Multi-objective adversarial gesture generation

Ylva Ferstl
Trinity College Dublin
yferstl@tcd.ie

Michael Neff
University of California Davis
mpneff@ucdavis.edu

Rachel McDonnell
Trinity College Dublin
ramcdonn@tcd.ie

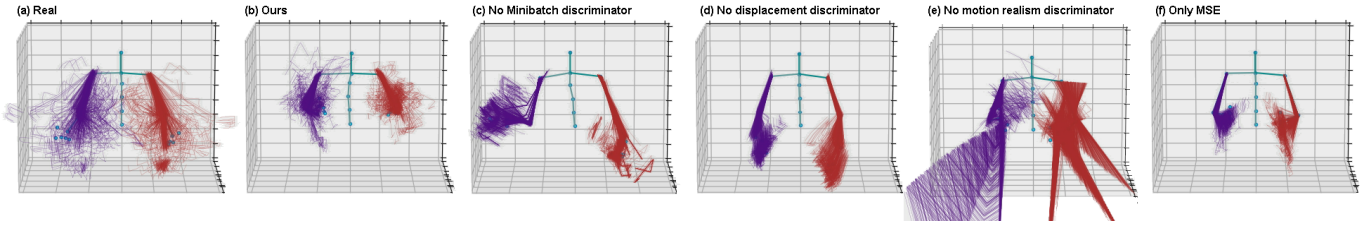

Figure 1: Motion distribution over 2 minutes, plotted at 4 fps. For one example clip, a) shows the real data distribution, b) the distribution with our method, c-e) examples of excluding specific training objectives, and f) the distribution for a model trained with a standard regression loss.

ABSTRACT

Applications for conversational virtual agents are on the rise, but producing realistic non-verbal behavior for spoken utterances remains an unsolved problem. We explore the use of a generative adversarial training paradigm to map speech to 3D gesture motion. We define the gesture generation problem as a series of smaller sub-problems, including plausible gesture dynamics, realistic joint configurations, and diverse and smooth motion. Each sub-problem is monitored by separate adversaries. For the problem of enforcing realistic gesture dynamics in our output, we train a classifier to automatically detect gesture phases. We find adversarial training to be superior to the use of a standard regression loss and discuss the benefit of each of our training objectives. We recorded a dataset of over 6 hours of natural, unrehearsed speech with high-quality motion capture, as well as audio and video recording.

CCS CONCEPTS

• **Computing methodologies** → **Animation**; *Machine learning*; *Motion processing*;

KEYWORDS

gesture generation, machine learning, generative adversarial networks, gesture segmentation

ACM Reference format:
Ylva Ferstl, Michael Neff, and Rachel McDonnell. 2019. Multi-objective adversarial gesture generation. In *Proceedings of Motion, Interaction and Games, Newcastle upon Tyne, United Kingdom, October 28–30, 2019 (MIG '19),* 10 pages.

MIG '19, October 28–30, 2019, Newcastle upon Tyne, United Kingdom

ACM ISBN 978-1-4503-6994-7/19/10…$15.00
https://doi.org/10.1145/3359566.3360053

https://doi.org/10.1145/3359566.3360053

1 INTRODUCTION

Interactive virtual agents are becoming increasingly common and people may enjoy interacting with them more than even realistic video-based characters [Kang et al. 2016]. However, they often remain feeling stiff and unnatural. Non-verbal behavior plays an important role in making these agents more appealing, and co-speech gestures specifically is a key component for increasing user engagement [Salem et al. 2011]. Automatic generation of such gesturing behavior for given utterances is appealing due to both cost-factors and time constrained animation needs. Despite much research in the area, automatically generating realistic gestural behavior remains an open problem. One of the difficulties in modelling the speech-to-gesture relation is the asynchronicity between the two channels; gesture precedes or co-incides with speech but rarely follows [McNeill 1992], making real-time prediction nearly impossible. A second difficulty is the highly non-deterministic mapping of speech to motion. Even the same speaker uttering the same phrase will likely perform different gesture motions on each repetition. Gestures may also communicate information not provided explicitly through speech, providing complementary not redundant information [De Ruiter et al. 2012; Melinger and Levelt 2004].

The non-deterministic mapping of speech to motion means for one utterance, multiple variations of a gesture (or no gesture at all) may be perceived as plausible by an observer. This presents a difficulty in training a speech-to-gesture model; even a plausible produced gesture may be penalized when it is numerically far from the exact gesture found in the dataset for this utterance. A standard regression loss in training a speech-to-gesture model is therefore not ideal.

In this work, we apply two novel techniques for training a recurrent neural network (RNN) producing gesture motion based on input speech. Firstly, we train a speech-input-motion-output RNN with a generative adversarial paradigm instead of a standard regression loss, and we specifically use multiple adversaries instead of a single one. Secondly, we aim to enforce a realistic gesture

phase structure by training one discriminator network to distinguish between real and generated motion based purely on its phase structure. As discussed in Section 2, natural gesture can be divided into distinct phases: preparation, stroke, hold, and retraction. We automatically extract the phase structure of real and generated motion with a phase classifier that we designed and trained specifically for this purpose.

Our multi-discriminator design allows the gesture generation problem to be defined with multiple smaller sub-problems. We discuss how each of our discriminator objectives improves the final result.

We will first introduce the phase classifier in Section 4, before discussing the speech-to-gesture model in Section 5.2 and its adversarial training in Section 7.1.

2 RELATED WORK

Gesture generation Various methods have been proposed for generating gesture from speech. Some approaches employ rule-based systems that rely on explicitly defined text-to-gesture rules [Cassell et al. 2001; Marsella et al. 2013; Thiebaux et al. 2008a]. Other works have used statistical modelling estimating conditional probabilities for speech features co-occurring with motion features [Bergmann and Kopp 2009a,b; Neff et al. 2008]. Various animation systems have been developed to produce gesture motion, such as SmartBody [Kallmann and Marsella 2005; Thiebaux et al. 2008b]; while is beyond the scope of this work to cover this area in detail, recent surveys provide an overview (e.g. Neff [2016]).

Machine learning approaches have both been used in a fully automatic manner without any need for hand annotating data [Bozkurt et al. 2016; Chiu and Marsella 2011, 2014; Levine et al. 2010, 2009], as well as in conjunction with hand-labelled, higher-level features such as gestural signs [Chiu and Marsella 2015].

