Motion Priors for Pose Estimation and Animation Workflows

Luke Smith

Master Thesis
April 2022

Prof. Dr. Robert W. Sumner

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

computer graphics laboratory

Abstract

TODO: Abstract

Zusammenfassung

TODO: translate to German

Contents

List of Figures

List of Tables

Introduction

Motion capture is an integral part of many modern animation pipelines. It can be defined simply as the act of recording, by automatic means, the motion of a person, animal or object. Throughout this this thesis we will primarily interest ourselves in the capturing of human motion. The advantages of motion capture, over traditional animation, are numerous and important in their scope. Motion capture allows for

- quasi-real time results
- high quality motion with realistic object interactions
- often reduced costs as compared to hand animation
- complexity that is constant with respect to the motion being captured

however also has it's inherent costs, notably

- uprfront costs for tailored software and hardware
- cumbersome motion capture suits
- lengthy setup times and a non-trivial capture systems requiring expert knowledge
- artifacts due to retargeting of skeletons.

These drawbacks mean that the technology is often prohibitive for small teams and individual artists/animators. It is clear though that overcoming these barries would provide huge benefits and open up the technology to a wide range of new users. For example individual artists could more efficiently prototype and develop animation sequences starting from a self captured motion, and small teams could readily make use of this technology to do more with less.

The important question therefore is simply; **how might we create a motion capture system without such upfront costs, specialized hardware and need for motion capture suits?** At Disney Research|Studios an approach is being investigated that aims to capture motion directly from RGB video. The system currently follows the following steps:

1. record a video of a human motion sequence (potentially from multiple angles)
2. perform per frame pose estimation using a machine learning model
3. run the resultant motion sequence through an optimiser to improve the quality of the motion (and potentially lift to 3d)

and shows great promise, however has a number of drawbacks. The most notable drawback comes from the fact that the per frame pose estimation system does not produce temporally consistent results. The predictions are made independantly per frame, hence the result motion is jittery, and can contain other artifacts. This is the motivation for the introduction of an optimisation system at the end of the pipeline to counteract these artifacts. However is must be noted that, although smoothness is mostly acheived, it does not fix all issues. The optimiser struggles to handle occluded motion sequences, it does not always deal with limb flips (where the left/right leg/arm are predictions are horizontally flipped for a frame), and cannot always fix completely wrong but confident predictions. It is of interest therefore to try and improve this aspect of the pipeline.

The data driven approach proposed in this thesis, in its most abstract form, is simpy that of learning a model that understands human motion. Depending on its conception, such a model could be employed in a number of manners; it could be directly applied to the task of rectifying a motion sequence, it could be used as a loss in the existing optimiser, or it could even simply be used to improve upon certain failings of the optimizer as an additional step to the pipeline, such as fixing occluded motion.

This thesis tackles the task of exploring such motion models with a primary goal of shedding light on the most promising model architecture and the most effictive manner in which such a model might be used.

TODO: DESCRIBE WHAT WAS DONE IN THE THESIS MORE

Related Work

The study of synthesising human motion has a long history motivated largely by the desire to create realistic and captivating media in the gaming and film industries. Early adaptive methods rely on motion matching [WH97] [Cla] in which interpolation between similar motions from a database of captured motion is performed. This however does not scale well to out of database motion and often generates generic, non stylized motion, though efforts have been made to introduce learned aspects to such methods [HKPP20] to improve upon their shortcomings.

TODO: MORE?

More recent branches of motion modelling commonly base themselves upon the use of machine learning techniques, notably deep learning, to learn a prior over plausible motion. This is a more general approach that can be applied to a wider range of tasks, and shows promise in overcoming some of the issues of motion inbetweening, as such systems can learn to better generalise to out of distribution motion sequences.

Within the area of deep learning, many techniques have been investigated, temporal convolutions [HSK16a], recurrent models [HYNP20], and reinforcement learning [CKP$^+$21] are but a few examples.

2.1. Motion AutoEncoders

A well explored model is that of the AutoEncoder (AE) [BKG21] or Variation-AutoEncoder (VAE) [KW22]. These are popular models as they encourage the learning of a latent representation [BKG21] of human motion, thus the intuition is that they learn not just to reproduce the data, but actually how humans move, thus providing a more robust prior.

TODO: MORE? c.f MEVA for nice related work section

Holden et al. [HSKJ15] [HSK16b] present simple CNN based autoencoder architectures that operates on motion sequences. The notion of skeletal aware convolutions and pooling/unpooling operations for a VAE, alongside a sliding window method for motion rectification are presented by the authors of [LVC+21]. [CSY+22] presents a novel approach of leaning a latent space, then projecting directly to this latent space from a motion sequence using a separate model, again operating directly on a motion sequence. The authors of MEVA [LGK20] postulate that a VAE often learns only smooth motion, as we are asking too much of the model, thus present a pipeline in which a smooth motion and coarse motion VAEs are jointly used. Holden et. al make another appearance with DeepPhase [SMK22], an autoencoder with a latent space enforced to match sinusoidal functions that represent periodic motion. Contrary to the common trope of sequence level models, the authors of [TWH+22] and of [LZCvdP21] operate in a frame to frame regime, predicting temporally local change of motion. Finally, a number of works present the Conditional-VAE architecture [SLY15] as a base with varying state representations, conditioning variables and loss terms, [RBH+21, TWH+22, LZCvdP21, MWFD21].

