly in 72.2 % of times for speaker John Oliver and 58.8 % for speaker Jonathan Gruber. This shows that our method can be applied successfully to speakers from different genres. The results show that especially rhythmic, beat gestures are predicted accurately.

6.2. Future Work

When extracting the 3D human pose from video data, the most difficult part is the estimation of the hand keypoints. The reasons are that the fingers are the smallest body part which is extracted and most of the time parts of the hand are occluded. In our approach, we tackle this problem by creating a heatmap with probabilities for each keypoint. We then fit a 3D hand model to the 2D heatmap prediction. The procedure can be improved by creating a large-scale hand dataset, similar to the dataset created by Mueller et al. (2018). This dataset already contains over 330,000 hand images with 3D keypoint ground truth. However, this dataset is missing images in which both hands of a person is visible. Speakers often clasp their hands and poses in which hands occlude each other also need to be covered by the 3D hand pose estimation model. Besides using a GAN for synthesizing images showing two hands, 3D data could be created using the Leap Motion sensor (Ultraleap, 2020). Another possibility would be to utilize motion tracking using gloves such as the Manus Prime 2 (Manus-VR, 2020) to create a 3D hand pose dataset. Once the dataset is created, the efficient ESPNetv2 encoder architecture by Mehta, Rastegari, Shapiro, and Hajishirzi (2019) can be used to develop a 3D hand pose estimation model.

A Recurrent Neural Network (RNN) could be implemented to consider long time sequences when predicting the gestures. In our approach, we process four seconds of speech when predicting the corresponding gestures. A longer history is useful to get more information about the complete sentence or paragraph the speaker is talking about. To carry long term information an LSTM or GRU cell can be implemented at the UNet bottleneck. At this position, the RNN processes encoded speech information.

Another idea would be to add audio and semantic features. Currently, we input the Mel spectrogram of the raw audio speech to the gesture generation model. Semantic features such as a transcript could improve the learning of iconic, metaphoric and deictic gestures. Besides semantic features, prosodic features such as MFCC could be added. Furthermore, the raw audio could be preprocessed such that the voice of the speaker is extracted and background noise is removed. Typical background noise includes laughs in the TV show genre and the voice of the audience in the academic genre.

Future work could also extend the gesture model such that it also generates facial expressions and complete body movement. Our gesture model focuses on shoulder, arm and hand motion. By adding more pose keypoints, facial expressions and the position of the speaker could also be predicted by the model. As with gesture prediction, lip synchronization also depends on the prosody of the input speech. The complete virtual character could be animated by speech.

A very interesting application for testing the generated gestures would be the animation of a humanoid robot. Our validation includes the animation of an avatar in virtual reality. To also animate a robot the encoding needs to be adapted such that it coincides with the motion Degrees Of Freedom (DOF) of the robot.

Another possibility for future research is to design and train a cross-speaker model. Our current model predicts speaker-specific gestures. In case it is not possible to obtain training data for a specific speaker, a cross speaker model would be able to predict gestures from the voice of a speaker not previously heard. According to Ginosar et al. (2019), using speech from a different speaker results in inaccurate gesture predictions. Hence, a large-scale speech-gesture dataset is needed which combines speakers from various genres.

It would also be interesting to investigate gesture from speakers of different disciplines. We have shown that we are able to generate plausible gestures for TV show and academic speakers, but our data-driven approach allows us to train a speaker model for any discipline in which video data exists. A promising additional experiment could compare speakers from different cultural backgrounds. This experiment could show if our model is able to detect minor differences in the cultural aspects of body language.

In summary, future areas of work include improving the accuracy of 3D hand pose estimation and the development of cross-speaker models. An interesting application would be to animate the generated gestures on a humanoid robot. In our work, we have shown that we accurately predict human body language in form of gestures. We achieve this by training a speaker-specific model that generates gestures from speech. When animating the gestures on a virtual character, people were not able to distinguish the generated from the original gestures.

Bibliography

Alexanderson, S., Henter, G., Kucherenko, T., & Beskow, J. (2020). Style-controllable speech-driven gesture synthesis using normalising flows. *Computer Graphics Forum*, 487–496.

