DTU Compute
Department of Applied Mathematics and Computer Science

3D human interaction synthesis for action recognition data augmentation

Master Thesis

3D human interaction synthesis for action recognition data augmentation

Master Thesis
June, 2024

By
Anders Bredgaard Thuesen

Copyright:	Reproduction of this publication in whole or in part must include the customary bibliographic citation, including author attribution, report title, etc.
Cover photo:	Vibeke Hempler, 2012
Published by:	DTU, Department of Applied Mathematics and Computer Science, Richard Petersens Plads, Building 324, 2800 Kgs. Lyngby Denmark www.compute.dtu.dk
ISSN:	[0000-0000] (electronic version)
ISBN:	[000-00-0000-000-0] (electronic version)
ISSN:	[0000-0000] (printed version)
ISBN:	[000-00-0000-000-0] (printed version)

Approval

This thesis has been prepared over six months at the Section for Indoor Climate, Department of Civil Engineering, at the Technical University of Denmark, DTU, in partial fulfilment for the degree Master of Science in Engineering, MSc Eng.

It is assumed that the reader has a basic knowledge in the areas of statistics.

Anders Bredgaard Thuesen - s183926

....

Signature

....

Date

Abstract

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Acknowledgements

Anders Bredgaard Thuesen, MSc Civil Engineering, DTU
Creator of this thesis template.

[Name], [Title], [affiliation]
[text]

[Name], [Title], [affiliation]
[text]

Contents

1 Introduction

In healthcare environments such as hospitals and care homes, data pertaining to critical incidents like falls are scarce due to their infrequent nature and the high privacy requirements surrounding such data. This scarcity poses significant challenges for training robust machine learning models, particularly in applications related to video classification where detailed understanding of such events is crucial. The primary goal of this thesis is to enhance the performance of video classification tasks by leveraging synthetic data, which is designed to closely mirror the underlying distribution of real-world incidents while enabling focused studies on specific, rare events.

To address the challenges inherent in collecting and utilizing real-world data from sensitive environments, this research proposes a novel approach using synthetic data generation. Synthetic data not only adheres to the distributional characteristics of genuine data but also provides flexibility to explore less common scenarios—specifically, those at the tail of the distribution which are typically underrepresented in available datasets.

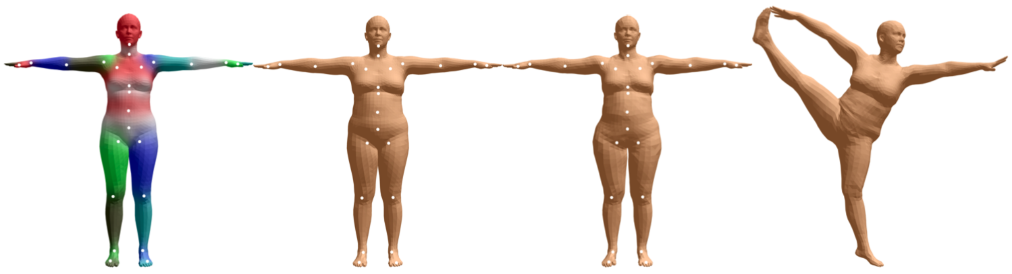
This work introduces a sophisticated framework for generating synthetic data by explicitly modeling three-dimensional (3D) environments. This includes detailed interactions both among humans and between humans and their surroundings. By integrating these complex interactions as a strong inductive bias, the proposed generative diffusion model enhances the realism and applicability of the synthetic data.

To construct and train this model, we utilize publicly available datasets such as HumanML3D and InterHuman. These datasets include motion capture data of individuals and pairs interacting, each accompanied by textual descriptions. These are combined with 3D scene reconstructions derived from video captures in actual hospital and care home settings. This integration of human motion and scene specifics forms the foundation for our synthetic data generation process.

