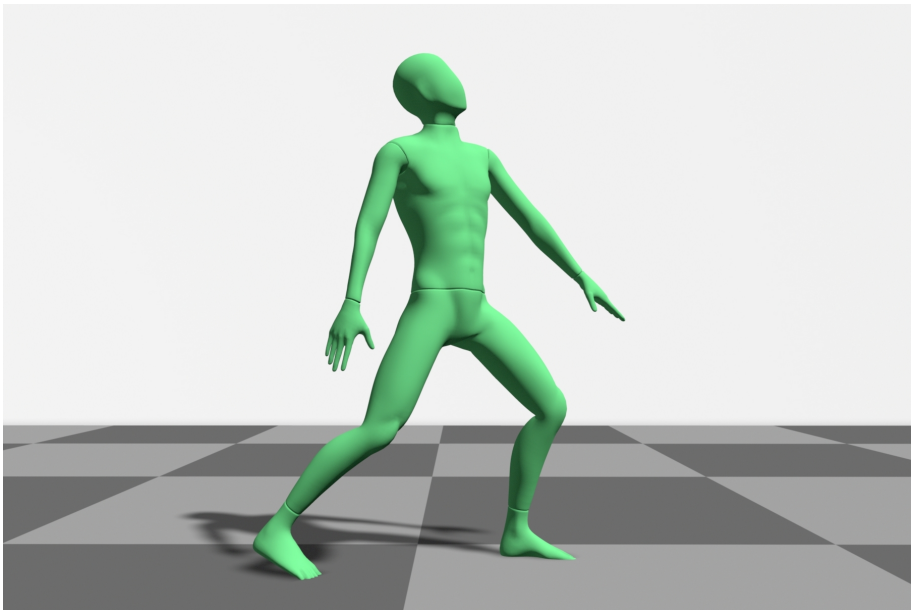
UNIVERSITY OF COPENHAGEN
DEPARTMENT OF COMPUTER SCIENCE



Ph.D. Thesis

Paul Schreiner

ADAM: Auto-Animation of Digital Humans Using AI Methods

Department of Computer Science

Advisors: Kenny Erleben, Sune Darkner, Matias Søndergaard

This thesis has been submitted to the Ph.D. School of The Faculty of Science, University of Copenhagen on December 6, 2023.

Voor Greet, zo jammer

Abstract

IMU based motion capture data notoriously lacks global positional information, due to inherent limitations in the sensing hardware. On top of that, it often suffers from physically implausible data artefacts, such as interpenetration of body parts, penetration of the floor and foot skating.

The research in this thesis presents methods for reconstructing global position trajectories from local pose information and improving IMU motion data quality in an automated fashion.

The first phase of this work introduces a novel method for reconstructing global positions using neural networks. A U-Net convolutional neural network was trained to process pose information for real-time position estimation. The work leveraged a diverse dataset, encompassing a wide range of activities and subjects, to train this network. Superior error properties were observed with the U-Net compared to a more standard convolutional neural network architecture, leading to more accurate global position predictions.

Building upon this foundation, subsequent research refined the global trajectory reconstruction process and added trajectory reconstruction in the vertical direction. A a lean U-Net model was developed, designed to integrate local pose information with acceleration signals from the IMU sensors. The model estimated short, character-centered trajectories over a sequence of frames, employing a weighted average approach to minimize estimation bias and noise. Tested on a novel dataset comprising actors not included in the training set, this enhanced method showed good accuracy in reconstructing ground truth trajectories. Acceleration signals were

shown to play a critical role in maintaining trajectory reconstruction quality when pose data quality declined.

The final aspect of this thesis tackled inherent limitations in IMU-based motion capture, such as self-penetrating body parts, foot skating, and floating. These issues significantly hamper the realism achievable with cost-effective IMU systems. To overcome this, reinforcement learning was utilised to train an AI agent that could mimic error-prone sample motions within a simulated environment. This approach could prevent these common distortions while preserving the unique characteristics of the sample motions. The agent was trained on a blend of faulty IMU data and high-quality optical motion capture data. By examining different configurations of observation and action spaces, optimal settings were identified for use on unseen data. The efficacy of this approach was validated employing a set of quantitative metrics. These tests, conducted on a benchmark dataset of IMU-based motion data from actors outside the training set, demonstrated the method's capability to enhance the realism and usability of IMU-based motion capture systems, narrowing the gap with marker-based alternatives.

Resumé

IMU-baseret bevægelsesfangstdata mangler notorisk global positionsinformation på grund af iboende begrænsninger i sensorens hardware. Derudover lider data ofte af fysisk umulige dataartefakter, såsom gennemtrængning af kropsdele, penetration af gulvet og fodskøjte.

Forskningen i denne afhandling præsenterer metoder til rekonstruktion af globale positionstrajektorier fra lokal kropspositionsinformation og forbedring af IMU-bevægelsesdata kvalitet på en automatiseret måde.

Den første fase af dette arbejde introducerer en ny metode til rekonstruktion af globale positioner ved hjælp af neurale netværk. Vi anvendte et U-Net konvolutionelt neuralt netværk til at behandle kropspositionsinformation til realtidspositionsestimering. Vores tilgang udnyttede et divers dataset, der omfattede et bredt udvalg af aktiviteter og emner, for at træne dette netværk. Vi observerede overlegne fejlegenskaber med U-Net sammenlignet med andre konvolutionelle neurale netværksarkitekturer, hvilket førte til mere nøjagtige globale positionsforudsigelser.

Med dette fundament som udgangspunkt forfiner vores efterfølgende forskning processen med rekonstruktion af den globale trajektori og tilføjede rekonstruktion af trajektori i den lodrette retning. Vi udviklede en slank U-Net-model designet til at integrere lokal kropspositionsinformation med accelerationssignaler fra IMU-sensorerne. Denne model var dygtig til at estimere korte, karaktercentrerede trajektorier over en sekvens af billeder, idet den anvendte en vægtet gennemsnitstilgang til at minimere estimationsbias og støj. Testet på et nyt datasæt bestående af skuespillere, der ikke var inkluderet i træningssættet, viste denne forbedrede metode god

nøjagtighed i rekonstruktionen af sandhedens trajektorier. Den løste effektivt problemer som drift i både horisontal og lodret bevægelse, hvilket understreger den kritiske rolle af accelerationssignaler i opretholdelse af kvaliteten af trajektorirekonstruktion, især når kropsposition datakvaliteten faldt.

