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
I would like to thank my supervisors for their feedback during my writing of this thesis.

Morten Rieger Hannemose, [Assistant Professor], [affiliation]
[text]

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

Contents

Preface . . . ii
Abstract . . . iii
Acknowledgements . . . iv

1 Introduction **1**

2 Related Work **2**
2.1 Human Pose Estimation . . . 2
2.2 Human Motion Generation . . . 3
2.3 Synthetic data augmentation . . . 4

3 Background theory **7**
3.1 Skinned Multi-Person Linear Model (SMPL) . . . 7
3.2 Denoising Diffusion Probabilistic Models (DDPM) . . . 8
3.2.1 Variance schedules . . . 10
3.2.2 Classifier and Classifier-Free Guidance (CFG) . . . 11
3.3 Tranformer architecture . . . 12
3.4 Human Motion Diffusion . . . 14

4 Data **17**
4.1 HumanML3D . . . 17
4.2 InterHuman . . . 17
4.3 Teton (raw) dataset . . . 17

5 Methods **19**
5.1 3D scene reconstruction . . . 19
5.2 Human pose sequence optimization . . . 19
5.3 Unification of datasets . . . 20
5.4 Motion and scene representation . . . 21
5.5 Diffusion model . . . 21

6 Results **23**

7 Discussion & Conclusion **25**

Bibliography **27**

A Appendices **31**
A.1 Derivation of posterior distribution . . . 31
A.2 L_{t-1} closed form derivation . . . 31

1 Introduction

Hospitals and care homes are facing pressure due to demographic changes, leading to fewer caregivers for an increasingly elderly population.

Teton [1] aims to alleviate this pressure by providing a privacy-preserving patient monitoring system that informs staff about current patient activities and alerts them to critical events, such as falls. The system consists of a camera sensor with built-in processing along with a mobile application. To preserve the privacy of patients, Teton has trained a deep neural network classification model on consensually obtained video footage from hospitals and care homes. This model runs on-device, ensuring that only the detections, and no video, ever leave the device.

As nurses and care home staff increasingly rely on the system in their daily work, Teton bears growing responsibility for ensuring the system's reliability to maintain staff trust and ultimately ensure patient safety. Critical events in the tail of the distribution, such as falls, are particularly challenging to address due to their rare occurrence, making it difficult to obtain sufficient data for training purposes.

A possible solution is to train on synthetic data from simulated scenarios; however, directly generating video is a very high-dimensional problem that requires vast amounts of data.

An alternative approach is to reduce the dimensionality of the problem by explicitly modeling the 3D scene and environment. This method enables precise control of the scene, as well as efficient rendering from different angles and under various lighting conditions, but requires solving the inverse problem of ”lifting” 2D videos into 3D.

Additionally, besides being able to synthesize realistic scenarios to augment the training data, accurate motion and scene understanding is crucial for applications like gait analysis, pressure ulcer detection, and automating much of the documentation burden by recognizing different human interactions such as changing diapers, intravenous injections, and more.

This thesis explores this approach in two parts. First, we investigate the use of monocular depth estimation and human pose estimation models to reconstruct the 3D scene, as well as 3D human locations and poses. Second, we train a generative diffusion model to conditionally sample new 3D sequences of humans interacting in a given 3D scene. Although essential for validating the approach, the final steps of ”skinning”, rendering, and evaluating the impact on model classification performance by training on these synthetically generated videos are left for future research.

[1] https://teton.ai

2 Related Work

2.1 Human Pose Estimation

Accurately estimating human poses is key in understanding their spatial relationships and interactions with the surrounding environment. Advances in this field have transitioned from 2D pose estimation, where joint positions are determined in image space, to 3D pose estimation, which involves inferring joint positions in three-dimensional space. The SMPL model (Loper et al. 2015) is often used in this context, providing a standardized framework for representing human body shapes and poses, thereby enabling more accurate and consistent mesh recovery. Additionally, the SMPL model reduces the number of parameters required to represent the human body, simplifying the computational complexity while maintaining a high level of detail.

Traditional approaches typically involve fitting 3D models to 2D keypoints extracted from images. This method, exemplified by techniques such as SMPLify (Bogo et al. 2016), first detects 2D joint positions (bottom-up) using a keypoint detector. These detected keypoints are then used (top-down) to fit a pre-defined 3D human body model, such as SMPL. This fitting process involves optimizing the model parameters to minimize the difference between the projected 3D keypoints and the detected 2D keypoints in the image. Such methods often require a strong prior to avoid generating highly implausible poses, as the optimization process can otherwise lead to unrealistic human body configurations. An example of such a prior is vPoser (Pavlakos et al. 2019), which uses a Variational Autoencoder (VAE) to learn a latent space of plausible human poses. Regularizing the latent pose space it is possible constrain the optimization process to more realistic pose configurations. Although these optimization-based methods provide accurate model-keypoint fits, they are often slow due to their iterative nature and sensitive to initialization.

Regression based methods such as HMR (Kanazawa et al. 2018) circumvent these issues by bypassing the intermediate 2D keypoint detection step, directly predicting the SMPL pose parameters from the input image. However, though the predicted poses are often fairly good, they fall short in terms of pixel-accuracy compared to optimization-based methods despite requiring large amounts of supervision. SPIN (Kolotouros, Pavlakos, Michael J Black, et al. 2019) on the other hand, is a hybrid method that combines both paradigms. From the input image a pose estimate is directly regressed and used as initialization to the SMPLify optimization process. After optimizing the joints to align with the predicted 2D keypoints, the optimized pose is then used as supervision signal to the regression network leading to iterative self-improvement. This collaborative approach inherits the accuracy of optimization based methods while being reasonably fast and less sensitive to initialization.

Despite these advancements, predicting poses frame-by-frame in video sequences poses significant limitations. Such methods often leads to temporal inconsistencies and jittery predictions because each frame is treated independently, ignoring the continuity and dynamics of human motion. To address these issues, VIBE (Video Inference for Body Estimation) (Kocabas, Athanasiou, and Michael J. Black 2020) incorporate both Gate Recurrent Units (GRUs) to improve temporal consistency and adversarial training using a discriminator model, exploting the large-scale AMASS motion capture dataset (Mahmood et al. 2019) to predict kinematically plausible, realistic motion sequences from the input video sequence. In 4D Humans (Goel et al. 2023) the authors ”transformarize” the orig-

inal HMR model and use the predicted poses and appearance features as input to the PHALP (Rajasegaran et al. 2022) tracking system, predicting their future 3D representations (spatial and visual) which are used to associated people across frames.

In common for above regression based methods is that they all only predict a single pose, assuming that the input image contains exactly one human. Hence, they rely on external bounding box annotations or object detection systems for extracing the relevant patches of the image, possibly leading to fragile detections. Recently, MultiHMR (Baradel* et al. 2024) based on a Visual Transformer (ViT) backbone has shown remarkable performance on single shot, multi-human pose estimation by predicting a 2D heatmaps of person centers used as input to a cross-attention module outputting the final estimated pose.

Deterministic models provide a single estimate of the human pose for a given input, without accounting for uncertainties. Above mentioned methods fall into this category, where the predicted pose is a fixed output of the model's inference process. Probabilistic models, in contrast, explicitly model the uncertainty in pose estimation, providing a distribution over possible poses rather than a single deterministic output. This approach allows the model to capture ambiguities inherent in 2D-to-3D mapping. Probabilistic models such as ProHMR (Kolotouros, Pavlakos, Jayaraman, et al. 2021) based on Normalizing Flows instead output a probility distribution over 3D poses for sampling, enabling it to better handle ambiguities.

