Bachelor Thesis

3D SKELETAL RECONSTRUCTION

Marios Piskopou

UNIVERSITY OF CYPRUS

DEPARTMENT OF COMPUTER SCIENCE

May 2025

UNIVERSITY OF CYPRUS

DEPARTMENT OF COMPUTER SCIENCE

3D SKELETAL RECONSTRUCTION

Marios Piskopou

Supervisor Professor

Andreas Aristeidou

A Thesis Submitted in Partial Fulfilment of the Requirements for the Degree of Bachelor of Science at the University of Cyprus

May 2025

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to all the people who supported me throughout the course of this thesis.

First and foremost, I would like to thank my supervisor, Andreas Aristeidou, for their continuous guidance, encouragement, and insightful feedback. Their expertise and support were instrumental in shaping the direction of this work and helping me overcome many technical and research challenges.

I would also like to thank the faculty members and peers who offered helpful advice and suggestions throughout this project, even if I am unable to mention each one by name. Your insights were greatly appreciated and helped guide my thinking during critical stages of the thesis.

Finally, a special thanks goes to my family and friends for their unwavering support, patience, and understanding during the demanding periods of this thesis. Their encouragement kept me motivated, especially during long training times and technical difficulties.

ABSTRACT

Motion capture (MoCap) is a widely used technique for capturing and analysing human movement in applications such as animation, gaming, and biomechanics. However, raw motion capture data often contains noise, marker occlusions, and inconsistencies—issues that are especially pronounced in expressive, high-energy movements such as dance. This thesis focuses on reconstructing and cleaning dance motion sequences from the DanceDB dataset using the MoCap-Solver, a deep learning-based motion reconstruction framework.

While the initial objective was to clean the dance data using the MoCap-Solver, the scope of the project expanded to develop a complete pipeline for processing and visualizing motion data. The final system includes: generating SMPL models from raw C3D files using MoSh++, converting SMPL sequences to BVH format, and visualizing the cleaned and reconstructed motions in Blender using three different methods—marker-based views, BVH skeleton animations, and SMPL mesh representations.

The results demonstrate that the MoCap-Solver significantly improves motion quality by reducing noise and enhancing joint stability. However, some limitations such as global body trembling and minor instability in hand motion were observed. Despite these issues, the toolbox developed in this thesis provides a functional and flexible workflow for cleaning and visualizing complex dance motions, and can serve as a foundation for future motion capture research and applications.

TABLE OF CONTENTS

Chapter 1

Introduction

1.1 Background and motivation

Motion capture (MoCap) is a widely-used technique in fields such as computer animation, game development, and biomechanics. By recording the movement of human performers using specialized tracking systems, MoCap provides detailed motion data that can be applied to digital characters or analysed for research purposes. However, raw motion capture data—particularly from optical systems using physical markers—is often noisy, incomplete, or imprecise. These issues are especially pronounced in fast, expressive movements such as dance, where occlusions, marker swaps, and sudden accelerations can lead to data artifacts. Dance data, in particular, presents unique challenges. Unlike controlled or repetitive actions, dance involves dynamic full-body motions, rapid transitions, and a high degree of expressiveness. These characteristics make it more prone to capture errors and harder to clean using traditional methods. High-quality, realistic dance animations require smooth, anatomically consistent motion data, which is often not achievable directly from raw capture sessions.

To address this, recent research has proposed neural motion reconstruction tools such as the MoCap-Solver, a deep learning-based system developed by NetEase Games and Tsinghua University. This tool is designed to reconstruct clean motion from noisy input markers by learning spatial and temporal motion patterns. It uses inverse kinematics and data-driven priors to generate high-quality skeletal motion sequences that align closely with realistic human movement.

The original motivation for this thesis was to use the MoCap-Solver to reconstruct and clean dance motion sequences from the DanceDB dataset collection of captured dance performances. However, as the project progressed, it became evident that a more complete motion processing pipeline was needed to handle the various steps involved in data cleaning, SMPL model creation, format conversion, and visualization. As a result, the scope of the thesis evolved into developing a toolbox of capabilities for handling motion capture data—from raw marker input to clean, visualized human motion.

1.2 Thesis Objective

The initial objective of this thesis was to reconstruct and clean dance motion sequences from the DanceDB dataset using the MoCap-Solver. DanceDB contains complex and expressive full-body motions that are particularly susceptible to common motion capture issues such as jitter, marker occlusion, and inconsistency in joint trajectories. By applying the MoCap-Solver to these sequences, the goal was to enhance the realism and smoothness of the captured motions, making them suitable for use in animation and other motion-driven applications.

As the work progressed, the scope of the thesis naturally expanded. Beyond cleaning motion sequences, there was a clear need to create a functional and modular pipeline for working with motion capture data in various forms. This led to the development of a broader toolbox that integrates several critical components for motion processing, transformation, and visualization.

The final objectives of the thesis can be summarized as follows:

- Apply the MoCap-Solver to noisy dance sequences from DanceDB in order to reconstruct clean skeletal motion.

- Generate SMPL models from raw C3D motion capture files using the MoSh++ fitting tool.

- Convert SMPL sequences to BVH format for compatibility with standard animation tools.

- Visualize the motion data using three methods in Blender: marker and joint visualization, BVH-based skeleton animation, and SMPL mesh animation.

- Evaluate and compare the outputs at each stage to assess motion quality, consistency, and visual realism.

In doing so, this thesis not only addresses the problem of cleaning complex dance motion but also establishes a reusable pipeline for future motion capture projects involving SMPL modelling and visualization.

