Text-conditioned 3D human motion synthesis

Clémence Grislain
ENS Paris Saclay
Ecole des Ponts ParisTech
clemence.grislain@eleves.enpc.fr

Roman Plaud
ENS Paris Saclay
Ecole des Ponts ParisTech
roman.plaud@eleves.enpc.fr

1. Introduction

In this report, we address the issue of 3D human motion generation from textual description. To do so, we rely on the work of `TEMOS`[3] which itself relies on `ACTOR`[2] from the same authors.
3D human motion generation from textual description is a very challenging task, which involves simultaneously modeling both the text and the human poses. The goal is to extract relevant information about the action from the text and generate realistic and believable motions of this action. We will first define the problem and present the method proposed in `TEMOS`[3]. Thanks to the provided code[3], we reproduce the results of the paper, both quantitatively and qualitatively. Even-though these results are State-of-the-art, there remain generated motions which present failures cases. For example, 3D sequence can suffer from "foot sliding" which can be defined as the avatar failing to lift its feet when he walks. `HuMoR`[5] proposes a method to avoid this phenomenon. We implement it and add it the `TEMOS` framework and present the obtained results.

2. Proposed method

2.1. Problem definition

We manipulate the following objects:
Textual description : A written sentence describing a human motion (*e.g. A person walks*)
3D Human motion : A sequence of frames of 3D human poses. In this report, we adopt an skeleton-based human representation based on the MMM (Master Motor Map)[1] [7] format.

The final goal is, given a textual description of an action, to generate 3D human motions corresponding to the textual description. We want the output motion to be probabilistic which will allow to have several generations from a single textual description.

[1]More precisely we use preprocessed MMM motions which are 21 interest points *xyz* coordinates

2.2. Architecture

The architecture proposed in [3] is a Variationnal Auto-encoder (VAE) which is composed of three main elements:
Text encoder: Pre-trained freezed DistilBert [6] is used to embed the textual description. This embeddings are then taken as inputs (as well as trainable distribution tokens as usual in a VAE) to the textual encoder whose architecture is Transformers-based.
Motion encoder: The same Transformers-based architecture is used as motion encoder (like in `ACTOR`[2], because of the multi-dimensionality of the inputs, each frame of the 3D motion is projected into a $\mathbb{R}^d$ space).
Motion decoder: A Transformers-based decoder which takes as inputs a latent vector (sampled from the output distribution of one of the encoder) and a series of positionnal encodings (the number of positionnal encodings is equal to the length of the motion sequence).
We display in Figure 1 the architecture overview, taken from the original paper [3].

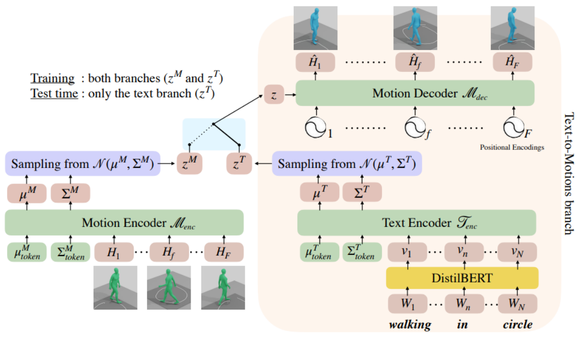
Training : both branches (z^M and z^T)
Test time : only the text branch (z^T)
$\hat{H}_1$
$\hat{H}_f$
$\hat{H}_F$
z
Motion Decoder $\mathcal{M}_{dec}$
Positional Encodings
Sampling from $\mathcal{N}(\mu^M, \Sigma^M)$
z^M
z^T
Sampling from $\mathcal{N}(\mu^T, \Sigma^T)$
μ^M
Σ^M
μ^T
Σ^T
Motion Encoder $\mathcal{M}_{enc}$
Text Encoder $\mathcal{T}_{enc}$
μ^M_{token}
Σ^M_{token}
H_1
H_f
H_F
μ^T_{token}
Σ^T_{token}
v_1
v_n
v_N
DistilBERT
W_1
W_n
W_N
walking
in
circle
Text-to-Motions branch

Figure 1. Architecture overview. Trainable parameters are displayed in green

2.3. Training strategy

At training time, both encoders produce a sample from their latent spaces. As we want to train both encoders, the loss contains two reconstruction terms: one from the reconstruction of the motion from the textual latent space vector and one from motion latent space vector. There are also regularizing terms: 4 KL-divergence terms, and a cross-model embedding similarity loss.