2.2 Human Motion Generation

The Human Motion Diffusion Model (MDM) (Tevet et al. 2023) represents a significant advancement in single-person, text-conditioned motion generation. Utilizing the CLIP (Contrastive Language-Image Pre-training) framework, this model can generate detailed and contextually appropriate human motion sequences based on textual descriptions. Unlike traditional denoising diffusion models that predict noise, MDM innovatively predicts the motion signal directly. This approach simplifies the computation of geometric losses, such as foot-ground contact, which are crucial for ensuring the physical realism of generated motions. At the heart of this model lies a transformer-based denoising network, which iteratively refines motion sequences to achieve high fidelity. Additionally, the model is capable of motion inbetweening, generating intermediate frames between given start and end frames to ensure smooth and continuous motion trajectories.

Building on the foundation of MDM, InterGen (Liang et al. 2024) extends the capabilities of motion generation to interactions between two individuals. This two-person, text-conditioned motion diffusion model also leverages the CLIP framework, enabling it to generate coordinated motion sequences based on textual inputs. The architecture of InterGen features a dual transformer-like model with shared weights, allowing it to predict the motion signals for both individuals concurrently. This shared-weight mechanism ensures that the generated motions are synchronized and contextually consistent. InterGen introduces new pairwise loss functions, designed to help the model learn the intricacies of human interactions, making it particularly suitable for applications involving social dynamics and collaborative activities. However, it is important to note that the model is specifically designed for generating interactions between exactly two persons, which limits its flexibility in more complex multi-person scenarios.

Addressing the critical challenge of physical plausibility in motion generation, PhysDiff Yuan et al. 2023 incorporates a physical projection layer into the diffusion sampling process. This ensures that the generated motions conform to the laws of physics, which is essential for applications where physical realism is paramount. PhysDiff strategically

applies these physical constraints later in the diffusion process, avoiding early imposition that could degrade motion quality. The primary concern is that constraining the diffusion process too much in the early steps can push it significantly off from the initial Gaussian distribution. Since the denoising network is not trained to handle these deviations, this can result in suboptimal motion quality. By applying physical constraints at a later stage, PhysDiff ensures that the generated motions remain within a plausible range while maintaining high fidelity and diversity.

2.3 Synthetic data augmentation

Synthetic data augmentation has emerged as a pivotal technique in the machine learning landscape, offering significant advantages for enhancing model performance. By generating diverse and realistic data variations, synthetic data augmentation addresses the limitations of real-world datasets, such as scarcity and imbalance. This method has proven particularly effective in improving the accuracy and robustness of models across various applications, from image recognition, natural language processing and human action recognition.

Studies like Peng and Saenko 2017 and Zheng et al. 2016 have underscored the value of synthetic data in aligning feature distributions and improving neural network stability. Additionally, techniques such as domain randomization discussed in Tremblay et al. 2018 have shown significant improvements in model robustness by generating a wide range of data variations.

A key challenge when using synthetic data augmentation is domain adaptation, as synthetic data might not resemble the real data distribution well enough to yield any improvements in real world tasks. Shrivastava et al. 2017 propose Simulated+Unsupervised learning to tackle this problem. Their method consists of adverserially training a refinement network to improve the realism of synthetic images while the discriminator's objective is to classify the refined and real input images as either synthetic or real. Employing S+U learning, the authors show significant improvements over training directly on synthetic images in the context of in-the-wild appearance-based gaze estimation.

Synthetic data generation methods can be roughly grouped into two categories: generation using handcoded simulators or using simulators learned from data. In the context of autonomous driving, hardcoded simulators, like CARLA (Dosovitskiy, Ros, et al. 2017), meticulously replicate real-world environments, while simulators learned from data, such as Wayze's GAIA-1 model (Hu et al. 2023), use extensive driving data to create realistic traffic scenarios.

Work by Hwang et al. 2020 introduce the ElderSim simulator for simulating elderlys daily activities in various environments from a variety of viewpoints in different lighting conditions. From evaluating three action recognition models trained on a mixture of real and synthetic data from the simulator, they find noticeable improvements in accuracy across subjects, viewpoints, age of subjects and datasets, demonstrating the effectiveness of synthetic data augmentation. However, the ElderSim simulator is inflexible as it is handcoded and depends on a static set of actions captured by an extensive motion capture setup.

Zhang et al. 2020 employ a Conditional Variational Auto-Encoder (CVAE) to generate realistic 3D human bodies integrated naturally within 3D scenes. This approach necessitates both a semantic understanding of the scene (e.g., placing a person sitting on a sofa) and adherence to physical constraints (such as ensuring human meshes do not intersect with

objects). However, the method only supports generating static 3D placements and hence cannot be used synthetically augmenting action recognition tasks.

3 Background theory

3.1 Skinned Multi-Person Linear Model (SMPL)

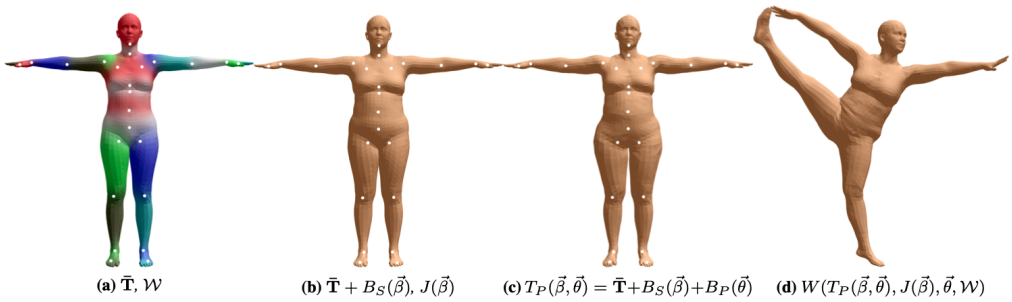


Figure 3.1: SMPL model (Image credit: Loper et al. 2015)

The Skinned Multi-Person Linear (SMPL) model (Loper et al. 2015) is a parametric model designed to accurately represent a diverse array of human body shapes and poses. The model is constructed on the principles of Linear Blend Skinning (LBS), augmented with corrective blend shapes derived from an extensive dataset of body scans. This approach allows the SMPL model to capture the nuanced deformations associated with different body shapes and poses, ensuring high fidelity and realism in representation. Furthermore, the model is compatible with standard graphics pipelines, facilitating its easy integration and rendering.

Since its inception, the SMPL model has been extended through various enhancements. Notable among these are SMPL-H (Romero, Tzionas, and Michael J. Black 2017), which models detailed hand movements, and SMPL-X (Pavlakos et al. 2019), which extends the model to include both detailed hand movements and facial expressions. Additionally, Dynamic-SMPL or DMPL (Loper et al. 2015) incorporates dynamic soft-tissue deformations, which simulate the natural movement of soft tissues during movement.

