PoseFix: Correcting 3D Human Poses with Natural Language

Ginger Delmas[1,2], Philippe Weinzaepfel[2], Francesc Moreno-Noguer[1], Grégory Rogez[2]

[1] Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona, Spain
[2] NAVER LABS Europe
[1]{gdelmas, fmoreno}@iri.upc.edu, [2]{name.surname}@naverlabs.com

Abstract

*Automatically producing instructions to modify one's posture could open the door to endless applications, such as personalized coaching and in-home physical therapy. Tackling the reverse problem (*i.e.*, refining a 3D pose based on some natural language feedback) could help for assisted 3D character animation or robot teaching, for instance. Although a few recent works explore the connections between natural language and 3D human pose, none focus on describing 3D body pose differences. In this paper, we tackle the problem of correcting 3D human poses with natural language. To this end, we introduce the PoseFix dataset, which consists of several thousand paired 3D poses and their corresponding text feedback, that describe how the source pose needs to be modified to obtain the target pose. We demonstrate the potential of this dataset on two tasks: (1)* text-based pose editing*, that aims at generating corrected 3D body poses given a query pose and a text modifier; and (2)* correctional text generation*, where instructions are generated based on the differences between two body poses. The dataset and the code are available at https://europe.naverlabs.com/research/computer-vision/posefix/.*

1. Introduction

How many puzzles could you solve with two human body poses and a description of their differences? Call this description a feedback. It could be automatically generated by a fitness application based on the comparison between the gold standard fitness pose and the pose of John Doe, exercising in front of their smartphone camera in their living room (*"straighten your back"*). In another context, the feedback can be considered a modifying instruction, provided by a digital animation artist to automatically modify the pose of a character, without having to redesign everything by hand. This feedback could be some kind of constraint, to be applied to a whole sequence of poses (make them run, but *"with hands on the hips!"*). It could also be a hint, to guide pose estimation from images in failure cases: start from an initial 3D body pose fit, and give step-by-step instructions for the model to improve its pose estimation (*"the left elbow should be bent to the back"*).

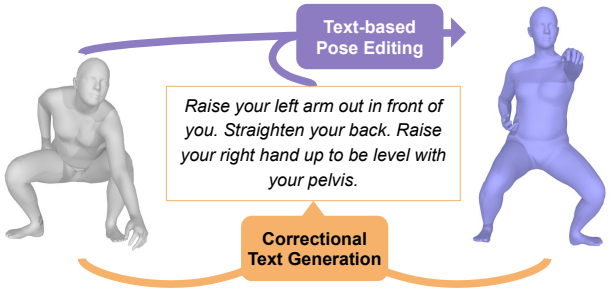

Figure 1: **Illustration of the tasks addressed with the new PoseFix dataset**, which consists of textual descriptions of the difference between two 3D body poses.

In this paper, we focus on free-form feedback describing the change between two static 3D human poses (which can be extracted from actual pose sequences). Why so static? There exist many settings that require the semantic understanding of fine-grained changes of static body poses. For instance, yoga poses are extremely challenging and specific (with a lot of subtle variations), and they are static. Some sport motions require almost-perfect postures at every moment: for better efficiency, to avoid any pain or injury, or just for better rendering *e.g.* in classical dance, yoga, karate, *etc*. What is more, the realization of complex motions sometimes calls for precise step-to-step instructions, in order to assimilate the gesture or to perform it correctly.

Natural language can help in all these scenarios, in that it is highly semantic and unconstrained, in addition of being a very intuitive way to convey ideas. While 3D poses can be manually edited within a design framework [40], language is particularly efficient for non-experts or when direct manipulation is not possible. The pose semantic we propose

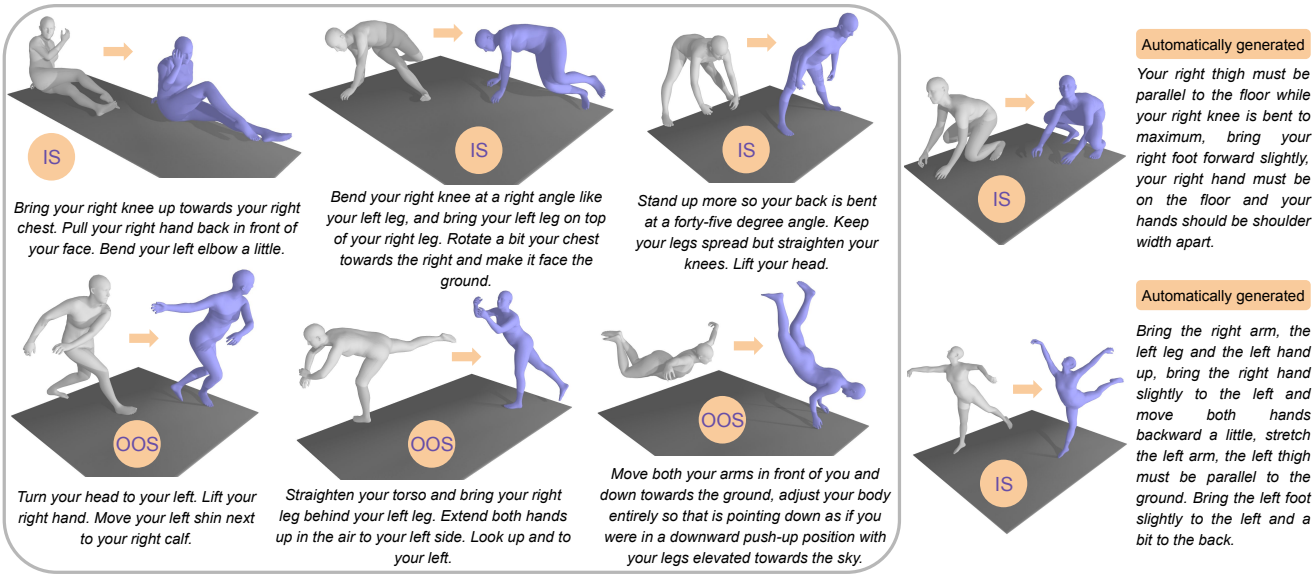

Figure 2: **Examples of pose pairs and their annotated modifier in PoseFix.** The source pose is shown in gray and the target pose in purple. Poses from in-sequence (*IS*) pairs are from the same motion clip; unlike out-of-sequence (*OOS*) pairs.

to learn here can be leveraged for other modalities (*e.g.* images) or in other settings (*e.g.* robot teaching).

While the link between language and images has been extensively studied in tasks like image captioning [34, 22] or image editing [66], the research on leveraging natural language for 3D human modeling is still in its infancy. A few works use textual descriptions to generate motion [18, 56], to describe the difference in poses from synthetic *2D* renderings [26] or to describe a single static pose [12]. Nevertheless, there currently exists no dataset that associates pairs of 3D poses with textual instructions to move from one source pose to one target pose. In this work, we thus introduce the PoseFix dataset, which contains over 6,000 textual *modifiers* written by human annotators for this scenario. In addition, we design a pipeline similar to [12], to generate modifiers automatically and increase the size of the data, see Figure 2 for some examples.

Leveraging the PoseFix dataset, we tackle two tasks: *text-based pose editing*, where the goal is to generate new poses from an initial pose and modification instructions, and *correctional text generation* where the objective is to produce a textual description of the difference between a pair of poses (see Figure 1). For the first task, we use a baseline consisting in a conditional Variational Auto-Encoder (cVAE). For the second, we consider a baseline built from an auto-regressive transformer model. We provide a detailed evaluation of both baselines, and show promising results.

In summary, our contributions are threefold:

- We introduce the PoseFix dataset (Section 3) that associates pairs of 3D human poses and human-written textual descriptions of their differences.
- We introduce the task of text-based pose editing (Section 4), that can be tackled with a cVAE baseline.
- We study the task of correctional text generation with a conditioned auto-regressive model (Section 5).

2. Related Work

3D pose and text datasets. AMASS [38] gathers several datasets of 3D human motions in SMPL [37] format. BABEL [49] and HumanML3D [18] build on top of it to provide free-form text descriptions of the sequences, similarly to the earlier and smaller Kit Motion-Language dataset [48]. These datasets focus on sequence semantic (high-level actions) rather than individual pose semantic (fine-grained egocentric relations). To complement, PoseScript [12] links static 3D human poses with descriptions in natural language about fine-grained pose aspects. However, PoseScript does not make it possible to relate two poses together in a straightforward way, as we attempt by introducing the new PoseFix dataset. In contrast to FixMyPose [26], the PoseFix dataset we introduce comprehends poses from more diverse sequences and the textual annotations were collected based on actual 3D data and not synthetic 2D image renderings (reduced depth ambiguity).

3D human pose generation. Previous works have mainly focused on the generation of pose sequences, conditioning on music [30, 31], context [10], past poses [63, 64], text labels [20, 46] and mostly on text descriptions [32, 1, 61, 2, 17, 47, 18, 56, 19, 27]. Some works push it one step further and also attempt to synthesize the mesh appearance [24, 62], leveraging large pretrained models like CLIP [50]. Similarly to PoseScript [12], we depart from generic actions and

focus on static poses and fine-grained aspects of the human body, to learn about precise egocentric relations. However, we consider two poses instead of one to comprehend detailed pose modifications. Different from ProtoRes [40], which proposes to manually design a human pose inside a 3D environment based on sparse constraints, we use text for controllability. As PoseScript and VPoser [44], an (unconditioned) pose prior, we use a VAE-based [29] model to generate the 3D human poses.

Pose correctional feedback generation. Recent advances in text generation have led to a shift from recurrent neural networks [55] to large pretrained transformer models, such as GPT [8]. These models can be effectively conditioned using prompting [41] or cross-attention mechanisms [51]. While multi-modal text generation tasks, such as image captioning, have been extensively studied [34, 22, 58] no previous work has focused on using 3D human poses to generate free-form feedback. In this regard, AIFit [15] extracts 3D data to compare the video performance of a trainee against a coach's and provides feedback based on predefined templates. [65] also outputs predefined texts for a small set of exercises and [35] does not provide any natural language instructions, either. Besides, FixMyPose [26] is based on highly-synthetic 2D images.