As we can see the literature is rich and diverse, but we found ourselves drawn to the HuMoR model [RBH+21], due to it's apparent performance and it's use for the exact task that we desire to solve, that of rectifying a motion sequence captured through frame by frame pose estimation. The authors of [RBH+21] present a C-VAE architecture that learns a distribution over latent transitions, conditioned on the previous pose. They use this architecture alongside an optimisation method that rectifies human motion obtained from, among other modalities, RGB video through pose estimation.

2.2. Motion Diffusion

With the notable success of diffusion models in the image generation literature [HJA20] [DN21] [RBL+21], diffusion models have begun to spread into many other fields within machine learning [YZS+23], and this also holds true of the task of human motion modelling.

The authors of Avatars grow legs [DKP+23] denoise a sequence of SMPL [LMR+15] parameters condition on sparse tracking inputs, notably taking the form of the orientation/translation of a headset and two hand controllers. They show that plausible motion can be generate from very sparse signals, thus indicating to us that the use of diffusion models in the rectification of occluded motion sequences is promising. Next, PhysDiff [YSI+22] provides a text conditioned diffusion model with the unique use of a physics based motion projection step in the difussion process that helps to ensure physical plausibility of the generated motion. The authors of EDGE [TCL22] propose a simple

HuMoR

The authors of HuMoR [RBH$^+$21] present a novel approach for learning and using a plausible motion prior. They train a conditional VAE that learns a distribution over latent transitions, in a canonical reference frame, between *states* that consist of a root translation, 3D joint positions, joint angles, and the respective velocities. They most notably use this model as a prior in a 'test time optimisation', which generates plausible sequence motions optimising for an initial state and a sequence of transitions starting from frame by frame estimates (2D/3D joints or points clouds). This optimisation includes, alongside others, a motion prior term based upon the conditional distribution $p(z_t|x_{t-1})$ that encourages plausible motion for the learned sequence. Note that the CVAE decoder also predicts foot contacts alongside change in state, which are used as regularisers during their main use case of 'test time optimisation'. The test time optimisation can operate on many modalities, 2D/3D joints, point clouds, etc., as the optimisation loss contains a Data Term ϵ_{data} that can be tailored to the modality as the HuMoR state is information rich, containing 3D joints (hence can fit to 2D joints through projection or directly to 3D) and can parametrise the SMPL model (hence the SMPL mesh can be correlated to point clouds). The initialisation for the test time optimisation is based upon VPoser TODO: Complete.

The performance of HuMoR as described in the paper [RBH$^+$21], alongside it's use in a problem that directly matches our own, lead us to evaluate and investigate the model Section 3.2, and subsequently try to extend and improve 3.3 upon it once the limitations became clear.

3.1. Model

TODO: Describe the model in more detail, include architecture diagrams etc., describe the rollout and the various stages of the model

3.2. HuMoR Investigation

3.2.1. Method

Our investigation began with a largely qualitative evaluation of the HuMoR model which had two main aims. First was to stress test the system, to see where it failed, where it succeded, and if the mentioned benefits in the HuMoR paper [RBH+21] were as described. Second was to evaluate the model with the defects of the current Disney Research|Studios system in mind to see if it could complement it's functionality, notably if it could improve upon occluded motion, joints flipping and confident but false predictions.

To acheive this goal, a selection of videos were taken in the Disney Research|Studios lab containing a variety of motion; fast, slow, and abnormal, and a including number of occluded scenes. The HuMoR system was ran on these videos and the results were investigated. The TestOps is performed in 3 stages, where only the 3rd makes use of the HuMoR motion model, we could therefore compare the stage 2 results to the stage 3 results to see where the model was providing an improvement over a more classical optimisation that includes smoothness and pose prior losses.

3.2.2. Advantages of HuMoR

We see a number of situations in which the model shows a clear improvement over the stage where the HuMoR model is not used.

In an occluded situation, as shown in Figure 3.1, where the 2d pose predictions don't have any information on the legs, the model manages to produce a realistic sitting motion.

(a) OpenPose **(b)** Stage 2 **(c)** Stage 3

Figure 3.1.: *HuMoR achieving occluded sitting*

We also note in many of the less difficult videos, that HuMoR produces clean motion and deals with many abnormal movements without obvious issues. It therefore doesn't seem to regress the easy situations, but can improve the more difficult situations, notably occlusions.

3.2.3. Drawbacks of HuMoR

We noted that while HuMoR manages to sit when occluded, it often fails to walk. This seems to be due to the fact that OpenPose often predicts both legs on the frames just before the occlusions where only one leg is actually un-occluded, as can be seen in Figure 3.2. This results in a sequence of poses where the frames before an occlusion indicate that the person is no longer walking, hence making it significantly more difficult for HuMoR to begin walking again during the occlusion. This issue is thus probably largely due to HuMoRs' TestOps' dependence on OpenPose and thus it's inheritence of OpenPoses' failure points.

(a) OpenPose **(b)** Stage 2 **(c)** Stage 3

Figure 3.2.: *Occluded walking failure point, 2 legs predicted instead of 1*

We also found that the choice of axis angle representation for various rotations led to the emergence of common known issue that arising from the discontinuities present in the representation [ZBL+18]. The shortest path between certain angles in axis angle space can lead to a 360 degree rotation, as seen in Figure 3.3, among other issues.

(a) **(b)** **(c)**

Figure 3.3.: *Axis angle issue: dubious rotations*

We found that the TestOps system would occasionally get itself into a tangle, as seen in Figure 3.4, we assume due to the autogregressive nature resulting in a potentially catastrophic cumulation of errors, and found this happended most notably for a sequence containing someone rolling on the floor. It is interesting to note that these sorts of motions were not often, if ever, present in the training data.