Aurand, A. M., Dufour, J. S., & Marras, W. S. (2017). Accuracy map of an optical motion capture system with 42 or 21 cameras in a large measurement volume. *Journal of Biomechanics*, 237–240.

Bergmann, K. & Kopp, S. (2009). Gnetic – using bayesian decision networks for iconic gesture generation. In *Intelligent virtual agents* (pp. 76–89).

Cao, Z., Hidalgo Martinez, G., Simon, T., Wei, S., & Sheikh, Y. A. (2019). Openpose: realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Cao, Z., Simon, T., Wei, S.-E., & Sheikh, Y. (2017). Realtime multi-person 2d pose estimation using part affinity fields. In *Ieee conference on computer vision and pattern recognition (cvpr)*.

Chen, Y., Wang, Z., Peng, Y., Zhang, Z., Yu, G., & Sun, J. (2018). Cascaded Pyramid Network for Multi-Person Pose Estimation. In *Ieee conference on computer vision and pattern recognition (cvpr)*.

Chiu, C.-C. & Marsella, S. (2011). How to train your avatar: a data driven approach to gesture generation. In *Intelligent virtual agents* (pp. 127–140).

Chiu, C.-C. & Marsella, S. (2014). Gesture generation with low-dimensional embeddings. *International Conference on Autonomous Agents and Multiagent Systems*, 781–788.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014, October). Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Conference on empirical methods in natural language processing (emnlp)* (pp. 1724–1734).

Ferstl, Y. & McDonnell, R. (2018). Investigating the use of recurrent motion modelling for speech gesture generation. In *International conference on intelligent virtual agents* (pp. 93–98).

Ferstl, Y., Neff, M., & McDonnell, R. (2019). Multi-objective adversarial gesture generation. In *Motion, interaction and games*.

Ge, L., Ren, Z., Li, Y., Xue, Z., Wang, Y., Cai, J., & Yuan, J. (2019). 3d hand shape and pose estimation from a single rgb image. In *Ieee conference on computer vision and pattern recognition (cvpr)*.

Geiger, A., Ziegler, J., & Stiller, C. (2011). Stereoscan: dense 3d reconstruction in real-time. In *Ieee intelligent vehicles symposium* (pp. 963–968).

Ginosar, S., Bar, A., Kohavi, G., Chan, C., Owens, A., & Malik, J. (2019, June). Learning individual styles of conversational gesture. In *Ieee conference on computer vision and pattern recognition (cvpr)*. IEEE.

Girshick, R., Radosavovic, I., Gkioxari, G., Dollár, P., & He, K. (2018). Detectron. https://github.com/facebookresearch/detectron.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... Bengio, Y. (2014). Generative adversarial nets. In *Proceedings of advances in neural information processing systems* (pp. 2672–2680).

Graves, A., Mohamed, A., & Hinton, G. (2013). Speech recognition with deep recurrent neural networks. In *2013 ieee international conference on acoustics, speech and signal processing* (pp. 6645–6649).

Greshko, M. (2020). Meet sophia, the robot that looks almost human. Retrieved from https://www.nationalgeographic.com/photography/proof/2018/05/sophia-robot-artificial-intelligence-science/

Hahnloser, R., Sarpeshkar, R., Mahowald, M., & Douglas, R. (2000). Digital selection and analog amplification co-exist in an electronic circuit inspired by neocortex. *Nature*.

Hanson-Robotics. (2020). Sophia. Retrieved from https://www.hansonrobotics.com/sophia/

Hasegawa, D., Kaneko, N., Shirakawa, S., Sakuta, H., & Sumi, K. (2018). Evaluation of speech-to-gesture generation using bi-directional lstm network. In *International conference on intelligent virtual agents* (pp. 79–86).

Henter, G. E., Alexanderson, S., & Beskow, J. (2019). MoGlow: Probabilistic and controllable motion synthesis using normalising flows. *ArXiv*.

Herculano-Houzel, S. (2009). The human brain in numbers: a linearly scaled-up primate brain. *Frontiers in Human Neuroscience*, 31.

Hochreiter, S. & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 1735–1780.

Huang, C.-M. & Mutlu, B. (2013). Modeling and evaluating narrative gestures for humanlike robots. In *Robotics: science and systems*.