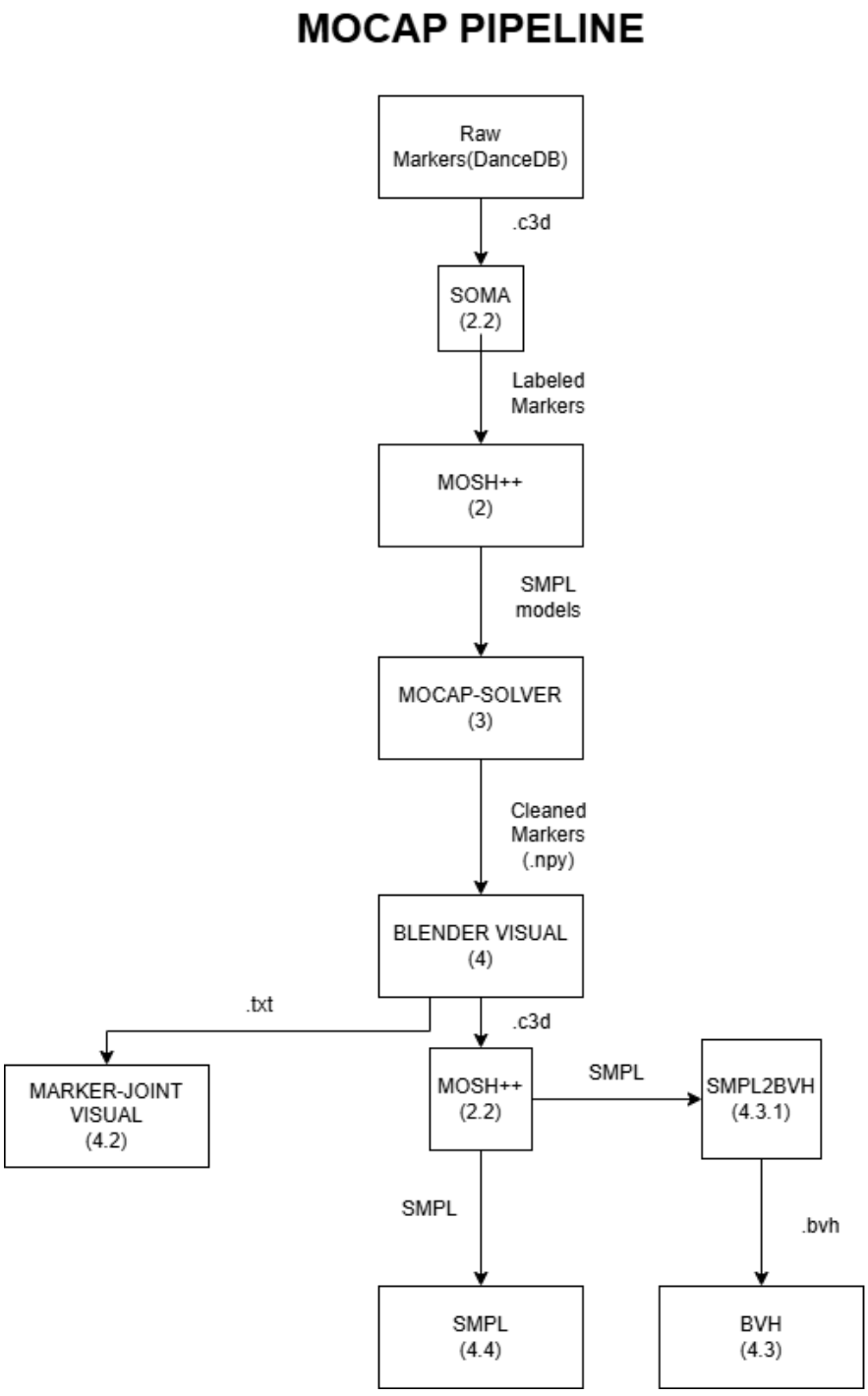


Figure1: **Overview of the full motion processing pipeline developed in this thesis.** Each module is annotated with the corresponding chapter or subchapter number. The pipeline starts from raw C3D motion capture files (DanceDB) and proceeds through labelling, SMPL model fitting, motion reconstruction with the MoCap-Solver, and finally visualization in Blender through three different formats (markers, BVH, SMPL).

1.3 Significance of the Work

This thesis contributes both a targeted application and a broader practical framework for working with motion capture data, particularly in the context of expressive dance motion. While the original focus was on using the MoCap-Solver to clean noisy motion sequences, the work evolved into building a modular and extensible pipeline that integrates motion reconstruction, SMPL model generation, format conversion, and visualization.

One of the key contributions of this thesis is the demonstration of how existing tools—such as the MoCap-Solver, MoSh++, and Blender visualization add-ons—can be combined into a functional workflow for motion clean-up and analysis. This is especially relevant for datasets like DanceDB, where the complexity of the motion introduces significant challenges that cannot be addressed by a single tool in isolation.

By validating the reconstruction of motion through both numerical evaluation and visual inspection, the thesis shows not only that the data can be cleaned, but that the cleaned results are meaningful, realistic, and suitable for reuse in animation or further analysis. The comparisons between raw and cleaned data, as well as between SMPL sequences generated using MoSh++ and those in the AMASS dataset, provide practical insight into the quality and reliability of these tools.

Furthermore, the ability to visualize the motion using multiple approaches—including marker-based views, skeletal BVH animations, and fully rendered SMPL meshes—offers flexibility for different end-user needs. This toolbox can serve as a foundation for future research or production environments that require robust motion capture pre-processing and visualization, especially for motion types that are non-repetitive and detail-sensitive, such as dance.

Chapter 2

Mosh++

2.1 Mosh++

Mosh++ is powerful tool for taking a c3d file (a file that contains that contain all the information needed to read, display, and analyze 3D motion data) and turning it into an SMPL model. This tool is useful for our work because the MoCap-Solver works only with SMPL models and the animation sessions that DanceDB is doing are stored in c3d files. This tool is an upgrade of the mosh.

2.1.1 Mosh

Mosh stands for Motion and Shape Capture from Sparse Markers and automatically extracts this detail from mocap data. Mosh estimates body shape and pose together using sparse marker data by exploiting a parametric model of the human body.

2.2 SOMA

For MoSh++ to operate correctly, it requires close integration with the SOMA (Solving Optical MoCap Automatically) tool. SOMA is a neural network-based model designed to process raw motion capture point cloud data (typically in .c3d or .pkl format), even when the

number of markers varies. It assigns marker labels at scale, without the need for calibration data or prior knowledge of the capture system. This independence from specific mocap hardware and the minimal human intervention it requires make SOMA a powerful pre-processing step for converting raw data into structured motion formats.

2.2.1 SOMA for unlabeled mocap point cloud

In many cases, motion capture point clouds either lack marker labels entirely or use naming conventions that differ from those expected by SOMA and MoSh++. Fortunately, this does not present a major obstacle. The developers of SOMA trained their model on a superset of 89 commonly used marker positions, covering a wide range of marker configurations found in commercial and research-grade mocap setups. As a result, SOMA is capable of assigning correct semantic labels to markers in unlabeled or unfamiliar layouts, automatically matching them to their anatomically correct positions. Once the point cloud is labeled, it can also be used to fine-tune or train a new SOMA model, which can then label future motion capture sessions recorded with the same marker layout — requiring only a small amount of additional time and effort. Importantly, the labeled point cloud produced by SOMA can be used directly as input to MoSh++, allowing the marker data to be fit to an SMPL model and exported in a consistent mesh format for further use.

2.3 SMPL

An SMPL, a Skinned Multi-Person Linear model, is a realistic 3D model that is based on blend shapes and skinning of the human body. It’s created from thousands of 3D body scans. Since each marker can represent a location on the mesh surface of the body, an SMPL can help obtain information such as pose and shape from Motion Capture markers. The downside of this, is that it’s prone to noise and error. Since the body shapes of the human are usually different, or the fact that MoCap suits have different placement for markers, there is an unreliable factor when trying to map the two together.