Recent work has explored recurrent networks for speech-to-gesture generation for English [Ferstl and McDonnell 2018] and Japanese speech [Hasegawa 2018; Kucherenko et al. 2019]. Such a network uses recurrent connections between network activations at consecutive time-steps to model data with temporal dependencies. Recurrent networks can, for example, capture the dynamics of a motion pattern well and have been successfully employed for human motion modelling tasks [Li et al. 2017; Pavllo et al. 2018]. However, recurrent networks trained with a standard error function tend to suffer from mean pose convergence, where longer term motion sequences quickly regress to the average pose (such as in Martinez et al. [2017] and Jain et al. [2015]). This may be due to error accumulation when feeding generated output back into the network [Holden et al. 2017], resulting in damped motion that may look constrained and unrealistic. Generative adversarial networks (GANs) have been proposed as one alternative training paradigm. Here, instead of minimizing a standard error function such as the mean squared error of joint positions or angles, the model's objective is to produce output that is qualitatively similar to real data, as judged by another model, the discriminator, that is trained simultaneously in conjunction with the generator. GANs have been successful in human motion modelling tasks [Barsoum et al. 2018; Kundu et al. 2018], as well as in a speech-to-head motion generation task [Sadoughi and Busso 2018].

Very recent work proposed a convolutional network combining a standard L1 regression loss with an adversarial discriminator for predicting 2D gesture motion from speech [Ginosar et al. 2019]. The authors represent audio visually as a spectrogram, which is then encoded by an audio encoder and subsequently processed by a UNet translation architecture [Ronneberger et al. 2015]. The authors created a large dataset of over 140 hours of 2D pose keypoints extracted from YouTube videos of 10 speakers. (This work and dataset was not yet available at the time of our work). The speakers are professional performers, such as John Oliver (*Last Week Tonight*) and Seth Meyers (*Late Night with Seth Meyers*), producing largely rehearsed speech and generally producing a relatively small set of clear gesture motions. Their speaker-specific models generate sequences rated equally good as mismatched real gesture samples, as measured by the rate it fooled human participants. The failure to surpass random real motion is an indication of the difficulty of the speech-to-gesture task. In our work, we focus on a different type of gesture motion, namely spontaneous, conversational speech gestures that appear more diverse and qualitatively different from the distinct gestures usually seen for professional performers (refer e.g. to John Oliver's performances in *Last Week Tonight*).

Gesture phase Natural gesture behavior consists of phases with qualitatively different dynamic characteristics [Kendon 1972] and these phases occur in specific patterns [Kita et al. 1997]. In the *preparation* phase, the hands are moved into position for the gesture. The *stroke* has the most focused energy, it is an "accented movement" with effort in the sense of Laban [Kita et al. 1997], and is the main meaning carrying movement of the gesture. The *retraction* moves the limbs back into a restful position (an incomplete retraction is noted as a *partial retraction*). *Holds* are segments with zero velocity and may occur before or after the stroke [Kita 1990]. All phases are optional except the stroke.

We aim to capture these specific dynamic phases in our gesture generation system. While these phases are present in any natural gesture data, capturing the phase structure implicitly would require a large dataset. Instead, we explicitly segment the phase structure of gesture motion.

Segmenting gesture motion into its phases is non-trivial and in many cases requires subjective judgment. Hence the labelling process cannot be seen as deterministic and 100% accuracy is unlikely, or even impossible. Different, automatic gesture phase annotation methods have been proposed, including the use of support vector machines [Madeo et al. 2016] and hidden Markov models [Alexanderson et al. 2016; Martell and Kroll 2007]. One limiting factor in training phase models is obtaining labelled data; segmenting just one minute of video into gesture phases may take one hour or more of work (e.g. Neff et al. [2008]). Previous work has therefore often focused on simpler sub-problems of detecting whether one specific phase is occurring (e.g. detection only of gesture strokes), or whether a gesture is being performed at all [Alexanderson et al. 2016; Bryll et al. 2001; Gebre et al. 2012].

3 DATASET

We recorded a high-quality dataset of natural speech and 3D motion specifically for the purpose of this work. We used a single male actor for the complete recording. The actor is a native English

speaker producing spontaneous conversational speech without interruptions, i.e., without verbal cues from a conversation partner. The actor was free to choose any topic in his speech but mostly covered personal stories and sports. We chose an actor with naturally frequent gesturing behavior, but he was unaware of the purpose of the recording. The actor addresses a person situated behind the camera in order to give him the visual feedback of a conversation partner. We recorded 25 takes, ranging between 10 and 20 minutes each, totalling over 370 minutes (more than 6 hours) of data. The actor's motion was captured with a 59 marker setup and 20 Vicon cameras at 120 fps (frames per second). Audio was recorded at 44 kHz. Video was captured with two cameras.

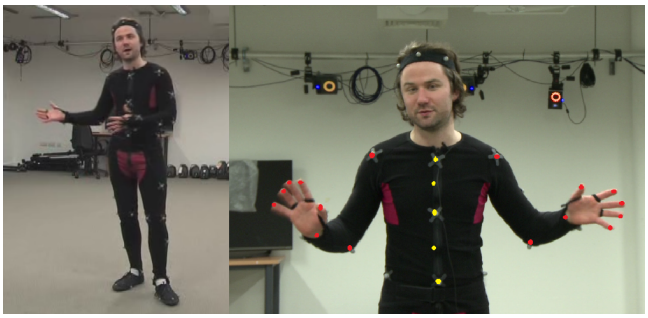

Figure 2: Capture setup and location of joints. The 16 red markings indicate the joints used for the gesture phase classification. The five yellow markings indicate the spinal joints added to the joint set for gesture motion prediction.

3.1 Data Pre-processing

We processed the recorded speech with openSMILE [Eyben et al. 2013] to extract 26 Mel Frequency Cepstral Coefficients (MFCCs), as well as the F0 (pitch) value. MFCCs are commonly used in speech recognition tasks and the F0 value as a prosodic feature carries information about emphasis. Speech features were extracted with a window size of 20 ms at steps of 0.01 ms, resulting in data of 100 fps.

We down-sampled the motion capture data from 120 to 100 fps to match the speech features. We center and lock the root node of the motion clips to the origin position with zero rotation and then extract the absolute positional values of the captured joints. Our actor remains fairly static in his lower body and we are therefore able to capture most of his dynamics from the joints upward of the locked root.

We normalize all speech and joint position features to zero mean and unit variance. We train all models on 20 fps; in order not to lose data, we take 20 fps data from 5 subsequent starting positions, resulting in 5 sets of 20 fps data.

3.2 Gesture phase annotation

We annotated the phase structure of a subset of 226 minutes of the complete dataset using the ANVIL annotation tool [Kipp 2001]. The 226 minutes were selected at random from the dataset. We aimed to annotate as much of our dataset as possible while ensuring annotation quality. For this purpose, we trained six annotators whose work was then repeatedly cross-checked at the start, before each annotator was assigned separate data clips. We annotated nine different gesture phases; (1) preparation, (2) stroke, (3) pre-hold, (4) hold, (5) independent hold, (6) rest hold, (7) partial retract, (8) retract, and (9) 'none'. Table 1 shows the frequency of each phase within the annotated data subset. *Pre-hold* and *hold* occur before and after the gesture. *Independent hold* occurs when a gesture has no stroke, but is defined by a held pose. *Rest hold* occurs when the hands are held in a relaxed position without being fully retracted to the sides of the body. *None* occurs when the arms are fully retracted to the sides of the body and no gesture is being performed.