The SMPL model is parameterized by $\vec{\beta}$, which encapsulates the deviations from a mean body shape, and $\vec{\theta}$, which specifies the axis-angle rotations of the 23 joints in the template skeleton. These rotations are local and are applied using a kinematic tree structure, which ensures that the hierarchical relationships between body parts are accurately represented. Mathematically, the model can be expressed as:

$$M(\vec{\beta}, \vec{\theta}) = W(T_P(\vec{\beta}, \vec{\theta}), J(\vec{\beta}), \vec{\theta}, \mathcal{W}) \tag{3.1}$$

where $T_P(\vec{\beta}, \vec{\theta})$ denotes the vertices of the rest pose, incorporating deformations due to body shape and pose, and is defined as:

$$T_P(\vec{\beta}, \vec{\theta}) = \bar{\mathbf{T}} + B_S(\vec{\beta}) + B_P(\vec{\theta}) \tag{3.2}$$

Here, $\bar{\mathbf{T}}$ represents the template mesh, $B_S(\vec{\beta})$ accounts for shape-dependent deformations, and $B_P(\vec{\theta})$ represents pose-dependent deformations.

The joint locations, $J(\vec{\beta})$, are derived from the shaped template vertices using a learned regression matrix $\mathcal{J}$, and can be formulated as:

$$J(\vec{\beta}) = \mathcal{J}(\bar{\mathbf{T}} + B_S(\vec{\beta})) \tag{3.3}$$

The skinning function, W, which can be either LBS or Dual-Quaternion Blend Skinning (DQBS), applies these deformations to the skeletal structure, with $\mathcal{W}$ representing the blend weights.

Overall, the SMPL model's parametric nature and its capability to accurately represent a wide range of human anatomies and movements establish it as a valuable tool in the fields of computer graphics, animation, and related areas.

3.2 Denoising Diffusion Probabilistic Models (DDPM)

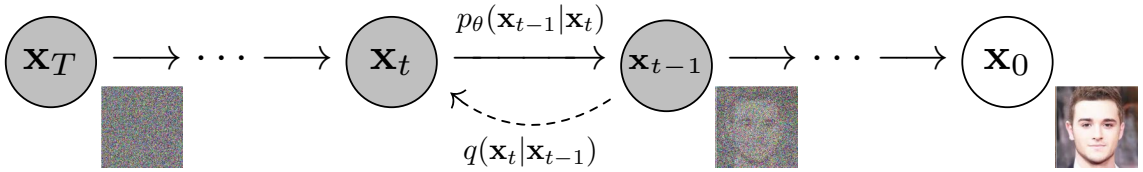


Figure 3.2: Diffusion forward and backward process (taken from Ho, Jain, and Abbeel 2020)

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. 3.2 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:

$$q(\mathbf{x}_{1:T} \mid \mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1}), \quad q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}\mathbf{x}_t, \beta_t \mathbf{I}) \tag{3.4}$$

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:

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1} \mid x_t), \quad p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)\right) \tag{3.5}$$

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:

$$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\epsilon_{t-1} \tag{3.6}$$

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

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

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:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{3.7}$$

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:

$$\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}\,(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t \overset{\text{Using 3.7}}{=} \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t\right) \tag{3.8}$$

$$\tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t \tag{3.9}$$

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

$$L_{VLB} = \mathbb{E}_q\left[\log\frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_q\left[\log p(\mathbf{x_T}) + \sum_{t\geq 1}\log\frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] \leq \mathbb{E}\left[\log p_\theta(\mathbf{x}_0)\right] \tag{3.10}$$

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

$$L_{VLB} = \mathbb{E}_q\left[\underbrace{D_{\mathsf{KL}}\left(q\left(\mathbf{x}_T \mid \mathbf{x}_0\right) \| p\left(\mathbf{x}_T\right)\right)}_{L_T} + \sum_{t=2}^{T}\underbrace{D_{\mathsf{KL}}\left(q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0\right) \| p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right)\right)}_{L_{t-1}} - \underbrace{\log p_\theta\left(\mathbf{x}_0 \mid \mathbf{x}_1\right)}_{L_0}\right] \tag{3.11}$$

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:

$$L_{t-1} = \mathbb{E}_q\left[\frac{1}{2\sigma_t^2}\|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t)\|^2\right] + C_t \tag{3.12}$$

where C_t is a constant depending on the choice of σ_t^2 and the noise schedule. Using eq. (3.8) Ho, Jain, and Abbeel 2020 reparameterize the expression in terms of predicting the noise:

$$L_{t-1} = \mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t\,(1-\bar{\alpha}_t)}\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta\left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t\right)\right\|^2\right] \tag{3.13}$$

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
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** $t = T, \dots, 1$ **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** $\mathbf{x}_0$

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:

$$L_{\text{simple}} = \mathbb{E}_{t \sim \mathcal{U}(1,T),\, \boldsymbol{\epsilon}} \left[\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}}\boldsymbol{\epsilon}, t) \right] \tag{3.14}$$

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.

3.2.1 Variance schedules

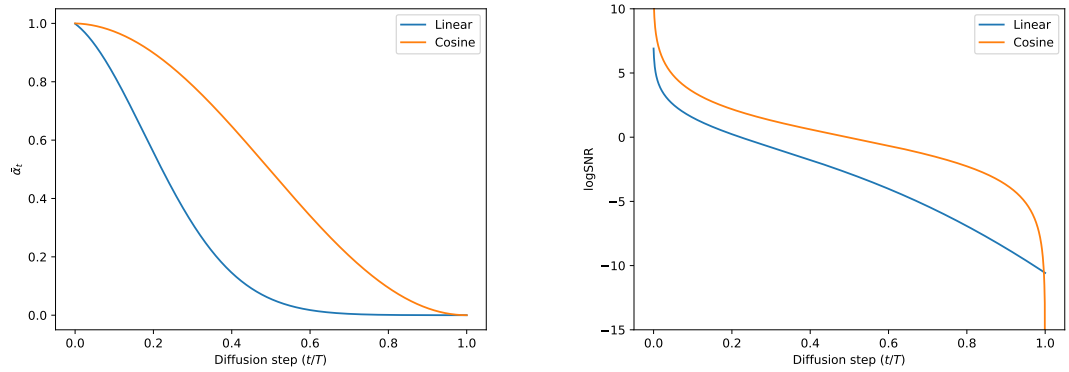


Figure 3.3: Linear and cosine schedule and signal-to-noise ratio.

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 problematized by A. Q. Nichol and Dhariwal 2021. The authors instead propose a cosine variance schedule, where the squared signal proportion is given by:

$$\bar{\alpha}_t = \frac{f(t)}{f(0)}, \quad f(t) = \cos\left(\frac{t/T + s}{1+s} \cdot \frac{\pi}{2} \right)^2 \tag{3.15}$$

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. 3.3.

3.2.2 Classifier and Classifier-Free Guidance (CFG)

In practice, it is often desirable to steer the generation process in order to output data belonging to a specific class, such as cats or dogs in the context of image generation, or incorporating some other information relevant for the output. This process can be framed within the context of probability theory as conditional generation, where the aim is to maximize the probability of the data, $\mathbf{x}$, given some conditioning signal, y, (e.g. a class label. According to Bayes' rule, the conditional probability can be expressed as:

$$p(\mathbf{x} \mid y) = \frac{p(\mathbf{x}, y)}{p(y)} \propto p(y \mid \mathbf{x}) p(\mathbf{x}) \tag{3.16}$$

