Adversarial gesture generation with realistic gesture phasing

Ylva Ferstl
Trinity College Dublin

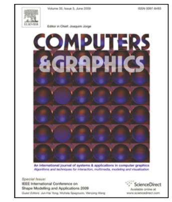
Rachel McDonnell
Trinity College Dublin

Computers & Graphics (2020)

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag



Adversarial gesture generation with realistic gesture phasing

Ylva Ferstl[a,], Michael Neff[b], Rachel McDonnell[a]

[a] *Trinity College Dublin*
[b] *University of California Davis*

ARTICLE INFO

Article history:
Received May 4, 2020

Keywords: gesture generation, machine learning, generative adversarial networks, gesture segmentation, virtual humans

ABSTRACT

Conversational virtual agents are increasingly common and popular, but modeling their non-verbal behavior is a complex problem that remains unsolved. Gesture is a key component of speech-accompanying behavior but is difficult to model due to its non-deterministic and variable nature. 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 three classifiers with different levels of detail to automatically detect gesture phases. We hand-annotate and evaluate over 3.8 hours of gesture data for this purpose, including samples of a second speaker for comparing and validating our results. 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.

1. Introduction

Interactive virtual agents are becoming increasingly common and people may enjoy interacting with them more than even with realistic video-based characters [1]. However, they often still feel stiff and unnatural. Non-verbal behavior plays an important role in making these agents more appealing, and co-speech gestures specifically are a key component for increasing user engagement [2]. 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 [3], 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 [4, 5].

e-mail: `yferstl@tcd.ie` (Ylva Ferstl), `mpneff@ucdavis.edu` (Michael Neff), `ramcdonn@scss.tcd.ie` (Rachel McDonnell)

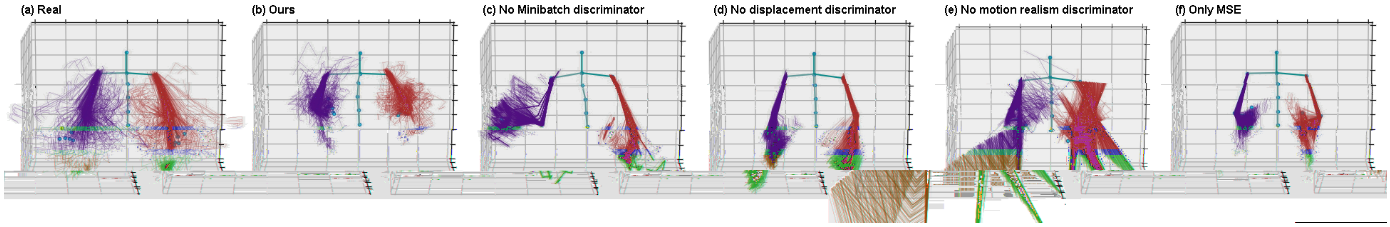


Fig. 1. Motion distribution over 2 minutes, plotted at 4 fps. For one example clip, a) shows the real data distribution, b) the distribution with our method, c-e) examples of excluding specific training objectives, and f) the distribution for a model trained with a standard regression loss.

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 study the phase structure of a gesture dataset and train a classifier to automatically detect these phases. The phase structure of natural gesture describes the dynamics and functions of motion segments within it, and can be divided into distinct parts: preparation, stroke, holds, and retraction. The expression of these phases and their sequencing may vary from speaker to speaker, making their labelling a difficult and at times ambiguous task.

In this work, we extend Ferstl et al. [6], with additional content regarding gesture phasing, including new results on our automatic phase classification, with a more speaker-flexible reduced phase model focusing on the stroke phase, the essential core of a gesture. We furthermore annotate gesture samples of a second speaker exhibiting a distinctly different gesture style in order to evaluate our automatic phase classification.

In an adversarial training paradigm, we use the automatic phase labelling to extract the phase structure of real and generated motion. Producing realistic phase structures becomes a training objective of the generator, enforced by a discriminator specifically designed for distinguishing phase sequences.

The set of training objectives further includes humanoid skeleton constraints, and utterance match and diversification objectives, each represented by separate discriminators.

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 and its adversarial training in Section 7.

2. Related work

2.1. 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 [7, 8, 9]. Other works have used statistical modelling estimating conditional probabilities for speech features co-occurring with motion features [10, 11, 12]. Many animation systems have been developed to produce gesture motion, such as SmartBody [13, 14]; while it is beyond the scope of this work to cover this area in detail, recent surveys provide an overview (e.g. [15]).

Machine learning approaches have both been used in a fully automatic manner without any need for hand annotating data [16, 17, 18, 19, 20], as well as in conjunction with hand-labelled, higher-level features such as gestural signs [21].

Recent work has explored recurrent networks for speech-to-gesture generation for English [22] and Japanese speech

[23, 24]. 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 [25, 26]. 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. [27] and Jain et al. [28]). This may be due to error accumulation when feeding generated output back into the network [29], 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 (MSE) 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 [30, 31], as well as in a head motion from speech generation task [32].

Recent work proposed a convolutional network combining a standard L1 regression loss with an adversarial discriminator for predicting 2D gesture motion from speech [33]. The authors represent audio visually as a spectrogram, which is then encoded by an audio encoder and subsequently processed by a UNet translation architecture [34]. 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*).

2.2. Gesture phase

Natural gesture behavior consists of phases with qualitatively different dynamic characteristics [35] and these phases occur in specific patterns [36]. In the *preparation* phase, the hands are moved into position for the gesture. The *stroke* is the expressive phase of a gesture and has the most focused energy; it is an "accented movement" with Effort in the sense of Laban [36], conveying a sense of intention and meaning of the motion. It is the main meaning-carrying movement of the gesture, often describing a specific shape that relates to the accompanying verbal phrase [3]. 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 [37]. 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 arguably 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. Often, gesture phases can be straightforward to identify, but in other cases, it may be more difficult. This tends to occur when one stroke goes directly into another or if a stroke starts from a retract position. Consider for example the ambiguous example of a gesture sequence in Figure 2, where both step (1) and (3) are considered a stroke phase: One could consider the motion to the middle transition frame (2) either a partial-retract of the first stroke in (1) or a preparation for the second stroke in (3).