Table 1: Frequency of the 9 annotated phases in the total annotation set of 226 minutes.

Gesture phase	Number of occurrences	Percent of annotated time
Preparation	5775	19.1%
Pre-hold	979	3.2%
Stroke	8655	39.6%
Hold	5100	24.8%
Independent hold	94	0.8%
Rest hold	474	3.1%
Partial retract	1077	3.8%
Retract	409	1.3%
'None'	475	4.2%
Total	23038	-

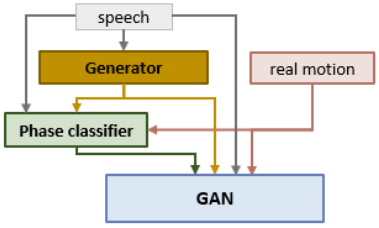

Figure 3: Overview of the system architecture. The generator receives speech features and produces gesture motion. The multi-discriminator GAN receives three different types of input: (1) the speech features belonging to a motion segment, (2) a motion segment (real or generated), and (3) the phase structure of the motion segment (determined by the phase classifier).

4 PHASE CLASSIFIER

Modelling gesture motion from speech directly is a hard problem. As described in Section 1, the same phrase may be plausibly accompanied by many different gesture shapes. Speech features may be more easily associated with the dynamics of gesture motion; the kinematics of gestures (e.g., speed and acceleration) have been shown to correlate with the prosodic features of speech [Valbonesi et al. 2002]. However, implicitly inferring gesture dynamics from raw positional data may be difficult and require a large amount

of data. We therefore model these dynamics explicitly. Namely, we extract gesture phases as higher-level representation of the characteristic dynamics of gesture motion. This representation is sufficiently low-dimensional (small set of different labels) to model its structure from a relatively small dataset. We hand-annotated the phase structure of 3.75 hours of data (as described in Section 3.2) and trained a classifier to detect gesture phases of a motion sequence. Our objective is to use this phase classification to enforce a realistic phase structure in the gesture generator's output. A classifier is necessary so that any new (un-annotated) motion can be segmented into phases and judged for its structural realism. After training the classifier on the annotated data subset, we never use the true hand-annotated phase labels, we always use the phase labels determined by the classifier and the full dataset. An overview of the phase classifier's role in the final architecture is shown in Figure 3, and will be discussed in more detail in Section 7.1.

4.1 Method

The classifier assigns one phase label to each time-step of an input sequence. For training the classifier, we reduce the annotated gesture phase label set from nine to six classes by combining all types of holds into one class. That is, we combine the labels 'pre-hold', 'hold', 'independent hold' and 'rest hold' into a super-class 'hold'. In effect, this simplifies the classification task by labelling all still frame sequences (sections with close-to-zero joint velocities) as one class, with the exception of the completely retracted 'none' position where the arms are relaxed by the side of the body. For the adversarial training, we decide to further reduce the set of phase labels to four classes: Preparation, holds (including pre-holds, independent holds, and rest-holds), strokes, and 'other'. The 'other' class combines retracts, partial retracts, and 'none' annotations. We choose this subset for the following reasons: first, we believe that holds and strokes are the most important representatives of gesture dynamics and their separation tends to get lost in standard training of recurrent networks (mean pose convergence leading to damped motion). Second, we separate the preparation phase due to its high frequency and relevance in the gesture structure. Retracts are relatively infrequent for our speaker, as is the 'none' phase (completely retracted position); we decided to pool these classes together to make for a higher confidence model and a more achievable task for the gesture generator. The phase labels produced by the classifier are used as pseudo ground-truth during adversarial training, and we therefore need the classifier to be as confident as possible in its decisions.

4.2 Network architecture and training

The classifier processes sequences of 5 seconds at a time, at 20 fps, and assigns a phase label to each of the resulting 100 time-steps. As input the classifier receives x, y and z velocities of 16 joints (total of 48 values), corresponding to the shoulder, elbow, wrist, and each fingertip, as well as the corresponding pitch value. The pitch value captures information about speech emphasis and using a single speech feature ensures that the motion input is not out-crowded. Including pitch improves our classification scores (see Table 3); in line with the finding that speech is associated with gesture phase [Yunus et al. 2019]. The network consists of a two-layer recurrent network with an additional feed-forward layer for input processing. The recurrent layers are Long Short Term Memory (LSTM) cells; specifically, a unidirectional LSTM in the first recurrent layer, and a bidirectional LSTM in the second recurrent layer. LSTM cells can handle sequential data, such as time series data, and bidirectional LSTMs specifically take both past and future data into account for predicting a time step. We regularize the network by applying dropout after each layer and batch normalization before the final output. Dropout rates are empirically determined to provide good performance without overfitting. The network details are visualized in Figure 4. Of our total of 226 minutes of annotated data, we separate 6.5 minutes of validation data by randomly selecting 13 start indices from which to take 30 seconds of data without overlap. The remaining annotations serve as training data.

4.3 Results

The classifier reaches an overall weighted F-score of 0.76. The detailed results can be seen in Table 2. The stroke and hold phases reach the highest scores; this is likely due to both their distinct dynamics as well as their high frequency in the training set (see Table 1). Lower frequency phases with less distinct dynamics, such as partial retracts, are more difficult to detect. Notably, the annotated phase labels are only pseudo ground truth, as determined by an annotator. We compare our results with the work of Madeo et al. [2016], who employ a hierarchical strategy of single-class classifiers, where e.g. a hold classifier first detects all holds, subsequently a stroke classifier detects all strokes, etc. Their results represent the best scores across multiple models rather than a single model encompassing all gesture classes. That is, they trained combinations of single-class classifiers and the here reported results represent the highest scores for each class across combinations. For example, the model achieving the score of 0.79 for detecting a preparation phase is not the same model that achieves the score of 0.79 for stroke detection.

We compare results for two classification models (4-class and 6-class), with and without speech pitch input. Table 3 shows the F1 scores of the classification models trained without pitch input. Comparing the results to the with-pitch models in Table 2, the benefit of including pitch in the input to the classifier is more obvious for the 6-class model, where all individual scores except 'partial retract' are improved by including pitch, as well as showing an improvement of 0.03 in the overall weighted F1 score. For the 4-class model, the individual class scores improve (all except stroke) or remain the same (stroke), but the weighted overall F1 remains the same when including pitch as input.

5 GESTURE GENERATOR

The gesture generator is the core of the system and models the speech-to-gesture translation. It receives speech features as input and produces the positions of the 21 joints shown in Figure 2.