Det sidste aspekt af vores forskning tog fat på iboende begrænsninger i IMU-baseret bevægelsesfangst, såsom forvrængninger som selvpenetrerende kropsdele, fodskøjte og svævning. Disse problemer hæmmer betydeligt den realisme, der kan opnås med omkostningseffektive IMU-systemer. For at overvinde dette anvendte vi forstærket læring til at træne en AI-agent, der kunne efterligne fejlbehæftede prøvebevægelser inden for et simuleret miljø. Denne tilgang gjorde det muligt for os at forhindre disse almindelige forvrængninger, samtidig med at de unikke karakteristika ved prøvebevægelserne bevares. Vi testede vores metode grundigt mod en blanding af fejlbehæftede IMU-data og højkvalitets optisk bevægelsesfangstdata. Ved at undersøge forskellige konfigurationer af observations- og handlingrum identificerede vi optimale indstillinger for vores brugssag. Effektiviteten af vores tilgang blev valideret ved hjælp af et omfattende sæt kvantitative metrikker og kvalitative vurderinger. Disse tests, udført på et benchmark-datasæt med IMU-baserede bevægelsesdata fra skuespillere uden for vores træningssæt, demonstrerede vores metodes evne til betydeligt at forbedre realisme og brugbarhed af IMU-baserede bevægelsesfangstsystemer og indsnævre kløften med marker-baserede alternativer.

Acknowledgements

In hindsight, I had no idea what I was getting myself into when we first started planning this project. Now, nearly four years later, I can conclude that it has been transformative for me as a person, not only academically and professionally, but also personally. I have learned valuable lessons about my limits, my capabilities, and my faults, and I will carry those with me wherever I go.

First of all, I want to express my gratitude to my advisors Kenny, Sune and Matias. They are the ones who set this all in motion and who were always there when I needed them, from tropical Caribbean islands to dark Danish basements during Covid. They made me feel trusted and taken serious from day one, and gave me the space I needed to complete this project.

I also want to extend my gratitute to Rokoko Electronics ApS and DIKU for facilitating this project. It has been a great journey and I am excited about what the future has in store. Even though I was a one-person project in the company, I have never felt alone with such a great crew of fantastic, curious, talented colleagues.

I want to express my gratitude to the Innovation Fund Denmark (Innovationsfonden Danmark) who partially funded this project (grant no. 9065-00085A).

Finally, I want to thank Anna for bearing with me this whole journey. I could not have done this without her love, support, and patience throughout.

List of Publications

- Paul Schreiner, Maksym Perepichka, Hayden Lewis, Sune Darkner, Paul G. Kry, Kenny Erleben, and Victor B. Zordan. Global position prediction for interactive motion capture. *Proc. ACM Comput. Graph. Interact. Tech.*, 4(3), sep 2021a. doi: 10.1145/3479985. URL `https://doi.org/10.1145/3479985`.

- Paul Schreiner, Sune Darkner, and Kenny Erleben. Root3D: Root position reconstruction in 3 dimensions for imu based motion capture. *Under review at the Symposium on Interactive 3D Graphics and Games*, 2024a.

- Paul Schreiner, Rasmus Netterstrøm, Sune Darkner, Hang Yin, and Kenny Erleben. ADAPT: Ai-driven artefact purging technique for imu based motion capture. *Under review at Computer Graphics Forum*, 2024b.

Table of Contents

Chapter 1

Introduction

This chapter provides an overview of and sets the context for the thesis. In section 1.1, the industrial context of the project is explored, along with a brief introduction to the technology underpinning inertial measurement unit based motion capture. The primary challenges addressed by this project are detailed in section 1.2. The key contributions of the project are outlined in section 1.3. Finally, section 1.4 offers insights into some of the unique challenges encountered as an industrial Ph.D. student.

1.1 Background

1.1.1 Industrial Setting of the Project

This project has been a collaboration between Rokoko Electronics ApS and the department of Computer Science at the University of Copenhagen (UCPH).

Rokoko Electronics ApS is a Copenhagen based company that develops motion capture hardware and software. It was founded in 2014 by students of the National Film School of Denmark as a traveling theater production for children. In this production, the children interacted with avatars projected on a large screen. The avatars were controlled by actors back stage wearing self made motion capture suits.

The suits were built using inertial measurement units (IMU) to record their motion in real-time. The choice for a self made suit came from the lack of affordable motion capture solutions at a time when even the few existing IMU based solutions were priced at over $10.000 per suit. Over time, the company realised that there was a broader need for affordable motion capture solutions. This eventually led to the founding of Rokoko as a motion capture company and the development and release of the first Smartsuit Pro back in 2016.

Since then a number of new hardware products have followed, with an improved full body suit, the Smartsuit Pro II, Smartgloves for hand motion capture, and a camera based face capture solution. Furthermore, a new product, the Coil Pro, is expected to be released in the beginning of 2024. The Coil Pro is a new type of motion capture product, that generates an electromagnetic field which is tracked by body mounted sensors. The advantage with a coil over camera based solutions is that the electromagnetic field does not suffer from occlusions. Rokoko also developed a software studio for recording, cleaning and editing of motion capture assets. It contains filter options to automate cleaning as well as an interface to do manual editing, all in a collaborative cloud based setting. Finally, it includes a market place for motion assets where users can find stock motion assets for easy integration into their projects.

One of the key objectives of Rokoko is to democratise motion capture; i.e. to create motion capture solutions that are accessible to anyone who has an interest in recording their motion, but does not have the resources for expensive high-end (optical) solutions. The company focuses on small production studios for games and film, and individual creatives; organisations or individuals that do not have the same resources as high end studios, but also have a lower quality requirement. The result is that a cornerstone of the company is to find the optimal balance between ease of use, quality and affordability. This is also a core principle of this project.

This project was conceived in 2019 by Rokoko's CPO Matias Søndergaard, together with professor Kenny Erleben and associate proffesor Sune Darkner from the University of Copenhagen. It has, in part, been funded by the Danish Innovation Fund (Innovationsfonden Danmark). The project facilitated the sharing of knowledge

and insights between industry and academia in Denmark by fostering collaboration in fundamental research. The areas of focus included IMU-based motion capture, physics simulation for computer graphics, and artificial intelligence. As a result, this collaboration has led to the publication of theory otherwise 'hidden' within the company at the one hand, and the materialisation of cutting-edge theory from academia into products on the other hand.

1.1.2 IMU Based Motion Capture

The animation industry has seen rapid growth over the last decades and is expected to remain a growth market in the years to come (Statista Research Department, 2023; Knowledge Sourcing, 2023). Motion capture is a corner stone of this industry, enabling creators to produces life like and highly detailed digital motion assets. This section of the animation market has traditionally been dominated by high end optical motion capture solutions, which has mostly been the result of their high quality output in combination with a lack of other viable alternatives. However, the last decade has seen the rise of a serious competitor to these technologies in the form of IMU based motion capture solutions. Companies such as Rokoko Electronics ApS and XSens have worked on popularising these types of solutions and bringing motion capture to a broader market.