Compositional learning consists in using a query made of multiple distinct elements, which can be of different modalities, as for visual question answering [4] or composed image retrieval [59]. Similarly to the latter, we are interested in bi-modal queries composed of a textual "modifier" which specifies changes to apply on the first element. Modifiers first took the form of single-word attributes [43, 39, 14] and evolved into free-form texts [60, 36]. While a large body of works focus on text-conditioned image editing [25, 7, 23] or text-enhanced image search [59, 5, 11], few study 3D human body poses. ClipFace [3] proposes to edit 3D morphable face models and StyleGAN-Human [16] generates 2D images of human bodies in very model-like poses. PoseTutor [13] provides an approach to highlight joints with incorrect angles on 2D yoga/pilate/kung-fu images. More related to our work, FixMyPose [26] performs composed image retrieval. Conversely to them, we propose to *generate* a *3D* pose based on an initial static pose and a modifier expressed in natural language.

3. The PoseFix dataset

To tackle the two pose correctional tasks considered in this paper, we introduce the new PoseFix dataset. It consists of 135k triplets of {*pose A*, *pose B*, *text modifier*}, where pose B (the target pose) is the result of the correction of pose A (the source pose), as specified by the *text modifier*. The 3D human body poses were sampled from AMASS [38]. All pairs were captioned in Natural Language thanks to our automatic comparative pipeline; 6157 pairs were additionally presented to human annotators on the crowd-source annotation platform Amazon Mechanical Turk. We next present the pair selection method, the annotations process and some dataset statistics.

3.1. Pair selection process

In- and Out-of-sequence pairs. Pose pairs can be of two types: "*in-sequence*" (IS) or "*out-of-sequence*" (OOS). In the first case, the two poses belong to the same AMASS sequence and are temporally ordered (pose A happens before pose B). We select them with a maximum time difference of 0.5 second, to have both textual modifiers describing precisely atomic motion sub-sequences and ground-truth motion. For an increased time difference between the two poses, they could be an infinity of plausible in-between motions, which would weaken such supervision signal. Out-of-sequence pairs are made of two poses from different sequences; to help generalize to less common motions and to study poses of similar configuration but different style, empowering "pose correction" beside "motion continuation".

Selecting pose B. As we aim to obtain pose B from pose A, we consider that pose B is guiding the most the annotation: while the text modifier should account for pose A and refer to it, its true target is pose B. Thus, to build the triplets, we first choose the set of poses B. So to maximize the diversity of poses, we follow [12], and get a set S of 100k poses sampled with a farthest-point algorithm. Poses B are then iteratively selected from S.

Selecting pose A. The paired poses should satisfy two main constraints. First, poses A and B should be similar enough for the text *modifier* not to become a complete *description* of pose B: if *A* and *B* are too different, it is easier for the annotator to just ignore *A* and directly characterize *B* [60, 36]. Yet, we aim at learning fine-grained and subtle differences between two poses. Hence, we rank all poses in S with regard to each pose B based on the cosine similarity of their PoseScript semantic pose features [12]. Pose A is to be selected within the top 100. Second, the two poses should be different enough, so that the modifier does not collapse to oversimple instructions like '*raise your right hand*', which would not compare to realistic scenarios. While we expect the poses to be quite different as they belong to S, we go one step further and leverage posecode information [12] to ensure that the two poses have at least 15 (resp. 20) low-level different properties for IS (resp. OOS) pairs.

One- and Two-way pairs. We consider all possible IS pairs $A \rightarrow B$, with A and B in S, that meet the selection constraints. Then, following the order defined by S, we sample OOS pairs: for each selected pair $A \rightarrow B$, if A was not already used for another pair, we also consider $B \rightarrow A$. We call such pairs '*two-way*' pairs, as opposed to '*one-way*' pairs. Two-way pairs could be used for cycle consistency.

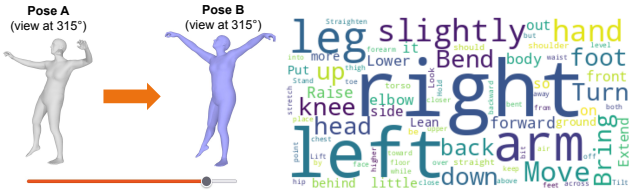

Figure 3: **Left: Data presented to the annotators.** The slider makes it possible to look at the poses under different viewpoints. **Right: word cloud** of the PoseFix annotations.

Property	Proportion	Example
Egocentric relations	74%	*Join your hands **in front of** your chest.*
Analogies	5%	*... **like you're** about to clap your hands.*
Implicit side description	25%	*Place your left toes on the ground and extend your Ø leg slightly.*

Table 1: **Semantic analysis** on 104 sampled human texts.

Splits. We use the same sequence-based split as [12], and perform pose pair selection independently in each subset. Since we also use the same ordered set S, some poses are annotated both with a description and a modifier: such complementary information can be used in a multitask setting.

3.2. Collection of human annotations

We collected the textual modifiers on Amazon Mechanical Turk from English-speaking annotators with a 95% approval rate who already completed at least 5000 tasks. To limit perspective-based mistakes, we presented both poses rendered under different viewpoints (see Figure 3, left). An annotation could not be submitted until it was more than 10 words and several viewpoints were considered. The orientation of the poses was normalized so they would both face the annotator in the front view. Only for in-sequence pairs, we would apply the normalization of pose A to pose B, to stay faithful to the global change of orientation in the ground-truth motion sequences.

The annotators were given the following instruction: "*You are a coach or a trainer. Your student is in pose A, but should be in pose B. Please write the instructions so they can correct the pose on at least 3 aspects.*". Annotators were required to describe the position of the body parts relatively to the others (*e.g.* '*Your right hand should be close to your neck.*'), to use directions (such as '*left*' and '*right*') in the subject's frame of reference and to mention the rotation of the body, if any. They were also encouraged to use analogies (*e.g.* '*in a push-up pose*'). For the annotations to size-agnostic, distance metrics were prohibited.

The task was first made available to any worker by tiny batches. Annotations were carefully scrutinized, and only the best workers were qualified to pursue to larger batches, with lighter supervision. In total, about 15% of the annotations were manually reviewed, and corrected when needed. We further cleaned the annotations by fixing misspelled and duplicated words, detected automatically. Figure 2 shows some pose pairs and their annotated modifiers.

	automatic	*human*
in-sequence	25,201	2,615
out-of-sequence	110,104	3,542
both-way	93,180	2,710
one-way	42,125	3,447
total	135,305	6,157

Table 2: **Number of pairs** of each set and type.

	automatic	*human*
different poses	99,231	7,433
different poses A	87,793	5,343
different poses B	98,939	5,922
in PoseScript	6,249	3,551
A in PoseScript	6,160	2,753
B in PoseScript	6,226	3,143

Table 3: **Number of poses** per type or shared with [12].

3.3. Generating annotations automatically

To scale up the dataset, we design a pipeline to automatically generate thousands of modifiers, by relying on low-level properties as in [12]. The process takes as input the 3D keypoint positions of two poses A and B, and outputs a textual instruction to obtain pose B from pose A. First, it measures and classifies the variation of atomic pose configurations to obtain a set of "*paircodes*". For instance, we attend to the motion of the keypoints along each axis ("*move the right hand slightly to the left*" (x-axis), "*lift the left knee*" (y-axis)), to the variation of distance between two keypoints ("*move your hands closer*") or to the angle change ("*bend your left elbow*"). We further define "*super-paircodes*", resulting from the combination of several paircodes or posecodes [12]; *e.g.* the paircode "*bend the left knee less*", associated to the posecode "*the left knee is slightly bent*" on pose A leads to the super-paircode "*straighten the left leg*". The super-paircodes make it possible to describe higher-level concepts or to refine some assessments (*e.g.* only tell to move the hands farther away from each other if they are close to begin with). The paircodes are next aggregated using the same set of rules as in [12], then they are structurally ordered, to gather information about the same general part of the body within the description. Ultimately, for each paircode, we sample and complete one of the associated template sentences. Their concatenation yields the automatic modifier. Please refer to the supplementary for more details. The whole process produced 135k annotations in less than 15 minutes. Some examples are shown in Figure 2. In this paper, we use the automatic data for pretraining only.

3.4. Statistics and semantic analysis

PoseFix contains 6157 (resp. 135k) human- (resp. automatically-) annotated pairs, split according to a 70%-10%-20% proportion. In average, human-written text modifiers are close to 30 words long with a minimum of 10 words. All together, they form a cleaned vocabulary of 1068 words, a wordcloud of which is shown in Figure 3 (right).

Negation particles were detected in 3.6% of the annotations, which makes textual queries with negations a bit harder, akin to similar datasets [60, 12]. A semantic analysis carried out on 104 annotations taken at random is reported in Table 1. We found that textual modifiers provide correctional instructions about 4 different body parts in average,

which vary depending on the context (pose A).

A few other annotation behaviors were found to be quite difficult to quantify, in particular "missing" instructions. Sometimes, details are omitted in the text because the context given by pose A is "taken for granted". For instance, in the 3rd example shown in Figure 2, the "*45-degree angle*" is to be understood with regard to the "0 degree" plan defined by the back of the body in pose A. Moreover, the annotator did not specify how the position of the arms have changed, supposedly because this change comes naturally once the back is straighten up, from the structure of the kinematic chain. These challenges are inherent to the task.

Detailed statistics are presented in Tables 2 and 3.

4. Application to Text-based Pose Editing

We introduce a VAE [29] baseline to perform text-based 3D human pose editing. Specifically, we aim to generate plausible new poses based on two input elements: an initial pose A providing some context (a starting point for modifications), and a textual modifier which specifies the changes to be made. Figure 4 gives an overview of our model.