It is important to note that the motion encoder is only used during training. At inference time it is removed to generate 3D motions from the sole textual description.

3. Provided implementation and evaluation

We reproduce the results of the paper from the provided pre-trained model.

3.1. Quantitative results

As in `TEMOS`, to quantitatively evaluate the model, we compute the APE (Average positionnal error) and the AVE (Average Variance error) metrics in different settings on the test set: (i) generating just one random sample per text, (ii) generating 10 motions per text and average the error, (iii) generating 10 samples per text and take the best error out of the 10. We show in Table 1 these results for the provided pre-trained model on the framework we work on (MMM preprocessed motions, see Section 2.1)

Samp.	Average Positional Error			
	root j	glob. traj.	mean loc.	mean glob
(i)	1.033	1.024	0.104	1.048
(ii)	1.041	1.033	0.104	1.057
(iii)	0.801	0.792	0.103	0.820
Samp.	**Average Variationnal Error**			
	root j	glob. traj.	mean loc.	mean glob
(i)	0.448	0.447	0.005	0.451
(ii)	0.452	0.452	0.005	0.456
(iii)	0.357	0.357	0.005	0.361

Table 1. Evaluation metrics of the provided pre-trained model (1784 training data, 520 test data, 1000 epochs, batch size of 32)

Results obtained in this framework are very similar to [3].

3.2. Qualitative results

We cannot perform a proper qualitative evaluation, as most of evaluation metrics proposed by [3] rely on comparison with other models based on human feedback. Yet, we use the animation pose editor `Blender` to visualize the human motions generations as well as some of the failures cases.

Figure 2 shows, in a single frame, motions generated by the pre-trained model (using the provided rendering code). Three samples are shown per textual description. We display in the first two rows, generations obtained with a variance factor of 1. To enhance diversity in the generated motions of *"A person is running"* we increase variance by a factor 5. As expected, the resulting 3D motions are more diverse in comparison with the first two rows whose generations are almost identical.

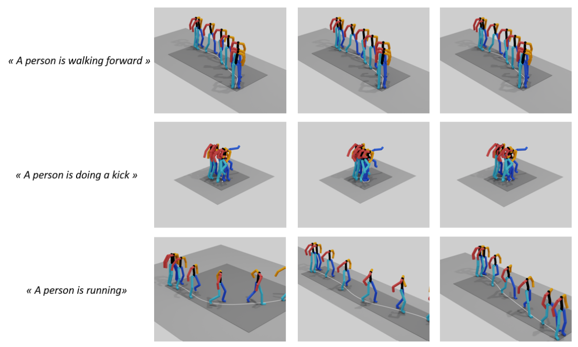

Figure 2. Generations produced by pretrained model

3.3. Pros and Cons of the method

Variationnal architectures aim at increasing diversity, that is being able to generate several plausible motions from a single textual description. The results in Figure 2 already show state-of-the-art reconstruction accuracy whereas the variationnal architecture is expected to lower the reconstruction performance. In fact, by increasing diversity, the generated motions tend to diverge from the ground truth data and thus increasing the final evaluation metrics. Thus, this method achieves both low reconstruction error and a wide diversity.

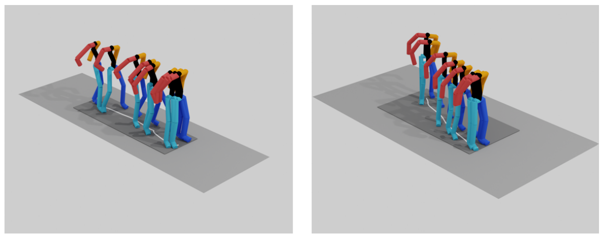

Figure 3. Two generations of *"A man is walking backwards"* with a variance factor of 3

However, there are some failures cases, as exposes on the video associated with the paper [3]. The one we are focusing on in this work is the foot sliding issue. In some generations, the 3D avatar does not lift his feet when moving. We see for instance in Figure 3 that for all generations of *"A man is walking backwards"* we observe this "moonwalk" phenomenon (the single frame rendering does not clearly show the issue but it is striking on videos).

4. Foot Sliding

4.1. Problem explanation

Foot sliding is characterized by a combination of two phenomena: a foot which is in contact with the floor <u>and</u>

which has a non-negative velocity. The paper HuMoR[5] addresses this issue by adding a contact loss. In this work, we aim at adding this feature to the TEMOS implementation.