Ionescu, C., Papava, D., Olaru, V., & Sminchisescu, C. (2014). Human3.6m: large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1325–1339.

Isola, P., Zhu, J., Zhou, T., & Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In *Ieee conference on computer vision and pattern recognition (cvpr)* (pp. 5967–5976).

Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., ... Fitzgibbon, A. (2011). Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual acm symposium on user interface software and technology* (pp. 559–568).

Jebara, T. & Meila, M. (2006). Machine learning: discriminative and generative. *The Mathematical Intelligencer*, 67–69.

Joo, H., Simon, T., & Sheikh, Y. (2018). Total capture: a 3d deformation model for tracking faces, hands, and bodies. In *Proceedings of the ieee conference on computer vision and pattern recognition*.

Karras, T., Aila, T., Laine, S., & Lehtinen, J. (2017). Progressive growing of gans for improved quality, stability, and variation. *ArXiv*.

Kingma, D. P. & Ba, J. (2015). Adam: a method for stochastic optimization. *International Conference on Learning Representations (ICLR)*.

Kirk, A. G., O'Brien, J. F., & Forsyth, D. A. (2005). Skeletal parameter estimation from optical motion capture data. In *Ieee conference on computer vision and pattern recognition (cvpr)* (pp. 782–788).

Kocabas, M., Athanasiou, N., & Black, M. J. (2020). Vibe: video inference for human body pose and shape estimation. In *Ieee conference on computer vision and pattern recognition (cvpr)*.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). 2012 AlexNet. *Proceedings of Advances in Neural Information Processing Systems*, 1–9.

Kucherenko, T., Hasegawa, D., Henter, G., Kaneko, N., & Kjellström, H. (2019). Analyzing input and output representations for speech-driven gesture generation. *International Conference on Intelligent Virtual Agents*.

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Ieee conference on computer vision and pattern recognition (cvpr)*.

Maass, W. (1997). Networks of spiking neurons: the third generation of neural network models. *Neural Networks*, 1659–1671.

Manus-VR. (2020). The manus glove. Retrieved from https://www.manus-vr.com/

McAuliffe, M., Socolof, M., Mihuc, S., Wagner, M., & Sonderegger, M. (2017). Montreal forced aligner: trainable text-speech alignment using kaldi. In *Interspeech conference*.

Mehta, S., Rastegari, M., Shapiro, L., & Hajishirzi, H. (2019). Espnetv2: a light-weight, power efficient, and general purpose convolutional neural network. In *Ieee conference on computer vision and pattern recognition (cvpr)*.

Moeslund, T. B. & Granum, E. (2001). A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 231–268.

Moeslund, T. B., Hilton, A., & Krüger, V. (2006). A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 90–126.

Mueller, F., Bernard, F., Sotnychenko, O., Mehta, D., Sridhar, S., Casas, D., & Theobalt, C. (2018). Ganerated hands for real-time 3d hand tracking from monocular rgb. In *Ieee conference on computer vision and pattern recognition (cvpr)*.

Nguyen, T., Nguyen, C., Nguyen, D. T., Nguyen, D., & Nahavandi, S. (2019). Deep learning for deepfakes creation and detection. *ArXiv*.

Panteleris, P., Oikonomidis, I., & Argyros, A. A. (2018). Using a single rgb frame for real time 3d hand pose estimation in the wild. *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 436–445.

Pavllo, D., Feichtenhofer, C., Grangier, D., & Auli, M. (2019). 3d human pose estimation in video with temporal convolutions and semi-supervised training. In *Ieee conference on computer vision and pattern recognition (cvpr)*.

Pfungst, O. (1911). *Clever hans:(the horse of mr. von osten.) a contribution to experimental animal and human psychology*. Holt, Rinehart and Winston.

Pollefeys, M., Nistér, D., Frahm, J.-M., Akbarzadeh, A., Mordohai, P., Clipp, B., … Towles, H. (2008). Detailed real-time urban 3d reconstruction from video. *International Journal of Computer Vision*, 143–167.

Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., … Vesely, K. (2011). The kaldi speech recognition toolkit.

Rasouli, A. & Tsotsos, J. (2019). Autonomous vehicles that interact with pedestrians: a survey of theory and practice. *IEEE Transactions on Intelligent Transportation Systems*, 1–19.