Data processing. Poses are characterized by their SMPL-H [52] body joint rotations in axis-angle representation. Their global orientation is first normalized along the y-axis. For in-sequence pairs, the same normalization that was applied to pose A is applied to pose B in order to preserve information about the change of global orientation.

Training phase. During training, the model encodes both the query pose A and the ground-truth target pose B using a shared pose encoder, yielding respectively features $\mathbf{a}$ and $\mathbf{b}$ in $\mathbb{R}^d$. The tokenized text modifier is fed into a pretrained embedding module to extract expressive word encodings. These are further processed by a learned textual model, to yield a global textual representation $\mathbf{m} \in \mathbb{R}^n$. Next, the two input embeddings $\mathbf{a}$ and $\mathbf{m}$ are provided to a fusing module which outputs a single vector $\mathbf{p} \in \mathbb{R}^d$. Both $\mathbf{b}$ and $\mathbf{p}$ then go through specific fully connected layers to produce the parameters of two Gaussian distributions: the posterior $\mathcal{N}_b = \mathcal{N}(\cdot|\boldsymbol{\mu}(\mathbf{b}), \boldsymbol{\Sigma}(\mathbf{b}))$ and the prior $\mathcal{N}_p = \mathcal{N}(\cdot|\boldsymbol{\mu}(\mathbf{p}), \boldsymbol{\Sigma}(\mathbf{p}))$ conditioned on $\mathbf{p}$ from the fusion of $\mathbf{a}$ and $\mathbf{m}$. Eventually, a sampled latent variable $\mathbf{z}_b \sim \mathcal{N}_b$ is decoded into a reconstructed pose $\hat{B}$.

The loss consists in the sum of a reconstruction term $\mathcal{L}_R(B, \hat{B})$ and the Kullback-Leibler (KL) divergence between $\mathcal{N}_b$ and $\mathcal{N}_p$. The former enables the generation of plausible poses, while the latter acts as a regularization term to align the two spaces. The combined loss is then:

$$\mathcal{L}_{\text{pose editing}} = \mathcal{L}_R(B, \hat{B}) + \mathcal{L}_{KL}(\mathcal{N}_b, \mathcal{N}_p). \quad (1)$$

We use the same negative log likelihood-based reconstruction loss as in [12]: it is applied to the output joint rotations in the continuous 6D representation [67], and both the joint and vertices positions inferred from the output by the SMPL-H [52] model.

Inference phase. The input pose A and the text modifier are processed as in the training phase. However, this time we sample $\mathbf{z}_p \sim \mathcal{N}_p$ to obtain the generated pose $\hat{B}$.

Evaluation metrics. We report the Evidence Lower Bound (ELBO) for the size-normalized rotations, joints and vertices, as well as the Fréchet inception distance (FID) which compares the distribution of the generated poses with the one of the expected poses, based on their semantic PoseScript features. The ELBO and the FID are mostly sensitive to complementary traits (support coverage and sample quality respectively). When some settings do not improve all metrics; we then base our decisions on the metrics with the highest differential. While the ELBO is better suited to evaluate generative models than reconstruction metrics, for intuitiveness, we also report the the MPJE (mean-per-joint error, in mm), the MPVE (mean-per-vertex error, in mm) and the geodesic distance for joint rotations (in degrees) between the target and the best (*i.e.*, closest) generated sample out of N=30 in all experiments.

Architecture details and ablations. We use the VPoser [44] architecture for the pose auto-encoder, resulting in features of dimension $d = 32$. The variance of the decoder is considered a learned constant [53]. We experiment with two different text encoders (Table 4, top): (i) a bi-GRU [9] mounted on top of pretrained GloVe word embeddings [45], or (ii) a transformer followed by average-pooling, processing frozen DistilBERT [54] word embeddings. We find that the transformer pipeline outperforms the other in terms of ELBO (+0.24 in average) when no additional pretraining is involved, supposedly because it uses already strong general-pretrained weights. Pretraining on our automatic modifiers brings the bi-GRU pipeline on par with the transformer one (+0.04). For simplicity, we will thereafter resort to the former.

For fusion, we use TIRG [59], a well-spread module for compositional learning. It consists in a gating mechanism composed of two 2-layer Multi-Layer Perceptrons (MLP) f and g balanced by learned scalars w_f and w_g such that:

$$\mathbf{p} = w_f f([\mathbf{a}, \mathbf{m}]) \odot \mathbf{a} + w_g g([\mathbf{a}, \mathbf{m}]). \quad (2)$$

It is designed to 'preserve' the main modality feature $\mathbf{a}$ while applying the modification as a residual connection.

Training data and augmentations ablations. We experiment with several kinds of data augmentations and training data. Corresponding results are reported in Table 4 (bottom). First, we try left/right flipping by swapping the rotations of the left and right body joints (*e.g.* the left hand becomes the right hand) and changing the text accordingly. This improves significantly the relevance of the generated poses (ELBO), especially when the model did not benefit

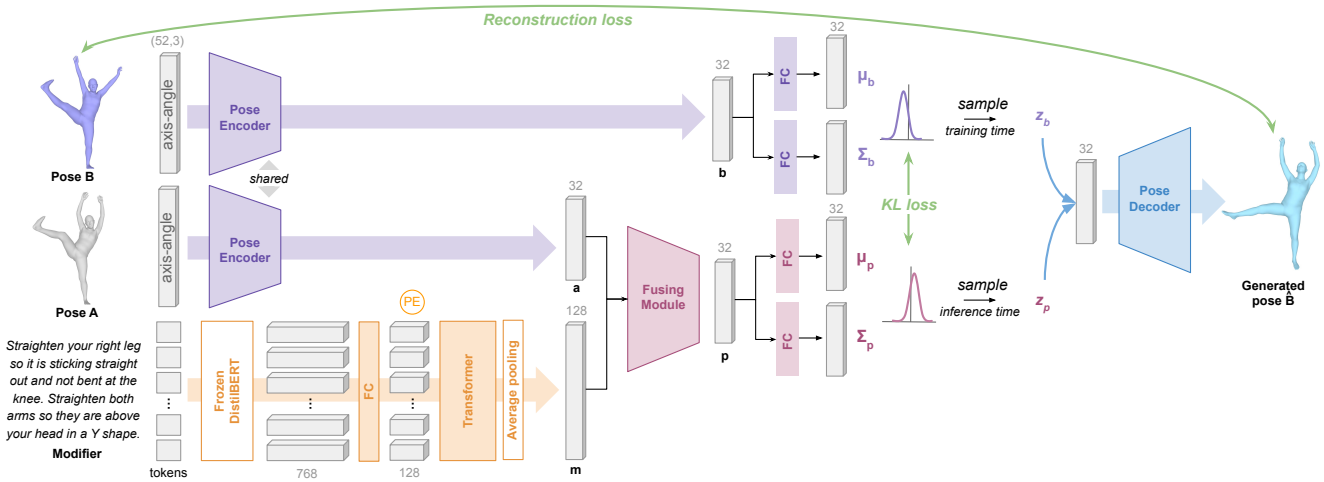

Figure 4: **Overview of our text-based pose editing baseline.** The top part represents a standard VAE, where poses are encoded into a Gaussian distribution. At training time, a latent variable is sampled and decoded into a pose to learn pose reconstruction. The bottom left part represents the conditioning: the text is encoded using a frozen DistilBERT with a small transformer on top. It is combined with source pose features in the *fusion module*, from which we predict a Gaussian distribution. A KL loss ensures the alignment of the distributions from the standard VAE and the conditioning. At test time, we sample from the latter to predict the target pose.

		FID ↓	ELBO ↑			Reconstruction ↓ *(best of 30)*		
			jts	v2v	rot	MPJE	MPVE	Geodesic
Text Encoder *(with/without pretraining)*								
without	GloVe + bi-GRU	0.19	0.61	**1.51**	0.50	278	217	9.89
	DistilBERT+transformer	**0.10**	**0.95**	**1.51**	**0.63**	**226**	**180**	**9.22**
with	GloVe + bi-GRU	**0.02**	**1.40**	**1.88**	**0.99**	**199**	**165**	**8.59**
	DistilBERT+transformer	**0.02**	1.37	1.84	0.93	201	167	8.70
Data augmentations *(with/without pretraining, GloVe+bi-GRU config)*								
without	no augmentation	0.19	0.61	1.51	0.50	278	217	9.89
	+ L/R flip	0.13	**1.10**	**1.73**	0.58	250	196	9.57
	+ paraphrases	0.19	0.90	1.45	0.58	233	186	9.44
	+ PoseMix	0.10	0.63	1.12	0.58	254	202	9.61
	+ PoseMix & PoseCopy	**0.04**	1.03	1.50	**0.78**	**221**	**178**	**9.07**
with	no augmentation	**0.02**	1.40	1.88	**0.99**	199	165	8.59
	+ L/R flip	**0.02**	**1.47**	**1.94**	0.97	197	163	8.65
	+ paraphrases	**0.02**	1.43	1.90	0.97	198	164	**8.58**
	+ PoseMix	0.06	0.68	1.13	0.91	214	174	8.74
	+ PoseMix & PoseCopy	0.03	1.23	1.71	0.98	208	172	8.75
	+ L/R flip & paraphrases	**0.02**	1.44	1.92	0.97	**196**	**162**	8.62

Table 4: **Text-based pose editing results** for various architectures, data augmentations and training strategies. We show the best result in bold and underline the second best.

from pretraining on diverse synthetic data (+37% average improvement of the ELBO).

Next, we use InstructGPT [41] to obtain 2 paraphrases per annotation. This form of data augmentation was found helpful, particularly when training on a small amount of data, *i.e.*, without pretraining (+20%).