Hence, it is proportional to the joint probability of the data and label, which from basic probability theory is equal to the product of the unconditional probability of the data, $p(\mathbf{x})$, and the probability of the conditioning signal given the data, $p(y \mid \mathbf{x})$ (for labels; that the data belonging to the given class). In the DDPM paper Ho, Jain, and Abbeel 2020 establishes connection between diffusion models and Noise-Conditioned Score Networks (NCSN) and shows how the reverse diffusion process can be seen as a temporal discritized type of annealed Langevin dynamics given by the stochastic differential equation:

$$\frac{d}{dt} \log p(\mathbf{x}_t) = \sqrt{\bar{\alpha}_t} s(\mathbf{x}_t, t) + \sqrt{1 - \bar{\alpha}_t} \mathbf{z}_t \tag{3.17}$$

where $z_t \sim \mathcal{N}(0, 1)$ and $s(\mathbf{x}_t, t) = \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)$ is referred to as the score function. It turns out that the denoising network can be used to approximate the score function:

$$s(\mathbf{x}_t, t) \approx -\frac{1}{\sqrt{1 - \bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \tag{3.18}$$

Returning to the factorization of the conditional probability, taking the gradient of the log of both sides reveals the relationship with the score function:

$$\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t \mid y) = \nabla_{\mathbf{x}_t} \log p(y \mid \mathbf{x}_t) + \underbrace{\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)}_{s(\mathbf{x}_t, t)} \tag{3.19}$$

Dhariwal and A. Nichol 2021 trains a classifier, $f_\phi(y \mid \mathbf{x}_t) \approx p(y \mid \mathbf{x}_t)$, seperately to predict the class label of noisy images. By using the reformulated denoising function:

$$\bar{\boldsymbol{\epsilon}}_\theta(\mathbf{x}_t, t) = \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) - \sqrt{1 - \bar{\alpha}_t} \nabla_{\mathbf{x}_t} \log f_\phi(y \mid \mathbf{x}_t) \tag{3.20}$$

where s controls guidance strength, it is possible to guide the diffusion sampling e.g. in order to generate images of a specific category, hence the name classifier guidance.

In contrast to classifier guidance, an alternative method known as classifier-free guidance, proposed by Ho and Salimans 2021, eliminates the need for a separate guidance network. This method leverages the same generative model to perform conditional generation by incorporating the conditioning signal, y, during the training process. Specifically, the model is trained with the conditioning signal but occasionally discards it with a certain probability, p_{uncond}. This technique allows the model to learn both the conditional and unconditional generation processes.

During the forward diffusion process, the model is trained to predict the noise added to the data both conditionally, $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, y)$, and unconditionally, $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, y = \emptyset)$. The conditional prediction uses the conditioning signal y (e.g., class labels), while the unconditional prediction is made without any conditioning.

At inference time, classifier-free guidance combines these predictions to steer the generation process towards the desired condition. This is achieved by interpolating between the conditional and unconditional predictions:

$$\bar{\epsilon}_\theta(\mathbf{x}_t, t, y) = \epsilon_\theta(\mathbf{x}_t, t, y = \emptyset) + s \cdot (\epsilon_\theta(\mathbf{x}_t, t, y) - \epsilon_\theta(\mathbf{x}_t, t, y = \emptyset)) \tag{3.21}$$

Here, s is a scaling factor that controls the strength of the guidance. This formulation allows the model to flexibly adjust the influence of the conditioning signal on the generation process, effectively guiding the reverse diffusion towards samples that match the desired condition.

By leveraging the same network for both conditional and unconditional noise predictions, classifier-free guidance simplifies the implementation and improves the efficiency of the conditional generation process. This method has demonstrated significant improvements in generating high-quality samples (Ho and Salimans 2021) that adhere to the specified conditions, making it a powerful alternative to traditional classifier guidance approaches.

3.3 Tranformer architecture

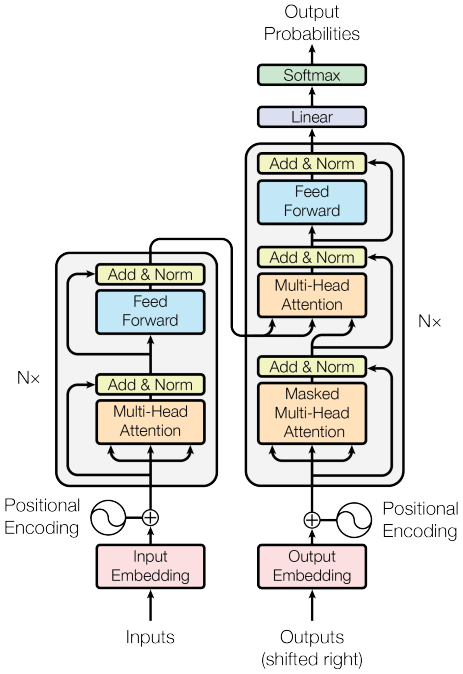


Figure 3.4: The transformer (Image from Vaswani et al. 2017.)

Ever since its publication, the Transformer (Vaswani et al. 2017) has revolutionized the field of deep learning. Despite being initially designed for Natural Language Processing (NLP), Transformers have demonstrated their versatility and effectiveness across a wide range of domains. Beyond NLP, Transformers have been successfully applied to other areas such as computer vision, where models like Vision Transformers (ViTs) (Dosovitskiy, Beyer, et al. 2021) leverage the same principles to process image data. In addition, fields such as speech recognition (Gulati et al. 2020), time-series forecasting (Zhou et al. 2021), and even protein folding (Jumper et al. 2021) have seen breakthroughs thanks to the adoption of Transformer-based models.

In contrast to models such as RNNs, GRUs and LSTMs relying on sequential processing of hidden states, the transformer model utilizes an attention mechanism for communica-

tion. The attention mechanism enables the model to capture long-range dependencies more effectively and has significantly improved the performance of various NLP tasks, including translation, summarization, and question answering (Devlin et al. 2018). Furthermore, due to the parallel nature of attention, the Transformer architecture has been subject for huge scaling with models exceeding billion of parameters (Radford et al. 2019, Touvron et al. 2023, Anil et al. 2023).

The architecture comprises an encoder and a decoder as illustrated in fig. 3.4. The encoder uses multi-headed self-attention to create semantic representations for each element in the input sequence, allowing the model to process the entire context simultaneously. Through cross-attention, the decoder integrates information from the encoder's output, while causal self-attention ensures that the model only considers previously generated tokens by masking out future tokens in the training phase. This enables autoregressive generation at inference time.

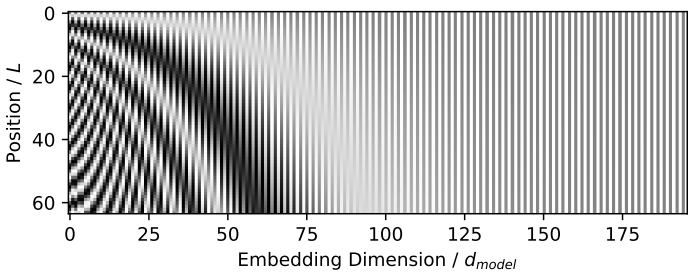
The transformer's attention mechanism is based on computing the scaled dot-product:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}, \tag{3.22}$$

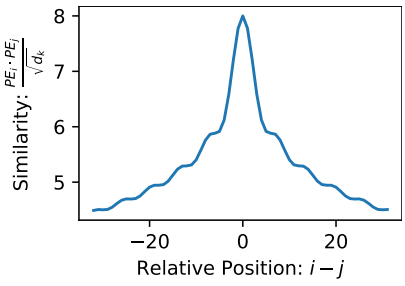
where **Q**, **K**, and **V** are the query, key, and value matrices, respectively, and d_k is the dimension of the key vectors. Hence, scaled dot-product attention calculates the attention score between all query and key vectors to identify the corresponding value vectors where the key-query alignment is strong. The softmax function and normalization constant, d_k, ensure that the attention output maintains an appropriate scale. However, one might want to allow the model to attend to multiple items. Multi-headed attention achieves this by using multiple attention mechanisms in parallel, each focusing on different parts of the input. Mathematically, it is formulated as:

$$\begin{aligned} &\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}\left(\text{head}_1, \ldots, \text{head}_\text{h}\right)\mathbf{W}^O \\ &\text{where head}_\text{i} = \text{Attention}\left(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V\right) \end{aligned} \tag{3.23}$$

where $\mathbf{W}^O \in \mathbb{R}^{hd_v \times d_\text{model}}$ is the output projection matrix and $\mathbf{W}_i^Q, \mathbf{W}_i^K \in \mathbb{R}^{d_\text{model} \times d_k}$, $\mathbf{W}_i^V \in \mathbb{R}^{d_\text{model} \times d_k}$ and $W_o \in \mathbb{R}^{hd_v \times d_\text{model}}$ are linear projection matrices reducing the dimensionality to $d_k = d_v = d_\text{model}/h$ where h is the number of heads. In other words, the query, key and value vectors are projected down to multiple heads between which the attention is computed before being concatenated and finally reprojected into the model dimensions.