4.2. Contact Loss

The contact loss proposed by HuMoR is composed of two terms so that $\mathcal{L}_{\text{contact}} = \mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{vel}}$.

- $\mathcal{L}_{\text{BCE}}$ consists of Binary cross entropy loss which supervises the ground contact classification of four *xyz* joints (right foot, right heel, left foot, left heel)
- $\mathcal{L}_{\text{vel}} = \sum_j \hat{c}_t^j \left\| \hat{\mathbf{v}}_t^j \right\|$ where j iterates through the 4 joints [2], $\hat{c}_t^j$ is the predicted probability that the joint j is in contact with the ground and $\hat{\mathbf{v}}_t^j$ is the velocity of the joint j at time t.

In an nutshell, $\mathcal{L}_{\text{vel}}$ penalizes motions that present foot sliding (high probability of contact and non-negative foot velocity).

4.3. TEMOS adaptation

We list here all modifications we do to plug this new feature into TEMOS implementation.

Contact labels. To supervise contact classification of the four feet joints we need the ground contact labels. To generate them we used the preprocessing code from HuMoR[5]. The computation is fitted to be done directly on AMASS [1] datasets. We therefore download KIT and CMU datasets from AMASS and preprocessed them to extract the contact labels. Yet, we train TEMOS on KIT-motion-language dataset whose motions (in MMM format) belong to the union of KIT and CMU (in SMPL format). As a consequence we match each data from KIT-motion-language [4] to its correspondence in AMASS. It resulted in 3405 matches over the 3911 available data in KIT-ML [3]. Furthermore, we transform the contact labels obtained in SMPL format to keep only the four joints corresponding to the feet (which match the MMM format).

Figure 4 shows an example of the resulting contact labels. We scatter the height of the four joints of interest over time and color them according to their contacts classification. The annotation corresponding to this motion is *"A person jumping and then walking four steps"*. Visually it seems that the generated labels fit. In this example, we see a big jump at the beginning with all four joints not being in contact with the ground and the four steps with right and left contacts being in opposition of phase.

[2]Right foot, right heel, left foot, left heel

[3]The losses come either from shape mismatches or preprocessing failures

Dataloader. We adapt the data module of [3] to deal with the new contact labels.

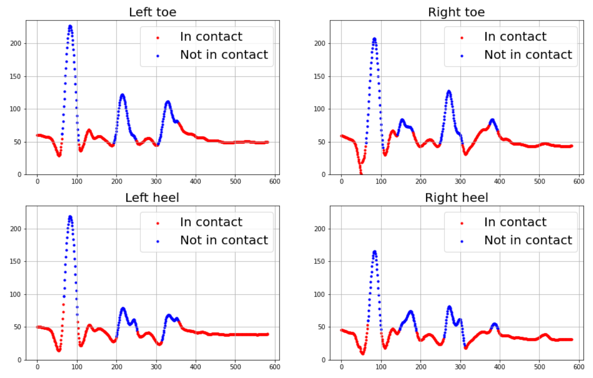

Figure 4. Visualization of generated contact labels for the motion annotated *"A person jumping and then walking four steps"* - joints height over time colored in red if classified "in contact" and in blue if "not in contact" for the four feet joints (left and right toe, left and right heel)

Architecture adaptation. We now output ground contact classification from motion decoder. We change output dimension from 64 (which corresponds to 3D points from MMM format) to 68.

Loss computation. Like there are two terms in the reconstruction loss, we add 4 contact terms to the global loss (two per encoder). One subtlety of our implementation is that HuMoR[5] reconstructs the velocities in addition to 3D coordinates. As a consequence, the term $\mathcal{L}_{\text{vel}}$ is easy to implement. In our case, as TEMOS does not reconstruct the velocities, we compute them directly from the resulting outputs with the standard finite differences formula: $\hat{\mathbf{v}}_t^j = \frac{\hat{\mathbf{x}}_{t+1}^j - \hat{\mathbf{x}}_{t-1}^j}{2}$, where $\hat{x}_t^j$ is the estimate of the position of the joint j at time t.