In order to encourage the model to fully leverage the textual cue, we define *PoseMix*, which gathers both the PoseScript [12] and the PoseFix datasets. When training with PoseScript data, which consist in pairs of poses and textual descriptions, we set pose A to 0. We notice a mitigated improvement, and even a drop in performance in the pretrained case. One possible reason for that is the difference in formulation between PoseScript descriptions ("*The person is ... with their left hand...*") and PoseFix modifiers ("*Move your left hand...*"). Another is that the model then learns to ignore A, which is nonetheless crucial in the PoseFix setting. To circumvent this last-mentioned issue, we improve the balance of the training data by introducing *PoseCopy*. This consists in providing the model with the same pose in the role of pose A and pose B, along with an empty modifier, assuming that a non-existent textual query will force the model to attend pose A. The *PoseMix & PoseCopy* setting yields a great improvement over all metrics for the non-pretrained case (+41%). This further shows that the formulation gap was not the main issue. As a side product, the fusing branch is now able to work as a pseudo auto-encoder, and to output a copy of the input pose when no modification instruction is provided.

Eventually, the pretraining has a more significant impact than using any kind of data augmentation (+84%). Besides, the data augmentations become much less effective in this setting (+1%). The model thus benefits better from pretraining on a large set of new pairs with synthetic instructions, than training on more human-written modifiers of the same pose pairs. We overall obtain our best model by combining pretraining, left/right flip and paraphrases (last row).

Detailed analysis. In Table 5, we evaluate our best pose editing model on several subsets of pairs and with different input types.

First, we notice higher ELBO performance on the out-of-sequence (OOS) pair set compared to the in-sequence (IS) set, suggesting that pairs in the latter are harder. This can

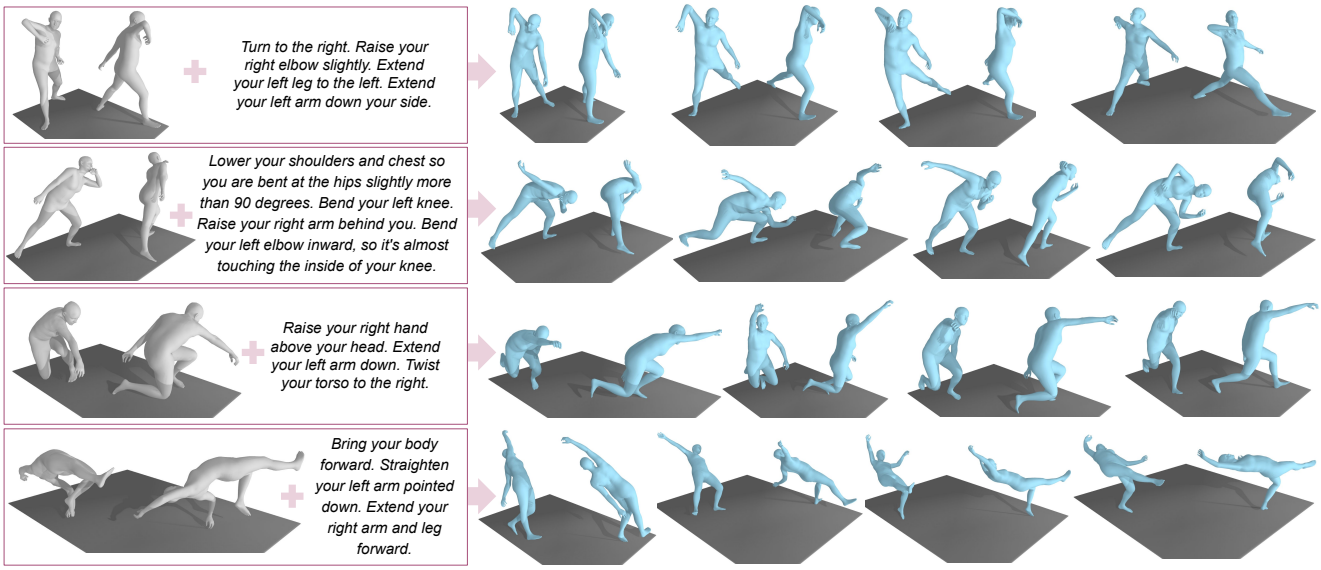

Figure 5: **Generated poses for the text-based pose editing task** on PoseFix queries from the left blocks. Two views of each pose are shown on the same ground plane for better visualization of the 3D. Generated poses are shown in blue. Original poses B from the PoseFix dataset are in the supplementary material.

	FID ↓	ELBO ↑			Reconstruction ↓ *(best of 30)*		
		jts	v2v	rot	MPJE	MPVE	Geodesic
Pair subset							
in-sequence (530)	0.04	1.33	1.78	0.88	**188**	**154**	**8.47**
out-of-sequence (709)	0.03	**1.53**	**2.02**	**1.04**	206	168	8.80
full PoseFix test set (1239)	**0.02**	1.44	1.92	0.97	196	162	8.62
Input type *(full PoseFix test set - 1239)*							
pose A only	0.04	1.43	**1.92**	**0.97**	219	180	8.91
modifier only	0.42	1.30	**1.92**	0.92	378	339	13.03
pose A + modifier	**0.02**	**1.44**	**1.92**	**0.97**	**196**	**162**	**8.62**

Table 5: **Pose editing results for various subsets and input types**, using the best model as per Table 4.

be due to pose A and pose B being more similar in IS than OOS, as they belong to the same sequence with a maximum delay of 0.5s. We indeed measure a mean per joint distance of 311mm between A and B in IS *vs*. 350mm in OOS: the differences between IS poses thus ought to be more subtle, yielding more complex modifiers. This drop in ELBO performance shows also that the model struggles more with IS modifiers, meaning that it most probably generates, in average, poses that are close to pose A, – in other words, it would takes guesses in the surroundings of pose A. This would actually be a good fall-back strategy, because the two poses are rather similar in general. In the IS case, since pose A and pose B are particularly close to each other, the model may end up finding, with enough guesses, a pose closer to pose B than it would in the OOS case, where the two poses are more different. This could explain why the reconstruction metrics using the best sample out of 30 are lower for the IS subset than the OOS subset.

Next, we compare the results when querying with the *pose A only* or the *modifier only*. The former achieves already high performance, showing that the initial pose A alone provides a good approximation of the expected pose B – indeed, the pair selection process constrained pose A and pose B to be quite similar. The latter yields poor FID and reconstruction metrics: the textual cue is only a modifier, and the same instructions could apply to a large variety of poses. Looking around pose A remains a better strategy than sticking to the sole modifier in order to generate the expected pose. Eventually, both parts of the query are complementary: pose A serves as a strong contextual cue, and the modifier guides the search starting from it (the pose being provided through the gating mechanism in TIRG). Both are crucial to reach pose B (last row).

Qualitative results. Last, we present qualitative results for text-based 3D human pose editing in Figure 5. It appears that the model has a relatively good semantic comprehension of the different body parts and of the actions to modify their positions. Some egocentric relations ("*Raise your right elbow slightly.*", first row) are better understood than others, in particular contact requirements ("*Bend your elbow so it's almost touching the inside of your knee*", second row). When missing some specifications, the model generates various pose configurations (*e.g.* the extent of the left leg extension in the first example). It can handle a number of instructions at once (third row), but may fail to attend all of them. Crouching and lying-down poses are the most challenging (see failure case in the last row, and how the crouch is hardly preserved in the third row).

5. Application to correctional text generation

We next present a baseline for correctional text generation. We aim to produce feedback in natural language ex-

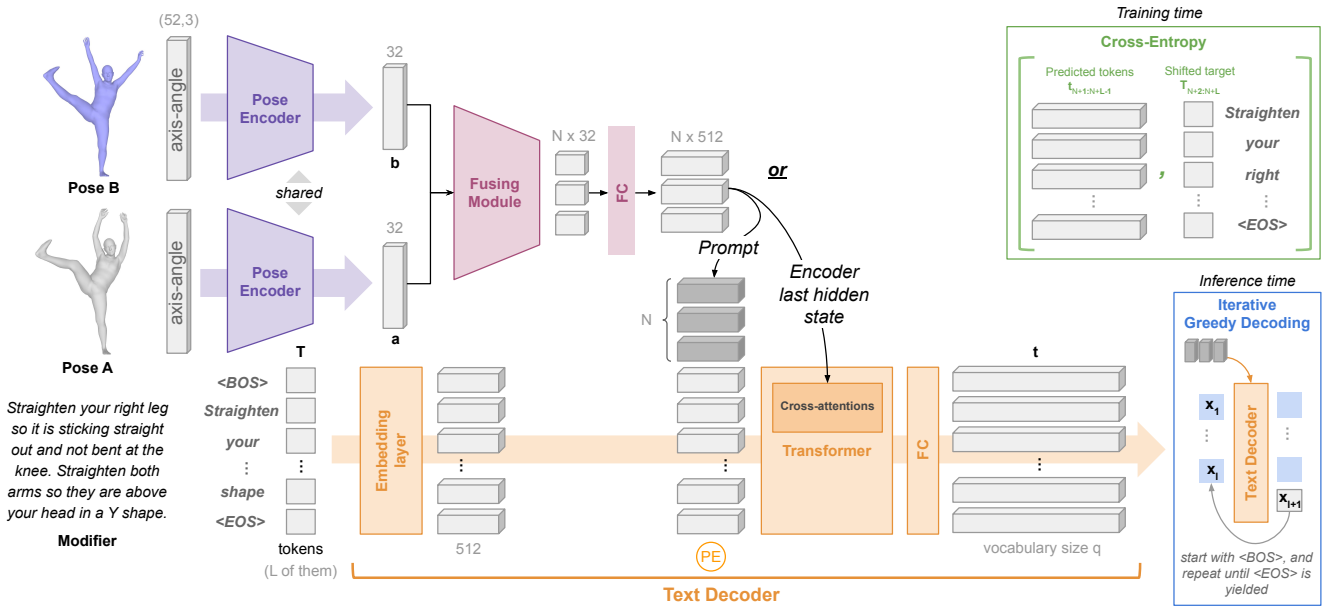

Figure 6: **Overview of our baseline for correctional text generation.** The bottom part represents a standard auto-regressive transformer model: the next word is predicted from the previously generated tokens. The decoder outputs a distribution of probabilities over the vocabulary for each token. The top part represents the conditioning on the pose pair: the two pose embeddings are fused together into a set of "pose tokens", further used for conditioning via prompting or via cross-attentions in the transformer. At inference, the modifier is generated iteratively using the greedy approach.

plaining how the source pose A should be modified to obtain the target pose B. We rely on an auto-regressive model conditioned on the pose pair, which iteratively predicts the next word given the previous generated ones (see Fig. 6).