5.1 Generator architecture

The generator receives 27 speech features as input, composed of 26 MFCC values and the speech pitch (F0) value. The generator then infers the x, y, and z positions of 21 joints, the hand, arm, and spine joints depicted in Figure 2. The 27 speech features are passed to a

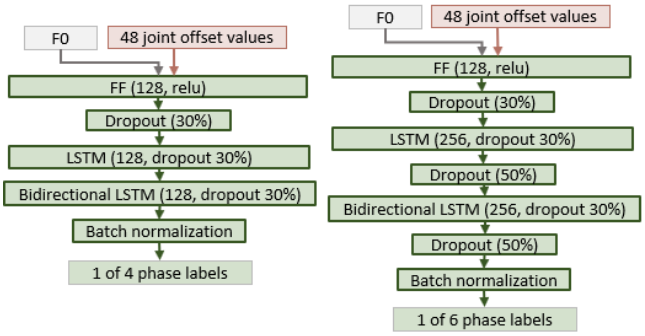

Figure 4: The two detailed network configurations for our 4-phase classifier and our 6-phase classifier. FF denotes a feed-forward layer with linear activation. In brackets are denoted the layer size or the dropout ratio. The 48 joint values refer to the x, y, and z offsets of the 16 joints shown in Figure 2.

Table 2: Scores of phase classifier. Our 'other' class combines the labels *retract*, *partial retract*, and *none*. The scores for Madeo et al. [2016] denote the best scores reached across multiple models.

Gesture phase	F-score 4 classes	F-score 6 classes	F-score Madeo et al. [2016]
Preparation	0.64	0.65	0.79
Stroke	0.79	0.79	0.79
Hold	0.83	0.81	0.58
Partial retract	-	0.47	-
Retract	-	0.73	0.5
'None'	-	0.75	-
'Other'	0.64	-	-
Overall	0.76	0.76	-

Table 3: Scores of phase classifier without pitch input.

Gesture phase	F-score 4 classes	F-score 6 classes
Preparation	0.63	0.64
Stroke	0.79	0.78
Hold	0.82	0.78
Partial retract	-	0.49
Retract	-	0.70
'None'	-	0.56
'Other'	0.60	-
Overall	0.76	0.73

feed-forward layer with size 256 and relu activation followed by 30% dropout during pre-training and 20% during adversarial training and batch normalization. The input is propagated to a Gated Recurrent Unit (GRU) of size 256 with a dropout of 50% during pre-training and 20% during adversarial training. A GRU is a variant of a recurrent network cell with fewer parameters than an LSTM, allowing faster training. The linear output layer of the generator produces the x, y and z position of 21 joints. The full generator architecture is visualized in Figure 5. During pre-training (described in the below Section 5.2), the dropout rate needs to be larger due to the fact that the mean squared error function used in pre-training poses a high probability of overfitting. The mean squared error gives the generator direct feedback on how far the predicted pose is from the ground truth. During later multi-adversarial training, the generator receives less direct output feedback and is therefore less likely to be able to overfit on the dataset. The adversarial loss merely tells the generator the likelihood of the discriminator(s) finding its output to be real data.

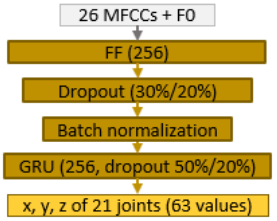

Figure 5: The generator network. The generator receives 27 prosodic speech features (26 MFCCs + F0) and produces the xyz position of 21 joints. In brackets are denoted the layer size or the dropout ratio; the larger dropout ratios apply to pre-training.

5.2 Generator pre-training

During later adversarial training (Section 7.1), the generator will receive feedback based on the phase structure of its motion output. This phase structure will be determined by the phase classifier previously described in Section 4. This automatic phase classification means that no matter what input, a phase label will be assigned to each time-step. Data points diverging from a skeleton structure and not resembling human motion may get assigned an indeterminable phase label. We do not want very unrealistic data to be assigned a potentially realistic phase labelling. This could allow for the following scenario: the generator generates effectively noise, the classifier produces a realistic phase structure based on this, the generator receives positive feedback for having produced motion with a realistic phase structure. We therefore first ensure a quality baseline of generator output that can reasonably be assigned phase labels by the phase classifier. Hence, before adversarial training, we initialize the generator to a baseline output resembling a skeleton structure.

We pre-train the generator with a standard mean squared error (MSE) loss of generated versus real motion:

$$MSE(m_g, m_r) = \frac{1}{T}\sum_{t=1}^{T}(m_g - m_r)^2 \quad (1)$$

MSE training allows for fast convergence towards a skeleton structure, but as expected, this training suffers from mean pose convergence and produces only very damped motions around the average joint positions. This is visualized in Figure 1 f), as well as in the supplemental video. We use this model as the starting point for the

adversarial training, and utilize the training history for pre-training the phase discriminator as described in Section 6.1.

6 ADVERSARIES

In the most basic terms, the generator's task is to fulfill two objectives: To produce joint positions that resemble human motion, and for this motion to be be appropriate with respect to the speech it accompanies. While on a higher level we want the generator to produce a realistic succession of gesture phases from speech, that objective is not restrictive enough to generate realistic motion. We discuss this step-by-step below.

6.1 Phase structure discriminator

The phase discriminator's job is to determine whether the generator's output follows a realistic gesture phase structure. This discriminator therefore only receives phase labels as input rather than joint positions. We additionally provide the phase discriminator with the pitch value at each time-step as an indicator of speech emphasis. The network architecture of the phase discriminator is detailed in Figure 6 a).

Phase labels are always determined by the phase classifier; that is, we never use the ground truth annotation during adversarial training. This ensures that any differences in the phase structure of real and generated data is not due to potentially noisy automatic classification. As the phase labels are automatically determined by the phase classifier, we need to ensure somewhat sensible input to the classifier, i.e. input resembling human motion. We utilize the training history of the generator's pre-training to prepare the phase discriminator. The training history of the generator are the generator weights saved periodically during its pre-training described in Section 5.2. The phase discriminator's pre-training utilizes this as follows: The phase discriminator receives the classified phase labelling of an untrained generator (i.e. noise input). When the phase discriminator achieves an accuracy score of at least 70% for three

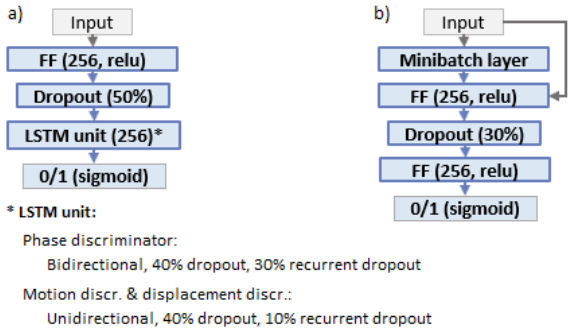

Figure 6: Network architecture of the adversaries. Left: Phase, motion, and displacement discriminators. Right: Minibatch discriminator. All discriminators apply input transformation via a feed-forward layer. (The minibatch layer applies Equation 2 before the input transformation.) Dropout is applied subsequently, followed by a recurrent unit (left) or another feed-forward transformation (right). The output layer applies a sigmoid activation.

batches in a row, the generator gets 'upgraded' with the next set of weights from the training history. This is repeated until the phase discriminator has reached the weights level of the fully pre-trained generator. This step-by-step upgrading of the generator's weights serves to not overwhelm the discriminator during pre-training.