Penalizing factor. We empirically choose the penalizing factor $\omega_{contact}$ so that original loss and contact loss have the same order of magnitude. The new loss is:

$$\mathcal{L} = \mathcal{L}_{recons} + \omega_{contact} \times \mathcal{L}_{contact}$$

4.4. Training settings

Thanks to the Google Cloud Virtual Machine provided we were able to train our models on a single Nvidia T4 GPU (16GB). In our framework, training takes about *37 seconds* per epoch (on the reduced dataset described in Section 4.3). As in [3] we train models with AdamW optimizer, a fixed learning rate of 10^{-4}, Transformers encoders and decoders of 6 layers. Due to memory and time constraints we train on a batch size of 16 and for 400 epochs, on 1584 training and 510 validation data (while the pre-trained models were trained on 1000 epochs with a batch size of 32 on training and validation sets of 1784 and 566 data).

4.5. Results

We show in Section 4.5.1 and Section 4.5.2 results for two different frameworks $\omega_{contact} \in \{0, 10^{-6}\}$.

4.5.1 Quantitative results

We here compute classical evaluation metrics, as in Section 3.1 as well as a new evaluation metric called Contact Weighted Velocity (CWV).

$$CWJ_j = \frac{1}{T}\sum_t c_t^j \left\|\hat{\mathbf{v}}_t^j\right\|$$

where j denotes one of the 4 contact joints, c_t^j is the contact label for joint j, and $\hat{\mathbf{v}}_t^j$ is the velocity of joint j at time t.

As explained before, we could only produce contact labels for a subset of the whole `KIT-ML` dataset. The original test set has 520 motions, whereas we only have 472 test motions with contact labels.

We display in Tables 2, 3 for the same sampling strategies described in Section 3.1.

Samp.	Average Positional Error			
	root j	glob. traj.	mean loc.	mean glob
(i)	1.189	1.181	0.101	1.202
(ii)	1.274	1.266	0.100	1.287
(iii)	0.882	0.872	0.099	0.899
Samp.	Average Variationnal Error			
	root j	glob. traj.	mean loc.	mean glob
(i)	0.594	0.594	0.004	0.596
(ii)	0.603	0.602	0.004	0.605
(iii)	0.440	0.440	0.004	0.444
Samp.	Contact Weighted Velocity			
	left toe	right toe	left heel	right heel
(i)	1.827	1.773	2.040	2.021
(ii)	1.761	1.692	1.968	1.931
(iii)	0.0020	0.0019	0.0023	0.0022

Table 2. Evaluation metrics for $\omega_{contact} = 0$ (1589 training data, 472 test data, 400 epochs, batch size of 16)

As expected, all CWV metrics are improved [4] by adding the contact loss which demonstrates quantitatively the effectiveness of the method to reduce the foot sliding effect for all the sampling methods (i), (ii) and (iii).

An additional and interesting result is the improvement of the APE and AVE metrics for (i) and (iii) frameworks. It means that adding the contact loss has not only improve contact metrics but also more generic metrics of reconstruction. Moreover time constraints did not allow us to tune the hyperparameter $\omega_{contact}$ [5] which could have improved even more the results.

[4]Improvement from Table 2 are in bold

Samp.	Average Positional Error			
	root j	glob. traj.	mean loc.	mean glob
(i)	**1.116**	**1.107**	**0.098**	**1.129**
(ii)	1.318	1.310	**0.098**	1.330
(iii)	**0.765**	**0.754**	**0.098**	**0.782**
Samp.	Average Variationnal Error			
	root j	glob. traj.	mean loc.	mean glob
(i)	**0.530**	**0.529**	0.005	**0.532**
(ii)	0.711	0.711	0.005	0.714
(iii)	**0.314**	**0.313**	**0.004**	**0.318**
Samp.	Contact Weighted Velocity			
	left toe	right toe	left heel	right heel
(i)	**1.628**	**1.589**	**1.823**	**1.814**
(ii)	**1.658**	**1.594**	**1.857**	**1.820**
(iii)	**0.0018**	**0.0017**	**0.0020**	**0.0020**

Table 3. Evaluation metrics for $\omega_{contact} = 10^{-6}$ (1589 training data, 472 test data, 400 epochs, batch size of 16)

It is important to note that these quantitative results are not yet comparable to those displayed in [3] as the test set is reduced. To be able to compare to them we train the model with the contact loss ($\omega_{contact} = 10^{-6}$) on 1000 epochs and evaluate it on the whole KIT-ML test set. The CWV metrics cannot be compute as we do not have the contact labels for the whole test set. We keep all other hyper parameters unchanged and display the results in Table 4.