Apart from a much lower price point, IMU based solutions come with a number of other advantages over optical systems. They are generally easier to set up, are not bound to external hardware constraining the recording volume, and do not suffer from marker occlusion or interference from daylight (Thewlis et al., 2013).

1.2 Problem Statement

The advantages described in the previous section come with their own unique set of limitations. Most prominently, the quality of IMU based motion capture is of a different order due to limitations in the precision of micro-electromechanical system (MEMS) IMU sensors.

Typically, an IMU measures nine degrees of freedom, incorporating three triple-axis sensors that are combined into a single package:

1. a gyroscope, measuring angular velocity in the sensor's frame of reference
2. an accelerometer, measuring accelerations working on the sensor body, including earths gravitational field
3. a magnetometer, measuring the surrounding magnetic field. Although not technically an inertial property, this helps with stabilising the other measurements as discussed further on

The two inertial measurement signals from IMU's, angular velocity and acceleration, can be used to infer the sensor's orientation and displacement in the world. However, as these signals are time derivatives of the quantities of interest, we then rely on integration of the measurements. This is where the problems with IMU sensor's originate. As any physical sensor, an IMU sensor contains inherent measurement biases and noise. Integration of a biased, noisy signal will lead to errors accumulating over time. These may not be noticeable over short periods of time, but as time progresses these become noticeable in the form of drift.

In the case of orientation estimates, these effects can be mitigated using measurements of earth's magnetic and gravitational fields from the other sensors. These measurements are then usually fused with the integrated angular velocities using sensor fusion algorithms such as a Kalman filter (Sabatelli et al., 2012). In the case of human motion capture the performance of these sensors can be further optimised by applying knowledge of the kinematic structure to which an ensemble of sensors is mounted (Roetenberg et al., 2009). However, these algorithms still have a limited precision due to inherent limitations in the hardware's precision. This leads to some loss in quality, but a generally acceptable level of precision.

For positional estimates, accelerations can be integrated twice to obtain global displacements (Floor-Westerdijk et al., 2012). However, this comes with additional complications, as drift effects grow exponentially and not linearly due to the double integration. Moreover, we do not have access to stabilising measurements, which

leads to potential unbounded drift over time. Finally, due to errors in the orientation measurements, we can not perfectly align the sensor with the global frame. This leads to errors when subtracting earths gravitational field from the measured accelerations. Due to these limitations, it is not feasible to track global positions from double integration of the acceleration sensors over longer periods of time. Solutions like incorporating GPS receivers, cameras, or magnetic field generators are often proposed to mitigate this problem (Roetenberg et al., 2009; Corrales et al., 2008; Schreiner et al., 2021b). However, they come with their own limitations including the necessity to record outdoors, a limited field of view, and/or a limited range. They would thus diminish some of the inherent advantages of IMU-based systems, rendering them suboptimal.

Apart from drift, the limitations in hardware precision lead to lower quality pose estimation, compared to poses recorded from optical systems. This in turn leads to physically implausible artefacts, examples of which are self-penetration of body parts due to orientation inaccuracies, and foot skating due to faulty global position estimates.

The works presented in this thesis aim at solving the problems stated above. In chapter 2, a method is presented for global position estimation in the horizontal plane using pose information and a simple convolutional neural network architecture. In chapter 3 an extension of this method is presented to also estimate displacements in the vertical direction, while increasing robustness for IMU based motion capture data by including acceleration information. Finally, chapter 4 presents a method using reinforcement learning to automate cleaning of IMU based motion capture data containing physically implausible artefacts. For a comprehensive discussion of the previous work related to the topics introduced here, the reader is referred to the corresponding chapters in this thesis.

1.3 Contributions

This section lists the main contributions from this work.

From the work *Global Position Prediction for Interactive Motion Capture* (Schreiner et al., 2021a):

1. We present a learning-based solution for estimating the global centre of mass position for horizontal planar motion of a character, based on short sequences of local pose information.

2. We show how a U-Net convolutional neural network architecture can be used as an estimation model for global positions. This choice outperforms standard convolutional neural network architectures, and makes our solution history independent as opposed to recurrent networks.

3. We compare the effects of training on datasets with specialised motions to training on datasets with a wide variety of motions. We show that training with specialised motions slightly improves the accuracy for those motion types, while training with varied motion makes the model more robust to unseen motions.

4. We show how estimating absolute character height, instead of displacements in the vertical direction, results in a better height estimate but also improves overall model performance by reducing the task complexity.

From the work *Root3D: Root Position Reconstruction in 3 Dimensions for IMU Based Motion Capture* (Schreiner et al., 2024a):

1. We present a robust method using a feedforward U-Net neural network for estimating global positions in 3 dimensions, from local pose information and acceleration signals from IMU sensors.

2. We present and empirically validate a method for improving estimations, by using a weighted aggregate of all the information from the time series data that is output by our U-Net.

3. We show how adding acceleration signals to the input of our network improves the accuracy of predictions and how it is essential when training on lower quality data such as IMU based data.

From the work *ADAPT: AI-Driven Artefact Purging Technique for IMU Based Motion Capture* (Schreiner et al., 2024b):

1. We present a physics-based framework for cleaning physically implausible artefacts from faulty motion capture data.
2. We show that mixing faulty and high quality training data is pivotal in generalising our method to unseen IMU based animation data from unseen actors.
3. We show that the choice of observation and actuation configuration greatly impacts the agent's ability to learn and produce quality output, and present an optimal choice for generalising to unseen data.

1.4 Reflections as an Industrial Ph.D. Student

Being an industrial Ph.D. student comes with a unique set of challenges. In this section I would like to reflect on some of these challenges, the way I dealt with them and what I would do differently with the knowledge I have now.

1.4.1 'Caught Between Two Worlds'

There are two worlds for an industrial Ph.D. student and while they largely overlap, they do have some distinct differences. I remember an illustrative conversation I had with a fellow Ph.D. student at the university. This was in the very beginning of my project when I had a temporary office at the department of computer science for a few months. I had been working for Rokoko for a few years prior to starting the project, and my principle supervisor, Kenny Erleben, wanted me to disconnect from my old role. He was worried that if I would stay at Rokoko's office full time from the start, I would have a hard time transitioning to my new role as an industrial researcher. So he had found me a small office at the top floor of the building, which housed a number of "regular" Ph.D students. On one of the first days, I met my next door neighbour out in the hallway and we started talking. He asked me who I was and what I was working on. So I, in all my virgin naivety, proudly explained to

him how I was set to change the world of character animation over the course of the next few years, revolutionise the state of the art and most importantly democratise motion capture, so that everybody could enjoy high end motion capture for no money at all. However naive this all was, the end user's experience was central in my point of view: they were the entire justification of my work.