To effectively reconstruct 3D scenes from the captured videos, we employ state-of-the-art models such as ProHMR and Depth Anything. These models are instrumental in generating per-frame human pose estimations and scene depth labels. These outputs, along with 2D keypoint annotations, are fed into a joint optimization process. This process is critical as it unifies the coordinate systems of the human models and the environment, ensuring that the motion trajectories are smooth and coherent. The result is a highly accurate 3D representation of the scenes, which serves as a vital input for our synthetic data generation.

2 Background

2.1 Skinned Multi-Person Linear Model (SMPL)



(a) $\bar{\mathbf{T}}$, $\mathcal{W}$ **(b)** $\bar{\mathbf{T}} + B_S(\vec{\beta})$, $J(\vec{\beta})$ **(c)** $T_P(\vec{\beta},\vec{\theta}) = \bar{\mathbf{T}} + B_S(\vec{\beta}) + B_P(\vec{\theta})$ **(d)** $W(T_P(\vec{\beta},\vec{\theta}), J(\vec{\beta}), \vec{\theta}, \mathcal{W})$

Figure 2.1: SMPL model

The Skinned Multi-Person Linear (SMPL) model is a parametric body shape model that accurately represents a wide range of human bodies and poses. It is built upon a foundation of linear blend skinning enhanced with corrective blend shapes, which are derived from a large dataset of body scans. The model captures the subtle deformations that occur with different body shapes and poses and can easily be rendered due to its compatibility with existing graphics pipelines. Since its publication, several extensions such as DMPL, incorporating dynamic soft-tissue deformation and SMPL-X, also modelling hands and facial expressions have been introduced. The model is parameterized by $\vec{\beta}$, capturing the variations from a mean body shape and $\vec{\theta}$, specifying the axis-angle rotation of 23 of the template skeleton joints. Mathematically, the model can be expressed as:

$$M(\vec{\beta},\vec{\theta}) = W(T_P(\vec{\beta},\vec{\theta}), J(\vec{\beta}), \vec{\theta}, \mathcal{W}) \tag{2.1}$$

where $T_P(\vec{\beta},\vec{\theta})$ returns the vertices of the rest pose, incorporating the deformations from the body shape and pose and is given by:

$$T_P(\vec{\beta},\vec{\theta}) = \bar{\mathbf{T}} + B_S(\vec{\beta}) + B_P(\vec{\theta}) \tag{2.2}$$

$J(\vec{\beta})$ returns the 3D joint locations from the shaped template vertices using a learned regression matrix $\mathcal{J}$ and is given by:

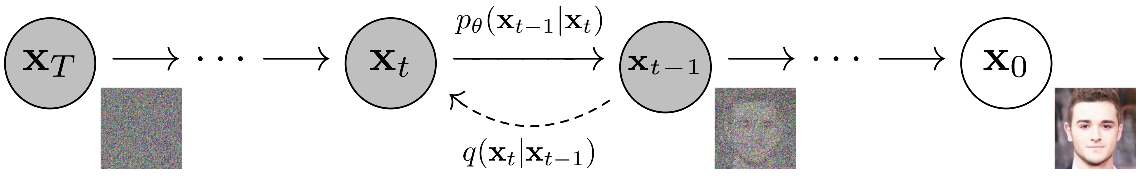
$$J(\vec{\beta}) = \mathcal{J}(\bar{\mathbf{T}} + B_S(\vec{\beta})) \tag{2.3}$$

W is the skinning function (e.g. Linear Blend Skinning (LBS) or Dual-Quaternion Blend Skinning (DQBS)) and $\mathcal{W}$ is the blend weights.

2.2 Human Mesh Reconstruction (HMR)

Recovering the mesh of humans have several applications…

2.3 Denoising Diffusion Probabilistic Models (DDPM)



$\mathbf{x}_T$
$\mathbf{x}_t$
$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$
$\mathbf{x}_{t-1}$
$q(\mathbf{x}_t|\mathbf{x}_{t-1})$
$\mathbf{x}_0$

Figure 2.2: Diffusion forward and backward process (taken from Ho, Jain, and Abbeel 2020)

In recent years, several types of generative models such as Variational Autoencoders (VAEs), Generative Adverserial Networks (GANs), autoregressive models and flow-based models have shown remarkable results in data generation of varying data modalities, such as images, audio, videos and text. Most recently, Denoising Diffusion Probabilistic Models (DDPMs) have gained large popularity especially within the field of image generation due to several reasons such as high-quality data generation, versitility in several data domains as well as controllability, allowing one to steer the generation towards desired outputs.