Training phase. Let $T_{1:L}$ be the L tokens of the text modifier. An auto-regressive generative model seeks to predict the next token $l+1$ from the first l tokens $T_{1:l}$. Let $p(\cdot|T_{1:l})$ be the predicted probability distribution over the vocabulary. The model is trained, via a cross-entropy loss, to maximize the probability of generating the ground-truth token T_{l+1} given previous ones: $p(T_{l+1}|T_{1:l})$.

To predict $p(\cdot|T_{1:l})$, the tokens $T_{1:l}$ are first embedded, and positional encodings are injected. The result is fed to a series of transformer blocks [57], and projected into a space whose dimension is the vocabulary size q. Let $\mathbf{t} \in \mathbb{R}^q$ denote the outcome. The probability distribution over the vocabulary for the next token $p(\cdot|T_{1:l})$ could be obtained from $Softmax(\mathbf{t})$.

Transformer-based auto-regressive models can be trained efficiently using causal attention masks which, for each token l, prevent the network from attending all future tokens $l' > l$, in a single pass.

Now, how do poses come into the picture? Pose A and pose B are encoded using a shared encoder, and combined in the fusing module, which outputs a set of N 'pose' tokens. To condition the text generation on pose information, we experiment with two alternatives: those pose tokens can either be used for prompting, *i.e.*, added as extra tokens at the beginning of the text modifier, or serve in cross-attention mechanisms within the text transformer.

Inference phase. For inference, we provide the model with the pose tokens and the special $<BOS>$ token which indicates the sequence beginning. We decode the output $\mathbf{t}_2$ in a greedy fashion, *i.e.*, we predict the next token as the word that maximizes the negative log likelihood. We proceed iteratively, giving previously decoded tokens $T_{1:l}$ to the model so to obtain the subsequent token $l+1$, until the $<EOS>$ token (denoting the end of the sequence) is decoded.

Evaluation metrics. We resort to standard natural language metrics: BLEU-4 [42], Rouge-L [33] and METEOR [6], which measure different kinds of n-grams overlaps between the reference text and the generated one. Yet, we notice that these metrics do not reliably reflect the model quality for this task. Indeed, we only have one reference text and, given the initial pose, very different instructions can lead to the same result (*e.g.* "*lower your arm at your side*" and "*move your right hand next to your hip*"); it is not just a matter of formulation. Thus, we report the top-k R-precision metrics proposed in TM2T [19]: we use contrastive learning to train a joint embedding space for the modifiers and the concatenation of poses A and B, then we look at the ranking of the correct pose pair for each generated text within a set of 32 pose pairs. We also report reconstruction metrics on the pose generated thanks to our best model from Section 4 using the generated text. These added metrics assess the semantic correctness of the generated texts.

		R Precision ↑			NLP ↑			Reconstruction ↓ *(best of 30)*		
		R@1	R@2	R@3	BLEU-4	ROUGE-L	METEOR	MPJE	MPVE	Geodesic
Control measures										
random text		3.13	6.25	9.38	7.11	26.33	26.88	225	185	9.07
original text		62.71	74.01	79.26	100.00	100.00	100.00	196	162	8.62
Injection type *(with/without pretraining)*										
without	prompt	3.63	7.10	10.73	9.74	**31.88**	27.72	226	184	8.94
	cross-attention	**6.78**	**12.27**	**17.35**	**10.62**	31.66	**28.74**	**220**	**180**	**8.85**
with	prompt	15.09	22.28	30.35	11.15	32.58	29.76	211	175	8.79
	cross-attention	**58.43**	**71.35**	**77.56**	**12.19**	**33.94**	**31.30**	**192**	**161**	**8.55**
Data augmentations *(with pretraining & cross-attention injection)*										
no augmentation		58.43	71.35	77.56	**12.19**	33.94	**31.30**	192	161	8.55
with L/R flip		**60.69**	**71.51**	**78.85**	12.14	**34.02**	30.90	**189**	**159**	**8.54**
with paraphrases		53.91	67.72	74.98	10.56	33.07	30.15	194	162	8.65
with PoseMix		45.12	56.66	64.89	10.94	33.22	30.12	197	164	8.74

Table 6: **Correctional text generation results** for various pose injections and data augmentations. For reference, we also provide numbers for the ground-truth texts and an annotated text chosen at random.

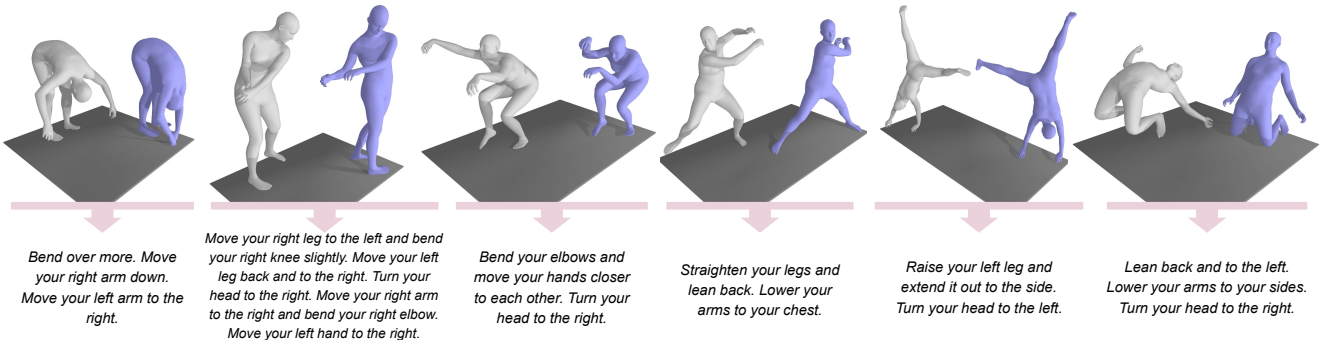

Figure 7: **Generated correctional texts** for PoseFix pose pairs (pose A is grey, pose B is purple). The original human annotations for these pose pairs are available in the supplementary material.

Quantitative results are presented in Table 6. We experiment with the same fusing module as before: TIRG [59], where the gating applied on the pair leading pose (pose B); thus using $N = 1$. We try prompting and cross-attention to inject the pose information in the text decoder, and found the latter to yield the best results. Pretraining on automatic modifiers significantly boosts the performance, *e.g.* with cross-attention injection, the R@2 increases from 12.27% to 71.35%. Regarding data augmentations, the left/right flip yields additional gains (+1.7% of average R Precision) with results close to those obtained with the ground-truth texts, both for R-precision and reconstruction. Even if the generated text does not have the same wording as the original text (low NLP metrics), combined with pose A, it achieves to produce a satisfactory pose $\hat{B}$, meaning that it carries the right correctional information. Of course, one should recall that the added metrics rely on imperfect models, which have their own limitations. Finally, we observe a decrease in performance with the paraphrases or the PoseMix settings: we hypothesize that these settings are harder than the regular one for this task, due to new words and formulations.

Qualitative results. Fig. 7 shows some generated texts. The model is able to produce satisfying feedback, it generates egocentric relations (third and fourth examples) and groups indications by body part (second column). However, it tends to mix up pose A and B (last two examples). It also sometimes describes only a subset of the differences.

6. Conclusion

This paper lays the groundwork for investigating the challenge of correcting 3D human poses using natural language instructions. Going beyond existing methods that utilize language to model global motion or entire body poses, we aim to capture the subtle differences between pairs of body poses, which requires a new level of semantic understanding. For this purpose, we have introduced PoseFix, a novel dataset with paired poses and their corresponding correctional descriptions. We also presented promising results for two baselines which address the deriving tasks of text-based pose editing and correctional text generation.

Acknowledgments. This work is supported by the Spanish government with the project MoHuCo PID2020-120049RB-I00, and by NAVER LABS Europe under technology transfer contract 'Text4Pose'.

References

[1] Hyemin Ahn, Timothy Ha, Yunho Choi, Hwiyeon Yoo, and Songhwai Oh. Text2action: Generative adversarial synthesis from language to action. In *ICRA*, 2018. 2

[2] Chaitanya Ahuja and Louis-Philippe Morency. Language2pose: Natural language grounded pose forecasting. *3DV*, 2019. 2

[3] Shivangi Aneja, Justus Thies, Angela Dai, and Matthias Nießner. Clipface: Text-guided editing of textured 3d morphable models. In *SIGGRAPH*, 2023. 3

[4] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *ICCV*, 2015. 3

[5] Alberto Baldrati, Marco Bertini, Tiberio Uricchio, and Alberto Del Bimbo. Conditioned and composed image retrieval combining and partially fine-tuning clip-based features. In *CVPRW*, 2022. 3

[6] Satanjeev Banerjee and Alon Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, 2005. 8

[7] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *CVPR*, 2023. 3

[8] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *NeurIPS*, 2020. 3

[9] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *EMNLP*, 2014. 5

[10] Enric Corona, Albert Pumarola, Guillem Alenya, and Francesc Moreno-Noguer. Context-aware human motion prediction. In *CVPR*, 2020. 2

[11] Ginger Delmas, Rafael Sampaio de Rezende, Gabriela Csurka, and Diane Larlus. Artemis: Attention-based retrieval with text-explicit matching and implicit similarity. In *ICLR*, 2022. 3

[12] Delmas, Ginger and Weinzaepfel, Philippe and Lucas, Thomas and Moreno-Noguer, Francesc and Rogez, Grégory. PoseScript: 3D Human Poses from Natural Language. In *ECCV*, 2022. 2, 3, 4, 5, 6, 13