I, of course, then went on to ask him about his work, and he started telling me an extensive story about higher order polynomials and abstract mathematical concepts I had never heard of before. I have to admit I had a hard time following what he was talking about, but I thought I could lead the conversation towards a more easily understandable direction and I asked him: "So what can this be used for in practise?" He stared at me for a few seconds, apparently for the first time considering that question, and then replied: "I am not sure... I haven't thought of that before..."

This conversation was pivotal for me in understanding that there is a difference between the academic and industrial realities. There are different mindsets involved, different motives and different incentives. Neither of these realities is better or worse than the other, they are just different.

Understanding this did not come immediately. I remember feeling a form of resentment at first. How could someone do something without having a clear idea of what its purpose could be, for themselves or for others? But that conversation made me think. It made me wonder about what it is that scientists do and what drives them. It made me think about the gathering of knowledge for the sake of understanding new things; of taking a tiny step ahead in the huge scheme of things, without maybe even knowing where exactly it will lead. It is in that sense that science as a whole is at the service of humanity, however small or seemingly insignificant something might look. Scientific value is not necessarily measurable by monetary means or user satisfaction.

And that is quite the contrast with the industrial reality, where everything is focused on direct and measurable effect. If there is no direct outlook on how a project can advance the state of the company - be that in terms of revenue, or

customer satisfaction, or potential new verticals and business models - then there is no place for it, especially in smaller companies. There are always exceptions, big tech giants such as Google and Nvidia have large departments of researchers who are quite free to explore what they deem interesting, but the leading idea is still that this will eventually benefit the company or produce a unicorn.

1.4.2 Balancing Industrial and Academic Expectations

Now, as an industrial Ph.D. student working for a small company, I have to navigate both the academic as well as the industrial reality, and I have found that quite challenging at times. In my experience, the company expects monetisable results, that are, as a figure of speech, packaged and ready to ship. On the other hand, the university expects novel research, that is publishable in high profile peer reviewed journals. These two expectations do not always integrate well, so part of my work as an industrial Ph.D. student has been to manage these expectations on both sides.

An example is the global position estimation project, discussed in detail further on in this thesis. The work is scientifically sound and was a great success from that perspective: it resulted in a peer reviewed publication at SCA 2021/PACMCGIT Schreiner et al. (2021a) and it shows a novel approach to solving the problem of global position estimation for IMU-based motion capture systems. We even successfully extended it from trajectories in the horizontal plane to multilevel 3D trajectory estimation, a work we have currently under review. However, academic success does not mean that this work can now directly be implemented as a product. The experiments were conducted under controlled circumstances: the work is a proof of concept. The step from there to implementation is huge and maybe even bigger than we realised beforehand. The result was that deployment did not go as hoped, while at the same time the reality at the company moved on. Other solutions that could solve part of the same cases solved by our work turned out easier to implement, and strategical changes in direction of the company all made the work less acutely relevant. We eventually decided to move away from implementation of the project and shelve it for the time being, leaving everyone involved with a feeling of

dissatisfaction.

I would not say now that the project was a waste of time, even though at the time I probably did feel that way for a while. But in hindsight it taught me some valuable lessons on how to conduct research in an industrial setting and how to safeguard against this kind of disappointment. One of the key take-aways in that regard are that it is crucial to organise a project so that it does not become a long runway to a single final result. In my experience, it is much safer to organise it around sub-deliverables, that each have their own value for the company, even if the end goal is not achieved. These deliverables in turn do not necessarily have to be of great scientific importance, and conversely the end-goal of the project may not be of huge impact to the company. As long as there is some kind of symbiosis where both sides benefit from the development of the other side.

Chapter 2

Global Position Prediction in the Horizontal Plane

This chapter presents the article:

Schreiner et al. (2021a):

Paul Schreiner, Maksym Perepichka, Hayden Lewis, Sune Darkner, Paul G. Kry, Kenny Erleben, and Victor B. Zordan. Global position prediction for interactive motion capture. *Proc. ACM Comput. Graph. Interact. Tech.*, 4(3), sep 2021a. doi: 10.1145/3479985. URL `https://doi.org/10.1145/3479985`,

It was presented at the Symposium on Computer Animation held from 7 to 10 September 2021 (online due to Covid-19). The full manuscript is contained in this chapter. The text and figures retain their original content as presented in the published version; however, they have been reformatted to comply with the style of this thesis.

2.1 Abstract

We present a method for reconstructing the global position of motion capture where position sensing is poor or unavailable. Capture systems, such as IMU suits, can provide excellent pose and orientation data of a capture subject, but otherwise need post processing to estimate global position. We propose a solution that trains a neural network to predict, in real-time, the height and body displacement given a short window of pose and orientation data. Our training dataset contains pre-recorded data with global positions from many different capture subjects, performing a wide variety of activities in order to broadly train a network to estimate on like and unseen activities. We compare training on two network architectures, a universal network (u-net) and a traditional convolutional neural network (CNN) - observing better error properties for the u-net in our results. We also evaluate our method for different classes of motion. We observe high quality results for motion examples with good representation in specialized datasets, while general performance appears better in a more broadly sampled dataset when input motions are far from training examples.

2.2 Introduction

Motion capture is making the transition from the studio to the home and consumer markets with virtual reality (VR) game consoles and related hardware demanding lower cost, less cumbersome, and interactive/real-time performance capture technologies. As the go-to commercial technology today, camera-based motion capture systems are quite common and offer attractive solutions for both marker-based and markerless capture solutions. However, consumer motion capture solutions, such as IMU suits, have the advantages of being untethered, do not suffer from occlusion, and avoid the need for dedicated space with carefully calibrated cameras. Additionally, in most cases, the equipment, such as inertial measurement units (IMUs), is considerably less expensive than camera-based hardware.

The problem with IMU-based motion capture is that it does not provide a direct

measurement of position. IMUs typically include accelerometers, magnetometers, and gyroscopes, which allow for an excellent measurement of rotation that can be used to reconstruct the pose of limbs as well as the orientation of the capture subject. In contrast, it is prohibitive to calculate useful position estimates through integration of the accelerometer signal due to noise and bias in their measurements. To the best of our knowledge, standard commercial (proprietary) solutions apply heuristics, such as reconstructing from assumed foot-ground contacts for interactive playback, and otherwise assume that errors can be corrected with post processing. But we note that this problem is also not unique to IMU capture, it exists for many measurement systems which focus on joint angle measurements, such as exoskeletons, strain sensors, and monocular camera systems.