6.2 Motion realism discriminator

Adversarial training between the generator and the phase discriminator alone will quickly lead to divergence from the skeleton structure due to the phase discriminator only judging the automatically classified phase labels. As described in Section 5.2, the phase classifier may assign a realistic phase structure to unrealistic input; when the generator is judged solely on this phase structure, it may receive positive discriminator feedback for entirely unrealistic output and increasingly diverge from producing skeleton-like joint positions. To address this problem, we employ a second discriminator that judges the output of the generator directly by receiving the pure generated joint positions, as well as the corresponding audio features. The 63 joint values (x, y, z of 21 joints) and 27 speech features are passed into the network architecture detailed in Figure 6 a).

The motion realism discriminator is pre-trained in a classic adversarial training setting with a new generator in order to learn to detect unrealistic point clouds not resembling a skeleton. This is necessary in order to not allow the already pre-trained generator to regress to non-humanoid point clouds.

6.3 Minibatch discriminator

Adversarial training is prone to suffering from mode collapse, where the generator produces repetitive patterns of output. While the discriminator can immediately learn that this specific pattern comes from the generator, the generator only needs to shift its repetitive output slightly to fool the discriminator. This may be repeated in an infinite cat and mouse game. One reason for this mode collapse is that a standard discriminator only judges one output sequence at a time, rather than in the context of a whole batch of data. A minibatch layer can be added to allow the discriminator to see this context and ensure that the generator cannot get away with even novel patterns when they are repetitive throughout the data batch [Salimans et al. 2016].

Instead of integrating minibatch discrimination into the motion realism discriminator, we achieved better performance when outsourcing the task to a separate discriminator. This discriminator receives 63 joint values (x,y,z of 21 joints) generated by the generator or taken from the ground truth and calculates a minibatch similarity measure:

$$sim(X) = L^1(W \cdot X), \tag{2}$$

where L^1 denotes the L1 norm and W is a 300-dimensional (trainable) weight tensor. The detailed architecture of the minibatch discriminator is shown in Figure 6 b).

6.4 Displacement discriminator

The generator's output at the beginning of adversarial training is the damped motion learned from the MSE pre-training. To encourage the generator towards less damped motion, we introduce a displacement discriminator that receives the same motion input as

the phase classifier, namely the per-frame x, y, and z offset of the 16 arm joints (48 values). That is, the displacement discriminator explicitly sees how much each joint has moved at each time-step; it can penalize a generator that produces very slow (or very fast) motion. The displacement discriminator also serves to reduce jitter in the motion (offset in one direction always followed by some offset to opposite direction).

The error from this discriminator receives a lesser weight and serves as a minor side objective of the generator training, helping to stabilize and speed up convergence and smooth output motion. The architecture of the displacement discriminator follows that of the motion realism discriminator and is visualized in Figure 6 a).

7 TRAINING PROCESS

During adversarial training, the generator's output is judged by all discriminators and an averaged error is computed, as detailed in Section 7.1 below. This is followed by a training step of objective numerical errors. The objective error functions speed up convergence and enable continuous prediction, as described in Section 7.2.

7.1 Adversarial training

The adversarial training is visualized in Figure 7 and summarized below:

- The **generator** receives 27 prosodic speech features as input and generates corresponding 3D positions of 21 joints.
- The **phase classifier** first converts the joint positions to frame offsets and subsequently predicts a sequence of gesture phase labels. The phase classifier also receives as input the F0 (pitch) value of each frame. The classifier's weights are fixed during adversarial training.
- The produced phase label sequence of the classifier, plus the F0 value, serve as input for the **phase structure discriminator**.
- The motion realism discriminator receives the joint positions directly, as well as all corresponding 27 speech features.
- The **displacement discriminator** receives the same motion input as the phase classifier, the per-frame joint offsets of the 16 arm and hand joints.
- The **minibatch discriminator** only receives the joint positions as input.

All three discriminators are trained with a binary cross-entropy loss to determine whether a motion sequence is real or generated. The loss of the generator with respect to the three discriminators is weighted and combined into a single value for the generator's training step. All models work with input sequences of 5 seconds, at 20 fps, resulting in 100 time-steps.

During adversarial training steps, the generator optimizes the binary cross-entropy of the discriminators' output. The generator's training error with respect to the four discriminators is averaged for each optimization step in the following manner:

$$\mathcal{L}_{GAN}(G) = \frac{w_p \mathcal{L}(G, D_p) + w_r \mathcal{L}(G, D_r) + w_m \mathcal{L}(G, D_m) + w_d \mathcal{L}(G, D_d)}{w_p + w_r + w_m + w_d}, \quad (3)$$

$$\text{with } w_p = 2, w_r = 4, w_m = 4, \text{ and } w_d = 1,$$

where w_p is the weight assigned to the phase discriminator's loss, w_r the weight for the motion realism discriminator, w_m the weight for the minibatch discriminator, and w_d the weight for the displacement discriminator. $\mathcal{L}(G, D)$ represents the generator's objective with respect to one discriminator. The weighting of 2:4:4:1 was chosen by empirically finding values that led to stable training with respect to all discriminator objectives, without the generator collapsing with respect to one or more objectives. The adversarial training of the generator is visualized in Figure 7. We use the RMSprop optimizer during adversarial training.

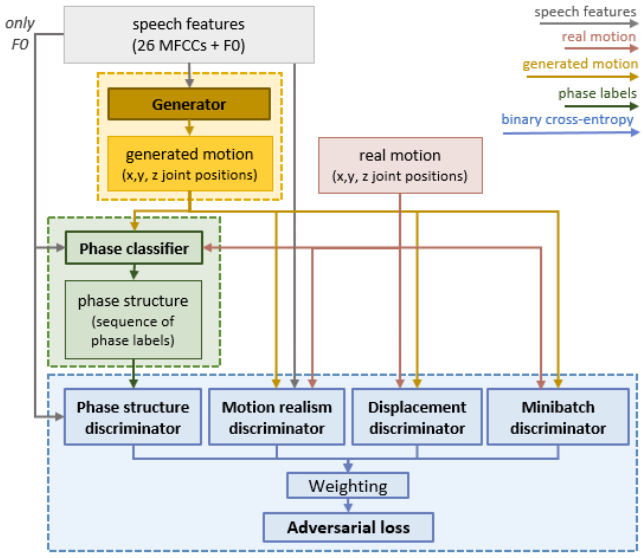

Figure 7: Adversarial training. The generator produces joint positions based on input speech features. Its output is judged by four discriminators with separate objectives, and a weighted error is computed with respect to all four evaluations. Each discriminator optimizes the binary cross-entropy objective, deciding if a given data sample is real or generated.