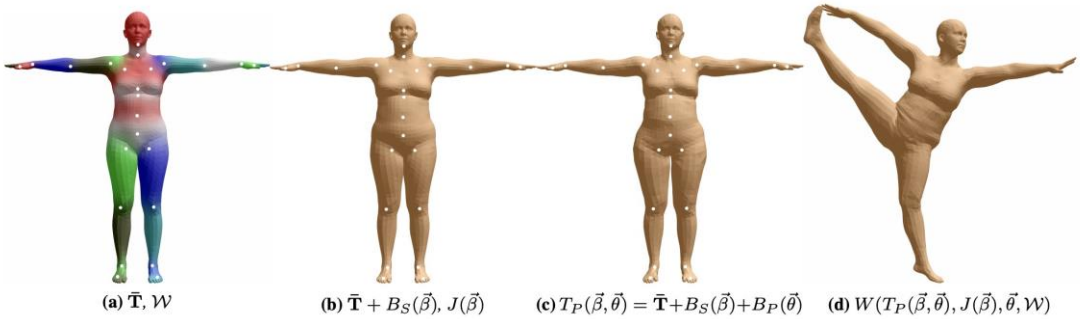


Figure 2: SMPL model. (a) Template mesh with blend weights indicated by color and joints shown in white. (b) With identity-driven blend shape contribution only; vertex and joint locations are linear in shape vector . (c) With the addition of pose blend shapes in preparation for the split pose; note the expansion of the hips. (d) Deformed vertices reposed by dual quaternion skinning for the split pose.

2.4 C3D TO SMPL

As mentioned earlier, the DanceDB dataset stores its motion capture sessions in the C3D format, and one of the goals of this thesis is to convert these sessions into SMPL models using MoSh++. However, this is not the only reason for using MoSh++ in this project. We also use it to convert the outputs of the MoCap-Solver back into SMPL format**,** allowing us to visually compare the cleaned results side by side with the original SMPL models—before the motion reconstruction process was applied.

2.4.1 DanceDB C3D

The marker labels used in the DanceDB dataset differ significantly from those expected by SOMA and MoSh++. For example, DanceDB uses generic labels such as Marker1, Marker2, etc., which are not standard anatomical labels and do not indicate the specific body part to which each marker belongs. As a result, these labels are not immediately compatible with MoSh++ or SOMA.

To address this, we used the pretrained SOMA model, which is based on a superset of 89 common marker positions, to automatically relabel the markers in DanceDB files. This process allows us to identify and assign appropriate anatomical labels to each marker. Once relabelled, the new C3D files can either be:

- used directly with MoSh++ to generate SMPL models, or

- used to train a custom SOMA model that matches the DanceDB marker layout, enabling fast, automatic labelling of all future sessions with minimal effort.

This pre-processing step is essential to ensure compatibility with the SMPL fitting pipeline and to maintain consistency across sessions.

2.4.2 MoCap-Solver Results C3D

In this section, we describe how the output of the MoCap-Solver was converted into SMPL models. The solver produces predicted marker positions, which were first saved into a text file. A simple script was then used to convert these coordinates into C3D format, enabling further processing.

Fortunately, the marker labels used by the MoCap-Solver were mostly standard and already recognized by SOMA and MoSh++, requiring little to no additional relabelling. With the labelled C3D file prepared, we proceeded to run it through MoSh++ to generate the corresponding SMPL sequence.

However, one limitation encountered was that MoSh++ currently runs only on the CPU, which made the fitting process relatively slow. As a result, we did not process the full-length dance sequences but selected shorter segments to evaluate the results. Despite the limited scope, the generated SMPL models were visually accurate and well-formed, although a few minor flaws were observed, which will be discussed in the following chapters.

Chapter 3

MoCap-Solver

3.1 MoCap-Solver

MoCap-Solver is a deep learning-based tool developed through a collaboration between NetEase Games and Tsinghua University. Its primary function is to process raw motion capture data—often noisy or incomplete due to marker occlusion or tracking inaccuracies—and convert it into high-quality, clean skeletal animations. By leveraging neural networks and principles such as inverse kinematics and temporal smoothing, the solver reconstructs plausible human motion that aligns closely with realistic movement patterns. One of the core motivations behind MoCap-Solver is to provide an automated, data-driven solution to a task that traditionally requires significant manual effort and domain expertise. The algorithm operates by mapping noisy marker positions to clean skeletal joint trajectories, producing animations that are not only visually coherent but also physically plausible. MoCap-Solver supports different skeletal models and can be adapted to datasets with varying marker configurations. This works in our favour because we have 38 markers, in contrast with the number of the markers they used for their paper which was 41. The performance of the solver is evaluated both on synthetic and real-world motion capture sequences to demonstrate its generalization capabilities.

3.2 Data Generation

In order to train and evaluate the MoCap-Solver effectively, it is essential to prepare the input data in a consistent and well-structured format. The MoCap-Solver can operate on two main types of data: real-time data and synthetic data. Real-time data refers to motion capture sequences acquired directly from optical motion capture systems during live recording sessions. In contrast, synthetic data is generated by simulating marker positions based on clean motion sequences from existing datasets. For the purposes of this thesis, the focus is placed on the use of synthetic data. This approach offers a controlled training environment, allowing the model to learn the mapping between noisy and clean motion in a more structured way, without the unpredictable characteristics often present in real-world recordings.

3.2.1 Real-time Data

To train the MoCap-Solver with real-time data, each motion capture sequence must be structured into the following three components:

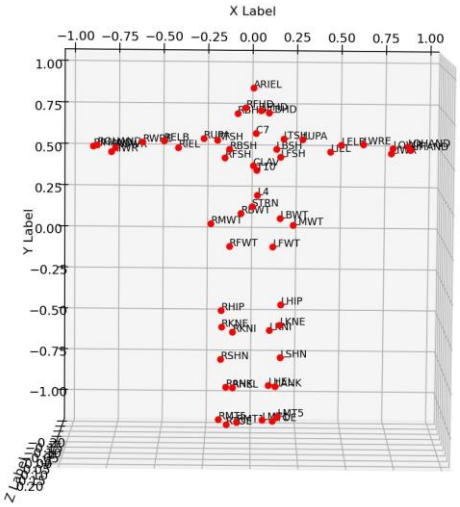
- **Raw Data**: The unprocessed marker animations captured directly by the optical motion capture system.
- **Clean Data**: The corresponding ground-truth skinned mesh animation, which includes clean markers and consistent skeleton animation. All skeletons across sequences must be homogeneous, meaning they should have the same number of joints and identical hierarchical structures. Clean markers must be properly skinned to the skeletons, and the skinning weights should remain consistent across all sequences.
- **Bind Pose**: A reference pose that defines the initial (rest) position of the skeleton and its associated markers before any animation is applied.