Remondino, F. (2004). 3-d reconstruction of static human body shape from image sequence. *Computer Vision and Image Understanding*, 65–85.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: convolutional networks for biomedical image segmentation. In *International conference on medical image computing and computer-assisted intervention (miccai)*.

Rössler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., & Niessner, M. (2019). Faceforensics++: learning to detect manipulated facial images. In *Ieee international conference on computer vision (iccv)* (pp. 1–11).

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., … Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 211–252.

Schmitz, A. (2012). *A primer on communication studies*.

Schneider, K. G., Hempel, R. J., & Lynch, T. R. (2013). That "poker face" just might lose you the game! the impact of expressive suppression and mimicry on sensitivity to facial expressions of emotion. *Emotion*, 852.

Semcon. (2020). Self driving car that sees you. Retrieved from https://semcon.com/smilingcar/

Seyama, J. & Nagayama, R. S. (2007). The uncanny valley: effect of realism on the impression of artificial human faces. *Presence: Teleoperators and virtual environments*, 337–351.

Sigal, L., Balan, A., & Black, M. J. (2010). HumanEva: synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International Journal of Computer Vision*, 4–27.

Simon, T., Joo, H., Matthews, I., & Sheikh, Y. (2017). Hand keypoint detection in single images using multiview bootstrapping. In *Ieee conference on computer vision and pattern recognition (cvpr)*.

Starner, T., Weaver, J., & Pentland, A. (1998). Real-time american sign language recognition using desk and wearable computer based video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1371–1375.

Starner, T. & Pentland, A. (1997). Real-time american sign language recognition from video using hidden markov models. In *Motion-based recognition* (pp. 227–243).

Sun, K., Xiao, B., Liu, D., & Wang, J. (2019). Deep high-resolution representation learning for human pose estimation. In *Ieee conference on computer vision and pattern recognition (cvpr)*.

Takeuchi, K., Kubota, S., Suzuki, K., Hasegawa, D., & Sakuta, H. (2017). Creating a gesture-speech dataset for speech-based automatic gesture generation. In *HCI international 2017 - posters extended abstracts* (pp. 198–202).

Turing, A. M. (2004). Computing machinery and intelligence (1950). *The Essential Turing: The Ideas that Gave Birth to the Computer Age. Ed. B. Jack Copeland. Oxford: Oxford UP*, 433–64.

Ultraleap. (2020). The leap motion sensor. Retrieved from https://www.ultraleap.com/

Wang, T.-C., Liu, M.-Y., Zhu, J.-Y., Tao, A., Kautz, J., & Catanzaro, B. (2018). High-resolution image synthesis and semantic manipulation with conditional gans. In *Ieee conference on computer vision and pattern recognition (cvpr)* (pp. 8798–8807).

Wei, S.-E., Ramakrishna, V., Kanade, T., & Sheikh, Y. (2016). Convolutional pose machines. In *Ieee conference on computer vision and pattern recognition (cvpr)*.

Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., & Girshick, R. (2019). Detectron2. https://github.com/facebookresearch/detectron2.

Xiang, D., Joo, H., & Sheikh, Y. (2019). Monocular total capture: posing face, body, and hands in the wild. In *Proceedings of the ieee conference on computer vision and pattern recognition*.

Yan, H. (2000). Paired speech and gesture generation in embodied conversational agents.

Yang, Y. & Ramanan, D. (2013). Articulated human detection with flexible mixtures of parts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2878–90.

Zimmermann, C. & Brox, T. (2017). Learning to estimate 3d hand pose from single rgb images. In *Ieee international conference on computer vision (iccv)*.

Appendix

Appendix A.

List Of Acronyms

ANN	Artificial Neural Network
AR	Augmented Reality
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DOF	Degrees Of Freedom
FPS	Frames Per Second
GAN	Generative Adversarial Network
GB	Gigabyte
GPU	Graphics Processing Unit
GRU	Gated Recurrent Unit
LSTM	Long Short-Term Memory
MFCC	Mel-Frequency Cepstral Coefficients
NN	Neural Network
PCK	Percentage of Correct Keypoints
RGB	Red Green Blue
RNN	Recurrent Neural Network
UMA	Unity Multipurpose Avatar
VR	Virtual Reality