7.2 Objective loss penalties

In addition to the adversarial updates of the generator, one MSE correction is performed per two adversarial steps. The MSE avoids major deviations of the generator's output from a realistic skeleton structure that would produce nonsensical phase label output and slow down the training overall.

The generator is trained to predict gesture motion for 5 seconds of speech input at a time rather than for continuous input. Gesture motion can therefore be visibly discontinuous between predictions. To avoid having to compute smooth transitions in post-processing, we introduce a penalty for the generator for discontinuous sequences within a training batch. The discontinuation penalty is computed as the mean squared distance between the start position of a sequence and the end position of the preceding sequence. The penalty for first sequence within a batch is always

set to zero and otherwise:

$$\mathcal{L}_{cont}(G) = \frac{1}{T}\sum_{t=1}^{T}(G(x)(t) - G(x)(t-1)))^2 \,. \quad (4)$$

We observed during adversarial training that the predicted finger positions often move far from the hand. To speed up the training process, we added a simple finger distance penalty restricting the predictions to realistic ranges. We compute the distance of each finger marker to the respective hand marker and calculate the MSE with respect to the real distances:

$$\mathcal{L}_{fingers}(G) = \frac{1}{n}\sum_{i=1}^{n}(\mathcal{D}_{fingers}(G(x)) - \mathcal{D}_{fingers}(Y(x)))^2 \quad (5)$$

with $Y(x)$ denoting the ground truth for sample x, and $\mathcal{D}_{fingers}$ computed as the concatenation of each finger marker's x, y, and z distance from the respective hand.

8 RESULTS

We conducted a series of evaluations to clarify the roles of each discriminator and their benefits for generator training.

8.1 Phase structure discriminator

The phase structure discriminator allows us to capture important gesture dynamics without having to rely on implicit learning from a larger dataset (such as in Ginosar et al. [2019]). During the pre-training described in Section 6.1, this discriminator easily learns to distinguish the (noisy) classified phase structures of real motion and motion produced by the pre-trained generator. During adversarial training, the phase discriminator's accuracy remains balanced with the generator's while the generator's output is improving in quality. We visualize the benefits of the phase discriminator for encouraging better gesture motion dynamics in the supplemental video; without the phase discriminator, the motion shows no clear holds or accelerations characteristic of the stroke phase. The motion appears to correspond less with the speech prosody.

8.2 Motion realism discriminator

The phase discriminator's judgment alone is not a sufficient constraint for the generator's output. As described in Section 6.2, the automatic phase label classification of the generator's output and the phase classifier's naivety with respect to non-human point clouds provides too much room for the generator to produce unrealistic data. The motion discriminator presents a better constraint for maintaining a skeleton structure as it sees the generator's output directly and successfully constrains the generator to data points resembling a skeleton structure. Figure 1 e) visualizes the output distribution produced by a generator unconstrained by a motion discriminator. The supplemental video also shows a sample of the motion produced without a motion realism discriminator; the joint positions move away from the skeleton structure, producing output not resembling human motion.

8.3 Minibatch discriminator

As a vanilla discriminator only judges output sequences in isolation, without taking the context of the data batch into consideration, the generator can suffer from mode collapse, as described in 6.3, and visualized by the plotted data distribution in Figure 1 c). Our minibatch discriminator successfully forces the generator to produce more diverse output. The supplemental video shows the repetitive motion generated under mode collapse, as well as the improved, diverse output with minibatch discrimination. We considered two alternative integrations of minibatch discrimination into our model, namely as part of the motion realism discriminator and as part of a separate discriminator. In practice, we find the adversarial training to be more stable by outsourcing the minibatch discrimination to a separate discriminator only receiving motion input. Generator training was less likely to collapse with respect to one discriminator when the adversarial objective was more distributed. The benefit of employing multiple discriminators has also been discussed in previous works [Albuquerque et al. 2019; Durugkar et al. 2016].

8.4 Displacement discriminator

Learning from the phase discriminator's feedback is potentially difficult for the generator due to the hidden layers between the generator and phase discriminator (i.e., the phase classifier's computations that are inaccessible to the generator). The generator's motion output is first converted to per-frame offsets of the joints and then passed to the classifier for higher level feature extraction. Introducing a discriminator receiving the same processed motion as the classifier can provide more direct feedback. In practice, we found that the addition of such a displacement discriminator sped-up learning and moved predictions away faster from the damped baseline motion produced by the pre-trained generator. We visualize this by plotting an example data distribution in Figure 1 d). The slow departure from the mean pose when training the model without the displacement discriminator is also shown in the supplemental video. We also illustrate the smoothing benefit of the displacement discriminator in the video: When training the generator without any discriminator receiving the joint offsets (i.e. with neither the displacement discriminator nor the phase classifier and discriminator), the motion output displays a great amount of jitter. We show that adding the displacement discriminator reduces jitter to a large degree. This discriminator receives the smallest weighting in the generator's objective.

8.5 Adversarial error weighting

We find a weighting of 2:4:4:1 for the error for the: phase discriminator, motion realism discriminator, minibatch discriminator, and the displacement discriminator respectively, to achieve the most stable training, measured by the accuracy of the binary cross-entropy objective for each discriminator. This weighting allows us to see stable accuracy improvements for the generator across all adversarial objectives without collapse with regard to one or more objectives.

8.6 Objective losses

The discontinuation penalty is largely successful in reducing the positional jumps between predicted motion sequences, making the model more applicable for continuous gesture generation for long sequences of speech input. The finger distance penalty proved a simple measure to avoid unrealistic finger positions without strongly constraining the generator in its predictions.

9 DISCUSSION

We explored generative adversarial networks for speech-to-gesture translation with higher level feature extraction. For this purpose, we first recorded a dataset of over six hours of natural, conversational speech with high-quality 3D motion capture. Gesture motion is marked by distinct dynamics, phases of acceleration and effort, of pause, and of relaxation. These higher-level dynamics can be difficult to capture implicitly. To enforce these dynamics more explicitly in a top-down manner, we train a classifier to detect these phases automatically, and then train a phase structure discriminator to detect realistic versus non-realistic phase sequences. To train the classifier, we hand-annotated the phases of a subset of over 3.5 hours of the dataset using 9 different phase labels.