Different, automatic gesture phase annotation methods have

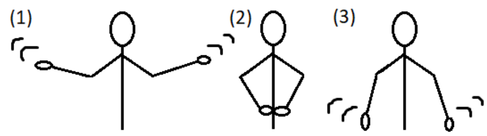


Fig. 2. Ambiguity in gesture sequence labelling. If steps (1) and (3) are each considered a gesture stroke, the motion to the transition step (2) may be labelled as either a partial-retract of the preceding stroke or a preparation phase for the following stroke.

been proposed, including the use of support vector machines [38] and hidden Markov models [39, 40]. 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. [10]). 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 [40, 41, 42].

Another difficulty in automatic phase detection is the difference in phase structure as well as phase expression between speakers and even within speaker. Phase structure differences can include overall gesture rate as well as differences in the distribution of phases; for example, one speaker may regularly produce two or more gesture strokes before returning to a rest position, while another speaker may average just one stroke before returning to rest [43]. Phase expression such as the stroke velocity profile can vary not only from speaker to speaker, but also between recordings of the same speaker [38]. This variability makes the task of automatic classification challenging, and, for a new, unseen speaker, particularly error-prone. Nevertheless, we consider even imperfect phase labelling a useful and reasonable way to explicitly describe different motion profiles present within a gesture, separating effortful, accented gesture strokes from less accentuated preparation and retraction as well as still hold phases. In this work, we focus on modelling just one speaker and his gesture dynamics to maximise training consistency of gesture dynamics in the training set.

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 addressed 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, one capturing a full body shot and the second camera capturing a higher-quality close-up shot of the face and parts of the upper body.

Fig. 3. Capture setup and location of joints. The 16 red markings indicate the joints used for the gesture phase classification. The five green markings indicate the spinal joints added to the joint set for gesture motion prediction.

3.1. Data pre-processing

We process the recorded speech with openSMILE [44] 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 are extracted with a window size of 20 ms at steps of 10 ms, resulting in data of 100 fps.

We down-sample 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 [45]. 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, respectively. *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 after a partial retract, without being fully retracted to the sides of the body. *None* occurs when no gesture is being performed; the arms are either fully retracted to the sides of the body or a no-gesture movement such as a self-adaptor is occurring.

Our speaker performs on average 38.1 gesture strokes per minute, or one gesture every 1.6 seconds. Assuming roughly the same gesture frequency in the remaining un-annotated 140 minutes of data, we estimate that our dataset contains approximately 14,000 gestures.

We computed pairwise coder agreement with ANVIL [45] by double-annotating five samples totalling 50 minutes of data, each with a different annotator combination. We found high segmentation agreement, averaging 98.5% (min=95.5%, max=99.9%), indicating high consistency in detecting phase boundaries. For the overall coding agreement that includes segment (or phase) labels, we achieved moderate agreement as defined by Krippendorff's alpha value [46], with a mean of $\bar{\alpha} = 0.46$ ($\alpha_{min} = 0.39, \alpha_{max} = 0.5$). As we pool all hold categories for the phase classifier in Section 4, we compare Krippendorff's alpha value for the case of treating post-stroke holds, pre-holds, rest-holds and independent holds all as a uniform hold category: $\bar{\alpha} = 0.47, \alpha_{min} = 0.43, \alpha_{max} = 0.53$.

In order to evaluate the robustness of our automatic phase classification in Section 4, we annotated a short sample of gesturing of a second speaker. For this, we took samples of just under 5 minutes of data from the Trinity Speech-Gesture dataset [22]. This sample was not included in the training set and only used for evaluation. The speaker in the Trinity Speech-Gesture dataset exhibits a qualitatively very different gesturing style to that of the speaker in this work, visualized in the supplemental video. This speaker often incorporates the whole body in a gesture and rarely stands still. This means that extracting the motion of the upper body joints does not fully describe the performed gesture, some information will be lost. Hold phases mark another observable difference between our speaker and the Trinity Speech-Gesture dataset; whereas holds tend to be associated with minimal movement in our speaker, the Trinity Speech-Gesture speaker's holds appear overall less still, with the speaker in seemingly constant motion.

Our annotated sample of the Trinity Speech-Gesture dataset suggest a similar gesture stroke frequency as in our database; we calculate 33.6 gesture strokes per minute. We annotated 160 strokes in this sample. All annotated phase frequencies are reported and compared to our speaker in Table 1.

An example of an annotated gesture sequence is given in Figure 4.