3.2.2 Synthetic Data

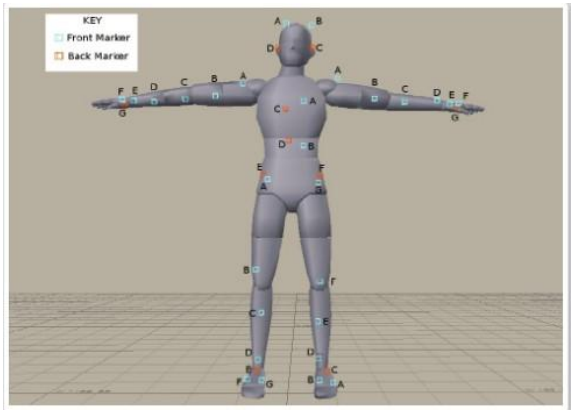
The developers of the MoCap-Solver provided a script that generates synthetic datasets from motion sequences in the AMASS database. This script prepares the data required for training, evaluation, and testing. The AMASS datasets are provided in SMPL format and contain several key parameters:

- **trans**: the global translation of the body,
- **gender**: the body model used (male, female, or neutral),
- **mocap framerate**: the frame rate of the recording,
- **betas**: body shape parameters,
- **dmpls**: parameters controlling soft tissue motion,
- **poses**: the root orientation and joint rotations per frame.

The script processes this SMPL data and produces a synthetic dataset containing 56 markers and 24 joints, structured into the same three components as the real-time data: raw data, clean data, and bind pose. Additionally, the script introduces artificial noise into the marker data using a fixed random seed. This allows the solver to be trained on corrupted input and evaluated on its ability to recover the original, clean motion. This synthetic noise serves as a controlled benchmark for assessing the solver's denoising and reconstruction capabilities.



(a) 56 Markers of the MoCap-Solver in T-Pose represented in a 3-D environment



(b) The 38 Markers captured in DanceDB

Figure 3: Similarities between the 56 markers of the MoCap-Solver and the 38 Markers from the
DanceDB Motion Captures

3.3 Training

MoCap-Solver employs neural networks to train models capable of cleaning and reconstructing motion capture data. The first step in the training process is to train the MoCap Encoders**,** which are neural modules designed to learn three key intrinsic components of motion: the template skeleton, the marker configuration, and the motion itself. The objective is to map the input raw markers to corresponding latent codes representing these intrinsic components. These latent codes are then decoded to compute clean marker positions using a linear blend skinning (LBS) function. One of the core strengths of this approach is its ability to capture and predict the spatio-temporal information embedded in the motion latent code, enabling the reconstruction of clean, plausible motion trajectories from noisy input.

3.3.1 Training Time

According to the original MoCap-Solver paper, the training process for both the encoders and the solver took approximately 50 hours. However, in this thesis, the training was conducted using the publicly available implementation provided on GitHub, which includes a higher number of full dataset passes (epochs). As a result, the overall training time increased significantly, taking approximately two full weeks to complete. It is suspected that memory limitations on the GPU led to overflow into system RAM, which severely impacted performance and contributed to the extended training duration.

3.3.2 Training Dataset

For training, the CMU dataset was used, accessed through the AMASS database. This dataset is the same one used by the original MoCap-Solver researchers in their experiments. The CMU dataset offers a rich and diverse set of human motion sequences, making it well-suited for training models that need to generalize across a wide range of movements. Its variety and

detail help the solver learn to reconstruct realistic and anatomically coherent motion across different body types and actions.

3.4 Evaluation

In this section, we evaluate the performance of the model trained using the CMU dataset. The goal is to determine how closely the results of our trained model align with those reported by the original developers of MoCap-Solver. In addition to evaluating the model on the CMU dataset, we also test it on a portion of the DanceDB dataset to observe how the model performs on previously unseen motion data. Since the solver was not trained on DanceDB and this dataset contains various noise and errors, we expect the error values to be higher.

The evaluation of MoCap-Solver is based on three key metrics:

- Marker Position Error (MPE)
- Joint Orientation Error (JOE)
- Joint Position Error (JPE)

These metrics allow us to assess both spatial accuracy and orientation consistency across the reconstructed motion.

Testing Dataset	MPE	JOE	JPE
Paper Results for CMU	14.57mm	4.75mm	13.15mm
CMU using the GitHub version of the Solver	17.04mm	6.34mm	15.04mm
DanceDB	26.98mm	12.01mm	24.03mm

Table 1 Evaluation results(MPE: Marker Position Error, JOE: Joint Orientation Error, JPE: Joint Position Error)

3.4.1 CMU Evaluation

In this section, we evaluate the performance of our trained model using the CMU dataset as both training and testing data. As mentioned earlier, this is the same dataset used by the original authors of MoCap-Solver in their experiments. As shown in Table 1, our results differ slightly from those reported in the original paper. This difference was expected, as the authors used a normalized bone length approach in their evaluation, which alters the joint positions. However, such normalization is not feasible in real-time applications where the ground-truth skeleton structure is unknown. This explains the small discrepancy in error values between our results and those from the original paper.

3.4.2 DanceDB Evaluation

Following the CMU evaluation, we tested our trained model on a subset of the DanceDB dataset to assess its performance on data outside the training distribution. As anticipated, the model exhibited higher error values compared to the CMU evaluation, as shown in Table 1. This can be attributed to two main factors. First, the solver was trained exclusively on the CMU dataset, and many of the dance movements in DanceDB were unfamiliar to the model. Second, the DanceDB dataset contains a number of noisy or corrupted sequences, which the solver attempted to correct. While the solver was able to fix many of these errors, the evaluation metrics naturally reflected the increased difficulty and variability in the test data.

Chapter 4

Visualization

4.1 Introduction

Visualization plays a crucial role in evaluating the performance of the MoCap-Solver, as it provides an intuitive and immediate way to assess the quality of the cleaned motion capture data. In this chapter, various methods are presented for visualizing the solver's outputs, focusing on the clarity, accuracy, and realism of the resulting motions. Three primary visualization approaches were employed. First, a marker- and joint-based representation was created using Blender, allowing for a direct comparison between raw, noisy inputs and the solver's cleaned outputs. Second, the solver's results were exported to the widely-used BVH (Bio Vision Hierarchy) format and subsequently imported into Blender for animation playback and analysis. Finally, the SMPL (Skinned Multi-Person Linear) model was utilized to render the motions on a realistic human mesh, providing a more complete and visually coherent evaluation. Each of these visualization methods offers distinct advantages and insights, which are detailed in the following sections.

4.1.1 Blender

Blender is a 3D graphics software that allows users to create high-quality 3D content for a variety of purposes. Blender has a comprehensive set of tools for creating and editing 3D models, including primitive shapes, deformers, and sculpting tools. It also has a robust set of

animation tools that allow users to create keyframe-based animations, rig characters, and create non-linear animations. Blender has advanced rendering capabilities that support ray tracing, global illumination, and physically-based rendering, and it includes a built-in game engine for creating interactive 3D applications and games. Blender has a set of sculpting tools that allow users to create detailed 3D models, and it has a range of compositing and post-processing effects that allow users to enhance the look and feel of their 3D graphics and animations. Blender is used by professionals in various industries and is also popular among hobbyists and 3D enthusiasts. Blender has the ability to use custom scripts in Python to generate objects, add animations to them and many more.