A DDPM is a parametarized Markov chain trained using variational inference to reverse a (forward) diffusion process, $q(\mathbf{x}_{1:T} \mid \mathbf{x}_0)$, as seen in fig. 2.2 wherein the signal of the data, $\mathbf{x}_0$, is gradually destroyed by adding gaussian noise according to predefined noise schedule $\{\beta_t \in (0,1)\}_{t=1}^T$ giving rise to increasingly noisy samples, $\mathbf{x}_1 \dots \mathbf{x}_T$.

The forward process is defined as follows:

$$q(\mathbf{x}_{1:T} \mid \mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1}), \quad q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}\mathbf{x}_t, \beta_t \mathbf{I}) \tag{2.4}$$

with T being the discritized number of diffusion steps before all original information is completely discarded. The goal of the inverse or backwards process then becomes to iteratively remove the noise, in order to arrivate at the original data. More formally, the process is defined as:

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1} \mid x_t), \quad p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)\right) \tag{2.5}$$

taking starting point in pure noise $p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$, incrementally removing the noise through the learned functions, $\boldsymbol{\mu}_\theta(\mathbf{x}_t, t)$ and $\boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)$ commonly parameterized by a deep neural network. Using the reparameterization trick, we are able to sample any noisy version of our data, $\mathbf{x}_t$, at time step t given our original data $\mathbf{x}_0$. Recall our forward transition probability function, $q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$. Letting $\alpha_t = 1 - \beta_t$, $\bar{\alpha}_t = \prod_{i=1}^t \alpha_i$ and using the reparameterization trick the expression can be rewritten as:

$$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\epsilon_{t-1} \tag{2.6}$$

where $\epsilon_{t-1} \sim \mathcal{N}(0,1)$. Expanding the recursive definition then gives:

$$\begin{aligned}
\mathbf{x}_t &= \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{\mathbf{t-1}} \\
&= \sqrt{\alpha_t}\left(\sqrt{\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_{t-1}}\boldsymbol{\epsilon}_{t-2}\right) + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{\mathbf{t-1}} \\
&= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{\alpha_t(1-\alpha_{t-1})}\boldsymbol{\epsilon}_{\mathbf{t-2}} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{\mathbf{t-1}} \\
&= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\bar{\boldsymbol{\epsilon}}_{t-2}
\end{aligned}$$

where $\bar{\boldsymbol{\epsilon}}_{t-2}$ merges the two independent Gaussians $\boldsymbol{\epsilon}_{t-1}$ and $\boldsymbol{\epsilon}_{t-2}$ into a single Gaussian with new variance as the sum of variances $\alpha_t(1-\alpha_{t-1})+(1-\alpha_t) = 1-\alpha_t\alpha_{t-1}$. Recursively applying the definition of $\mathbf{x}_t$ and merging the gaussian noise terms results in the simplified expression:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{2.7}$$

Conversely, given $\mathbf{x}_0$ the reverse conditional probability:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t\mathbf{I}\right)$$

becomes tractable to compute using Bayes rule (derivation in appendix A.1) with mean and variance given by:

$$\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t}\mathbf{x}_t \overset{\text{Using 2.7}}{=} \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t\right) \tag{2.8}$$

$$\tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t \tag{2.9}$$

The model is trained by optimizing the variational lower bound on the log likelihood:

$$L_{VLB} = \mathbb{E}_q\left[\log\frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_q\left[\log p(\mathbf{x_T}) + \sum_{t\geq 1}\log\frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] \leq \mathbb{E}\left[\log p_\theta(\mathbf{x}_0)\right] \tag{2.10}$$