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

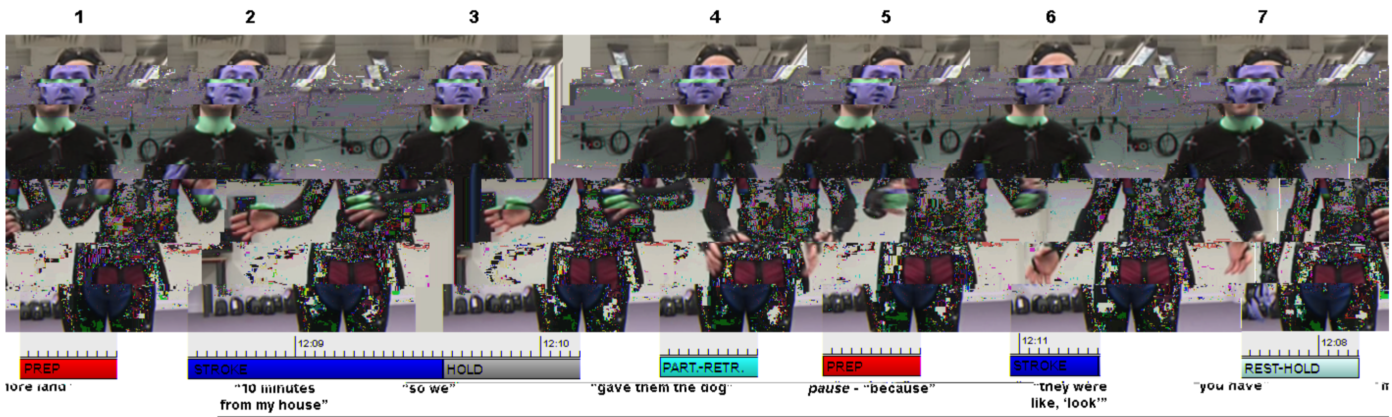


Fig. 4. Sample of an annotated gesture sequence. For each annotated gesture phase, the speaker's accompanying phrase is given. (1) The hands start in a resting position. (2) The preparation phase brings the hands into position for the gesture. (3) The stroke phase carries the meaning of the gesture (the act of giving). (4) The hands stay in position, the speaker pauses for a moment. (5) The hands are retracted partially towards a restful position. (6) A new preparation phase immediately initializes the next gesture. (7) Another gesture stroke is performed, describing "more".

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	
	Our speaker	TSG speaker	Our speaker	TSG speaker
Preparation	5775	130	19.1%	14.9%
Pre-hold	979	17	3.2%	1.6%
Stroke	8655	160	39.6%	28.5%
Hold	5100	110	24.8%	26.1%
Independent hold	94	3	0.8%	0.7 %
Rest hold	474	27	3.1%	10.3%
Partial retract	1077	48	3.8%	6.5%
Retract	409	13	1.3%	2.1%
'None'	475	14	4.2%	9.3%
Total	23038	522	100%	100%

prosodic features of speech [47]. 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 5, and will be discussed in more detail in Section 7.1.

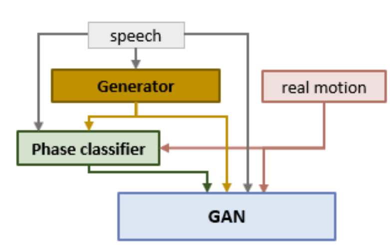


Fig. 5. Overview of the system architecture. The generator receives speech features and produces gesture motion. The multi-discriminator GAN receives three different types of input: (1) the speech features belonging to a motion segment, (2) a motion segment (real or generated), and (3) the phase structure of the motion segment (determined by the phase classifier).

We furthermore train a robust 1-phase classifier for gesture stroke detection as a useful tool for future gesture analysis. The stroke phase represents the core, meaning-carrying part of a gesture, and hence its segmentation is essential for gesture form analysis.

We validate all phase classification models on a second speaker with different gesture style.

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 that capture the main phase types by combining all types of holds into one class. This reduces the problem of unbalanced class frequencies (e.g. only 94 independent holds out of 23,038 phases), as well as removing some redundant information (e.g. a hold occurring between preparation and stroke can be assumed to be a pre-hold; a hold after a partial-retract is a rest-hold). Hence, 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. As discussed later, the partial-retract phase proved difficult to classify, so for training our generative network, we decided to combine it with the retract class, and due to its rarity we furthermore combine the fully retracted 'none' class into the retract group. For our adversarial training we therefore have four phase 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 believe this subset captures the most essential dynamics of gesture motion; we consider holds and strokes 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 smoothed, 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 100 time steps (5 seconds at 20 fps), and assigns a phase label to each step. The input of the classifier are the x, y and z directional 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 we are not increasing the input space significantly and hence minimize the network's ability to overfit. Including pitch improves our classification scores (see Table 2), in line with the finding that speech is associated with gesture phase [48].

The network is visualized in more detail in Figure 6, but generally consists of a two-layer recurrent network with an additional densely connected NN (neural network) 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.

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. Composing the validation data of snippets from multiple takes this way ensures that the validation performance is not annotator- or take-specific. The remaining annotations serve as training data.

We trained three classification models for segmenting gesture. Firstly, we train a 6-class model distinguishing all annotated phases (pooling all hold categories), second, a 4-class classifier pooling rare phases into an ‘other’ class, and third, a 1-class model for detecting only the core stroke phase with increased confidence. For the two multi-phase models, we train a version each with and without speech pitch input; the network details are visualized in Figure 6. For the stroke classifier, we predict a single class, the stroke phase, which is the essential phase in gesture. This allows for more confident classification when dealing with different speaker styles, extending the applicability of this work. The stroke classifier is visualized in Figure 7. The output layer applies a softmax activation in the case of the 4- and 6-class model, and a sigmoid activation in the single-class stroke classifier. The differences in network architecture between the 3 classifiers results from empirically finding the best performing configuration for each number of classes. The number and size of recurrent layers was chosen based on the best found trade-off between modelling capacity and generalizability, i.e. reaching good performance without overfitting.

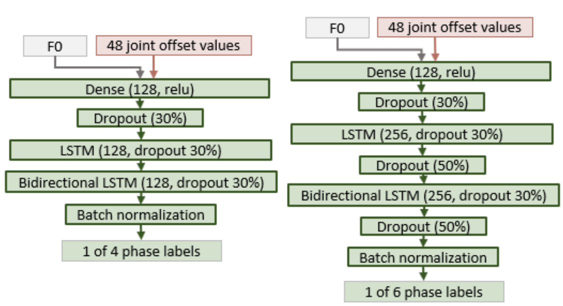


Fig. 6. The two detailed network configurations for our 4-phase classifier and our 6-phase classifier. ‘Dense’ denotes a standard densely connected NN layer. In brackets are denoted the layer size or the dropout ratio. The 48 joint values refer to the x, y, and z offsets of the 16 joints shown in Figure 3.