4.2 Marker and Joint Visualization

The first visualization technique I used was the Marker and Joint Visualization script created by Dr. Andreas Aristeidou. This script is creating a humanoid skeleton by taking the positions of the markers and the joints of the human body and connect the joints with bones and the markers with the joints with thin lines. However, this script was originally designed for an older version of Blender, which led to several compatibility issues when attempting to use it in a newer version. These issues included deprecated APIs, changes in Blender's data structure, and differences in the way animations and object hierarchies are handled. As a result, several modifications had to be made to the script to ensure it functioned correctly with the updated Blender environment. Once adapted, the script successfully displayed both raw and cleaned marker and joint positions, allowing for direct visual comparison between noisy input data and the solver's output.

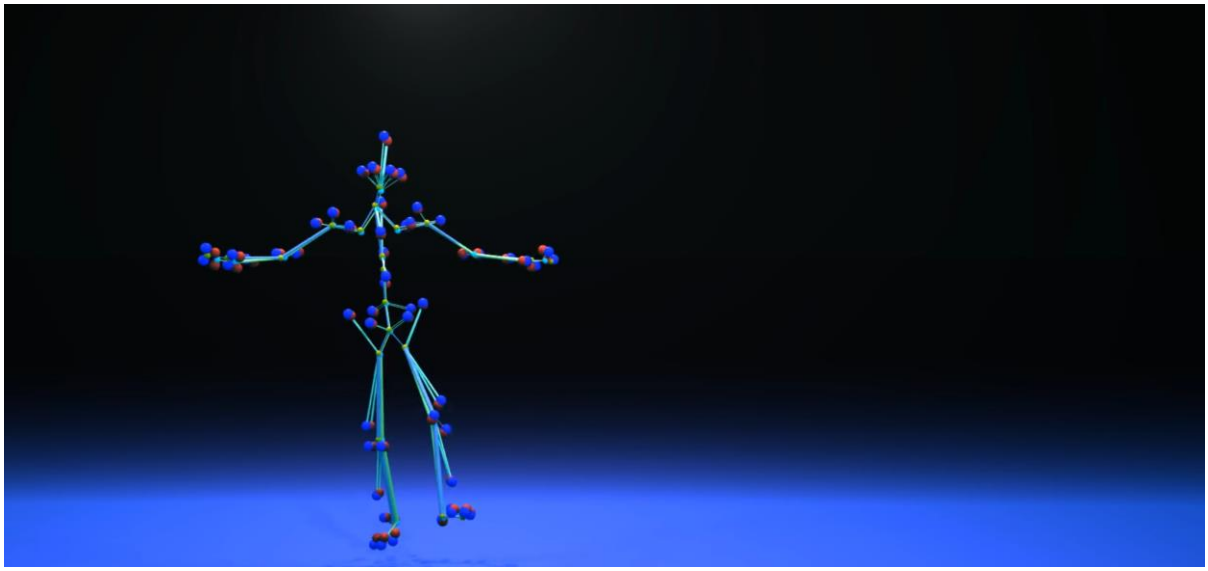


Figure 4: Marker and Joint Visualisation with two actors. Red are the raw markers and blue are the predicted markers

4.2.1 Data Input

Firstly, we needed to bring the data we got from the solver to the appropriate form so the blender script can display the results we wanted. So, we saved the predicted and raw markers and joints from the solver and we saved them in NumPy files. But the blender script needs text files in a specific format to work. Thus, we created a script that takes the NumPy files and creates a text file for each one the markers and joints for all the time of the animation.

4.2.2 Actors

We created two versions of the blender script, the one script represents one actor and the second represent two actors. First, we created the one actor scripts to check if the script works fine and shows markers and the joints in the right order. We encounter some problems with the right order of the joints which created a body that did not represented a human, so we had to make the order right and connect the joints again so we could see the human figure. The second version of the script we created was the two-actor representation. The two actors were the ground truth of the markers and joints and the second one was the predicted one that was created by the solver. We changed the colors of the markers and joints to sperate the two actors visually and placed the one actor on top of the other to see the real time differences.

4.3 BVH

The Biovision Hierarchy (BVH) format is a widely-used file format in the motion capture and animation industry. It was originally developed by Biovision for storing motion data and has since become a standard for representing skeletal animations. A BVH file contains two main components: a hierarchical definition of a skeleton (including joint names, parent-child relationships, and initial positions) and a motion section that describes the position and rotation of each joint over time, typically across a sequence of frames. BVH files are particularly useful for visualization because they are supported by many 3D animation tools, including Blender. By making the MoCap-Solver's results to BVH, it becomes possible to easily import and animate the reconstructed skeleton in Blender. This allows for a frame-by-frame inspection of the generated motion and enables visual comparison between the solver’s

output and the original noisy input, especially in terms of joint stability and trajectory smoothness.

4.3.1 SMPL TO BVH

Before we import the results of the solver to the blender, we first need to make them into BVH. As we mentioned in the third chapter, we take the results of the MoCap-Solver and we made them into SMPL. So now the only step that is left is to make them into BVH. For that purpose, we used the character animation tools by Kosuke Fukazawa in GitHub which worked perfectly and got the expected results. This tool provides functionality for retargeting SMPL motion data to a skeleton hierarchy compatible with BVH, effectively bridging the gap between statistical human body models and standard animation pipelines. It processes the SMPL pose and shape parameters and reconstructs a skeletal animation, which is then exported into BVH format while preserving the joint hierarchy and motion fidelity. This conversion enables SMPL-based motion data to be visualized and analysed within animation tools like Blender, which natively support BVH files.

4.3.2 BVH to Blender

For visualizing the BVH animations in Blender, the Deep Motion Editing tool developed by Peizhuo Li. This tool provides a convenient Blender interface for importing BVH files and assigning them to a pre-rigged character model. It streamlines the process of loading, retargeting, and visualizing skeletal animations, making it easier to analyse motion sequences frame by frame. Using this tool, the motions generated by the MoCap-Solver—after being converted to BVH—could be effectively rendered on a human-like figure within Blender, allowing for a more intuitive assessment of motion quality and realism.