(a) OpenPose **(b)** Stage 2 **(c)** Stage 3

Figure 3.4.: *HuMoR in a tangle*

Next, we saw overly smoothed motion for certain clips, most notably for particularly stylised motion such as dancing. We also found that if there is a single frame without Open Pose predictions then as of that frame the TestOps fails. Finally, and most importantly, we found that the TestOps was extremely slow. It took around 20mins per 2s clip, and that we could batch at most 4 2s clips together, so 20mins per 8s on a Nvidia GeoForce 1080ti with 11Gbs memory.

We concluded that there were a number of potential benifits, notably in occluded situations, and that many of the issues, including the axis angle representation issue, the dependence on humor, and smooth motion, have a good chance of being fixed with different design choices. The glaring issue however was the unresonable amount of time the system took, rendering it unusable for our purposes as we wish the system to be close to real time.

3.2.4. Profiling

To investigate this speed issue, we profiled the code, as can be seen in Figure 3.5. We found that the program spent 90% of it's time in the Stage 3 optimiser closure, 56% of it's time performing the backwards step and 32% of it's time in the rollout function. Hence it was clear that the speed issue was due to the slow act of rolling out and the large computation necessary to perform the backwards step on the large computation graph resulting from the rollout.

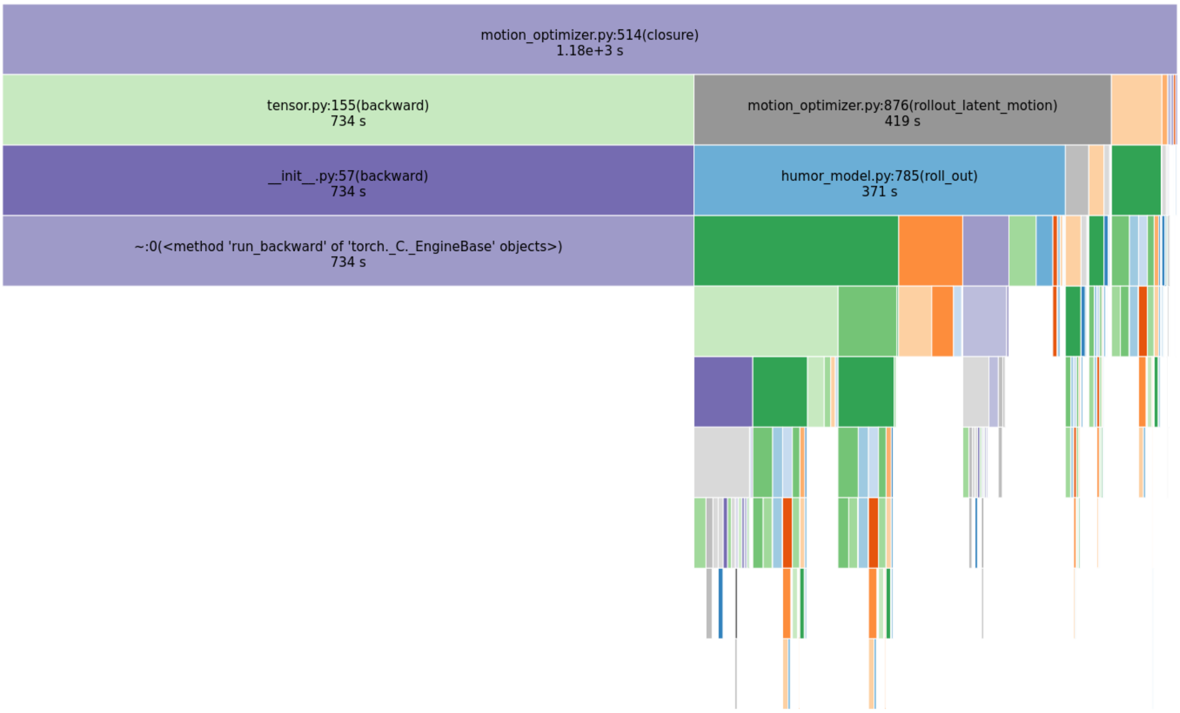


Figure 3.5.: *TestOps profiling*

3.2.5. Conclusion

We came to the final conclusion that the HuMoR motion model alongside the TestOps optimisation showed potential, but that it must be sped up before we can conclusively say if it is of particular use to us. We note that to speedup the method, we must primarily focus on improving/removing the rollout. Thus, in the next section, an attempt to speed up the TestOps is made.

3.3. Improving HuMoR TestOps

This section presents the attempt that was made at modify the TestOps of the HuMoR paper [RBH+21] in such a way that we attain comparable results but in a significantly shorter timespan.

TODO: See if I have introduced the terms properly (decoded sequence, x's, x''s, etc.)

3.3.1. Speeding up the TestOps

Realising that the main speed limitations are due to the concept of rollout over the entire sequence, we decided to break this long range dependence. Through short overlapping rollouts, it is possible to acheive more parrallelisation and smaller computation graphs, and our intuition suggests that the need for such a large context window is not necessary, that only a small number of autoregressed frames are needed to be able to propagate information sensibly and to get meaningful gradients. Anecdotaly, it would seem intuitive that if a video contains sitting, then walking, then sitting again, the second act of sitting would not depend in on the early act of sitting, hence rolling out over the whole sequence seems unnecessary.

The HuMoR TestOps rollout is presented graphically in Figure 3.6. x is the state, and z are the latent variables, the optimised variabels are highlighted in pink, and the losses are coloured, gray x's indicate they are calculated during the optimisation from the optimised variables. As can be seen, first all the states are autoregressed from the initial state and the latent variables, and then the losses are applied. This can be seen to result in very long range dependencies in the computation graph, thus making the gradients time consuming to calculate.