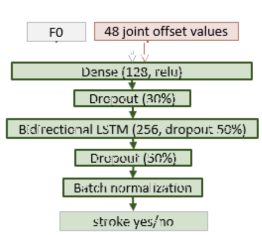


Fig. 7. The network configurations for our 1-phase (stroke) classifier.

4.3. *Results*

4.3.1. *Multi-phase classifiers*

The multi-phase classifiers reach an overall weighted F-score of 0.76 for both the 4-class and the 6-class model. 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. Furthermore, partial-retracts and preparation phases both average a length of less than 500 ms, making them potentially harder to catch as well as align; at our training sample rate of 20 fps, a prediction with just one frame of erroneous shift would only yield an 80% score. Notably, the annotated phase labels are only pseudo ground truth, as determined by an annotator, resulting in some inconsistencies and errors. Inter-rater category agreement for our evaluation samples averages 64.4%, capping the realistically achievable score for the phase classifier.

Since the input is always a sequence of 5 seconds from a randomly drawn starting point, the classifier has limited context information for predicting the phase label of a time step. Providing the label of the phase preceding a sequence or increasing sequence length may improve classification results.

Validating our classifiers on the annotated sample of the Trinity Speech-Gesture dataset (denoted as ‘TSG speaker’), the 4-class model proves more robust with an F-score of 0.69. The 6-class model reaches a score of 0.65, with the weakness lying in the less common classes, particularly partial-retract. The most confidently predicted class throughout all model versions and across both speakers is the ‘hold’ class; this may be the easiest class to extract as it contains almost all sections of zero velocity. Possible exceptions are the no-gesture sections annotated as ‘none’, though our speaker tends to swing his arms during these and indeed not stay still.

We compare results for the two multi-phase classification models (4-class and 6-class), with and without speech pitch input (Table 2). The benefit of including pitch in the input to the classifier is more pronounced 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 F-score. For the 4-class model, the individual class scores improve (all except stroke) or remain the same (stroke), but the weighted overall F remains the same when including pitch as input. We also report the performance of the no-pitch models on the second speaker. No benefit is apparent for including pitch of the second speaker; this may be due to

Table 2. F-scores of phase classifier. Results without pitch input are reported in brackets behind the results with pitch input. Our 'other' class combines the labels *retract*, *partial retract*, and *none*. The results denoted as TSG correspond to our validation speaker taken from the Trinity Speech-Gesture dataset.

Gesture phase	4 classes	4 classes TSG speaker	6 classes	6 classes TSG speaker	F-score Madeo et al. [38]
Preparation	0.64 (0.63)	0.56 (0.55)	0.65 (0.64)	0.56 (0.51)	0.79
Stroke	0.79 (0.79)	0.72 (0.7)	0.79 (0.78)	0.71 (0.71)	0.79
Hold	0.83 (0.82)	0.76 (0.76)	0.81 (0.78)	0.74 (0.77)	0.58
Partial retract	-	-	0.47 (0.49)	0.39 (0.35)	-
Retract	-	-	0.73 (0.70)	0.54 (0.52)	0.5
'None'	-	-	0.75 (0.56)	0.51 (0.59)	-
'Other'	0.64 (0.6)	0.58 (0.54)	-	-	-
Overall	**0.76 (0.76)**	**0.69 (0.67)**	**0.76 (0.73)**	**0.65 (0.66)**	

Table 3. F-scores of the stroke classifier.

Gesture phase	Our speaker	TSG speaker
Stroke	0.79	0.72
No stroke	0.85	0.86
Overall	**0.83**	**0.82**

speech features as input and produces the positions of the 21 joints shown in Figure 3.

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 3.

The generator architecture is visualized in Figure 8. The speech input is processed by a densely connected NN layer (size 256, relu activation), followed by a dropout layer (30% during pre-training, 20% during adversarial training) and batch normalization. The network core is a Gated Recurrent Unit (GRU, size 256, 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 output layer (densely connected NN layer with linear activation) of the generator produces the x, y and z position of 21 joints.

During pre-training (described in the below Section 5.2), the dropout rate is larger due to the MSE function used in pre-training posing a high probability of overfitting. The MSE gives the generator direct feedback on how far each 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, without per-pose numerical error feedback.

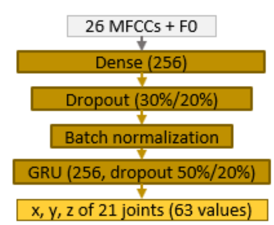


Fig. 8. The generator network. The generator receives 27 prosodic speech features (26 MFCCs + F0) and produces the xyz position of 21 joints. In brackets are denoted the layer size or the dropout ratio; the larger dropout ratios apply to pre-training with MSE.

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. The 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:

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

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 1f, 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.

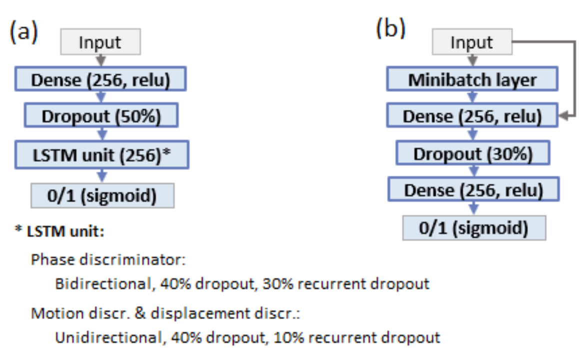


Fig. 9. Network architecture of the adversaries. Left: Phase, motion, and displacement discriminators. Right: Minibatch discriminator. All discriminators apply input transformation via a standard densely connected NN layer. (The minibatch layer applies Equation 2 before the input transformation.) Dropout is applied subsequently, followed by a recurrent unit (left) or another densely connected NN layer (right). The output layer applies a sigmoid activation.