(a) Sinusoidal positional encoding with values from 1 (white) to -1 (black). d_model = 196 and max length, $L = 64$.



(b) Normalized dot-product of encodings at positions i and j.

Since the transformer architecture does not naturally encode positional information, Vaswani

et al. 2017 introduce an additive sinusoidal encoding (fig. 3.5a) to the model's input embeddings:

$$\mathbf{PE}(i,\delta) = \begin{cases} \sin\left(\frac{i}{10000^{2\delta'/d_{\text{model}}}}\right) & \text{if } \delta = 2\delta' \\ \cos\left(\frac{i}{10000^{2\delta'/d_{\text{model}}}}\right) & \text{if } \delta = 2\delta' + 1 \end{cases}. \tag{3.24}$$

for $\delta = 1, \ldots, d_{\text{model}}$ and $i = 1, \ldots, L$. Considering the the query-key matrix dot-product with added positional encodings:

$$(\mathbf{Q} + \mathbf{PE})(\mathbf{K} + \mathbf{PE})^T = \mathbf{Q}\mathbf{K}^T + \mathbf{Q}\mathbf{PE}^T + \mathbf{PE}\mathbf{K}^T + \mathbf{PE}\mathbf{PE}^T. \tag{3.25}$$

This expansion reveals the various interactions between the query, key, and positional encodings. Specifically, the term $\mathbf{Q}\mathbf{K}^T$ computes the standard attention score based on the query and key. The term $\mathbf{Q}\mathbf{PE}^T$ captures the interaction between the content of the query and positional encodings, while $\mathbf{PE}\mathbf{K}^T$ captures the interaction between the positional encodings and the content of the key. Finally, $\mathbf{PE}\mathbf{PE}^T$ represents the self-interaction of the positional encodings. An example of this is illustrated in fig. 3.5b showing the normalized dot-product similarity of positional encodings at relative distances. These combined interactions enable the model to learn to attend to items at both absolute and content-dependent offsets, relavant for the objective at hand.

3.4 Human Motion Diffusion

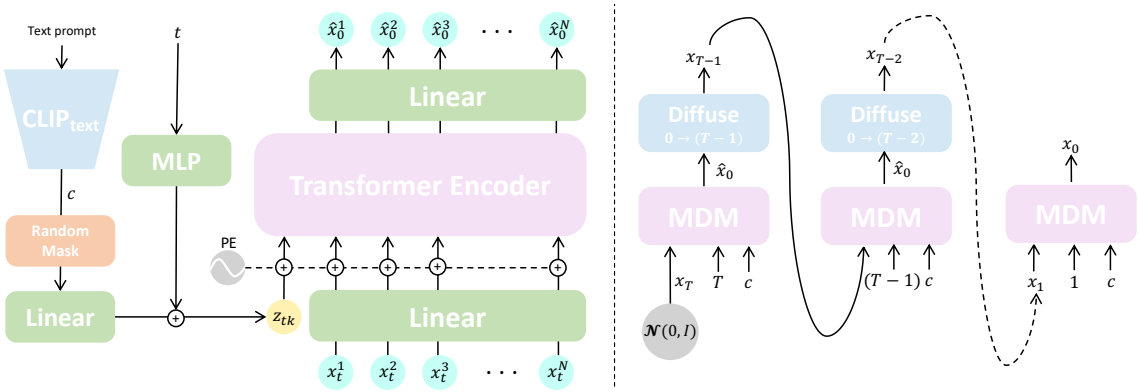


Figure 3.6: **(Left)** MDM architecture. **(Right)** MDM Sampling process.

The Motion Diffusion Model (MDM) (Tevet et al. 2023) provides a generative model for human motion conditioned on natural language descriptions using classifier-free guided diffusion models. The authors model human motion as a sequence of poses, $x^i \in \mathbb{R}^{J \times D}$, represented by either their joint rotations or positions where J is the number of joints and D is the dimensions of the representation and $i = 1, \ldots, N$ is the current frame of the motion.

Throught the diffusion process

$$q(\mathbf{x}_t^{(1:N)} \mid \mathbf{x}_{t-1}^{(1:N)}) = \mathcal{N}\left(\sqrt{\alpha_t}\mathbf{x}_{t-1}^{(1:N)} \mid (1-\alpha_t)\mathbf{I}\right), \tag{3.26}$$

the motion sequence is gradually degraded until becoming pure noise $x_T^{1:N} \sim \mathcal{N}(0,1)$ at the final timestep $t = T$ of the noise schedule, in correspondence with eq. (3.4). To reverse the diffusion process, the authors suggest parameterizing the score function,

$G(\mathbf{x}_t^{(1:N)}, t, c)$ to predict the signal instead of the noise, where c is the conditioning signal. In addition to the *simple* loss function given by:

$$\mathcal{L}_{\text{simple}} = E_{x_0 \sim q(x_0|c), t \sim [1,T]} \left[\|x_0 - G(x_t, t, c)\|_2^2 \right] \tag{3.27}$$

the authors introduce several geometric loss functions. This aligns with previous work on motion generative models (Petrovich, Michael J. Black, and Varol 2021, Shi et al. 2020) to help enforce physical constraints necessary for realistic motion, preventing artifacts such as teleportation and foot sliding. These include joint velocities and foot ground contact:

$$\begin{aligned} \mathcal{L}_{\text{pos}} &= \frac{1}{N} \sum_{i=1}^{N} \left\| FK\left(x_0^i\right) - FK\left(\hat{x}_0^i\right) \right\|_2^2 \\ \mathcal{L}_{\text{foot}} &= \frac{1}{N-1} \sum_{i=1}^{N-1} \left\| \left(FK\left(\hat{x}_0^{i+1}\right) - FK\left(\hat{x}_0^i\right) \right) \cdot f_i \right\|_2^2 \\ \mathcal{L}_{\text{vel}} &= \frac{1}{N-1} \sum_{i=1}^{N-1} \left\| \left(x_0^{i+1} - x_0^i\right) - \left(\hat{x}_0^{i+1} - \hat{x}_0^i\right) \right\|_2^2 \end{aligned} \tag{3.28}$$

where FK is the forward-kinematic function in case joint rotation represention and the identity function otherwise and $f_i \in \{0, 1\}$ is the binary foot contact indicators, punishing non-zero foot velocities when in contact with the ground. Hence, the final loss object becomes:

$$\mathcal{L} = \mathcal{L}_{\text{simple}} + \lambda_{\text{pos}} \mathcal{L}_{\text{pos}} + \lambda_{\text{vel}} \mathcal{L}_{\text{vel}} + \lambda_{\text{foot}} \mathcal{L}_{\text{foot}}. \tag{3.29}$$

Conforming to classifier-free guidance, the conditioning signal is dropped $p = 10\%$ of the time by setting $c = \emptyset$

In training the denoising network, parameterized by a transformer network, the conditioning is dropped 10% of the time by setting $c = \emptyset$

The denoising network is parameterized by a transformer network taking

4 Data

Dataset	# Sequences	Multi-person	Text descriptions	Scene object
HumanML3D		✗	✓	✗
InterHuman	7779	✓(exactly 2)	✓	✗
Teton (processed)	27.656	✓	✗	✓

Figure 4.1: Overview of different dataset properties.