TODO: Update Figure 3.7 and remove the 'NEW' norm, labels on green loss, and potentially red loss?

The most obvious way to break this autogregression is to maintain and optimise a separate sequence of x's, updating this seperate sequence with reference to the x''s decoded from the latent variables as in Figure 3.7. We can take into account the decoded x''s in several manners:

- Copy over
  - Directly replace the optimised sequence of x's with the decoded x''s.
- Blend
  - Perform a weighted addition of the optimised x's and decoded x''s.
- Loss term
  - Add a loss term on the difference between the optimised x's and decoded x''s.

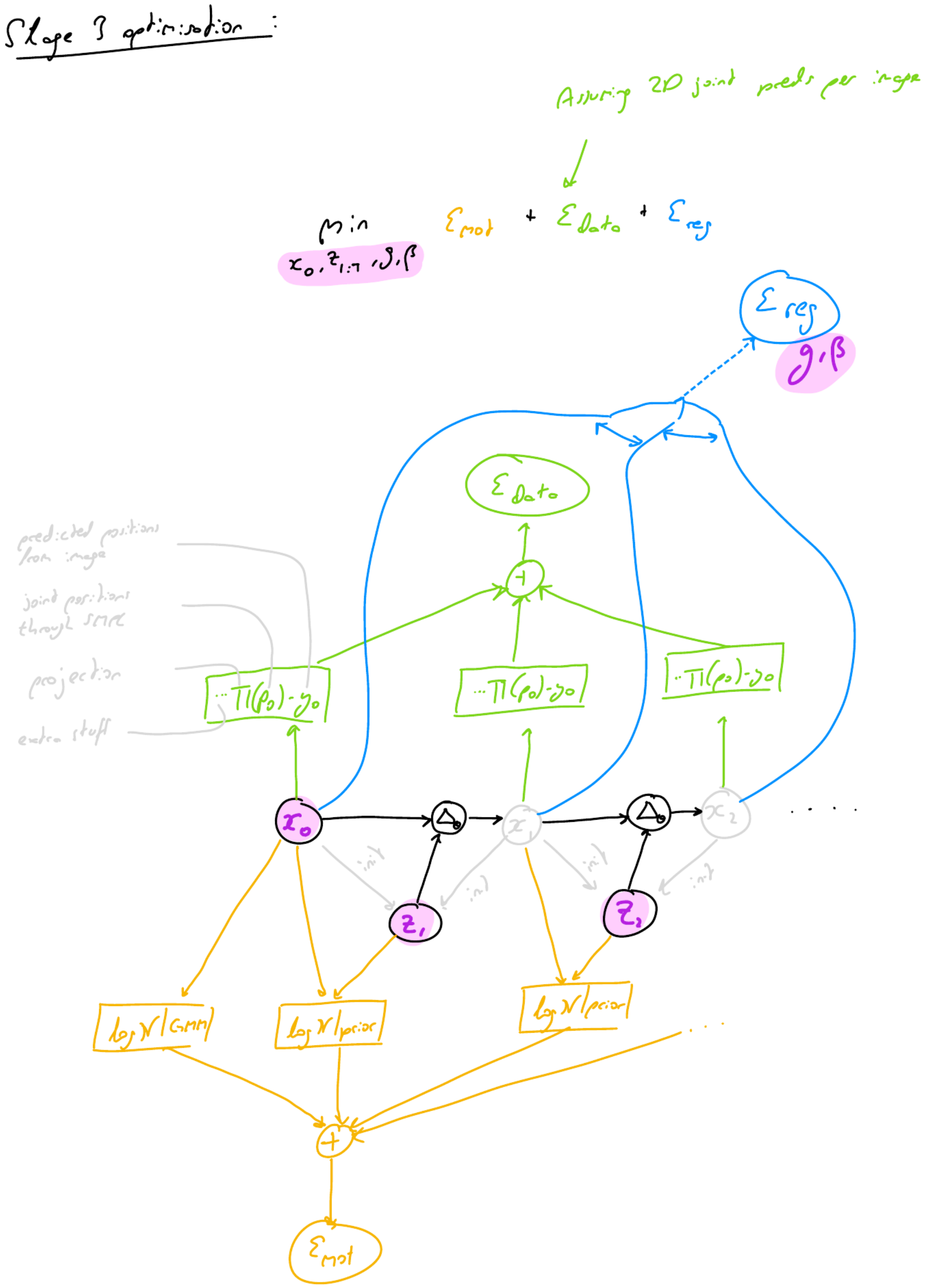


Figure 3.6.: *TestOps Computation Graph*

All decoding steps can now be performed in parrallel, rather than sequentially, which greatly reduces the computation time.

This new method also allows us to experiment with different amounts of rollout. We can once again decode the decoded sequence of x''s to get x'', and so on and so forth. These extra sequences are all decoded with the same latent variables thus the updates to the latent variables will take into account the losses on all the decoded sequences. Graphically this can be seen in Figure 3.8, where each next line . TODO: describe the figure better

The information contained in each x can flow forwards through the short range rollouts, and over the iterations of the optimiser. Note however that the information flow will be slower than the HuMoR TestOps, as in the TestOps all subsequent x's are regressed from the initial x_0. In the HuMoR TestOps information can flow backwards from any future frame's x in one update through the gradients, as all x's are linked in the computation graph through the auto-regression (though our intuition suggests that the information does not actually need to flow so far, as previously discussed). In the new method however, the information flow will be much more limited, as the rollouts will be significantly shorter, though again the information should flow through the iterations fo the optimiser.