6. Adversaries

A training objective with a standard regression loss can be problematic for gesture generation due to the variability of speech gesture. The same or a similar utterance may reasonably be associated with various different gestures; the generator may produce a subjectively valid gesture that is nonetheless objectively far from the ground-truth pose sequence, resulting in a high training error. A common result is mean pose convergence, where the generator produces damped motion around the mean, minimizing error across all possibilities. Our adversarial training paradigm removes the tight constraint of predicting exact poses while still enforcing higher-level descriptors of natural gesture, as well as lower-level humanoid skeleton configuration constraints.

Specifically, in an adversarial training paradigm, the generator receives as feedback only a single value per generated gesture sequence, representing the decision of the discriminator whether the presented sequence looks real or not. Therefore, rather than receiving a numerical error for every pose in a sequence as is the case in a standard regression loss, the generator receives a single, more qualitative judgement about the entire pose sequence.

Our chosen descriptors of natural gesture can be summarized as three basic objectives: (1) The generator should produce sequences of joint positions that represent valid human skeleton configurations. (2) The produced pose sequences should describe realistic gesture dynamics, including distinct phases of e.g. acceleration as well as stillness. (3) The output pose sequences should be appropriate with respect to the speech they accompany. With this selection of objectives, we aim to ensure that our output can both be considered speech gesture (valid human skeleton moving according to speech), as well as addressing the problems in previous works of overly smooth or lethargic motion, by explicitly enforcing some characteristics of gesture motion dynamics.

In this Section, we will discuss how we represent the above output objectives with a set of training adversaries, called discriminators, each enforcing a different part of the objectives. Each discriminator is a separate neural network, with its own training loss feedback. Their architectures are detailed in Figure 9; we will describe each discriminator one-by-one 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 9a.

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 want 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 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 we found this to lead to increasing divergence from skeleton-like joint positions. To address this problem, we employ a second discriminator that judges the output of the generator directly by receiving the raw 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 9a.

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 [49].

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 9b.

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. In effect, the displacement discriminator judges the directional velocity of the generated joint positions. 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 9a.

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 10 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 discriminators learn independently from each other, sharing no weights and receiving individual training loss feedback. 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:

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

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

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 10, representing a more detailed version of the previously presented Figure 5. We use the RMSprop optimizer during adversarial training.

7.2. Objective loss penalties

In addition to the adversarial updates of the generator, one MSE correction is performed per two adversarial steps. The

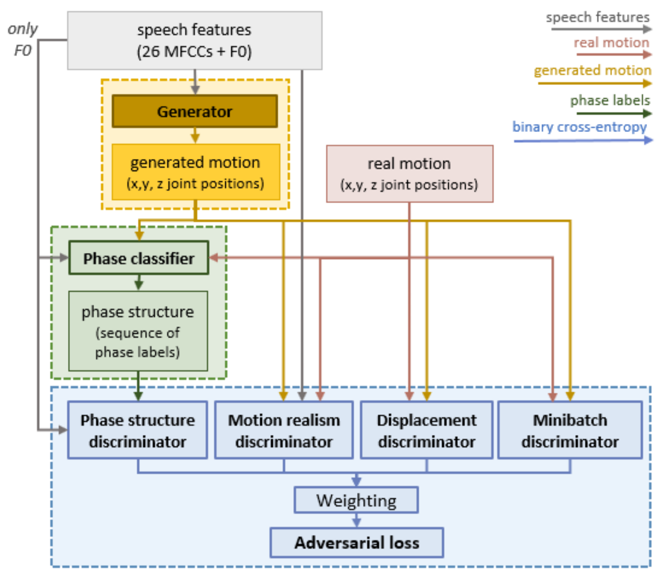


Fig. 10. Adversarial training. The generator produces joint positions based on input speech features. Its output is judged by four discriminators with separate objectives, and a weighted error is computed with respect to all four evaluations. Each discriminator optimizes the binary cross-entropy objective, deciding if a given data sample is real or generated.

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. An alternative, similar approach would be to restrict joint positions to realistic ranges.

The generator is trained to predict gesture motion for 5 seconds of speech input at a time rather than for continuous input. Gesture motion is therefore continuous within 5 second prediction intervals, but can be visibly discontinuous between intervals. 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:

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

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:

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

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 qualitative evaluations to clarify the roles of each discriminator and their benefits for generator training, and quantitative evaluations of the resulting generator output.

8.1. *Qualitative evaluation*

In this section, we discuss how each discriminator as well as the objective loss penalties affects the output of the generator qualitatively.

8.1.1. *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. [33]). 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.

8.1.2. *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 1e 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.

8.1.3. 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 1c. 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, more 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 when 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 [50, 51].

8.1.4. 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 1d. 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.

8.1.5. Adversarial error weighting

We find a weighting of 2:4:4:1 for the error of 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.

8.1.6. 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.