In this paper, we propose a learning-based solution to compute the global position of joint or "pose" based motion capture by exploiting a large collection of previously recorded (optical) motion capture data. To this end, we train a neural network offline to predict the global body displacement from pose data based on a short-horizon trajectory of current pose data. We hypothesize that the information present in this short trajectory contains sufficient detail about the dynamics being captured that a short temporal window of such data will provide key information to predict the character's global motion. Namely, following training, the network predicts the vertical position of the center of mass and its horizontal displacements per frame. The latter is integrated to reconstruct the global motion. The use of a fixed temporal window makes the solution history independent, in contrast to, for instance, a recurrent neural network or other global optimization solution which considers a full trajectory. Further, because our solution requires only a short window of data, it is ideal for real-time use. Specifically, we showcase the use of a universal network (u-net) Ronneberger et al. (2015) to accomplish this effort, while we contrast it to other network architectures as well as their sensitivity to training data, window length, and a number of other system and network hyper-parameters. Once trained, the u-net prediction is much faster than real-time with our unoptimized code running at around 200 fps (further discussion on the timing appears later in the paper.)

Figure 2.1 shows a preview of our results. In addition to the qualitative evaluation

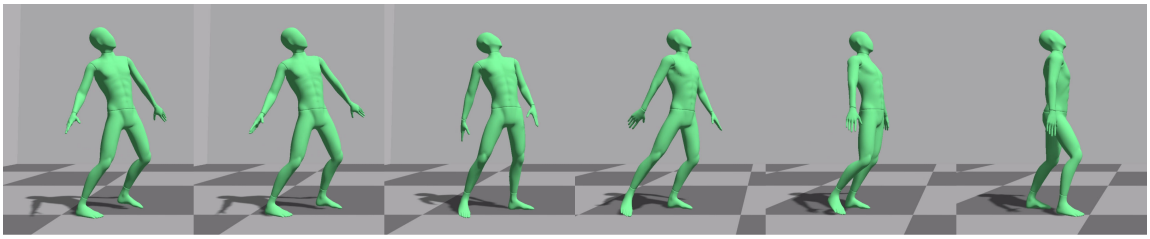


Figure 2.1: Example of captured motion where our method can create global position for unseen motion capture data. Because we train our u-nets with a large corpus of motion capture data, we are able to reconstruct global position for a wide variety of behaviors, even this unusual zombie-style walk.

of our animations, we compute errors for our reconstructions compared to (held-out) test data and independent reconstructions. We evaluate our design decisions through ablation studies, and choice of data representations. Finally, we present results and comparisons on multiple data subsets, and discuss the limitations and advantages as well as future improvements.

2.3 Related Work

Motion capture data recording and reuse has received a tremendous amount of attention in the research of human computer animation and analysis. We refer readers to additional sources, such as the book by Menache (2011), for the basics of the topic as a research domain, and instead focus here on the specific competing technology and approaches related to this work. Namely, we see our effort as an alternative to the common marker-based optical recording technologies, which require controlled environments and a relatively cumbersome set up Thewlis et al. (2013). While optical systems provide precision data, they require expert operators and are (still) rather expensive, relegating them to specialized and high-end studios or labs that are well funded and have ample space to devote to motion capture.

Because of their low cost, and relative ease-of-use, inertial motion recording systems in particular have attracted the attention of practitioners in commercial applications including sports, medicine, and entertainment. Research to address the

position problem, the focus of our paper, also appears in sports and medicine, where the aim is high accuracy under very specific conditions. A number of solutions have been proposed to meet the needs of the specific domain Lapinski et al. (2009); Coyte et al. (2013); Kok et al. (2017); Suh (2014); Li et al. (2020); Widagdo et al. (2017). For example, biomedical researchers have proposed to correct special cases of errors for rehabilitation Coyte et al. (2013). Others have focused on reducing drift in specialized highly dynamic behaviors of interest Fasel et al. (2017).

In the animation research community, the requirement for accuracy may be relaxed in favor of lower cost and more flexible solutions. For example, in support of recording alternatives, animation researchers have offered a variety of solutions for systems that compete with marker-based optical systems Vlasic et al. (2007); Slyper and Hodgins (2008); Shiratori et al. (2011).

Alternative motion capture systems have been commercially available for many years, e.g., IMU suits Roetenberg et al. (2009), and researchers have been offering solutions for computing and improving collected data from such for nearly as long Roetenberg et al. (2009); Schwarz et al. (2012); Floor-Westerdijk et al. (2012). However, to date, non-optical motion capture solutions have not become mainstream due to their limitations.

Finding global positions from pose information has also been studied extensively in the field of computer vision, where global trajectories are estimated from images Mehta et al. (2017); Zhou et al. (2018); Véges and Lőrincz (2019); Pavllo et al. (2019); Shimada et al. (2020). These techniques rely on 2D pose representations in an image space, i.e., with respect to a (normalized) camera focal point. This representation implicitly encodes 3D space in a 2D projection and is intrinsically different to our problem where we try to estimate 3D global coordinates from local 3D pose parameters.

2.3.1 Inertial Motion Capture

For inertial systems, motion capture data is extracted readily to provide body orientations Roetenberg et al. (2009) and, from these, joint angles and body orientation for

a given skeleton. While one can estimate sensor position displacement by integration of the acceleration from the IMUs, the calculation is prone to errors Floor-Westerdijk et al. (2012). Some researchers have proposed methods, for example particle filters, to extract better position data Schwarz et al. (2012). Others suggest the addition of complementary sensors, such as global positioning systems (GPS) Roetenberg et al. (2009) or ultraWide band (UWB) sensors for indoor positioning Corrales et al. (2008).