which in practice means minimizing the negative variational lower bound. Ho, Jain, and Abbeel 2020 rewrites this into a sum of KL-divergences:

$$L_{VLB} = \mathbb{E}_q\left[\underbrace{D_{\mathsf{KL}}\left(q\left(\mathbf{x}_T \mid \mathbf{x}_0\right) \| p\left(\mathbf{x}_T\right)\right)}_{L_T} + \sum_{t=2}^{T}\underbrace{D_{\mathsf{KL}}\left(q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0\right) \| p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right)\right)}_{L_{t-1}} - \underbrace{\log p_\theta\left(\mathbf{x}_0 \mid \mathbf{x}_1\right)}_{L_0}\right] \tag{2.11}$$

where the authors model L_0 from seperate discrete decoder. As L_T doesn't depend on our parameters, θ, and is therefore constant, it can be ignored during optimization. The rest of the L_{t-1} terms can be efficiently computed in the closed form, by fixing the variance $\boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t) = \sigma_t^2\mathbf{I}$ to only depend on the current timestep (authors propose $\sigma_t^2 = \beta_t$ or $\sigma_t^2 = \tilde{\beta}_t$). L_{t-1} can then be written in closed form (derivation in appendix A.2) as:

$$L_{t-1} = \mathbb{E}_q\left[\frac{1}{2\sigma_t^2}\|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t)\|^2\right] + C_t \tag{2.12}$$

where C_t is a constant depending on the choice of σ_t^2 and the noise schedule. Using eq. (2.8) Ho, Jain, and Abbeel 2020 reparameterize the expression in terms of predicting the noise:

$$L_{t-1} = \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t\left(1-\bar{\alpha}_t\right)}\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta\left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t\right)\right\|^2\right] \tag{2.13}$$

as they find it leads to better unconditional sample quality when training on the CIFAR10 dataset. Furthermore, they report the best sample quality when using the "simple" objective, ignoring the weighing:

$$L_{\text{simple}} = \mathbb{E}_{t\sim\mathcal{U}(1,T),\ \boldsymbol{\epsilon}}\left[\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}}\boldsymbol{\epsilon}, t)\right] \tag{2.14}$$

Algorithm 1 Training

```
1: repeat
2:   $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3:   $t \sim \text{Uniform}(\{1, \ldots, T\})$
4:   $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5:   Take gradient descent step on
         $\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$
6: until converged
```

Algorithm 2 Sampling

```
1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: for $t = T, \ldots, 1$ do
3:   $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4:   $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$
5: end for
6: return $\mathbf{x}_0$
```

The final training and sampling scheme, as outlined in algorithm 1 and algorithm 2 can be summarized as follows. The training algorithm optimizes the model to predict and remove noise from data, while the sampling algorithm uses the trained model to iteratively transform random noise into structured data.

To summarize, diffusion models are generative models that create data by reversing a diffusion process. They are trained using a simplified version of the variational lower bound on the log-likelihood. Starting from pure noise, the model iteratively removes the predicted noise through a step-by-step denoising process, effectively reconstructing the original data. This approach allows DDPMs to produce high-quality and diverse outputs.