3.3.2. Implementation notes

A number of issues were encountered during the implementation that are worth mentioning. After implementing the new method, attempts were made to get the optimiser to produce sensible results. The general approach was to use a small number of losses, so as to get a better intuition of each loss, with the goal of better understanding how to balance the losses in the optimiser, but some errors in individual losses were encountered.

Firstly, as expected, the Axis-Angle representation became a problem. When running through functions to convert to root-local reference frame for the HuMoR model for decoding (as it operates on said reference frame), and then back to world reference frame for the opftimiser, the Axis-Angle vector flipped direction. Though the flipped vector still represented the same rotation, the angle representations of the same angle in the optimised sequence of x's and the decoded sequence of x''s were different, hence encorporating the information from decoded sequence cause the angle to move somewhere between the two flipped values which was no longer a sensible angle representation, thus resulting in root flips and other strange effects.

Secondly, an issue due to the SMPL model was found. When using a 2d reprojection loss (comparing the projected SMPL joints to the OpenPose 2d predictions) on a sequence where only the left arm and face had any OpenPose predictions, the legs were being moved by the optimiser. This was wrong as there were only losses on the arm and face joints, hence no gradients should flow to the legs. It was eventually found that the skinning weights of the SMPL model [LMR+15] contain spurious long range connections. 3% of the LBS (linear blend skinning) weights in the SMPL model are non-zero but less than $1e-2$, which seems rather too low to have any meaningful effect on the skinning, but which allows for spurious gradients to flow.
For the comparison to the OpenPose predictions the OpenPose skeleton must be obtained from the SMPL mesh. Certain joints are regressed, other are simply taken directly as vertices from