[13] Bhat Dittakavi, Divyagna Bavikadi, Sai Vikas Desai, Soumi Chakraborty, Nishant Reddy, Vineeth N Balasubramanian, Bharathi Callepalli, and Ayon Sharma. Pose tutor: An explainable system for pose correction in the wild. In *CVPR*, 2022. 3

[14] Hazel Doughty, Ivan Laptev, Walterio Mayol-Cuevas, and Dima Damen. Action Modifiers: Learning from Adverbs in Instructional Videos. In *CVPR*, 2020. 3

[15] Mihai Fieraru, Mihai Zanfir, Silviu Cristian Pirlea, Vlad Olaru, and Cristian Sminchisescu. AIFit: Automatic 3D human-interpretable feedback models for fitness training. In *CVPR*, 2021. 3

[16] Jianglin Fu, Shikai Li, Yuming Jiang, Kwan-Yee Lin, Chen Qian, Chen Change Loy, Wayne Wu, and Ziwei Liu. Stylegan-human: A data-centric odyssey of human generation. In *ECCV*, 2022. 3

[17] Anindita Ghosh, Noshaba Cheema, Cennet Oguz, Christian Theobalt, and Philipp Slusallek. Synthesis of compositional animations from textual descriptions. In *ICCV*, 2021. 2

[18] Chuan Guo, Shihao Zou, Xinxin Zuo, Sen Wang, Wei Ji, Xingyu Li, and Li Cheng. Generating diverse and natural 3d human motions from text. In *CVPR*, 2022. 2

[19] Chuan Guo, Xinxin Zuo, Sen Wang, and Li Cheng. Tm2t: Stochastic and tokenized modeling for the reciprocal generation of 3d human motions and texts. In *ECCV*, 2022. 2, 8

[20] Chuan Guo, Xinxin Zuo, Sen Wang, Shihao Zou, Qingyao Sun, Annan Deng, Minglun Gong, and Li Cheng. Action2motion: Conditioned generation of 3D human motions. In *ACMMM*, 2020. 2

[21] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 14

[22] Simao Herdade, Armin Kappeler, Kofi Boakye, and Joao Soares. Image captioning: Transforming objects into words. In *NeurIPS*, 2019. 2, 3

[23] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. In *ICLR*, 2023. 3

[24] Fangzhou Hong, Mingyuan Zhang, Liang Pan, Zhongang Cai, Lei Yang, and Ziwei Liu. Avatarclip: Zero-shot text-driven generation and animation of 3d avatars. *ACM TOG*, 2022. 2

[25] Yuming Jiang, Ziqi Huang, Xingang Pan, Chen Change Loy, and Ziwei Liu. Talk-to-edit: Fine-grained facial editing via dialog. In *ICCV*, 2021. 3

[26] Hyounghun Kim, Abhay Zala, Graham Burri, and Mohit Bansal. FixMyPose: Pose correctional captioning and retrieval. In *AAAI*, 2021. 2, 3

[27] Jihoon Kim, Jiseob Kim, and Sungjoon Choi. Flame: Free-form language-based motion synthesis & editing. In *AAAI*, 2023. 2

[28] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 14

[29] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014. 3, 5, 14

[30] Hsin-Ying Lee, Xiaodong Yang, Ming-Yu Liu, Ting-Chun Wang, Yu-Ding Lu, Ming-Hsuan Yang, and Jan Kautz. Dancing to music. In *NeurIPS*, 2019. 2

[31] Ruilong Li, Shan Yang, David A. Ross, and Angjoo Kanazawa. Ai choreographer: Music conditioned 3d dance generation with aist++. In *ICCV*, 2021. 2

[32] Angela S. Lin, Lemeng Wu, Rodolfo Corona, Kevin W. H. Tai, Qixing Huang, and Raymond J. Mooney. Generating animated videos of human activities from natural language descriptions. In *NeurIPS workshops*, 2018. 2

[33] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, 2004. 8

[34] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence

Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 2, 3

[35] Jingyuan Liu, Nazmus Saquib, Zhutian Chen, Rubaiat Habib Kazi, Li-Yi Wei, Hongbo Fu, and Chiew-Lan Tai. Posecoach: A customizable analysis and visualization system for video-based running coaching. *IEEE trans. VCG*, 2022. 3

[36] Zheyuan Liu, Cristian Rodriguez-Opazo, Damien Teney, and Stephen Gould. Image retrieval on real-life images with pre-trained vision-and-language models. In *ICCV*, 2021. 3

[37] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. SMPL: A skinned multi-person linear model. *ACM TOG*, 2015. 2

[38] Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. AMASS: Archive of motion capture as surface shapes. In *ICCV*, 2019. 2, 3, 12

[39] Tushar Nagarajan and Kristen Grauman. Attributes as operators: factorizing unseen attribute-object compositions. In *ECCV*, 2018. 3

[40] Boris N Oreshkin, Florent Bocquelet, Felix G Harvey, Bay Raitt, and Dominic Laflamme. Protores: Proto-residual network for pose authoring via learned inverse kinematics. In *ICLR*, 2021. 1, 3

[41] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*, 2022. 3, 6

[42] Kishore Papineni, Salim Roukos, Todd Ward, and Wei jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, 2002. 8

[43] Devi Parikh and Kristen Grauman. Relative attributes. In *ICCV*, 2011. 3

[44] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed AA Osman, Dimitrios Tzionas, and Michael J Black. Expressive body capture: 3D hands, face, and body from a single image. In *CVPR*, 2019. 3, 5, 14

[45] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014. 5

[46] Mathis Petrovich, Michael J. Black, and Gül Varol. Action-conditioned 3D human motion synthesis with transformer VAE. In *ICCV*, 2021. 2

[47] Mathis Petrovich, Michael J Black, and Gül Varol. Temos: Generating diverse human motions from textual descriptions. In *ECCV*, 2022. 2

[48] Matthias Plappert, Christian Mandery, and Tamim Asfour. The kit motion-language dataset. *Big data*, 2016. 2

[49] Abhinanda R Punnakkal, Arjun Chandrasekaran, Nikos Athanasiou, Alejandra Quiros-Ramirez, and Michael J Black. BABEL: Bodies, action and behavior with english labels. In *CVPR*, 2021. 2

[50] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 2

[51] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. *arXiv preprint arXiv:2212.04356*, 2022. 3

[52] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: Modeling and capturing hands and bodies together. In *SIGGRAPH Asia*, 2017. 5, 14

[53] Oleh Rybkin, Kostas Daniilidis, and Sergey Levine. Simple and effective vae training with calibrated decoders. In *ICML*, 2021. 5

[54] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019. 5

[55] Ilya Sutskever, James Martens, and Geoffrey E Hinton. Generating text with recurrent neural networks. In *ICML*, 2011. 3

[56] Guy Tevet, Brian Gordon, Amir Hertz, Amit H Bermano, and Daniel Cohen-Or. Motionclip: Exposing human motion generation to clip space. In *ECCV*, 2022. 2

[57] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 8

[58] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *CVPR*, 2015. 3

[59] Nam Vo, Lu Jiang, Chen Sun, Kevin Murphy, Li-Jia Li, Li Fei-Fei, and James Hays. Composing text and image for image retrieval-an empirical odyssey. In *CVPR*, 2019. 3, 5, 9

[60] Hui Wu, Yupeng Gao, Xiaoxiao Guo, Ziad Al-Halah, Steven Rennie, Kristen Grauman, and Rogerio Feris. Fashion iq: A new dataset towards retrieving images by natural language feedback. In *CVPR*, 2021. 3, 4

[61] Tatsuro Yamada, Hiroyuki Matsunaga, and Tetsuya Ogata. Paired recurrent autoencoders for bidirectional translation between robot actions and linguistic descriptions. *IEEE RAL*, 2018. 2

[62] Kim Youwang, Kim Ji-Yeon, and Tae-Hyun Oh. Clip-actor: Text-driven recommendation and stylization for animating human meshes. In *ECCV*, 2022. 2

[63] Ye Yuan and Kris Kitani. Dlow: Diversifying latent flows for diverse human motion prediction. In *ECCV*, 2020. 2

[64] Yan Zhang, Michael J. Black, and Siyu Tang. We are more than our joints: Predicting how 3d bodies move. In *CVPR*, 2021. 2

[65] Ziyi Zhao, Sena Kiciroglu, Hugues Vinzant, Yuan Cheng, Isinsu Katircioglu, Mathieu Salzmann, and Pascal Fua. 3d pose based feedback for physical exercises. In *ACCV*, 2022. 3

[66] Xingran Zhou, Siyu Huang, Bin Li, Yingming Li, Jiachen Li, and Zhongfei Zhang. Text guided person image synthesis. In *CVPR*, 2019. 2

[67] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *CVPR*, 2019. 5

Supplementary Material

In this supplementary material, we first provide additional details and statistics on the PoseFix dataset in Section A. The original triplets from PoseFix for the generated results presented in the main paper are available in Section B. Additional visualizations are provided in Section C. Finally, we give implementation details in Section D.

A. PoseFix complementary information

In this section, we provide additional details about the creation of the PoseFix dataset.

A.1. Human annotations

Sequences of origin. The poses in PoseFix were extracted from AMASS [38]. In Figure A1, we present the proportion of poses coming from each of the datasets included in AMASS. We notice that most poses belong to the DanceDB dataset (44%), presumably because this is where the poses are the most diverse. Recall that poses were chosen following a farther-point sampling algorithm to ensure we would get a various subset of poses. Besides, we note that most of the sequences available in DanceDB (94%) and MPI-limits (83%) provided at least one pose to PoseFix, which suggests that PoseFix could help in apprehending very complex, extreme poses.

Turkers qualifications and statistics. The annotations were collected on Amazon Mechanical Turk. Participating workers ("Turkers") had to come from English-speaking countries (Australia, Canada, New Zealand, United Kingdom, USA), have completed at least 5,000 other tasks, and have an approval rate greater than 95%. In total, 105 different annotators participated. We qualified 20 of them for access to the larger batches, on the basis of at least 3 good annotations. Other 50 workers were excluded from our annotation task because of poor writing, misunderstanding of the task or cheating. The remaining participants did not complete enough annotations of good quality to be qualified for accessing more. Eventually, over 90% of the annotations were made by 8 annotators.