Clearly, an attractive option is to avoid additional hardware and instead employ a software solution derived from the input. Several solutions have been proposed in this area as well. The most straightforward approach is to employ the knowledge of the kinematics, for example extracting foot contact phases and then using this as a root to dictate forward motion through kinematics Yuan et al. (2011); Zheng et al. (2014). While we cannot be certain, we believe proprietary solutions for commercial packages use heuristics, kinematics, and Kalmann filtering, to extract foot contacts. Furthermore, various methods have been proposed to determine locomotion phase including special sensors to detect foot and heal strikes Foxlin (2005); Ju et al. (2015). Unfortunately, both heuristic and sensor techniques are prone to errors when contact changes occur and in behaviors that include flight phases, e.g., running Suh (2014). Our technique aims to follow the current trends in machine learning (ML) to address the root positioning problem of such systems, especially for real-time applications, such as gaming and direct playback. While some IMU-based research has employed ML, for example to handle noisy IMU placement Xiao and Zarar (2018), to our knowledge, no research has been published concerning use of ML in position estimation for real-time capture to date. Zhou et al. (2020) present a synthesis tool for keyframing which includes a global path predictor with some like characteristics to our own, although they solve a synthesis problem for 'in-betweening" while we solve a reconstruction problem in comparison. Notably, they report a precision of 0.7 cm/frame which is on the order of 100 times larger than our results.

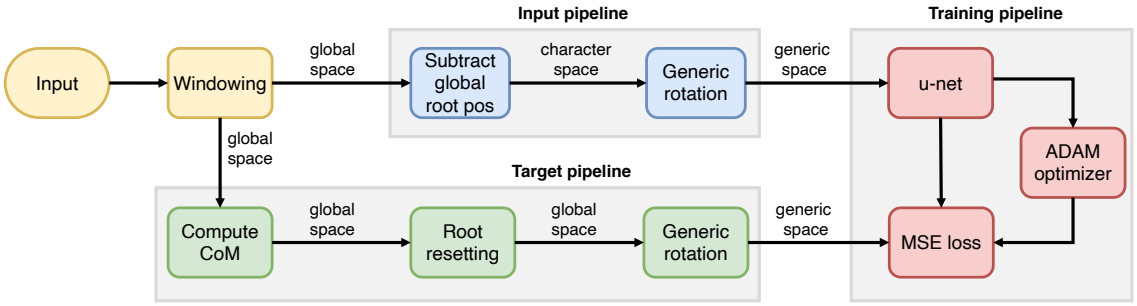


Figure 2.2: Input data preprocessing pipeline for motion capture data imported from Rokoko's Motion Library and targets for training a neural network, u-net, to predict center of mass positions given a time-window of relative joint data input.

2.3.2 Neural Networks for Motion Capture

The fields of computer animation and motion capture have had a multitude of ML techniques employed in both academia and industry. Recurrent neural networks (RNNs) are an evident choice for dealing with time-series data and have been employed for motion learning Fragkiadaki et al. (2015); Martinez et al. (2017). However, RNN-based approaches often suffer from noisy outputs, a problem that has been addressed by a multitude of authors. Ghosh et al. (2017) combine dropout autoencoders with LSTM (DAE-LSTM), i.e., using LSTMs to predict motion poses and denoising autoencoders (DAEs) to filter output reducing accumulative error. Recently, Wang et al. (2019) propose spatio-temporal recurrent neural networks (STRNN) consisting of three networks: a network to encode temporal dependencies, a spatial network to learn body-part dependencies, and a residual component to smooth out high frequency noise.

Holden et al. (2015, 2016) introduced the usage of convolutional neural nets (CNNs) to the domain of computer animation, using temporal convolutional layers to learn motion manifolds on the CMU motion capture dataset. Overlapping windows of motion are computed on animation data and provided as input to the network. CNNs have the advantage of being typically easy to train and producing smooth motion output, but suffer from their inability to deal with long-term dependencies. Recently, specialized models designed for animation data have achieved

state-of-the-art results, such as phase function neural networks (PFNN) Holden et al. (2017), which explicitly use footstep phase information to dynamically change network weights to drive character controllers. However, this method requires prior explicit manual labeling of motion phase information. Mode adaptive neural networks (MANN) Zhang et al. (2018) address this issue by automatically extracting phase information. Starke et al. (2020) improve upon this by considering local motion phases, determined by joint contacts with external objects.

Processing of time-series (e.g. motion capture) with neural networks has primarily been performed using RNNs which are specifically designed for this purpose.

Although their application has been successful in the context of text and speech Chiu and Nichols (2016); Graves et al. (2013) and EEG series in sleep classification Biswal et al. (2017), they are notoriously hard to train Pascanu et al. (2013). To overcome the challenges of RNNs Perslev et al. (2019) propose the u-net architecture Ronneberger et al. (2015) for time-series data as an alternative, and report superior performance, significantly increased stability, and ability to be trained on very large data sets Perslev et al. (2021). The scale-space convolutional structure of the u-net enables the modeling of temporal-spatial correlation over a fixed time-window and by modifying the u-net architecture to a regression output provides us with an ensemble of predictors. Ensembles are know for robustness and accuracy Hastie et al. (2009) and the average the set of predictions from our ensemble provides the final prediction. The choice of u-net offers faster training compared RNNs, increased stability during training and real-time inference Perslev et al. (2019).

2.4 Method

In this section, we introduce the u-net architecture and present the data pre- and post-processing pipeline that we use to optimize the learning capacity of the network. A core idea behind the use of this type of network architecture is the ability to learn correlations between pose data and its spatial-temporal correlation structure at multiple temporal scales. Figure 2.2 provides an overview of different parts of our training pipeline that we describe throughout the remainder of this section.

2.4.1 U-net Architecture

Through empirical experimentation we opt to employ the u-net architecture for our specialized problem. To adapt this tool for our needs, our u-net is modified for regression and acts as an ensemble of regression models from which we construct our prediction. Its layout is summarized in Figure 2.3. This network consists of an encoder stage and a decoder stage with skip connections relaying information at different temporal scales. In the encoder stage, the input data is encoded in the temporal dimension while being expanded in the feature dimension using convolutional layers. The input to the network is a 2D tensor, with time in the vertical dimension and features in the horizontal dimension.

We use T to denote the time-window size and N for the dimension of the combined feature vectors. A description of the layers, sizes, and features of the network architecture are as follows:

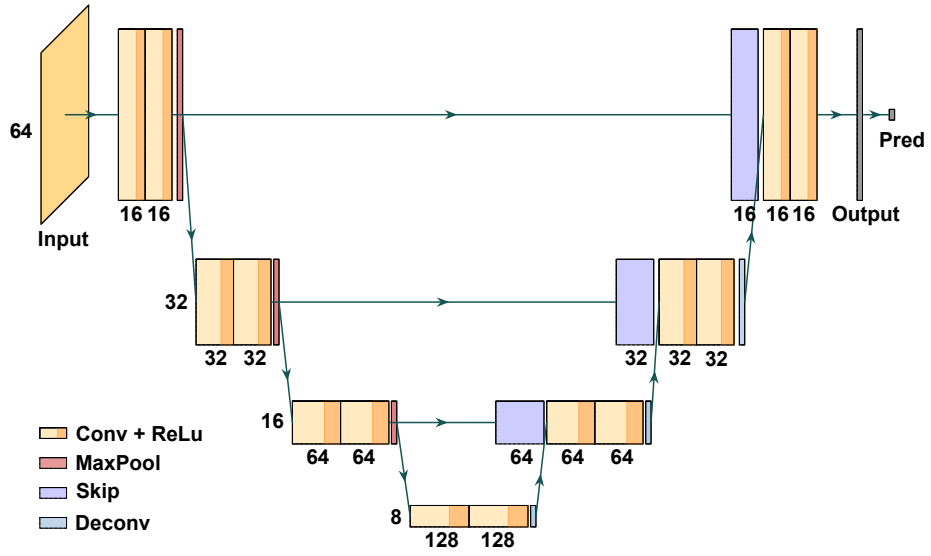


Figure 2.3: Illustration of our u-net layout, observe that skip connections and up/down sampling allow the u-net to handle time-series data and perform analysis of data at different frequency levels.