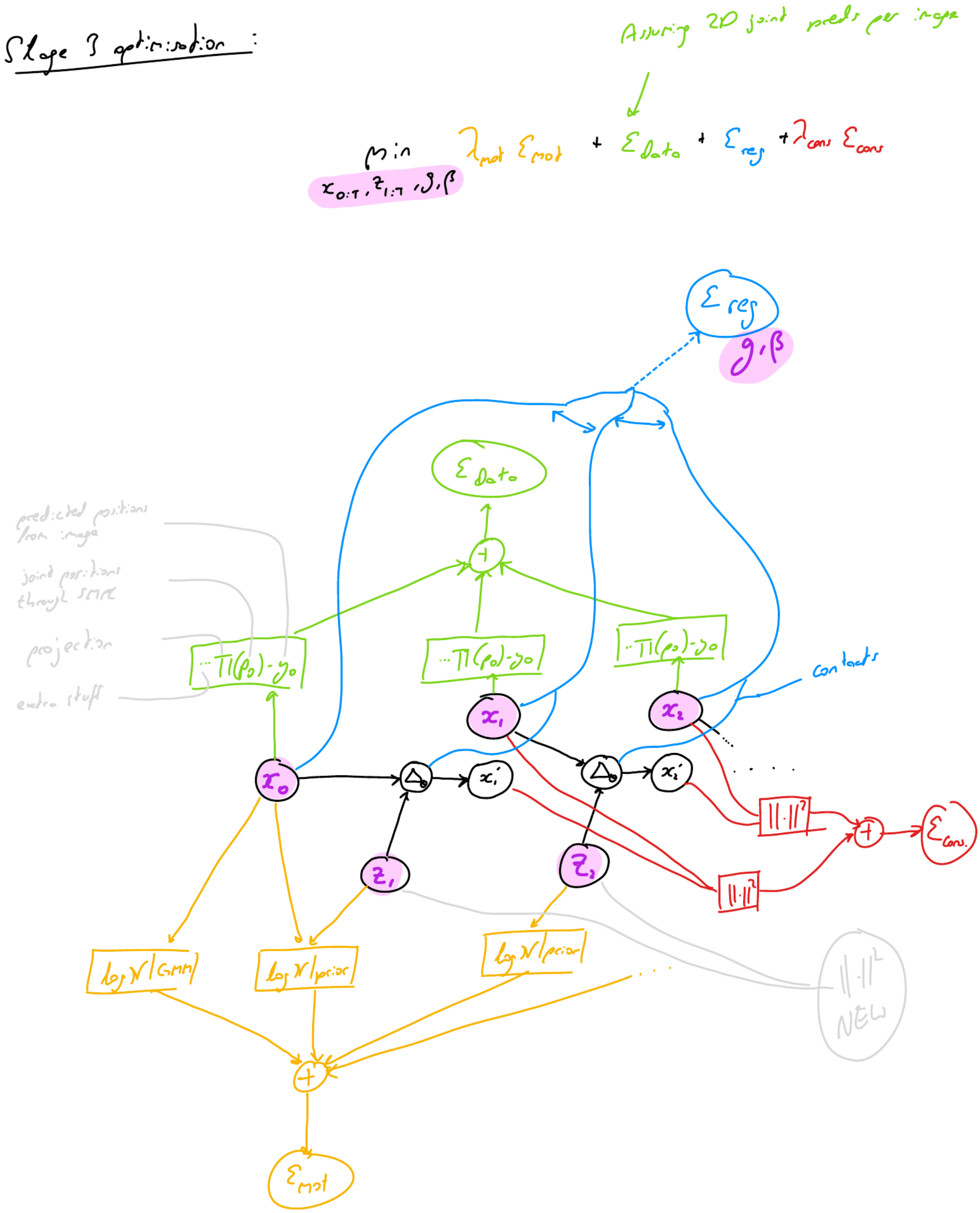


Figure 3.7.: *Decoupled Computation Graph*

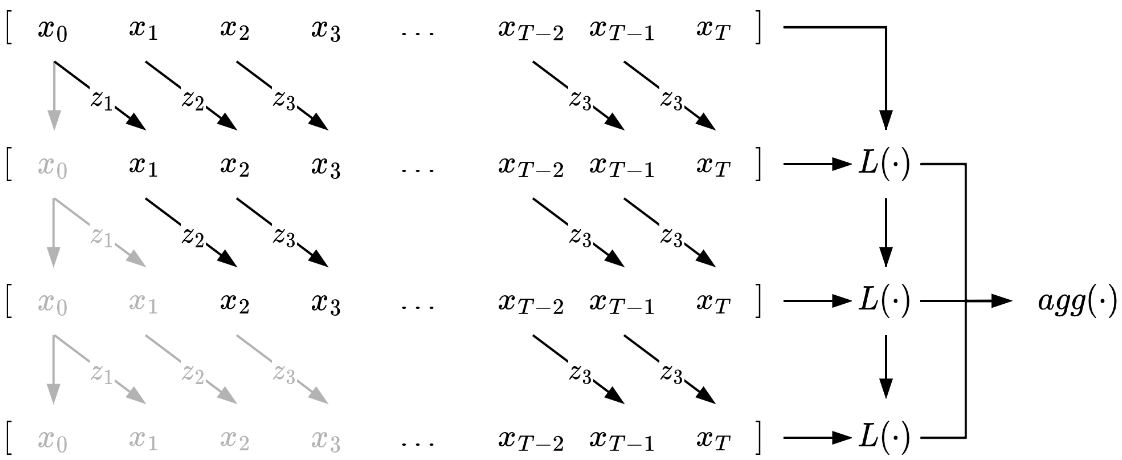


Figure 3.8.: *Decoded sequences*

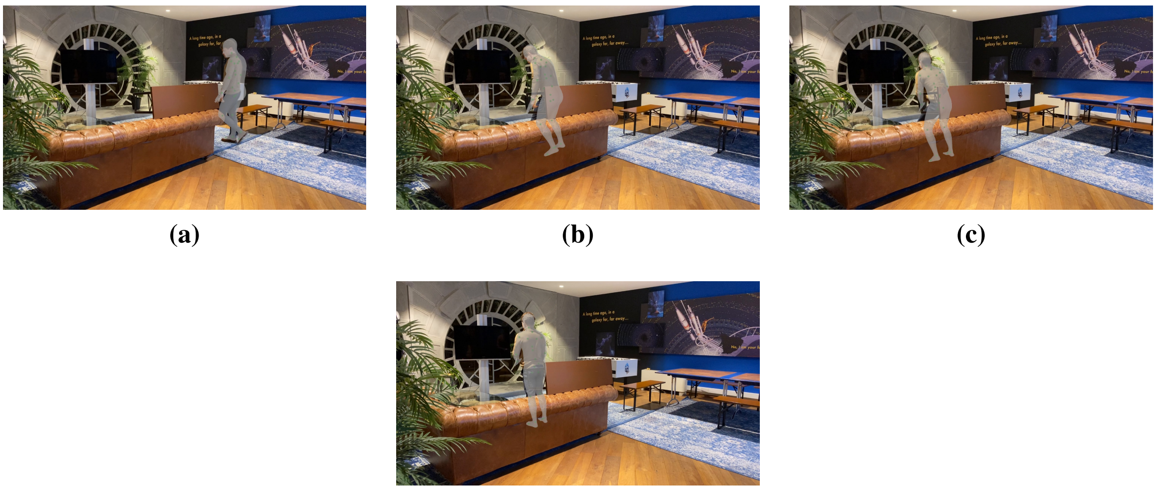
the mesh, as in the case of the nose joint in the OpenPose skeleton which is taken to be vertex 332 [SMP]. We noted that this vertex 332 is skinned 99.8% by the 'head' joint, but also 0.2% by the right hip, left knee and right knee. This issue was solved by pruning all weights below 1e-2. It is interesting to note that there are known spurious connections in the SMPL blend shapes [OBB20], but that we have found no reference to spurious connections in the LBS weights.

3.3.3. Experiments

TODO: explain why the initial rollout was bad, and the compounding of errors forward TODO: maybe we should discuss more in the humor investagtion section about the bad initial rollout TODO: Check everything is explicitely named, e.g have a graphic clearly naming the optimised x's and the decoded x's, and what the z's are

To begin with, we looked into the initial z's that were obtained from projecting the Stage 2 results through the HuMoR encoder. We found that in occluded situations these z's encode a sitting motion without any optimisation as seen in Figure 3.9, that simply the projection leads them to encode the necessary occluded motion. However we note that when autoregressively rolled out over a longer sequence the results, in Figure 3.11, deviate largely from the initial sequence of x's seen in Figure 3.10. This shows us that though the initial z's encode some sensible motion locally, and can be used to accurately recover the next frame, when they are chained together small deviations from the x motion accumulate to create an overall deviation. For example if the arm is moved slightly too much by a single z, then all the future frames will have the arm in slightly the wrong position, and multiple z's are wrong in the same direction, then the arm will raise up. This deviating sequence is the initial starting rollout in the Stage 3 of the HuMoR TestOps in which the rollout is optimised, though it seems unfortunate to us that from a sensible sequence of x's obtained from Stage 2 we get such a bad sequence through the autoreggresive rollout. We do however note that though while this starting point represents a sequence that deviates largely, the z's might not need to move so far to avoid the accumulation effect. The intuition in our case was that we are starting from some sensible x's and z's, and therefore that it should be possible to refine the x's by take into account the HuMoR model through the z's whilst updating the z's.