2.3.1 Variance schedules

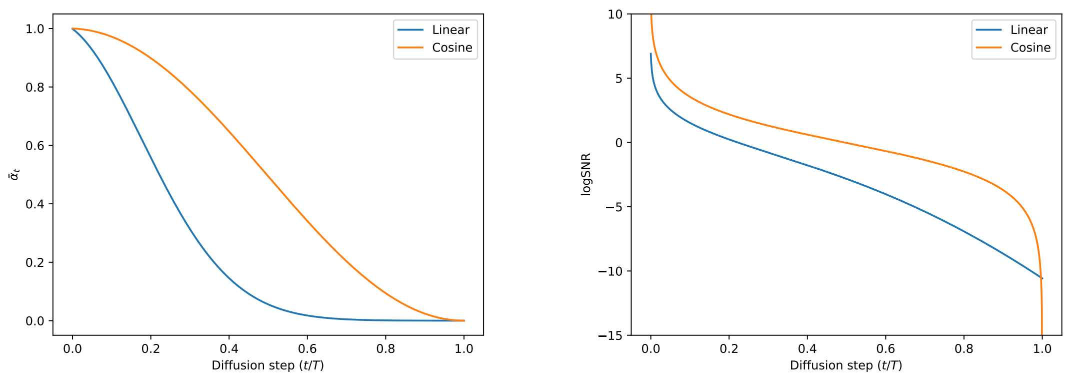


Figure 2.3: Linear and cosine schedule and logSNR.

In the DDPM paper Ho, Jain, and Abbeel 2020 choose a variance schedule with β_t linearly increasing from $\beta_1 = 10^{-4}$ to $\beta_T = 0.02$. However, this results in $\mathbf{x}_t$ almost entirely losing its signal in the last quarter of the schedule as shown by A. Q. Nichol and Dhariwal 2021. The authors instead propose a cosine variance schedule, where the squared signal proportion is given by:

$$\bar{\alpha}_t = \frac{f(t)}{f(0)}, \quad f(t) = \cos\left(\frac{t/T + s}{1+s} \cdot \frac{\pi}{2}\right)^2 \tag{2.15}$$

where s controls the offset of the noise schedule. The authors set $s = 0.008$ as they found that $s = 0$ resulted in minuscule noise near $t = 0$ making it hard for the model to predict ϵ. From $\bar{\alpha}_t$ one can then compute $\beta_t = \min\left(1 - \frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}}, 0.999\right)$ with the min to prevent singularities. A comparison of the cosine and linear schedules can be seen in fig. 2.3.

2.3.2 Classifier and Classifier-free Guidance (CFG)

When sampling from generative models, one often wants to conditionally generate samples from the data distribution of different sub categories. One method for doing so with diffusion models is classifier guidance. Here the generation process relies on an external model guiding the reverse diffusion process towards the sub category date manifold. We first define the score-function as the gradient of the log-likelihood:

$$s_\theta(\mathbf{x}) = \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) \tag{2.16}$$

$$\dot{\mathbf{x}} = \tag{2.17}$$

$$\nabla_{\mathbf{x}_t} \log p_\theta\left(\mathbf{x}_t\right) = -\frac{1}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t\right) \tag{2.18}$$

Dhariwal and A. Nichol 2021

When sampling from generative models, one often wants to conditionally generate samples from the data distribution of different sub categories. Dhariwal and A. Nichol 2021 trains a classifier $p_\phi(y \mid \mathbf{x}_t, t)$ on noisy images and uses its gradient $\nabla_{\mathbf{x}_t} p_\phi(y \mid \mathbf{x}_t, t)$ in order to guide the reverse diffusion process.

In order to control the diffusion process, one can introduce a drift term in the sampling process as follows:

2.4 Tranformer architecture

2.5 Human Motion Generation

Generating realistic human motion has several applications

- Human Motion Diffusion -

3 Data

3.1 HumanML3D

The HumanML3D dataset combines the HumanAct12 and AMASS datasets, integrating human motion captured using advanced motion capturing systems and converting the data to a unified parameterization. It covers a broad range of daily human activities, providing 14,616 motions in total, accompanied by 44,970 single-sentence descriptions. Each motion clip includes 3-4 descriptions, and the entire dataset amounts to 28.59 hours of recorded motion. Additionally, the data is augmented by mirroring all motions, with corresponding adjustments to descriptions, such as changing “clockwise” to “counterclockwise.”

3.2 InterHuman

The InterHuman dataset is a comprehensive, large-scale 3D dataset designed to capture human interactive motions involving two individuals. It includes approximately 107 million frames detailing a wide range of human interactions, from professional activities to daily social behaviors. Each motion sequence is paired with natural language annotations, totaling 23,337 descriptions, which provide context and detail for the captured interactions, enhancing the dataset’s utility for training and evaluating models.