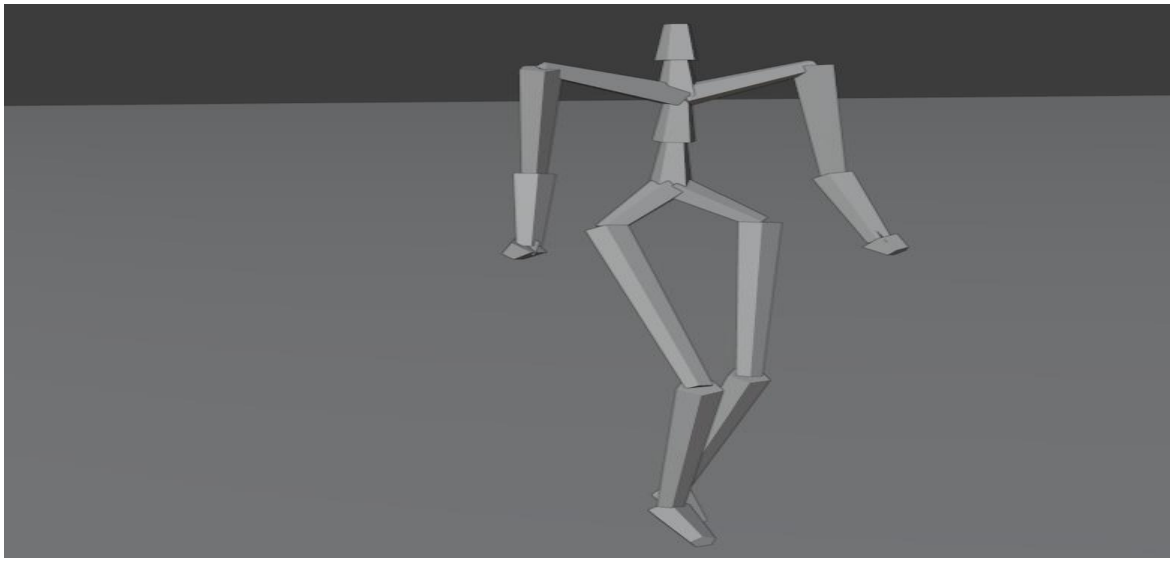


Figure 5: BVH file rendered in the Blender app.

4.4 SMPL to Blender

To visualize SMPL-based outputs directly in Blender, the SMPL-X Blender Add-on was used. Developed by Pavlakos et al. as part of the official SMPL-X project, this tool provides functionality for importing and animating SMPL, SMPL+H, and SMPL-X mesh sequences using corresponding pose, shape, and translation parameters. The add-on loads model files and parameter sequences, generating realistic animated human meshes directly in Blender's 3D environment. This enabled the high-quality visualization of the MoCap-Solver's results in human form, offering a more natural and complete understanding of motion quality compared to skeletal or marker-only representations.

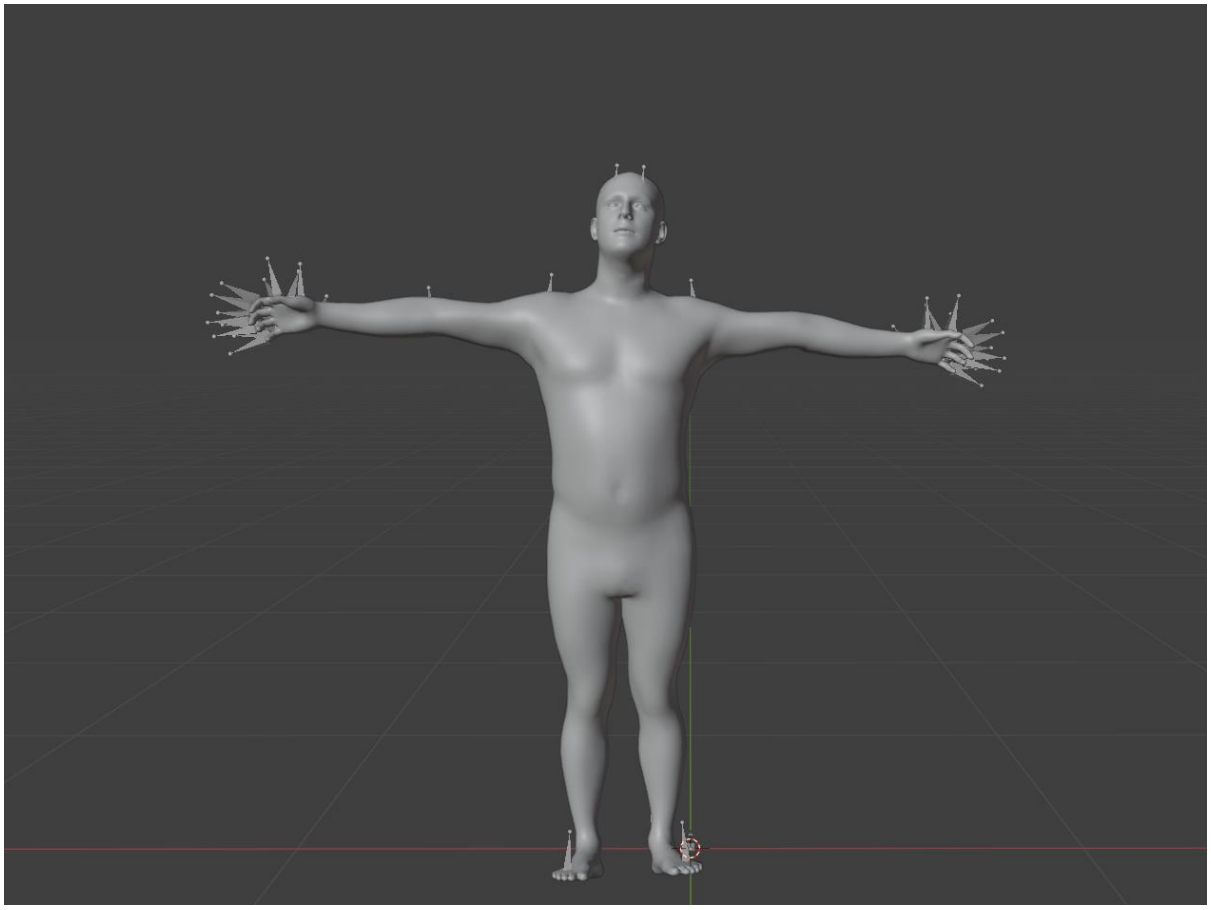


Figure 6: SMPL Visualization in blender of a male model

Chapter 5

Results

5.1 Overview

While previous chapters presented the numerical results of the MoCap-Solver, such as reconstruction error and evaluation metrics, this chapter focuses on the visual assessment of its performance. Visual analysis offers an intuitive understanding of how effectively the solver improves motion quality by reducing noise, correcting marker inconsistencies, and producing smoother skeletal trajectories. Two primary visualization methods are used to compare the results of the solver with the original raw data: marker and joint-based visualization and SMPL-based mesh visualization. Each method provides unique insights into motion quality and realism. Marker-based views allow direct comparison of positional data and motion continuity, while SMPL-based visualizations show how the motion would appear on a full human body mesh, enabling a more holistic evaluation. Additionally, this chapter compares the SMPL sequences generated using MoSh++ with the SMPL models provided in the AMASS dataset. This comparison highlights the differences in mesh fitting quality and consistency, offering further insight into the pre-processing steps involved in generating synthetic training data for the solver.