Figure 3.9.: *Stage2 z's encode a sitting motion*

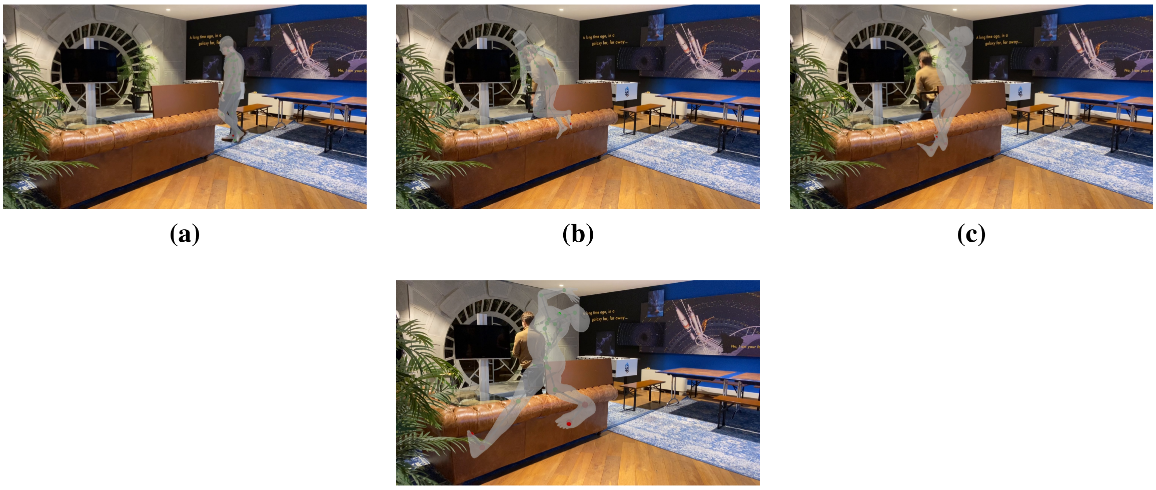


(a) **(b)** **(c)**

(d)

Figure 3.10.: *Stage 2 x's*

With some preliminary experiments in which we began optimising x's and z's we found that the most successful approach to blending the decoded and optimised x's was to have an L1 loss on the distance between them. We saw that when simply copying over, the propagation of small errors forward was too strong (as desribed in Figure 3.11) and the system failed to recover as the long range gradients present in the HuMoR TestOps method weren't present, and we found when blending rather than completely copying over that the weighting of the blending



(a) **(b)** **(c)**

(d)

Figure 3.11.: *Deviation due to rollout of Stage 2 z's*

was another parameter that was difficult to choose, hence having a loss was the path of least resistence.

Next we experimented with jointly optimising x and z with a larger variety of losses. It was noted that the sitting down motion was not acheived consistenly. However as previously mentioned, the initial z's encode a sitting motion, thus we conject that the x's and z's were fighting and this sitting motion was lost, that is to say the z's move to match the x's motion in which the skeleton simply clips into the floor without bending legs, thus the z's begin encoding less leg bending. To avoid this issue in future experiments, the z's were detached from the consistency loss making the optimised x's match the decoded x's.

This led us on to some experiments in which we either optimise z whilst fixing x, or vice versa. With the z's fixed at the stage 2 results, we managed to acheive a sensible sitting motion for a clip containing occluded sitting, but when we applied the same optimiser settings to a longer clip the results were no longer satisfactory. We also noted that when it did work, it required a large number of iterations, in the 1000s, which took 10mins or more, thus the speedup was not as evident as hoped (though our method would continue scaling better). In the inverse case, fixing x to the final HuMoR optimised states and optimising z, we noted that we consistently converge in the direction of the final HuMoR z's, which indicates that we are optimising in the right direction, with the best results acheived at a rollout of 5 decoding steps.

All these experiments were performed with varying numbers of decoding steps, 1, 2, 5, and 10. We found that the effect of more stable results but slower compute times with longer rollouts did manifest as intuition would suggest.

These experiments suggest that the next direction to pursue would be an optimisation scheme in which we flip flop optimising z then x. However considering our difficulty in achieving consistent results for the optimisation of the x's with fixed z's, our feeling that this might not be the optimal solution to the problem, my desire to try a new direction, and the time constraints, we deemed it time to move on and try a new method.

3.3.4. Conclusion

We found that jointly optimising the x's and z's did not prove an easy task to solve, which may indicate that

- it is a particularly difficult problem, and therefore that the optimiser struggles to satisfy the constraints and find a good minimum for both x and z at the same time
- the coupling of the x's and z's in the HuMoR decoder is an important issue. We found it to be at least a minor issue as it was necessary to decouple the z's from the consistency loss (matching the optimised xs to the decoded x''s) to avoid the z's being modified in such a way as to fit the unoptimised x's at the beginning of the optimisation
- the optimiser is badly balanced

This at the very least suggests that a new decoder that does not couple x's and z's would be useful and worth investigating if a similar method is pursued.

In the investigations we found it possible to acheive sensible results on a given clip (for example fixing the unoptimised initial z's, we can a plausible sitting motion in a small video of sitting in occlusion) but that the same settings for the optimiser failed to acheive a good result on an alternative clip. This may indicate that

- again, this is a difficult problem that the optimiser will struggle to solve
- the optimiser is badly balanced

as before.