3.3 Teton dataset

10s of video with 10 frames per second. Pseudo dataset of SMPL pose predictions from HMR2.0 for each frame

4 Methods

4.1 Tracking & Matching

We track the predicted staff and patients across multiple frames, indicated by $t = 1 \dots T$, by iteratively assigning the predictions, $\{P_i^{(t)}\}_{i=1}^{M_t}$, to the latest known tracks, $\{Q_j^{(t)}\}_{j=1}^{N_t}$ by greedily picking the assignment with the minimum cost:

$$\operatorname*{argmin}_{i,j} \mathcal{L}_{\text{track}}(P_i^{(t)}, Q_j^{(t-1)}), \tag{4.1}$$

until either all predictions or latest tracks have been assigned exactly once. In the case of any unassigned predictions, a new track is initialized. After tracks have been assigned, the latest track $Q_j^{(t)}$ is updated with the assigned predictions. We choose the following composite loss function:

$$\mathcal{L}_{\text{track}}(P, Q) = \alpha \left\| P_{\text{3D kpts}} - Q_{\text{3D kpts}} \right\|_2 + \beta \left\| P_{\text{class}} - Q_{\text{class}} \right\|_\infty , \tag{4.2}$$

incorporating both the Euclidean distance between the 3D joint keypoints as well as the predicted person class (staff/patient), with α and β weighting the influence of each term. The tracks, T_i, are then greedily matched to the ground truth annotations G_j, minimizing the loss:

$$\mathcal{L}_{\text{match}}(T, G) = \sum_t^T \begin{cases} \left\| a_{\text{track,2D kpts}}^{(t)} - b_{\text{track,2D kpts}}^{(t)} \right\|_2 & \text{if } a_{\text{track}}^{(t)} \in a_{\text{track}} \\ \gamma & \text{otherwise} \end{cases} \tag{4.3}$$

over the trajectory time horizon $t = 1, 2, \dots, T$ where γ is the punishment for not detecting the person.

4.2 Human pose sequence optimization

4.3 3D scene reconstruction

Restoring depth scale and offsets

We utilize pseudo ground truth disparity maps generated by the Depth Anything model, inversely proportional to the scene depth, in order to reconstruct a point cloud of the scene. To recover the depth scale and offset we rasterize the predicted SMPL poses, extract the z-buffer, compute the inverse depth and regress the intersection with the normalized disparity maps using the Random Sample Consensus (RANSAC) algorithm robust to outliers.

$$\begin{bmatrix} px \\ y \\ 1/z \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & 0 & p_x \\ 0 & f_y & 0 & p_y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{4.4}$$

$$X = \frac{Z}{f_x}(x - p_x), \quad Y = \frac{Z}{f_x}(y - p_y) \tag{4.5}$$

where f_x, f_y are the horizontal and vertical focal lenghts respectively and (p_x, p_y) is the principal point often chosen as the center $(w/2, h/2)$ of the image.

4.4 Motion and scene representation

Previous work on single human motion generation uses a canonical representation

We use a 6D continuous representation of the joint angles as according to Zhou et al. 2019.

4.5 Diffusion model

5 Results

6 Discussion & Conclusion

Bibliography

Ho, Jonathan, Ajay Jain, and Pieter Abbeel (2020). “Denoising diffusion probabilistic models”. In: *Advances in neural information processing systems* 33, pp. 6840–6851.

Nichol, Alexander Quinn and Prafulla Dhariwal (18–24 Jul 2021). “Improved Denoising Diffusion Probabilistic Models”. In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 8162–8171. URL: https://proceedings.mlr.press/v139/nichol21a.html.

Dhariwal, Prafulla and Alexander Nichol (2021). “Diffusion Models Beat GANs on Image Synthesis”. In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., pp. 8780–8794. URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/49ad23d1ec9fa4bd8d77d02681df5cfa-Paper.pdf.