8.2. Quantitative evaluation

We provide a quantitative evaluation of our generation results based on the wrist motion in Figure 11. We present these results

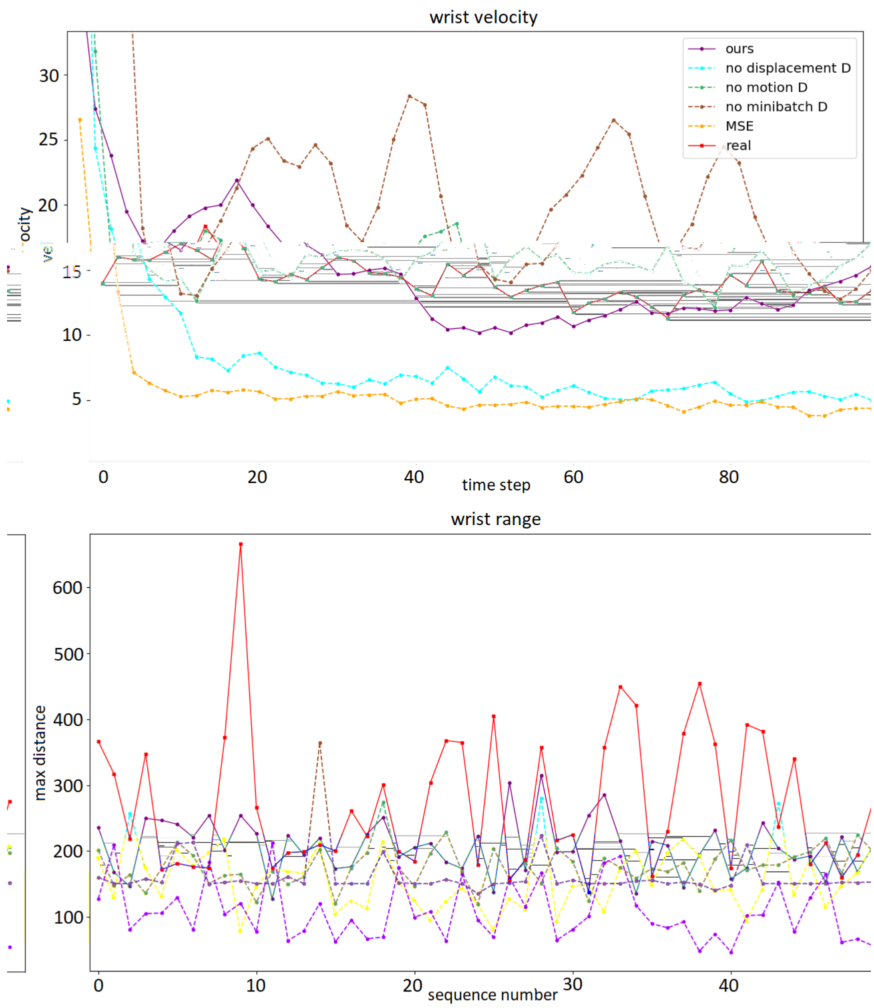


Fig. 11. Quantitative gesture generation evaluation. Top: Wrist velocity for each predicted time step, median across 150 sequences (see Equations 6 and 7). Bottom: Maximum distance of the wrists from mean pose for 50 randomly selected sequences.

in an ablation manner, as in Figure 1, evaluating how removal of a specific discriminator in training affects the generation result.

The top graph plots the wrist velocity per predicted time step, each representing the median over 150 predicted gesture sequences. This 1-dimensional velocity of the 3-dimensional x, y, z joint coordinates of a time step t and a sequence i is more specifically calculated as follows:

$$velocity(t^i) = |x_{t^i} - x_{t^{i-1}}| + |y_{t^i} - y_{t^{i-1}}| - |z_{t^i} - z_{t^{i-1}}| \quad (6)$$

$$velocity(t) = median(t^0, t^1, ...t^i, ...t^n) \quad (7)$$

We can see that one of the closest matches of real motion (red) are achieved by our model (purple) and the system configuration removing the motion discriminator (green). However, the latter configuration generates joint positions that heavily violate human skeleton constraints. Removing the minibatch discriminator (brown) produces faster than real motion, as well as resulting in highly repetitive output. The output under removal of the displacement discriminator (blue) as well as the output the generator trained solely with a mean squared error loss (yellow) exhibits very slow motion, much below realistic levels.

The bottom graph in Figure 11 plots the maximum distance travelled away from the mean pose, for 50 example sequences. The closest match to real wrist position ranges is achieved by our model, though it does not reach the wide ranges of real motion. The MSE-trained generator and the no-displacement-discriminator condition show a comparable level of variation to real motion, but the gestures are overall closer to the body both than real motion and than for our model. The no-motion-discriminator condition similarly produces lower ranges than real motion. The no-minibatch-discriminator condition produces very stable ranges, indicative of the repetitive gesture sequences generated.

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, including 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 gesture phases automatically, and then train a phase structure discriminator to detect realistic versus non-realistic phase sequences.

To train the phase classifier, we hand-annotated the phases of an over 3.7 hour long subset of our dataset using 9 different phase labels. We validate our results on a second speaker, for whom we annotate an additional small sample of gesture sequences. We compare three models of phase classification with different levels of detail (1-, 4-, and 6-class classification). We achieve good results, and we conclude that our error rate may to a relatively large extend be due to inter-coder inconsistencies. This leads to the dilemma of weighing data quantity against data quality; the large time requirement of hand-annotation (1 hour or more work for 1 minute of data) tempts distributing the work load across a number of people, but this may lead to increased problems with annotation consistency. When motion capture is available, we suggest that automatically pre-annotating all sections with close to zero velocity as 'hold' could speed up the annotation process as well as increase inter-coder agreement in future work.

Our 1-class stroke classifier performs similarly well on both our speaker and the validation speaker. 4- and 6-class classification reaches equal scores for our speaker; for the validation speaker, the 4-class model achieves a significantly higher score. One reason for the drop in performance on the validation speaker for the multi-phase models may be differences in speaker style, leading to different expressions of gesture phase. The higher the level of detail, the larger are the expected inter-speaker differences. Ideal phase classification may therefore always be speaker-specific.

For training our gesture generator, instead of using a standard regression loss, we construct a generative adversarial setting with multiple discriminators. We observe a clear advantage of adversarial training over using a standard regression loss; the produced motion has a larger positional range, more realistic velocity, and appears much less damped.