While many of the issues encountered may simply be due to a badly balanced optimiser, considerable effort was made to mitigate this potential issue, and therefore it is our impression that this problem is at the very least a difficult one. We have improved our intuition, fixed several issues, and understood more about the problem and our goals. We therefore can make a more informed decision with respect to which direction to pursue next.
We found that taking a system that was designed with autogregression in mind, and trying to parrallelise it, proved more difficult than expected. The experiments lead us to beleive that jointly optimising a sequence of poses and latent variables obtained through a single frame decoder may be a more difficult formulation of the problem than others. We therefore conclude that a method in which we model longer motion sequences may be more fruitful.

TODO: Double check through Notes/Current_experiments.md to see if there's anything else we can learn from the experiments

Motion Diffusion Models

4.1. Method

TODO: Describe diffusion models

4.2. Experiments

4.3. Conclusion

Conclusion and Outlook

TODO: Chpt 1

Appendix

Appendix intro

A.1. Section

TODO:

Bibliography

[BKG21] Dor Bank, Noam Koenigstein, and Raja Giryes. Autoencoders, 2021.

[CKP+21] Kyungmin Cho, Chaelin Kim, Jungjin Park, Joonkyu Park, and Junyong Noh. Motion recommendation for online character control. *ACM Trans. Graph.*, 40(6), dec 2021.

[Cla] Simon Clavet. Motion matching and the road to next-gen animation.

[CSY+22] Xin Chen, Zhuo Su, Lingbo Yang, Pei Cheng, Lan Xu, Bin Fu, and Gang Yu. Learning variational motion prior for video-based motion capture, 2022.

[DKP+23] Yuming Du, Robin Kips, Albert Pumarola, Subastian Starke, Ali Thabet, and Artsiom Sanakeyeu. Avatars grow legs: Generating smooth human motion from sparse tracking inputs with diffusion model, 2023.

[DN21] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. *CoRR*, abs/2105.05233, 2021.

[HJA20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *CoRR*, abs/2006.11239, 2020.

[HKPP20] Daniel Holden, Oussama Kanoun, Maksym Perepichka, and Tiberiu Popa. Learned motion matching. *ACM Trans. Graph.*, 39(4), aug 2020.

[HSK16a] Daniel Holden, Jun Saito, and Taku Komura. A deep learning framework for character motion synthesis and editing. *ACM Trans. Graph.*, 35(4), jul 2016.

[HSK16b] Daniel Holden, Jun Saito, and Taku Komura. A deep learning framework for character motion synthesis and editing. *ACM Trans. Graph.*, 35(4), jul 2016.

[HSKJ15] Daniel Holden, Jun Saito, Taku Komura, and Thomas Joyce. Learning motion manifolds with convolutional autoencoders. In *SIGGRAPH Asia 2015 Technical*

Briefs, New York,NY,USA, 2015. Association for Computing Machinery.

[HYNP20] Félix G. Harvey, Mike Yurick, Derek Nowrouzezahrai, and Christopher Pal. Robust motion in-betweening. *ACM Transactions on Graphics*, 39(4), aug 2020.

[KW22] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022.

[LGK20] Zhengyi Luo, S. Alireza Golestaneh, and Kris M. Kitani. 3d human motion estimation via motion compression and refinement. *CoRR*, abs/2008.03789, 2020.

[LMR+15] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Smpl: A skinned multi-person linear model. *ACM Trans. Graph.*, 34(6), nov 2015.

[LVC+21] Jiaman Li, Ruben Villegas, Duygu Ceylan, Jimei Yang, Zhengfei Kuang, Hao Li, and Yajie Zhao. Task-generic hierarchical human motion prior using vaes. *CoRR*, abs/2106.04004, 2021.

[LZCvdP21] Hung Yu Ling, Fabio Zinno, George Cheng, and Michiel van de Panne. Character controllers using motion vaes. *CoRR*, abs/2103.14274, 2021.

[MWFD21] Mathieu Marsot, Stefanie Wuhrer, Jean-Sébastien Franco, and Stephane Durocher. Multi-frame sequence generator of 4d human body motion. *CoRR*, abs/2106.04387, 2021.

[OBB20] Ahmed A. A. Osman, Timo Bolkart, and Michael J. Black. STAR: sparse trained articulated human body regressor. *CoRR*, abs/2008.08535, 2020.

[RBH+21] Davis Rempe, Tolga Birdal, Aaron Hertzmann, Jimei Yang, Srinath Sridhar, and Leonidas J. Guibas. Humor: 3d human motion model for robust pose estimation. In *International Conference on Computer Vision (ICCV)*, 2021.

[RBL+21] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021.

[SLY15] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[SMK22] Sebastian Starke, Ian Mason, and Taku Komura. Deepphase: Periodic autoencoders for learning motion phase manifolds. *ACM Trans. Graph.*, 41(4), jul 2022.

[SMP]

[TCL22] Jonathan Tseng, Rodrigo Castellon, and C. Karen Liu. Edge: Editable dance generation from music, 2022.

[TWH+22] Xiangjun Tang, He Wang, Bo Hu, Xu Gong, Ruifan Yi, Qilong Kou, and Xiaogang Jin. Real-time controllable motion transition for characters. *ACM Transactions on Graphics*, 41(4):1–10, jul 2022.

[WH97] D. J. Wiley and J. K. Hahn. Interpolation synthesis of articulated figure motion. *IEEE Computer Graphics and Applications*, 17(06):39–45, nov 1997.

[YSI+22] Ye Yuan, Jiaming Song, Umar Iqbal, Arash Vahdat, and Jan Kautz. Physdiff: Physics-guided human motion diffusion model, 2022.

[YZS+23] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications, 2023.

[ZBL+18] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks, 2018.