Pricing. Properly completing an annotation, after a bit of training, was timed to take approximately 1'10". Annotations from the smaller qualifying batches were rewarded $0.25. Once a worker completed 3 of them correctly, s/he was granted access to the larger batches, where annotations were rewarded $0.32 each, based on the minimum wage in California for 2023. We additionally paid a 10% bonus for every 30 annotations.

Quality assessment. Annotations from the early smaller qualifying batches which were opened to any worker were systematically reviewed. In contrast, only up to 10% of the trusted worker annotations were randomly selected for manual review. The quality of the annotations was assessed based on the following criteria:

- *completeness*: most of the differences between pose A and pose B were addressed in the annotation;
- *direction accuracy*: the annotation explains how to go from pose A to pose B, and not the reverse;
- *left/right accuracy*: the words '*left*' and '*right*' were used in the body's frame of reference;

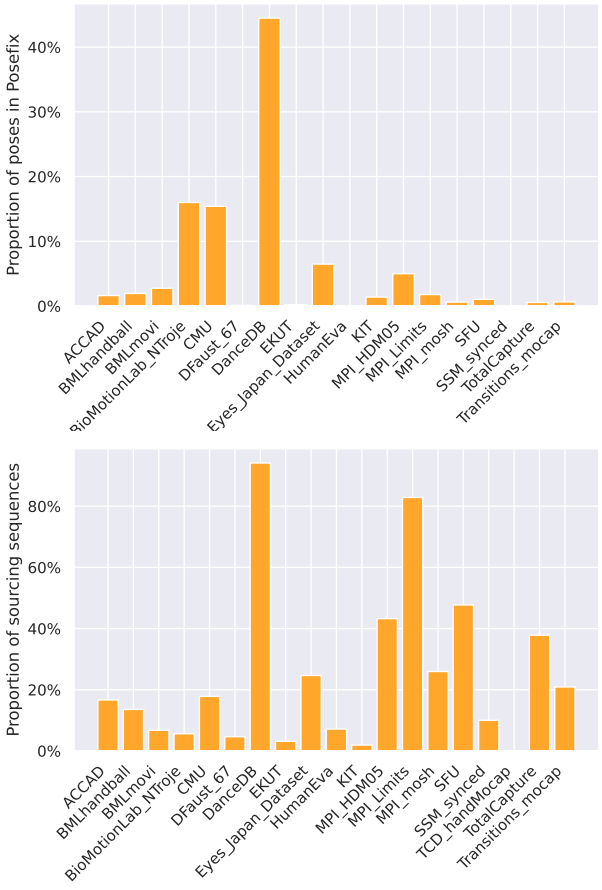

Figure A1: **Origin of the human-annotated poses in PoseFix.** The top plot shows the proportion of poses in PoseFix that come from each sub-dataset in AMASS [38]. The lower plot shows the proportion of sequences, in each of the sub-dataset, that provided at least one pose to PoseFix.

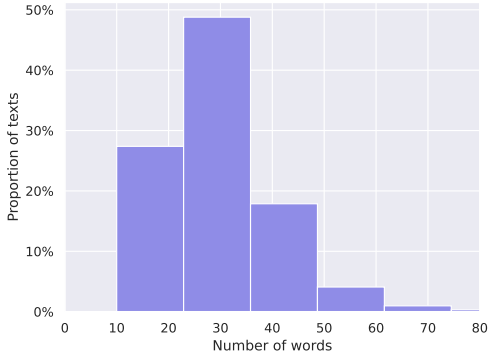

Figure A2: **Distribution of the number of words in the human-written annotations from PoseFix**.

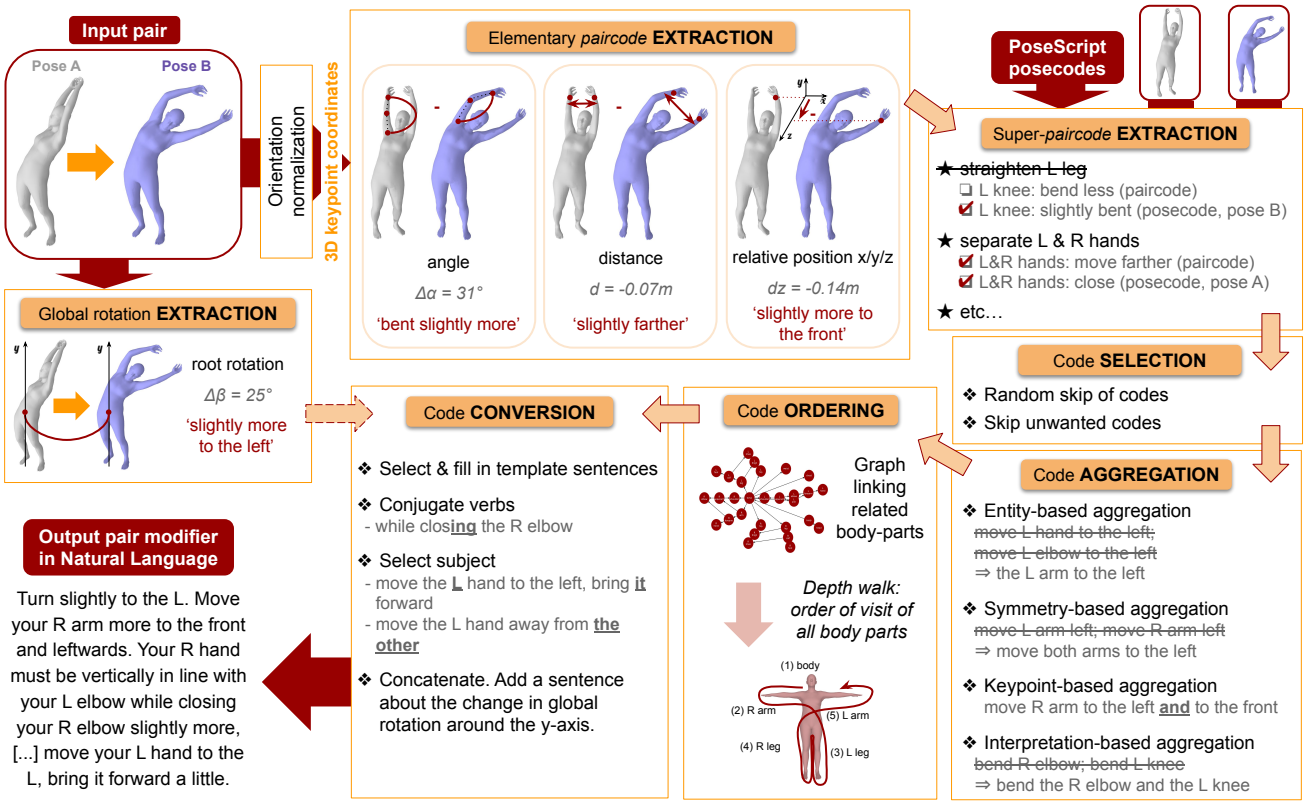

Figure A3: **Automatic Comparative Pipeline**, which generates modifiers based on the 3D keypoint coordinates of two input poses. L (resp. R) stands for 'left' (resp. 'right').

- *3D consideration*: the annotation fits the 3D information, no guess was taken on occluded body parts, or ambiguous postures;
- *no distance metric*: the annotation does not contain any distance metric (*e.g.*, '*one meter apart*'), which would not scale to bodies of different size;
- *writing quality*: correct grammar and formulation.

Length of the human-written annotations. Figure A2 shows the length distribution of the collected annotations. We here refer to the length as the number of words, excluding punctuation. While the annotations were constrained to be at least 10-word long, they tend to count about 30 words, suggesting that the differences between two similar poses A and B are both subtle and several.

A.2. Automatic annotations

We explain here in more details the learning-free process to automatically generate modifiers. The different steps of the pipeline are illustrated in Figure A3. We comment on some of those steps.

Code extraction. Two of the elementary paircodes are basically variation-versions of the initial posecodes [12]: we look at the change in angle posecode or distance posecode between pose A and pose B. The third kind of paircode studies the variation in position of a keypoint along the x-, y- or z- axis. All three paircodes are computed on the orientation-normalized bodies, so that the produced instructions would not depend on the change in global orientation of the body between pose A and pose B. This last part is treated separately, and yields a sentence that is added at the beginning of the modifier.

We also resort to the posecodes of both poses A and B to define super-paircodes, and thus gain in abstraction or formulation quality. There can be several ways to achieve the same paircode, each way comprising at least two conditions (posecode and paircode mixed together). Some posecodes of pose B, if statistically rare, are also included in the final modifier, *e.g.* '*the hands should be shoulder-width apart*', '*the left thigh should be parallel with the ground*'. Posecodes of pose A are only useful for super-paircode computations.

Code selection and aggregation. We proceed as in [12]. Trivial codes are removed. The codes (paircodes + posecodes) are aggregated based on simple syntactic rules depending on shared information between codes.

Code ordering. The final set of codes is semantically ordered to produce modifiers that are easier to read and closer to what a human would write (*i.e.*, describe about everything related to the right arm at once, instead of scattering pieces of information everywhere in the text). This step did not exist in the PoseScript automatic pipeline. Specifically, we design a directed graph where the nodes represent the body parts and the edges define a relation of inclusion or proximity between them (*e.g. torso→left shoulder*, *arm→forearm*). For each pose pair, we perform a randomized depth walk through the graph: starting from the *body* node, we

choose one node at random among the ones directly accessible, then reiterate the process from that node until we reach a leaf; at that point, we come back to the last visited node leading to non-visited nodes and sample one child node at random. We use the order in which the body parts are visited to order the paircodes.

Code conversion. Codes are converted to pieces of text by plugging information into a randomly chosen template sentence associated to each of them. The pieces of text are next concatenated thanks to transition texts. Verbs are conjugated accordingly to the chosen transition (*e.g.* "while + gerund") and code (*e.g.* posecodes lead to "[...] should be" sentences).