5.2 Visual Comparison of MoCap-Solver Output

To better understand the effectiveness of the MoCap-Solver beyond numerical metrics, this section presents a visual comparison between the solver's output and the original raw motion capture data. The comparison is being done with two methods: a marker and joint-based visualization, which directly reflects the raw and cleaned positional data, and an SMPL-based

visualization, which shows how the same motion is expressed on a full human body mesh. Both visualization methods provide complementary perspectives. The marker-based view highlights positional noise and structural consistency, while the SMPL-based rendering emphasizes the overall realism and continuity of the full-body motion. Together, they offer a comprehensive evaluation of the solver's performance in reconstructing plausible human motion.

5.2.1 Marker-Joint Visualization

The marker and joint-based visualization provide a direct representation of the raw and cleaned motion capture data. This method focuses on displaying individual markers and their associated skeletal joints in Blender, allowing for a frame-by-frame comparison of how the MoCap-Solver processes and improves noisy motion sequences. In the raw data, common issues such as jitter and abrupt position changes. These artifacts appeared both during fast and slow motions. By contrast, the cleaned data produced by the MoCap-Solver exhibits much smoother trajectories, with reduced noise and more natural movement patterns. Joints maintain consistent spacing, and limb motion appears more continuous and physically plausible. This visualization also highlights how the solver corrects structural distortions, such as misaligned limbs or unstable joint orientations. In several test sequences, the solver effectively restored the underlying skeleton structure, resulting in a visually coherent animation that preserved the intent of the original motion. By comparing the raw and cleaned sequences side by side, it becomes evident that the MoCap-Solver solves the abrupt position changes of some of the markers and joints. However, we came across an unexpected problem that made the whole body of the cleaned actor to tremble a lot which is making the dance less pleasant to watch and not completing the purpose of using the solver to reconstruct the motion capture data.

5.2.2 SMPL-Based Visualization

The SMPL-based visualization offers a high-level, intuitive view of the motion by rendering the movement on full 3D human meshes. This method allows for a direct side-by-side comparison between the original (noisy) and the cleaned (solver-processed) motion sequences, showing how a realistic human body would perform the actions before and after reconstruction. By visualizing both motions on structured SMPL body models, it becomes easier to assess the impact of the solver on the overall motion quality. This representation helps bridge the gap between abstract joint trajectories and real-world motion, making the

evaluation more relatable and visually grounded. We can see that the MoCap-Solver made improvements in the posture alignment and made smoothest transitions. But, still we can witness the same trembling effect in the cleaned motion of the actor that makes the whole structure of the body.

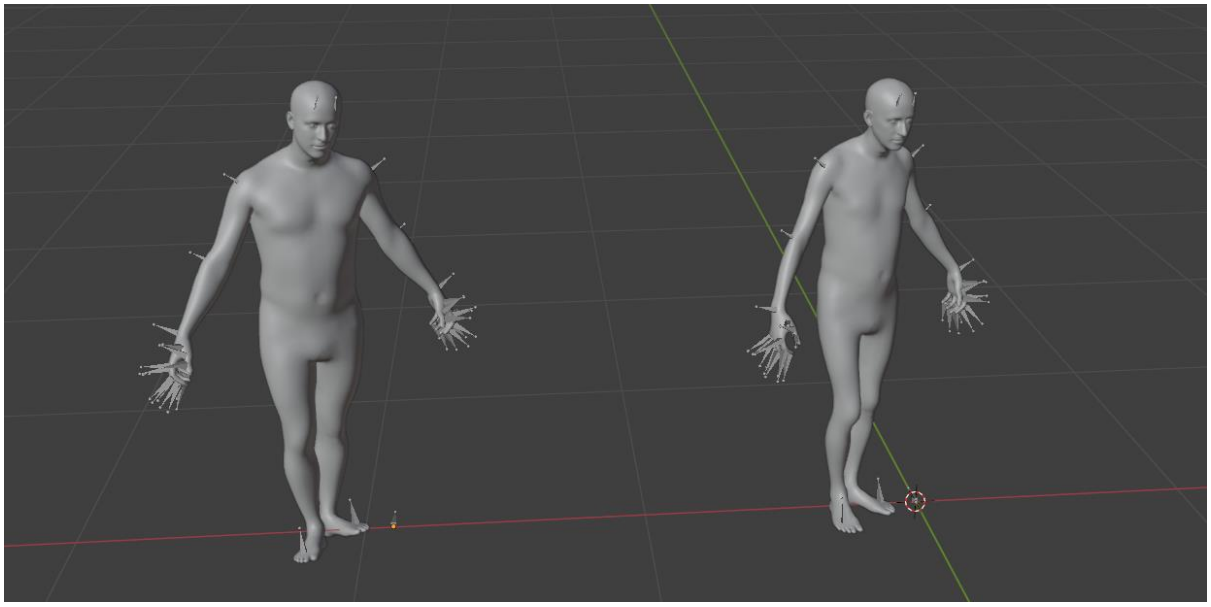


Figure 7: Comparison between AMASS DanceDB model and after the MoCap-Solver Prediction. Left: AMASS, Right: MoCap-Solver prediction

5.3 Comparison Between MoSh++ and AMASS SMPL Models

To ensure the correctness of the SMPL models generated during this project, a comparison was conducted between the outputs of MoSh++ and the corresponding SMPL models provided in the AMASS dataset. The goal of this comparison was to verify whether the models produced by running MoSh++ on raw marker data are consistent with the high-quality reference models already available in AMASS. This validation step is important for two main reasons. First, it confirms that the process of generating SMPL sequences using MoSh++ is functioning as expected. Second, it builds confidence in using MoSh++ to create new SMPL models from future motion capture sessions, particularly in cases where AMASS-style data is not already available. Visually, the SMPL meshes generated by MoSh++ closely matched those from AMASS in terms of body shape and joint articulation, but when the animations start the Mosh++ version begins to show some problems. The whole body is moving as expected but the hands are behaving inconsistently with erratic or unnatural movement. This may be due to the fact that we did not mange to create the perfect marker layout for our motion capture session to use as reference for the Mosh++ model.

5.4 Discussion

The visual comparison methods presented in this chapter provided valuable insights into the performance and limitations of the MoCap-Solver when applied specifically to dance motion capture data. Since dance movements are typically expressive, complex, and dynamic, they present a unique challenge for motion reconstruction algorithms. Both the marker-based and SMPL-based visualizations revealed clear improvements in motion smoothness, joint stability, and anatomical coherence in the solver's output compared to the original raw dance sequences. However, in many cases, the solver introduced a trembling effect that affected the entire body, particularly during sequences with fast or continuous movement. This unintended shaking diminished the overall visual quality of the motion and undermined the expressive clarity that is essential in dance, ultimately compromising the intended outcome of producing smooth and realistic reconstructed performances.