4.1 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.”

4.2 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.

7779 sequences resulting in 6.56h

4.3 Teton (raw) dataset

Figure 4.2: Examples from the Teton dataset

The Teton dataset contains approximately 50,000 image sequences hierarchically organized by hospital/carehome site, room and timestamp. Each sequence consists of 100 frames recorded at approximately 10 frames per second, resulting in a sequence duration of about 10 seconds and totaling roughly 140 hours of data.

Each person in the frame has been manually annotated with their class (person, staff, patient), center point, bounding box, keypoints and current action (e.g., ”laying in bed”, ”standing on floor”). Using an image similarity threshold, the 100 frames are reduced to a smaller set of key frames. Each key frame is annotated with segmentation masks for people in the frame as well as the floor, walls and furniture (e.g., beds, sofas and chairs).

As the dataset is inheriently two dimensional and does not contain 3D annotations such as poses or depth maps, we instead rely on pseudo-ground truth data derived from off-the-shelf models. In the case of human poses the HMR2.0 model by Goel et al. 2023 is initially run on the image sequences to predict a set of poses. Poses with high reprojection error are filtered out, and the remaining poses with low reprojection error are used as pseudo-ground truth.

For reconstructing depth maps the monocular metric depth estimation model, Depth Anything by Yang et al. 2024, trained on the indoor NYUv2 dataset (Nathan Silberman and Fergus 2012), is applied to the set of key frames.

5 Methods

5.1 3D scene reconstruction

Extracting SMPL poses

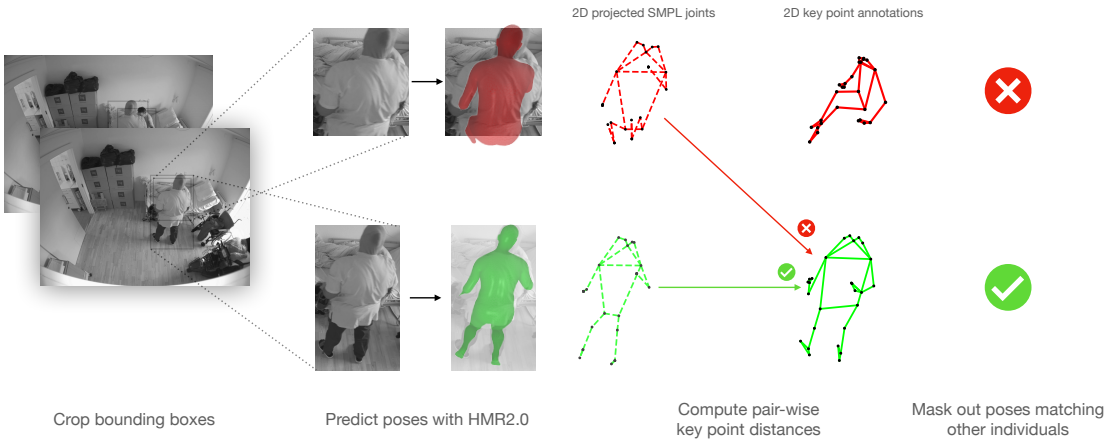


Figure 5.1: Illustration of pose extraction process with occlusion handling.

We leverage the pretrained HMR2.0 model from Goel et al. 2023 for pose extraction, applying it frame-wise to crops of each human bounding box in the frame. This approach ensures proper alignment with existing dataset annotations without the need for additional tracking. To handle potential occlusions that may cause duplicate predictions of the occluding pose, pairwise euclidean keypoint distances are computed for all edges in the bipartite graph of annotated 2D keypoints and SMPL joint 2D projections for individuals in the current frame. Poses with the minimum keypoint distance to other individuals' annotated keypoints are masked out.

Restoring scene depth

We combine both metric and relative variants of the Depth Anything model (Yang et al. 2024) to restore the depth of the scene. Using the low-resolution 384x512 metric depth predictions, $m_{1:K}$ (upsampled to match higher resolution disparity maps), we restore the scaling and offset from the higher resolution predicted disparity maps, $d_{1:K}$, by linearly fitting the disparity scaling α and offset β according to argmin $\alpha \cdot d_{1:K} + \beta - 1/m_{1:K}$ for all K key frames to get our upscaled metric depths, $m^*_{1:K} := 1/(\alpha \cdot d_{1:K} + \beta)$.

5.2 Human pose sequence optimization

Due to poses being predicted independently per frame, the combined pose sequences often lack temporal consistency, resulting in janky or erratic movements. To address this issue, we refine the pose sequences by optimizing for smooth trajectories across all $1 \dots T$ frames by minimizing a composite smooth loss objective assuming constant velocity:

$$\mathcal{L}_{\text{smooth}} = \sum_t \|\hat{\mathbf{j}}^{(t)} - \mathbf{j}^{(t)}\|_2 + \lambda_{\hat{j}} \sum_t \mathcal{R}^{(t)}_{\hat{j}} \tag{5.1}$$

where $\hat{\mathbf{j}}^{(t)} = \mathbf{j}^{(t-1)} + \mathbf{v}^{(t-1)}$ is the predicted joint location at time t assuming constant velocity $\mathbf{v}^{(t)} = \mathbf{j}^{(t)} - \mathbf{j}^{(t-1)}$, and $\mathcal{R}_{\bar{j}}^{(t)} = \|\mathbf{j}^{(t)} - \bar{\mathbf{j}}^{(t)}\|_2$ a regularization term preventing the optimized location $\mathbf{j}^{(t)}$ to not diverge too far from the initially predicted joint location $\bar{\mathbf{j}}^{(t)}$.

5.3 Unification of datasets

Before concatenating the HumanML3D, InterHuman and our own Teton dataset we make sure they all conform to the same coordinate system and scaling by implicitly representing the floor as the XY-plane with positive Z indicating the upwards direction using metric units for coordinates. This removes the need to explicitly condition the model on the floor plane possibly simplifying the learning task for more efficient training.