Zhou, Yi et al. (June 2019). “On the Continuity of Rotation Representations in Neural Networks”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

A Appendices

A.1 Derivation of posterior distribution

$$\begin{aligned}
q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0) &= q(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{x}_0)\frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)} \\
\Leftrightarrow \log q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0) &= \log q(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{x}_0) + \log q(\mathbf{x}_{t-1}|\mathbf{x}_0) - \log q(\mathbf{x}_t|\mathbf{x}_0) \\
&= \frac{1}{2}\left(\frac{\left(\mathbf{x}_t - \sqrt{\alpha_t}\mathbf{x}_{t-1}\right)^2}{\beta_t} + \frac{\left(\mathbf{x}_{t-1} - \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0\right)^2}{1-\bar{\alpha}_{t-1}} + \frac{\left(\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}_0\right)^2}{1-\bar{\alpha}_t}\right) + K \\
&= \frac{1}{2}\left(\frac{\mathbf{x}_t^2 - 2\sqrt{\alpha_t}\mathbf{x}_t\mathbf{x}_{t-1} + \alpha_t\mathbf{x}_{t-1}^2}{\beta_t} + \frac{\mathbf{x}_{t-1}^2 - 2\sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_{t-1}\mathbf{x}_0 + \bar{\alpha}_{t-1}\mathbf{x}_0^2}{1-\bar{\alpha}_{t-1}} + C(x_t,x_0)\right) +
\end{aligned}$$

We collect all terms for $\mathbf{x}_{t-1}$ and $\mathbf{x}_{t-1}^2$.

A.2 L_{t-1} closed form derivation

The closed form of the KL-divergence between two Gaussians $\mathcal{N}_0(\boldsymbol{\mu}_0,\Sigma_0),\mathcal{N}_1(\boldsymbol{\mu}_1,\Sigma_1)$ is given as:

$$D_{\mathsf{KL}}(\mathcal{N}_0\|\mathcal{N}_1) = \frac{1}{2}\left(\mathrm{tr}(\Sigma_1^{-1}\Sigma_0) - k + (\mu_1-\mu_0)^T\Sigma_1^{-1}(\mu_1-\mu_0) + \ln\frac{\det\Sigma_1}{\det\Sigma_0}\right) \tag{A.1}$$

Inserting $q(\mathbf{x}_{t-1}\mid\mathbf{x}_t,\mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_{t-1};\tilde{\mu}_t(\mathbf{x}_t,\mathbf{x}_0),\tilde{\beta}_t\mathbf{I}\right)$ and $p(\mathbf{x}_t\mid\mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_{t-1};\mu_\theta(\mathbf{x}_t,t),\sigma_t^2\mathbf{I}\right)$ yields:

$$D_{\mathsf{KL}}(q\|p) = \frac{1}{2}\left(k\frac{\tilde{\beta}_t}{\sigma_t^2} - k + \frac{1}{\sigma_t^2}\|\tilde{\mu}_t(\mathbf{x}_t,\mathbf{x}_0) - \mu_\theta(\mathbf{x}_t,t)\|^2 + \ln(\sigma_t^2) - \ln(\tilde{\beta}_t)\right) \tag{A.2}$$

Letting $\sigma_t^2 = \tilde{\beta}_t$ $(= \beta_t = \frac{1-\bar{\alpha}_t}{1-\bar{\alpha}_{t-1}}\tilde{\beta}_t)$ simplifies the above to:

$$D_{\mathsf{KL}}(q\|p) = \frac{1}{2\sigma_t^2}\|\tilde{\mu}_t(\mathbf{x}_t,\mathbf{x}_0) - \mu_\theta(\mathbf{x}_t,t)\|^2\ (+\,C_t) \tag{A.3}$$

where C_t is a constant depending only on the noise schedule.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Technical
University of
Denmark

Richard Petersens Plads, Building 324
2800 Kgs. Lyngby
Tlf. 4525 1700

www.compute.dtu.dk