Chapter 6

Conclusion

6.1 Summary of Objectives

This thesis set out to clean and reconstruct complex dance motion data from the DanceDB dataset using the MoCap-Solver. The initial aim was to improve motion quality by removing noise, inconsistencies, and structural errors commonly found in raw motion capture data. As the work progressed, the thesis expanded into developing a practical toolbox for handling and visualizing SMPL-based motion data, offering a broader contribution to motion capture processing workflows.

6.2 Achievements

Over the course of this thesis, several key components were successfully developed and integrated into a coherent motion processing pipeline. The MoCap-Solver was applied to noisy dance sequences from the DanceDB dataset, resulting in visibly cleaner and more stable motion. A reliable method for generating SMPL models from raw C3D files was implemented using MoSh++, enabling the transformation of marker-based data into skinned human body models. Additionally, SMPL sequences were successfully converted into the BVH format, allowing for compatibility with standard animation tools. The project also introduced three distinct visualization methods within Blender: marker and joint visualization, BVH-based skeleton animation, and full-body SMPL mesh animation. These tools made it possible to compare raw and cleaned motion side-by-side, facilitating a more intuitive evaluation of motion quality. Finally, by comparing the generated SMPL sequences

with those provided in the AMASS dataset, the reliability of the MoSh++ fitting process was validated, confirming its suitability for future motion capture applications.

6.3 Observations and Insights

The visual evaluation of the reconstructed dance motions offered valuable insights into the performance and limitations of the overall pipeline. The MoCap-Solver proved highly effective in reducing local marker noise and improving joint consistency, particularly in the torso, legs, and arms. The cleaned motions appeared more anatomically correct and visually stable compared to the raw input. However, a recurring issue observed to all the sequences was a subtle trembling or jitter affecting the entire body. This instability disrupted the natural flow of the dance and indicated a need for improved global motion consistency in the solver. Additionally, while the SMPL models generated using MoSh++ were largely consistent with those in the AMASS dataset, the hand movements occasionally appeared erratic or unnatural. These irregularities were minor but noticeable, highlighting the importance of detailed fitting in motion reconstruction. Overall, the combination of marker-based and SMPL-based visualizations was essential in uncovering these issues—insights that would have been difficult to obtain through numerical metrics alone.

6.4 Limitations

While the developed pipeline achieved its core goals, several limitations were encountered throughout the project. A key issue was the subtle trembling effect in some of the cleaned dance sequences produced by the MoCap-Solver, which affected the perceived naturalness of the motion. This indicated a lack of global temporal stability in certain high-energy or expressive motions, such as those common in dance. Additionally, the hand and finger articulation in the SMPL models generated via MoSh++ was occasionally unstable or unnatural, which could impact applications that require detailed upper-body or gestural accuracy. Another practical limitation was the significant time required to train the MoCap-Solver. Due to the model's complexity and the volume of data, training took approximately two weeks to complete, which presents a barrier to iteration and experimentation, especially on systems with limited hardware resources. Finally, some of the scripts used for visualization—particularly those developed for older versions of Blender—had to be manually updated to function correctly in modern software environments, adding to the technical workload.

6.5 Future Work

Several directions can be explored to further improve and expand the work presented in this thesis. First, addressing the global motion stability of the MoCap-Solver output should be a priority, particularly for complex and expressive motion types like dance. Enhancing the solver's temporal smoothing or integrating additional constraints may help reduce the trembling effect observed in the cleaned sequences. Improving hand and finger articulation in SMPL model generation is another important step, especially for applications that rely on accurate upper-body motion or gesture recognition. In addition, the training time of the MoCap-Solver, which took approximately two weeks in this project, poses a significant barrier to scalability. Future work should investigate methods to reduce training duration, such as model optimization, data sampling strategies, or leveraging more powerful hardware or cloud-based GPU solutions. Lastly, the current pipeline, while functional, could benefit from further automation and user-friendly integration to support real-time or batch processing of new motion capture data with minimal manual effort.

6.6 Conclusion

In summary, this thesis began with the goal of cleaning dance motion data using the MoCap-Solver and evolved into a comprehensive exploration of motion reconstruction, SMPL model generation, and visualization. Through the development of a flexible and modular toolbox, the project not only improved the quality of dance motion data but also provided a foundation for future work in motion capture processing. While challenges such as trembling effects, hand instability, and long training times remain, the insights and tools produced through this work offer a practical starting point for researchers and developers aiming to enhance the realism and usability of complex motion sequences.

Bibliography

[1] Kang Chen, Yupan Wang, Song-HaiZhang, Sen-ZheXu, Weidong Zhang, and Shi-MinHu. 2021. MoCap-solver: a neural solver for optical motion capture data. ACM Trans. Graph. 40, 4, Article 84 (August 2021)

[2] Loper, M., Mahmood, N. and Black, M.J., 2014. MoSh: Motion and shape capture from sparse markers. ACM Transactions on Graphics (ToG), 33(6), pp.1-13.

[3] Mahmood, N., Ghorbani, N., Troje, N.F., Pons-Moll, G. and Black, M.J., 2019. AMASS: Archive of motion capture as surface shapes. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 5442-5451).

[4] Ghorbani, N. (2021). *MoSh++: Motion and Shape Capture* [Computer software]. GitHub. https://github.com/nghorbani/mosh

[5] Andreas Aristidou, Daniel Cohen-Or, Jessica K. Hodgins, and Ariel Shamir. 2018. Selfsimilarity analysis for motion capture cleaning. CGF 37, 2 (2018), 297–309.

[6] Andreas Aristeidou, Yiorgos Chrysanthou and Joan Lasenby Extending FABRIK with model constraints. Comp. Anim. Virtual Worlds 2016

[7] Blender Foundation. (2024). *Blender (Version 4.2)* [Computer software]. https://www.blender.org

[8] Pavlakos, G., Choutas, V., Ghorbani, N., Bolkart, T., Osman, A. A. A., Tzionas, D., & Black, M. J. (2019). *SMPL-X Blender Add-on* [Software]. GitHub. https://smpl-x.is.tue.mpg.de

[9] Li, P. (2021). *deep-motion-editing* [Software]. GitHub. https://github.com/PeizhuoLi/deep-motion-editing

[10] Pavlakos, G., Choutas, V., Ghorbani, N., Bolkart, T., Osman, A. A. A., Tzionas, D., & Black, M. J. (2019). Expressive body capture: 3D hands, face, and body from a single image. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 10975–10985. https://doi.org/10.1109/CVPR.2019.01123

[11] Fukazawa, K. (2021). *CharacterAnimationTools* [Software]. GitHub. https://github.com/KosukeFukazawa/CharacterAnimationTools