- The u-net operates at 3 different scales in the encoding and 3 scales in the

decoding. At each scale 2 consecutive convolutions of the input to that scale are performed.

- The first convolution is 2 dimensional with a kernel spanning the entire feature dimension N. In the temporal dimension we have experimented with kernels of size 3 and 5 and found that a kernel of size 5 generally provides the best results.

- With an input of [Batch $\times$ Channels in $\times$ $T \times N$] the output of the first convolution looks like [Batch $\times$ Channels out $\times$ $T \times 1$]. *Channel in* is generally 1 in our use case.

- The number of output channels of the first convolution doubles for each up sampling layer and is halved for each down sampling layer.

- The second convolution is exclusively in the temporal dimension, over all the output channels from the first convolution.

- The activation functions used throughout the network are rectified linear units (ReLU).

- After each set of convolutions the output of that step is reshaped so that the input to the next layer is again of the form [Batch $\times$ $1 \times T \times F$]. Here $F =$ *Channels out* can be seen as a new abstract feature dimension.

- At the end of the layer the current output is stored for later use in the skip connections. Then the output is down sampled in the temporal dimension using a maxpool operation with a length of 2. The feature dimension is kept constant during this step.

- Between each down and up sampling layer of the same temporal scale, there is a skip connection which passes the output of the encoder directly to its temporal counter part in the decoder side of the network. This ensures that the network can extract information and process it in the output for multiple timescales.

- The decoder structure follows an inverse description of the encoding process, where the up sampling is performed using linear interpolation.

In our implementation, we use the ADAM optimizer with a weight decay of $1e^{-5}$ as the only non-default parameter. We use the MSE loss function from the PyTorch library with all parameters set to default values.

2.4.2 Source Data

Our raw data is a rich collection of assets each containing the performance of a single motion or a short sequence of motions. Our training motion set comes from commercially available databases for human motion data, with 577 different assets from 61 unique actors, totalling 629,093 frames. Note, we purposefully include data from different motion capture studios (authors) and individuals (actors) to support diversity with respect to capture variance, and character size, shape, and gender. Each asset also may have individually distinct bone dimensions. Further, the range of motion types contained is also diverse, examples include walking, dance, jumps, martial arts, idle motion, and more. The animation data is encoded in a hierarchical format using a skeleton consisting of 19 bones plus a root bone (the pelvic bone). Each bone's motion is stored as an offset from its parent and a rotational trajectory. The root's motion is represented by positional and rotational trajectories with respect to a global frame.

For training the data was split into 3 subsets: a training set, a validation set and a left-out test set. The training set is used to train the model, the validation set is used to calculate a validation loss at the end of each training epoch. Finally, the left-out test set is used after training has converged to determine error statistics and all other results in the result section of this paper.

2.4.3 Pre-Processing of Input Data

In preparation for training, we re-sample the data to a uniform frame rate of 100 Hz as is typical with IMU motion capture systems (similar to Perslev et al. (2019)). This

is necessary because the input to the u-net requires a consistent temporal frequency, that is, each pose window must be the same size and span the same period. However, our training motion assets come from different sources with different frame rates. Subsequently, we preprocess the data by extracting short temporal windows, and mapping data into a generic forward facing reference frame (see also the pipeline diagram in Figure 2.2).

The data is passed to the network in short sequences of frames which we call *windows*. Each window is conceptually a short (less than one second) animation, with a fixed length of time. We chose a window size of $T = 64$ throughout this work. This windowing is performed online at training time, and has the advantage that we do not need to store duplicate frame data, hence reducing memory usage during training. This has negligible impact on training time as it is simply an array of pointers to memory. The windows overlap with a stride of 1 causing every frame in the data to be passed to the network in T consecutive windows. During training, the windows are shuffled in order to avoid bias from temporal correlation.

In addition, rather than using the local hierarchical joint angles, we preprocess each asset based on its skeletal dimensions by calculating 3D position vectors that indicate joint positions with respect to the root (the pelvic bone) and use those to represent poses. For our pose data, we assume the root that has a fixed position at the origin of a world frame. Further, as motion within the physical world is invariant to facing direction in the horizontal plane, we also define a *generic rotation* in which our model is to be trained. We align the vertical axis of our reference frame to match the the global vertical, opposite the direction of gravity. Next, the axes of the horizontal plane of our reference frame is set from the orientation of the root at the first frame of each window. That is, the root forward axis is projected to the global forward direction and the lateral axis is orthogonal to both the forward and the vertical.

The 19 joint position vectors are stacked and zero padded to form a 1D array $f \in \mathbb{R}^{64}$ for each frame. The root is not included in the input tensor because it is assumed to be poor or missing in the input data. The padding is done to maintain identical dimensions across the down- and up-sampling layers of the u-net. Finally,

each series of 64 consecutive frames are concatenated to form $[64 \times 64]$ tensors which are fed to the network in batches of 16 windows, thus $[16 \times 64 \times 64]$ is the size of the input tensors to our networks.

2.4.4 Processing of Training Targets

To compute the training targets, we adjust the input samples by: 1) computing the center of mass and treating this signal as the target; and 2) zeroing the root displacement at the start of the temporal window.

The root of our character is defined as the pelvic body which is subject to its own oscillations, for example, as the hips shift left and right in forward walking. Instead of estimating the root, we predict the center of mass (CM) positions, as it is less oscillatory and more indicative of the dynamics of the behavior. The CM estimates of the training targets are computed by summing a weighted approximation of each limb's center of mass. The weighting of each limb was performed using a re-targeting of the parameters from Dumas and Wojtusch (2017). In our experiments, we found this simple approach of computing the CM to be sufficient. In addition, to make the network invariant to the starting position of a time window, the trajectory in the horizontal plane of each window is reset to start at the origin. In the result, the training target is a time series representing the displacement of the character over the time period of the window.