Aligning floor with XY-plane

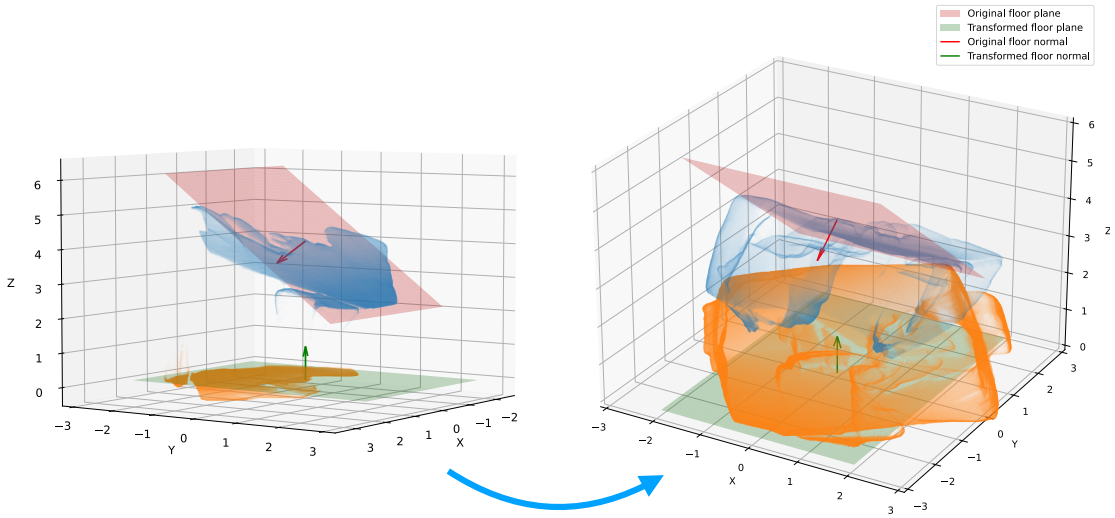


Figure 5.2: Translation and rotation of room point cloud aligning the floor with the XY-plane.

Using the floor segmentation masks we isolate the floor point cloud and use the outlier robust RANSAC regressor with inline threshold of 10cm to fit the plane equation:

$$z = \beta_z + \beta_x x + \beta_y y \tag{5.2}$$

To compute the rotation matrix that aligns the normal vector $\hat{\mathbf{n}}_{\text{floor}} = \mathbf{n}_{\text{floor}}/\|\mathbf{n}_{\text{floor}}\|$ (where $\mathbf{n}_{\text{floor}} = (0, 1, b_y)^T \times (1, 0, b_x)^T$) of the floor plane in the camera coordinate system with the normal vector $\mathbf{n}_{XY} = (0, 0, 1)$ of the XY-plane, we can utilize Rodrigues' rotation formula.

First, we construct the skew-symmetric matrix **K** from the components of the cross product vector $\mathbf{v} = \hat{\mathbf{n}}_{\text{floor}} \times \mathbf{n}_{XY}$. The skew-symmetric matrix **K** is defined as:

$$\mathbf{K} = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix} \tag{5.3}$$

Next, we compute the rotation matrix **R** using Rodrigues' rotation formula. The formula incorporates the identity matrix **I**, the skew-symmetric matrix **K**, and a scaling term based on the angle between the vectors. The resulting rotation matrix is given by:

$$\mathbf{R} = \mathbf{I} + \mathbf{K} + \mathbf{K}^2 \left(\frac{1-c}{s^2} \right) \tag{5.4}$$

Here, **I** is the identity matrix, c is the dot product of **a** and **b**, and s is the norm of the cross product vector **v**. This formula ensures that the rotation matrix **R** not only aligns **a** with **b** but also preserves the orthogonality and orientation of the coordinate system.

Finally, we transform the point cloud by translating it to align the floor plane intercept with the origin, followed by rotating it using the computed rotation matrix:

$$\mathbf{p}^* = \mathbf{R} \left(\mathbf{p} - (0, 0, b_z)^T \right) \tag{5.5}$$

In a similar fashion, we transform our poses by applying eq. (5.5) to our pose root joint translations. Since our joint rotations are relative to the root global orientation, **O**, we first compute its inverse global-to-local rotation matrix, $\mathbf{O}^{-1}$. After applying the new rotation matrix **R**, we then invert the result to convert back from local-to-global orientation:

$$\mathbf{O}^* = \left(\mathbf{R}\mathbf{O}^{-1} \right)^{-1} \tag{5.6}$$

The resulting transformation of the point cloud and poses in aligning the floor with the XY-plane is illustrated in fig. 5.2.

5.4 Motion and scene representation

We follow the work by Liang et al. 2024 and represent each pose in the motion sequence using their proposed non-canonical representation, but disregard the binary foot-ground contact features:

$$\mathbf{x}_t = \left[\mathbf{j}_g^p,\ \mathbf{j}_g^v,\ \mathbf{j}^r \right] \tag{5.7}$$

where $\mathbf{j}_g^p$ are the global joint positions, $\mathbf{j}_v^p$ the global joint velocities and $\mathbf{j}^r$ the relative SMPL 6D joint rotations.

5.5 Diffusion model

otion data for each person is flattened into a single array of poses. Each pose in the sequence is embedded using a linear layer. To embed the motion timestamp, we add positional encoding to the latent variables (z_t).

Identity encoding is achieved by randomly selecting (without replacement) a learned ID embedding for each person and adding it to each pose embedding. Additionally, class information (such as medical staff, patients, etc.) is encoded by adding a learned class embedding.

For textual descriptions, we use the CLIP model to encode the text, and the resulting output is embedded using a linear layer. This embedded text is then prepended to the input sequence for the transformer.

Object point clouds are processed by clustering the coordinates into groups of 64, which are then embedded using a linear layer. These embeddings are integrated into the model to provide spatial context.

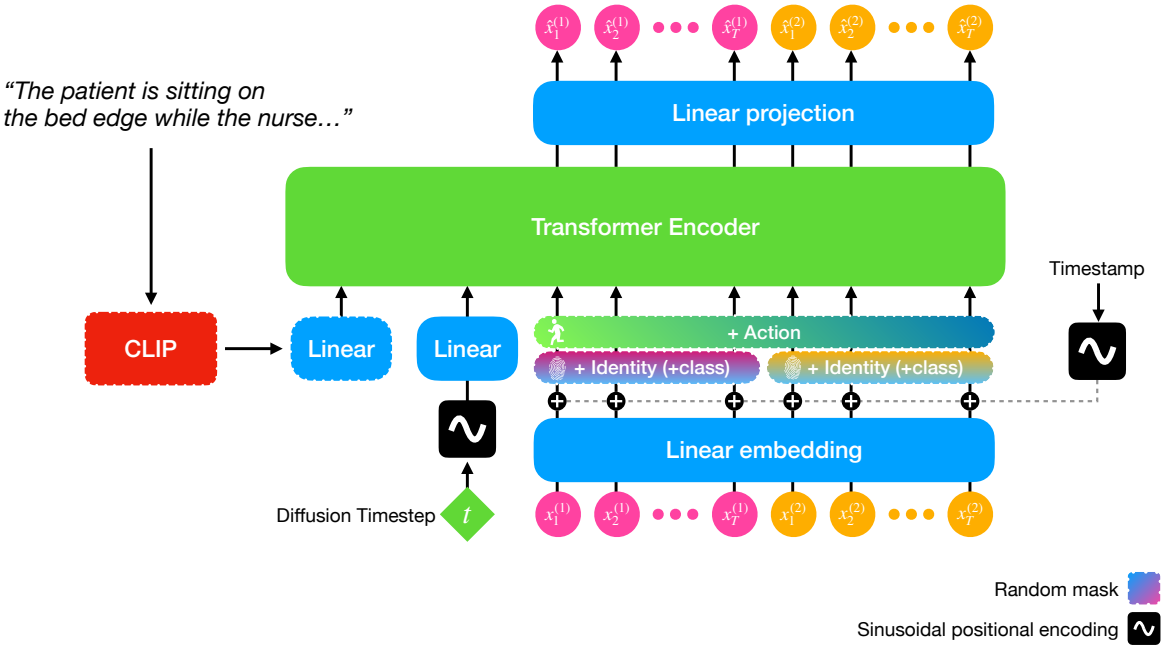


Figure 5.3: Diffusion architecture

6 Results

7 Discussion & Conclusion

Bibliography

Loper, Matthew et al. (Oct. 2015). “SMPL: A Skinned Multi-Person Linear Model”. In: *ACM Trans. Graphics (Proc. SIGGRAPH Asia)* 34.6, 248:1–248:16. DOI: 10.1145/2816795.2818013.