Instead of using a standard regression loss for our gesture generator, we trained our model in a generative adversarial setting with multiple discriminators. We see a clear advantage of adversarial training over using a standard regression loss in that the produced motion has a larger range and appears much less damped.

Using multiple discriminators, we phrase the speech-to-gesture generation problem as a series of sub-problems. We use our automatic phase labelling to enforce a more realistic gesture phase structure in our output; this is the task of the phase structure discriminator. The phase structure discriminator enables the enforcement of higher level characteristics in the output without having to rely on implicit learning from a large amount of data.

Because an automatic phase classifier will always assign some phase label to even random point clouds, we constrain the motion output with a second discriminator judging the joint positions as real or fake; this is the task of the motion realism discriminator. Because the motion realism discriminator's task is to judge one generated motion sequence at a time, it can allow for the same sequence to be generated repeatedly. A minibatch discriminator detects such repetitive patterns, ensuring diversity in the output. Generated motion can often look jittery; we address this by including a the training objective of realistic joint displacement per frame, monitored by the displacement discriminator.

To our knowledge, this is the first work using adversarial training for generating 3D gesture motion from natural speech, and the first work exploring the use of multiple discriminators. We observe a benefit of using multiple discriminators to stabilize adversarial training, and we report how each discriminator addresses a distinct sub-problem in the gesture generation task. We employ explicit modelling of the dynamics of gesture motion to allow learning of these higher level features from a smaller dataset. We see our work as a further step towards enabling automatic animation of realistic conversational agents.

Our results are limited to gesture generation for the single speaker we recorded and more data of various speakers would be necessary to make generalizations. Due to the high variance of gesture behavior across speakers, this is a very difficult task. Because we generate gesture motion from prosodic speech features, semantically meaningful gestures can hardly be inferred without explicitly employing speech recognition methods. Speech recognition, however, would likely only yield a benefit when using a much larger dataset, ensuring a number of examples of the same phrases.

10 FUTURE WORK

While the generated motion improved greatly with respect to standard regression loss training, the motion still lacks realism. Looking forward, we will explore other measures of realism that may complement adversarial training.

We are furthermore interested in explicitly enforcing gesture phase by first determining the appropriate gesture phase and then using this as a conditional input for the generator, similar to the approach proposed by Holden et al. [2017], who use locomotion phase as input in a character control system. This would require gesture phase extraction from only speech, rather than motion data. In this regard, Yunus et al. [2019] report interesting initial results in predicting gesture phase from prosodic speech features. We will also explore the use of convolutional networks within a generative adversarial paradigm, such as in Ginosar et al. [2019], exploring visual data representations of speech as well as motion.

ACKNOWLEDGMENTS

This research was funded by Science Foundation Ireland under the ADAPT Centre for Digital Content Technology (Grant 13/RC/2016). Partial funding was also provided through NSF grant 1748058.

REFERENCES

Isabela Albuquerque, João Monteiro, Thang Doan, Breandan Considine, Tiago Falk, and Ioannis Mitliagkas. 2019. Multi-objective training of Generative Adversarial Networks with multiple discriminators. *arXiv preprint arXiv:1901.08680* (2019).

Simon Alexanderson, David House, and Jonas Beskow. 2016. Automatic annotation of gestural units in spontaneous face-to-face interaction. *Proceedings of the Workshop on Multimodal Analyses enabling Artificial Agents in Human-Machine Interaction - MA3HMI '16* (2016), 15–19. https://doi.org/10.1145/3011263.3011268

Emad Barsoum, John Kender, and Zicheng Liu. 2018. HP-GAN: Probabilistic 3D human motion prediction via GAN. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 1418–1427.

Kirsten Bergmann and Stefan Kopp. 2009a. GNetIc - Using bayesian decision networks for iconic gesture generation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 5773 LNAI (2009), 76–89. https://doi.org/10.1007/978-3-642-04380-2_12

Kirsten Bergmann and Stefan Kopp. 2009b. Increasing the expressiveness of virtual agents: autonomous generation of speech and gesture for spatial description tasks. *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1* (2009), 361–368. http://portal.acm.org/citation.cfm?id=1558013.1558062

Elif Bozkurt, Yücel Yemez, and Engin Erzin. 2016. Multimodal analysis of speech and arm motion for prosody-driven synthesis of beat gestures. *Speech Communication* 85 (2016), 29–42. https://doi.org/10.1016/j.specom.2016.10.004

Robert Bryll, Francis Quek, and Anna Esposito. 2001. Automatic hand hold detection in natural conversation. In *IEEE Workshop on Cues in Communication*. 1–4.

Justine Cassell, Hannes Högni Vilhjálmsson, and Timothy Bickmore. 2001. BEAT: the Behavior Expression Animation Toolkit. *ACM Transactions on Graphics* (2001), 477–486. https://doi.org/10.1007/978-3-662-08373-4_8

Chung Cheng Chiu and Stacy Marsella. 2011. How to train your avatar: A data driven approach to gesture generation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 6895 LNAI (2011), 127–140. https://doi.org/10.1007/978-3-642-23974-8_14

Chung-cheng Chiu and Stacy Marsella. 2014. Gesture Generation with Low-Dimensional Embeddings. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. 781–788.

Chung-Chen Chiu and Stacy Marsella. 2015. Predicting Co-verbal Gestures: A Deep and Temporal Modeling Approach. *International Conference on Intelligent Virtual Agents* 9238 of th, August 2015 (2015), 152–166. https://doi.org/10.1007/978-3-319-21996-7_17

Jan P De Ruiter, Adrian Bangerter, and Paula Dings. 2012. The interplay between gesture and speech in the production of referring expressions: Investigating the tradeoff hypothesis. *Topics in Cognitive Science* 4, 2 (2012), 232–248.

Ishan Durugkar, Ian Gemp, and Sridhar Mahadevan. 2016. Generative multi-adversarial networks. *arXiv preprint arXiv:1611.01673* (2016).

Florian Eyben, Felix Weninger, Florian Gross, and Björn Schuller. 2013. Recent developments in opensmile, the munich open-source multimedia feature extractor. In

Proceedings of the 21st ACM international conference on Multimedia. ACM, 835–838.

Ylva Ferstl and Rachel McDonnell. 2018. Investigating the use of recurrent motion modelling for speech gesture generation.. In *IVA '18: International Conference on Intelligent Virtual Agents (IVA '18)*. 93–98.

Binyam Gebrekidan Gebre, Peter Wittenburg, and Przemyslaw Lenkiewicz. 2012. Towards automatic gesture stroke detection. In *LREC 2012: 8th International Conference on Language Resources and Evaluation*. European Language Resources Association, 231–235.

Shiry Ginosar, Amir Bar, Gefen Kohavi, Caroline Chan, Andrew Owens, and Jitendra Malik. 2019. Learning Individual Styles of Conversational Gesture. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3497–3506.