By using multiple discriminators, we can 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 dynamic 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 generated 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. Lastly, 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 for the purpose. 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 generated motion improved greatly with respect to standard regression loss training, the produced motion still lacks desirable levels of realism. Looking forward, we will explore other measures of realism that may complement adversarial training.

We are interested in working towards explicit enforcement of gesture phase by using the gesture phase as a conditional input for the generator, comparable to the approach proposed by Holden et al. [29], who use locomotion phase as input in a character control system. This may require gesture phase extraction solely from input speech, rather than motion data. In this regard, Yunus et al. [48] report interesting initial results in predicting gesture phase from prosodic speech features.

Using our gesture phase extraction, we want to analyze speech gesture further to understand better the relationship of gesture characteristic and accompanying speech. Considering the suggested differences in phase expression, as well previously found differences in gesture style (e.g. Ginosar et al. [33]), we want to investigate how gesture meaning can, or cannot, be compared across speakers.

We are also looking to explore the use of convolutional networks within a generative adversarial paradigm, such as in Ginosar et al. [33], 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

[1] Kang, SH, Feng, AW, Seymour, M, Shapiro, A. Smart mobile virtual characters: Video characters vs. animated characters. In: Proceedings of the Fourth International Conference on Human Agent Interaction. ACM; 2016, p. 371–374.

[2] Salem, M, Rohlfing, K, Kopp, S, Joublin, F. 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;:247–252doi:10.1109/ROMAN.2011.6005285.

[3] McNeill, D. Hand and mind: What gestures reveal about thought. University of Chicago press; 1992.

[4] Melinger, A, Levelt, WJ. Gesture and the communicative intention of the speaker. Gesture 2004;4(2):119–141.

[5] De Ruiter, JP, Bangerter, A, Dings, P. The interplay between gesture and speech in the production of referring expressions: Investigating the tradeoff hypothesis. Topics in Cognitive Science 2012;4(2):232–248.

[6] Ferstl, Y, Neff, M, McDonnell, R. Multi-objective adversarial gesture generation. In: Motion, Interaction and Games. 2019, p. 1–10.

[7] Cassell, J, Vilhjálmsson, HH, Bickmore, T. BEAT: the Behavior Expression Animation Toolkit. ACM Transactions on Graphics 2001;:477–486URL: http://dl.acm.org/citation.cfm?id=383315. doi:10.1007/978-3-662-08373-4_8.

[8] Thiebaux, M, Marsella, S, Marshall, AN, Kallmann, M. SmartBody: behavior realization for embodied conversational agents. In: Proceedings of International Joint Conference on Autonomous Agents and Multiagent Systems. Aamas. ISBN 978-0-9817381-0-9 LK; 2008, p. 151–158. URL: http://dl.acm.org/citation.cfm?id=1402409. doi:10.1016/j.ins.2009.01.020.

[9] Marsella, S, Xu, Y, Lhommet, M, Feng, A, Scherer, S, Shapiro, A. Virtual character performance from speech. In: Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation. ISBN 9781450321327; 2013, p. 25–35. URL: http://dl.acm.org/citation.cfm?doid=2485895.2485900. doi:10.1145/2485895.2485900.

[10] Neff, M, Kipp, M, Albrecht, I, Seidel, HP. Gesture modeling and animation based on a probabilistic re-creation of speaker style. ACM Transactions on Graphics 2008;27(1):1–24. URL: http://portal.acm.org/citation.cfm?doid=1330511.1330516. doi:10.1145/1330511.1330516.

[11] Bergmann, K, Kopp, S. 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) 2009;5773 LNAI:76–89. doi:10.1007/978-3-642-04380-2_12.

[12] Bergmann, K, Kopp, S. 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;:361–368URL: http://portal.acm.org/citation.cfm?id=1558013.1558062.

[13] Kallmann, M, Marsella, S. Hierarchical motion controllers for real-time autonomous virtual humans. In: International Workshop on Intelligent Virtual Agents. Springer; 2005, p. 253–265.

[14] Thiebaux, M, Marsella, S, Marshall, AN, Kallmann, M. 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; 2008, p. 151–158.

[15] Neff, M. Hand gesture synthesis for conversational characters. Handbook of Human Motion 2016;:1–12.

[16] Levine, S, Theobalt, C, Koltun, V. Real-time prosody-driven synthesis of body language. ACM Transactions on Graphics 2009;28(5):1. doi:10.1145/1618452.1618518.

[17] Levine, S, Krähenbühl, P, Thrun, S, Koltun, V. Gesture controllers. ACM Transactions on Graphics 2010;29(4). URL: http://dl.acm.org/citation.cfm?id=1778861. doi:10.1145/1833351.1778861.

[18] Chiu, Cc, Marsella, S. Gesture Generation with Low-Dimensional Embeddings. In: Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems. 2014, p. 781–788.

[19] Chiu, CC, Marsella, S. 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) 2011;6895 LNAI:127–140. doi:10.1007/978-3-642-23974-8_14.

[20] Bozkurt, E, Yemez, Y, Erzin, E. Multimodal analysis of speech and arm motion for prosody-driven synthesis of beat gestures. Speech Communication 2016;85:29–42. URL: http://dx.doi.org/10.1016/j.specom.2016.10.004. doi:10.1016/j.specom.2016.10.004.

[21] Chiu, CC, Marsella, S. Predicting Co-verbal Gestures: A Deep and Temporal Modeling Approach. International Conference on Intelligent Virtual Agents 2015;9238 of th(August 2015):152–166. URL: http://link.springer.com/chapter/10.1007/978-3-319-21996-7{_}17. doi:10.1007/978-3-319-21996-7_17.

[22] Ferstl, Y, McDonnell, R. Investigating the use of recurrent motion modelling for speech gesture generation. In: IVA '18: International Conference on Intelligent Virtual Agents (IVA '18). 2018, p. 93–98.