Bogo, Federica et al. (Oct. 2016). “Keep it SMPL: Automatic Estimation of 3D Human Pose and Shape from a Single Image”. In: *Computer Vision – ECCV 2016*. Lecture Notes in Computer Science. Springer International Publishing.

Pavlakos, Georgios et al. (2019). “Expressive Body Capture: 3D Hands, Face, and Body from a Single Image”. In: *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Kanazawa, Angjoo et al. (2018). “End-to-end Recovery of Human Shape and Pose”. In: *Computer Vision and Pattern Recognition (CVPR)*.

Kolotouros, Nikos, Georgios Pavlakos, Michael J Black, et al. (2019). “Learning to Reconstruct 3D Human Pose and Shape via Model-fitting in the Loop”. In: *ICCV*.

Kocabas, Muhammed, Nikos Athanasiou, and Michael J. Black (June 2020). “VIBE: Video Inference for Human Body Pose and Shape Estimation”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Mahmood, Naureen et al. (Oct. 2019). “AMASS: Archive of Motion Capture as Surface Shapes”. In: *International Conference on Computer Vision*, pp. 5442–5451.

Goel, Shubham et al. (2023). “Humans in 4D: Reconstructing and Tracking Humans with Transformers”. In: *International Conference on Computer Vision (ICCV)*.

Rajasegaran, Jathushan et al. (2022). “Tracking People by Predicting 3D Appearance, Location and Pose”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2740–2749.

Baradel*, Fabien et al. (2024). “Multi-HMR: Multi-Person Whole-Body Human Mesh Recovery in a Single Shot”. In: *ECCV*.

Kolotouros, Nikos, Georgios Pavlakos, Dinesh Jayaraman, et al. (2021). “Probabilistic Modeling for Human Mesh Recovery”. In: *ICCV*.

Tevet, Guy et al. (2023). “Human Motion Diffusion Model”. In: *The Eleventh International Conference on Learning Representations*. URL: https://openreview.net/forum?id=SJ1kSyO2jwu.

Liang, Han et al. (2024). “Intergen: Diffusion-based multi-human motion generation under complex interactions”. In: *International Journal of Computer Vision*, pp. 1–21.

Yuan, Ye et al. (2023). “PhysDiff: Physics-Guided Human Motion Diffusion Model”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Peng, Xingchao and Kate Saenko (2017). *Synthetic to Real Adaptation with Generative Correlation Alignment Networks*. arXiv: 1701.05524 [cs.CV].

Zheng, Stephan et al. (June 2016). “Improving the Robustness of Deep Neural Networks via Stability Training”. In: *2016 IEEE Conference on Computer Vision and Pattern Recog-*

nition (CVPR). IEEE. DOI: 10.1109/cvpr.2016.485. URL: http://dx.doi.org/10.1109/CVPR.2016.485.

Tremblay, Jonathan et al. (2018). *Training Deep Networks with Synthetic Data: Bridging the Reality Gap by Domain Randomization*. arXiv: 1804.06516 [cs.CV]. URL: https://arxiv.org/abs/1804.06516.

Shrivastava, Ashish et al. (July 2017). "Learning from Simulated and Unsupervised Images through Adversarial Training". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. DOI: 10.1109/cvpr.2017.241. URL: http://dx.doi.org/10.1109/CVPR.2017.241.

Dosovitskiy, Alexey, German Ros, et al. (2017). "CARLA: An Open Urban Driving Simulator". In: *Proceedings of the 1st Annual Conference on Robot Learning*, pp. 1–16.

Hu, Anthony et al. (2023). *GAIA-1: A Generative World Model for Autonomous Driving*. arXiv: 2309.17080 [cs.CV]. URL: https://arxiv.org/abs/2309.17080.

Hwang, Hochul et al. (2020). *ElderSim: A Synthetic Data Generation Platform for Human Action Recognition in Eldercare Applications*. arXiv: 2010.14742 [cs.CV]. URL: https://arxiv.org/abs/2010.14742.

Zhang, Yan et al. (June 2020). "Generating 3D People in Scenes without People". In: *Computer Vision and Pattern Recognition (CVPR)*. URL: https://arxiv.org/abs/1912.02923.

Romero, Javier, Dimitrios Tzionas, and Michael J. Black (Nov. 2017). "Embodied Hands: Modeling and Capturing Hands and Bodies Together". In: *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*. 245:1–245:17 36.6.

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 (July 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.

Ho, Jonathan and Tim Salimans (2021). "Classifier-Free Diffusion Guidance". In: *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*. URL: https://openreview.net/forum?id=qw8AKxfYbI.

Vaswani, Ashish et al. (2017). "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Dosovitskiy, Alexey, Lucas Beyer, et al. (2021). "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *ICLR*.

Gulati, Anmol et al., eds. (2020). *Conformer: Convolution-augmented Transformer for Speech Recognition*.

Zhou, Haoyi et al. (2021). “Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting”. In: *The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Conference*. Vol. 35. 12. AAAI Press, pp. 11106–11115.

Jumper, John et al. (Aug. 2021). “Highly accurate protein structure prediction with AlphaFold”. In: *Nature* 596.7873, pp. 583–589. ISSN: 1476-4687. DOI: 10.1038/s41586-021-03819-2. URL: https://doi.org/10.1038/s41586-021-03819-2.

Devlin, Jacob et al. (2018). “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *arXiv preprint arXiv:1810.04805*.

Radford, Alec et al. (2019). *Language models are unsupervised multitask learners*. URL: https://www.semanticscholar.org/paper/Language-Models-are-Unsupervised-Multitask-Learners-Radford-Wu/9405cc0d6169988371b2755e573cc28650d14dfe (visited on 01/06/2023).

Touvron, Hugo et al. (2023). *Llama 2: Open Foundation and Fine-Tuned Chat Models*. arXiv: 2307.09288 [cs.CL]. URL: https://arxiv.org/abs/2307.09288.

Anil, Rohan et al. (2023). *PaLM 2 Technical Report*. arXiv: 2305.10403 [cs.CL]. URL: https://arxiv.org/abs/2305.10403.

Petrovich, Mathis, Michael J. Black, and Gül Varol (2021). *Action-Conditioned 3D Human Motion Synthesis with Transformer VAE*. arXiv: 2104.05670 [cs.CV]. URL: https://arxiv.org/abs/2104.05670.

Shi, Mingyi et al. (Sept. 2020). “MotioNet: 3D Human Motion Reconstruction from Monocular Video with Skeleton Consistency”. In: *ACM Transactions on Graphics* 40.1, pp. 1–15. ISSN: 1557-7368. DOI: 10.1145/3407659. URL: http://dx.doi.org/10.1145/3407659.

Yang, Lihe et al. (2024). “Depth Anything: Unleashing the Power of Large-Scale Unlabeled Data”. In: *CVPR*.

Nathan Silberman Derek Hoiem, Pushmeet Kohli and Rob Fergus (2012). “Indoor Segmentation and Support Inference from RGBD Images”. In: *ECCV*.

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:

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

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