Dai Hasegawa. 2018. Evaluation of Speech-to-Gesture Generation Using Bi-Directional LSTM Evaluation of Speech-to-Gesture Generation Using Bi-Directional LSTM Network. In *IVA '18: International Conference on Intelligent Virtual Agents (IVA '18)*. https://doi.org/10.1145/3267851.3267878

Daniel Holden, Taku Komura, and Jun Saito. 2017. Phase-functioned neural networks for character control. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 42.

Ashesh Jain, Amir R. Zamir, Silvio Savarese, and Ashutosh Saxena. 2015. Structural-RNN: Deep Learning on Spatio-Temporal Graphs. (2015), 5308–5317. https://doi.org/10.1109/CVPR.2016.573 arXiv:1511.05298

Marcelo Kallmann and Stacy Marsella. 2005. Hierarchical motion controllers for real-time autonomous virtual humans. In *International Workshop on Intelligent Virtual Agents*. Springer, 253–265.

Sin-Hwa Kang, Andrew W Feng, Mike Seymour, and Ari Shapiro. 2016. Smart Mobile Virtual Characters: Video Characters vs. Animated Characters. In *Proceedings of the Fourth International Conference on Human Agent Interaction*. ACM, 371–374.

Adam Kendon. 1972. Some relationships between body motion and speech. *Studies in dyadic communication* 7, 177 (1972), 90.

Michael Kipp. 2001. Anvil-a generic annotation tool for multimodal dialogue. In *Seventh European Conference on Speech Communication and Technology*.

Sotaro Kita. 1990. The temporal relationship between gesture and speech: A study of Japanese-English bilinguals. *MS, Department of Psychology, University of Chicago* 90 (1990), 91–94.

Sotaro Kita, Ingeborg Van Gijn, and Harry Van Der Hulst. 1997. Movement Phases in Signs and Co-speech Gestures , and Their Transcription by Human Coders. *Gesture and sign language in human-computer interaction : international gesture workshop Bielefeld* (1997), 23–35. https://doi.org/10.1007/BFb0052986

Taras Kucherenko, Dai Hasegawa, Gustav Eje Henter, Naoshi Kaneko, and Hedvig Kjellström. 2019. Analyzing Input and Output Representations for Speech-Driven Gesture Generation. In *IVA '19: International Conference on Intelligent Virtual Agents (IVA '19)*.

Jogendra Nath Kundu, Maharshi Gor, and R Venkatesh Babu. 2018. BiHMP-GAN: Bidirectional 3D Human Motion Prediction GAN. *arXiv preprint arXiv:1812.02591* (2018).

Sergey Levine, Philipp Krähenbühl, Sebastian Thrun, and Vladlen Koltun. 2010. Gesture controllers. *ACM Transactions on Graphics* 29, 4 (2010). https://doi.org/10.1145/1833351.1778861

Sergey Levine, Christian Theobalt, and Vladlen Koltun. 2009. Real-time prosody-driven synthesis of body language. *ACM Transactions on Graphics* 28, 5 (2009), 1. https://doi.org/10.1145/1618452.1618518

Zimo Li, Yi Zhou, Shuangjiu Xiao, Chong He, and Hao Li. 2017. Auto-conditioned lstm network for extended complex human motion synthesis. *arXiv preprint arXiv:1707.05363* 3 (2017).

Renata Cristina Barros Madeo, Sarajane Marques Peres, and Clodoaldo Aparecido De Moraes Lima. 2016. Gesture phase segmentation using support vector machines. *Expert Systems with Applications* 56 (2016), 100–115. https://doi.org/10.1016/j.eswa.2016.02.021

Stacy Marsella, Yuyu Xu, Margaux Lhommet, Andrew Feng, Stefan Scherer, and Ari Shapiro. 2013. Virtual character performance from speech. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 25–35. https://doi.org/10.1145/2485895.2485900

Craig Martell and Joshua Kroll. 2007. Corpus-based gesture analysis: an extension of the form dataset for the automatic detection of phases in a gesture. *International Journal of Semantic Computing* 1, 04 (2007), 521–536.

Julieta Martinez, Michael J. Black, and Javier Romero. 2017. On human motion prediction using recurrent neural networks. (2017). https://doi.org/10.1109/CVPR.2017.497 arXiv:1705.02445

David McNeill. 1992. *Hand and mind: What gestures reveal about thought*. University of Chicago press.

Alissa Melinger and Willem JM Levelt. 2004. Gesture and the communicative intention of the speaker. *Gesture* 4, 2 (2004), 119–141.

Michael Neff. 2016. Hand gesture synthesis for conversational characters. *Handbook of Human Motion* (2016), 1–12.

Michael Neff, Michael Kipp, Irene Albrecht, and Hans-Peter Seidel. 2008. Gesture modeling and animation based on a probabilistic re-creation of speaker style. *ACM Transactions on Graphics* 27, 1 (2008), 1–24. https://doi.org/10.1145/1330511.1330516

Dario Pavllo, David Grangier, and Michael Auli. 2018. Quaternet: A quaternion-based recurrent model for human motion. *arXiv preprint arXiv:1805.06485* (2018).

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*. Springer, 234–241.

Najmeh Sadoughi and Carlos Busso. 2018. Novel realizations of speech-driven head movements with generative adversarial networks. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 6169–6173.

Maha Salem, Katharina Rohlfing, Stefan Kopp, and Frank Joublin. 2011. A friendly gesture: Investigating the effect of multimodal robot behavior in human-robot interaction. *Proceedings - IEEE International Workshop on Robot and Human Interactive Communication* (2011), 247–252. https://doi.org/10.1109/ROMAN.2011.6005285

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved techniques for training gans. In *Advances in neural information processing systems*. 2234–2242.

Marcus Thiebaux, Stacy Marsella, Andrew N Marshall, and Marcelo Kallmann. 2008a. SmartBody: behavior realization for embodied conversational agents. In *Proceedings of International Joint Conference on Autonomous Agents and Multiagent Systems*. 151–158. https://doi.org/10.1016/j.ins.2009.01.020

Marcus Thiebaux, Stacy Marsella, Andrew N Marshall, and Marcelo Kallmann. 2008b. Smartbody: Behavior realization for embodied conversational agents. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems, 151–158.

Lucia Valbonesi, Rashid Ansari, David McNeill, F Quek, Susan Duncan, Karl E Mc-Cullough, and R Bryll. 2002. Multimodal signal analysis of prosody and hand motion: Temporal correlation of speech and gestures. In *2002 11th European Signal Processing Conference*. IEEE, 1–4.

Fajrian Yunus, Chloé Clavel, and Catherine Pelachaud. 2019. Gesture Class Prediction by Recurrent Neural Network and Attention Mechanism. In *Proceedings of the 19th ACM International Conference on Intelligent Virtual Agents*. ACM, 233–235.