Samp.	Average Positional Error			
	root j	glob. traj.	mean loc.	mean glob
(i)	1.083	1.073	0.111	1.099
(ii)	1.053	1.042	0.111	1.068
(iii)	0.802	**0.790**	0.110	**0.820**
Samp.	Average Variationnal Error			
	root j	glob. traj.	mean loc.	mean glob
(i)	0.449	0.448	**0.005**	**0.451**
(ii)	**0.429**	**0.427**	**0.005**	**0.431**
(iii)	**0.331**	**0.330**	**0.005**	**0.334**

Table 4. Evaluation metrics for $\omega_{contact} = 10^{-6}$ (1589 training data, 520 test data, 1000 epochs, batch size of 16)

One can note that the results of the model trained with the additional contact loss almost match the results[6] of the pre-trained model display in Table 1 for all the evaluation metrics on all the sampling frameworks whereas the model is trained on lower batch size with fewer training data. Thus, the quantitative state-of-the-art results are not deteriorated by the additional contact loss while the foot sliding

[5]Only $\omega_{contact} = 3 \times 10^{-6}$ was tested but without significant improvements

[6]Improvement from Table 1 are in bold

effect is reduced (as shown by the previous experiment).

4.5.2 Qualitative results

Rendering show a clear reduction of the foot sliding effect on the motions generated by the model trained with the contact loss. We focus our study on the motion generated by the textual description *"A person is walking backwards"*, for which the foot sliding was striking on the generations of the pre-trained model. Figure 5 shows frames of the motions generated by our model, on which the avatar lifts his feet when walking backwards - which was not possible before. Even though, this demonstrates the qualitative improvement of the method we leave here a link on which one can watch rendered motions videos, which are way more indicative.

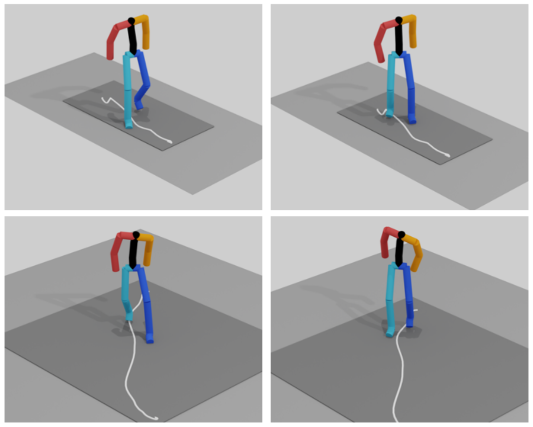
Figure 5. Visualization of two frames from two generated motions from the textual description *"A person is walking backwards"* on which the avatar lifts its feet

4.5.3 Conclusion

In this work, we studied the algorithm `TEMOS`[3] which generated diverse motions from textual description. Yet, some generations suffer unrealistic foot sliding effect. To solve the issues, based on the work `HuMoR`[5], we modified the provided implementation of `TEMOS` to add a ground contact loss which penalises this sliding effect. The obtained result, both qualitative and quantitative, show that this method does prevent the sliding effect while keeping the quantitative state-of-the-art results of the original work.

References

[1] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: archive of motion capture as surface shapes. *CoRR*, abs/1904.03278, 2019. 3

[2] Mathis Petrovich, Michael J. Black, and Gül Varol. Action-conditioned 3D human motion synthesis with transformer VAE. In *International Conference on Computer Vision (ICCV)*, 2021. 1

[3] Mathis Petrovich, Michael J. Black, and Gül Varol. TEMOS: Generating diverse human motions from textual descriptions. In *European Conference on Computer Vision (ECCV)*, 2022. 1, 2, 3, 4, 5

[4] Matthias Plappert, Christian Mandery, and Tamim Asfour. The KIT motion-language dataset. *CoRR*, abs/1607.03827, 2016. 3

[5] Davis Rempe, Tolga Birdal, Aaron Hertzmann, Jimei Yang, Srinath Sridhar, and Leonidas J. Guibas. Humor: 3d human motion model for robust pose estimation, 2021. 1, 3, 5

[6] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2019. 1

[7] Ömer Terlemez, Stefan Ulbrich, Christian Mandery, Martin Do, Nikolaus Vahrenkamp, and Tamim Asfour. Master motor map (mmm) - framework and toolkit for capturing, representing, and reproducing human motion on humanoid robots. In *Humanoids*, pages 894–901. IEEE, 2014. 1