We refer to the code for the detailed and complete list of paircodes and super-paircodes definition.

B. Original triplets of the generation examples

In this section, we provide the original triplets for the generation results presented in Figure 5 (see Figure A4) and in Figure 7 (see Figure A5). While this ground truth may ease the comparison, it is not the only true answer for a generative model: multiple valid results could be produced. The GT was purposely omitted to prevent judgment bias, but is added here for reference.

C. Miscellaneous visualizations

Robot teaching application. The choice of modifiers in Natural Language to learn the difference between two poses proves especially useful in applications where direct manipulation is not possible, for instance in the case of robot teaching. Figure A6 shows a snapshot of a demo where a two-arm robot pose is optimized to match SMPL keypoints obtained from textual instructions.

The PoseCopy behavior. The PoseCopy setting for the text-based pose editing task consists in training the model with a proportion of the data where the text is emptied and pose B becomes a copy-paste of pose A. This training configuration makes it possible for the model to yield the exact same pose as the initial one, when no correctional instruction is specified, see Figure A7 for an example. Besides, we hypothesize that this setting encourages the model to better pay attention to pose A.

D. Implementation details

Architecture details. We follow the VPoser [44] architecture for our pose encoder, modified to account for the 52 joints of the SMPL-H [52] body model. In the 'glove+bigru' configuration of our pose editing baseline, GloVe word embeddings are of size 300 and we use a bidirectional GRU with one layer and hidden state features of size 512. In the transformer configuration, we use a frozen pretrained DistilBERT model to encode the text tokens. The transformer afterwards is composed of 4 layers with 4 heads and feed-forward networks with 1024 dimensions. It relies on GELU [21] activations and uses a 0.1 dropout. The text embedding is eventually obtained by performing an average pooling. The transformer in our correctional text generation baseline is the same as for pose editing, except that we use 8 heads. In our models for both tasks, the poses and texts are encoded in latent spaces of dimensions d=32 and n=128 (n=512 for the text generation task) respectively.

Optimization and training details. We trained our models with the Adam [28] optimizer, a batch size of 128, a learning rate of 10^{-5} (10^{-4} for pretraining; and 10^{-6} for finetuning in the case of pose editing) and a weight decay of 10^{-4} (10^{-5} for finetuning in the case of pose editing). The pose editing model was trained for 10,000 epochs (half for pretraining and half for finetuning, or 10,000 straight if no pretraining was involved), while the text generation model was trained for 3,000 epochs for pretraining and 2,000 for finetuning. In the *PoseCopy* setting, 50% of the batch is randomly used in "copy" mode (*i.e.*, empty text, with poses A and B being the same).

Why using the ELBO metric? The ELBO is well suited to VAEs [29]: it balances reconstruction and KL into a lower bound on the data log likelihood, a universal quantity for comparing likelihood-based generative models. It accounts for the probabilistic nature of the model, by evaluating the target under the output distribution. In a VAE framework, reporting reconstruction errors only does not penalize the model for storing a lot of information in the latent variable produced by the encoder. The extreme case of an encoder that learns an identity function would appear optimal, yet fail at test time when the ground truth is no longer available for encoding. By contrast, the ELBO takes both reconstruction and the amount of information given by the encoder (the KL term) into account, and combines them into a lower bound on the data log likelihood.

Hand data. We used the hand data (fingers joints) for all ours experiments, but note that this was not necessary, given that the hands all have the same pose for PoseFix human-annotated pose pairs. In case more data with relevant hand information is annotated in the future, we suggest to keep the original hand data for the pairs annotated in this version of the dataset, as some annotators may have referred to them in their instructions.

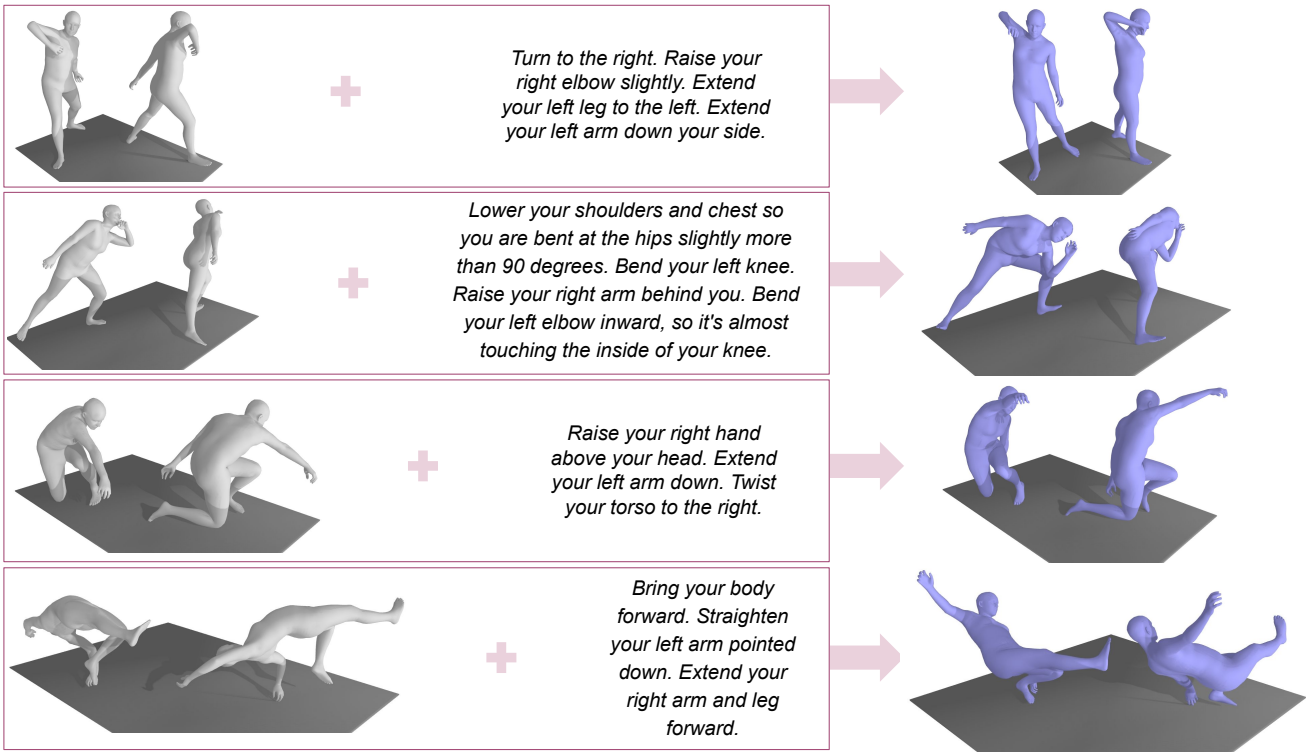

Figure A4: **Original poses** B for the text-based pose editing task and PoseFix queries presented in Figure 5. Two views of the each pose are shown on the same ground plane. Pose A is shown in grey, pose B in purple.

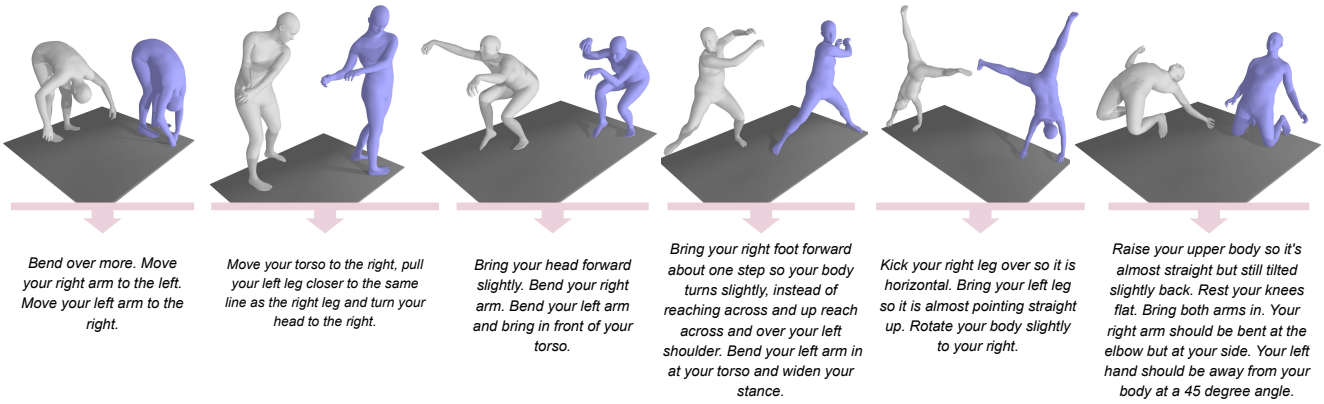

Figure A5: **Original correctional feedback annotation** for PoseFix pose pairs presented in Figure A5. Pose A is shown in grey, pose B in purple.

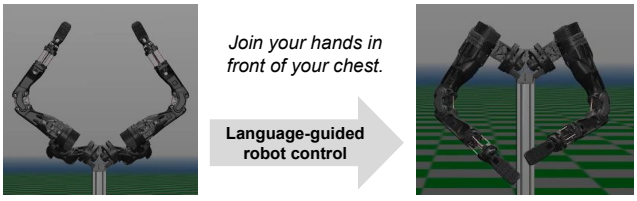

Figure A6: **Robot teaching application.**

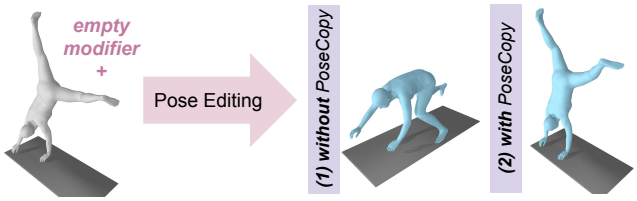

Figure A7: **Effect of training with PoseCopy.**