[23] Hasegawa, D. 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). November. ISBN 9781450360135; 2018,doi:10.1145/3267851.3267878.

[24] Kucherenko, T, Hasegawa, D, Henter, GE, Kaneko, N, Kjellström, H. Analyzing input and output representations for speech-driven gesture generation. In: IVA '19: International Conference on Intelligent Virtual Agents (IVA '19). 2019,.

[25] Pavllo, D, Grangier, D, Auli, M. Quaternet: A quaternion-based recurrent model for human motion. arXiv preprint arXiv:180506485 2018;.

[26] Li, Z, Zhou, Y, Xiao, S, He, C, Li, H. Auto-conditioned lstm network for extended complex human motion synthesis. arXiv preprint arXiv:170705363 2017;3.

[27] Martinez, J, Black, MJ, Romero, J. On human motion prediction using recurrent neural networks 2017;URL: http://arxiv.org/abs/1705.02445. doi:10.1109/CVPR.2017.497. arXiv:1705.02445.

[28] Jain, A, Zamir, AR, Savarese, S, Saxena, A. Structural-RNN: Deep Learning on Spatio-Temporal Graphs 2015;:5308–5317URL: http://arxiv.org/abs/1511.05298. doi:10.1109/CVPR.2016.573. arXiv:1511.05298.

[29] Holden, D, Komura, T, Saito, J. Phase-functioned neural networks for character control. ACM Transactions on Graphics (TOG) 2017;36(4):42.

[30] Barsoum, E, Kender, J, Liu, Z. Hp-gan: Probabilistic 3d human motion prediction via gan. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2018, p. 1418–1427.

[31] Kundu, JN, Gor, M, Babu, RV. Bihmp-gan: Bidirectional 3d human motion prediction gan. arXiv preprint arXiv:181202591 2018;.

[32] Sadoughi, N, Busso, C. Novel realizations of speech-driven head movements with generative adversarial networks. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE; 2018, p. 6169–6173.

[33] Ginosar, S, Bar, A, Kohavi, G, Chan, C, Owens, A, Malik, J. Learning individual styles of conversational gesture. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019, p. 3497–3506.

[34] Ronneberger, O, Fischer, P, Brox, T. U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. Springer; 2015, p. 234–241.

[35] Kendon, A. Some relationships between body motion and speech. Studies in dyadic communication 1972;7(177):90.

[36] Kita, S, Gijn, IV, Hulst, HVD. 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.

[37] Kita, S. The temporal relationship between gesture and speech: A study of japanese-english bilinguals. MS, Department of Psychology, University of Chicago 1990;90:91–94.

[38] Madeo, RCB, Peres, SM, Lima, CADM. Gesture phase segmentation using support vector machines. Expert Systems with Applications 2016;56:100–115. URL: http://dx.doi.org/10.1016/j.eswa.2016.02.021. doi:10.1016/j.eswa.2016.02.021.

[39] Martell, C, Kroll, J. Corpus-based gesture analysis: an extension of the form dataset for the automatic detection of phases in a gesture. International Journal of Semantic Computing 2007;1(04):521–536.

[40] Alexanderson, S, House, D, Beskow, J. 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;:15–19URL: http://dl.acm.org/citation.cfm?doid=3011263.3011268. doi:10.1145/3011263.3011268.

[41] Gebre, BG, Wittenburg, P, Lenkiewicz, P. Towards automatic gesture stroke detection. In: LREC 2012: 8th International Conference on Language Resources and Evaluation. European Language Resources Association; 2012, p. 231–235.

[42] Bryll, R, Quek, F, Esposito, A. Automatic hand hold detection in natural conversation. In: IEEE Workshop on Cues in Communication. 2001, p. 1–4.

[43] Kipp, M, Neff, M, Kipp, KH, Albrecht, I. Towards Natural Gesture Synthesis : Evaluating gesture units in a data-driven approach to gesture synthesis. Synthesis 2007;:1–14URL: http://dx.doi.org/10.1007/978-3-540-74997-4{_}2. doi:10.1007/978-3-540-74997-4_2.

[44] Eyben, F, Weninger, F, Gross, F, Schuller, B. Recent developments in opensmile, the munich open-source multimedia feature extractor. In: Proceedings of the 21st ACM international conference on Multimedia. ACM; 2013, p. 835–838.

[45] Kipp, M. Anvil-a generic annotation tool for multimodal dialogue. In: Seventh European Conference on Speech Communication and Technology. 2001,.

[46] Artstein, R, Poesio, M. Inter-coder agreement for computational linguistics. Computational Linguistics 2008;34(4):555–596.

[47] Valbonesi, L, Ansari, R, McNeill, D, Quek, F, Duncan, S, McCullough, KE, et al. Multimodal signal analysis of prosody and hand motion: Temporal correlation of speech and gestures. In: 2002 11th European Signal Processing Conference. IEEE; 2002, p. 1–4.

[48] Yunus, F, Clavel, C, Pelachaud, C. Gesture class prediction by recurrent neural network and attention mechanism. In: Proceedings of the 19th ACM International Conference on Intelligent Virtual Agents. ACM; 2019, p. 233–235.

[49] Salimans, T, Goodfellow, I, Zaremba, W, Cheung, V, Radford, A, Chen, X. Improved techniques for training gans. In: Advances in neural information processing systems. 2016, p. 2234–2242.

[50] Durugkar, I, Gemp, I, Mahadevan, S. Generative multi-adversarial networks. arXiv preprint arXiv:161101673 2016;.

[51] Albuquerque, I, Monteiro, J, Doan, T, Considine, B, Falk, T, Mitliagkas, I. Multi-objective training of generative adversarial networks with multiple discriminators. arXiv preprint arXiv:190108680 2019;.