In post processing, to recover global root data, the system performs an inverse of the target pre- processing. As the same frame in time is present in multiple target windows (due to the windowing), the network provides multiple predictions for the CM target of the same frame. So, we opt to collect all estimations and compute a mean value for our final CM position prediction.

2.5 Implementation and Results

Our u-net framework is implemented in Python, using PyTorch 1.7.0 for training the neural networks and NumPy/SciPy for data pre-processing. The experiments were

all trained on a cluster using NVIDIA TITAN RTX GPUs. The memory use on the GPU during training is approximately 3 GB. The focus of the experiments is to show that we can solve the global positioning problem using a neural network in real time. After training, the u-net runs above speeds needed for real time, in excess of 200 fps. However, there is an inherent delay due to the window size, as follows. If a window has size T, then the first T frames the u-net makes no estimation. Between frames T and $2T$ it can make a suboptimal estimation, and from frame $2T$ it can make the reported estimate - in real time - with a T frame delay. In our results we chose a value of T as 64 frames, or approximately 0.64 s.

We compare the u-net with conventional CNNs to demonstrate its benefits and drawbacks. A 4 layer CNN with Batch Normalization and Leaky ReLU activations is used for comparison. One-dimensional convolution is performed along the time dimension, with sliding windows identical to those of our u-net. The input/output channels of the 4 layers are as: (1) in: $T \times N$, out: 40; (2) in: 40, out: 20; (3) in: 20, out: 10; and (4) in: 10, out: 3.

We conducted the following experiments and discuss their results later in this section:

- **All data vs specialized dataset.** We investigate use of a more specialized dataset than the whole set (ALL), expecting that this would result in a better performance for selected motion types. To this end, we produce a reduced dataset containing only assets that are either *Run*, *Walk*, or *Idle* (RWI).

- **Absolute vertical vs vertical displacement.** An inherent problem with estimating displacement is that integration is needed. Any bias in the estimation, however small, will therefore eventually grow without bounds. In this experiment, we estimated the absolute height of the character against vertical displacements to avoid drift on the vertical axis.

- **Center of mass vs root position estimation.** We proposed the use of the CM in the estimation as a more stable signal than the root position for global displacement prediction. We compare the two to ascertain the impact of this decision.

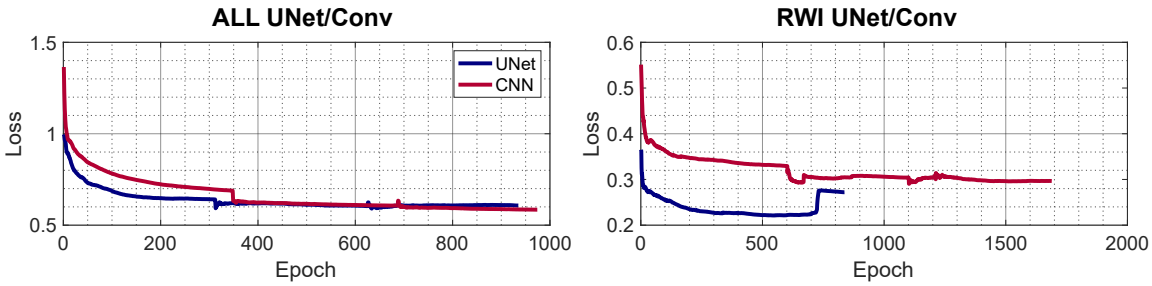


Figure 2.4: Validation loss plots for all data (ALL) or run walk idle (RWI), and different network architectures. Note the discontinuities in most plots caused by restarting the ADAM optimizer. The jump in the curve of the of the u-net in the RWI plot is caused by over fitting. Note that the blue (u-net) and red (CNN) curves are consistently in the same loss range within experiments.

To improve the visual quality of the results, we optionally post process the results with inverse kinematics (as noted in the video). To this end, foot ground contact labels are extracted from the motion using a method based on Lee et al. (2002), comparing each foot joint's positions and velocities against predefined thresholds. Ground contact labels are subsequently cleaned using thresholds to avoid false positives and false negatives by removing short contacts less than 5 frames and filling gaps between contacts that are less than 15 frames. Using the contact labels, feet target positions are generated. For frames where contact is detected, the target is set to a static point where the horizontal plane coordinates correspond to the position of the foot, and vertical coordinate corresponds to the previously determined floor height. Cubic Hermite spline interpolation is used to blend in and out from this contact point. Finally, the feet are made to track the target positions using analytical two joint inverse kinematics. Note that this is not a full foot skate cleanup, but provides a moderate improvement to our final results.

2.5.1 Validation

The typical training times for all experiments were in the order one to two weeks. Figure 2.4 shows the validation loss for both conventional nets and u-nets, and serves as a sanity check that training converges in all cases. Table 2.1 provides an overview

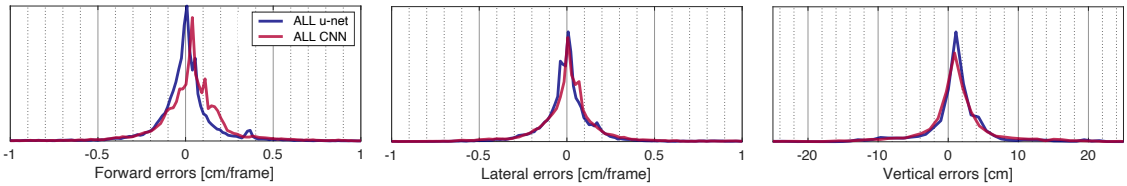


Figure 2.5: Probability density functions showing the distribution of the per frame error on each axis for the all data experiment.

of the error statistics of the different experiments. The u-net version of the ALL dataset outperformed the other scenarios. Mean errors result in drift when integration is performed. While all models had some degree of mean error, albeit low, our u-net performed best for all parameters that contribute to drift, with mean errors in the sub millimeter range. We did see a larger mean error in the vertical direction, however this does not contribute to any drift as these estimates are not integrated. Observe how the standard deviation is always an order of magnitude larger than the mean error, indicating that the errors are predominantly local and less systematic.

To take a closer look at the somewhat large mean errors of the height estimates, we refer to the plots in Figure 2.5. Here probability density functions are shown for the errors of both networks. We can see that while the main body of errors for the u-net model in the forward direction is centered around zero, the errors from the CNN model are more biased. For the vertical axis we see a bias in the distribution of both models, however this bias is significantly smaller than μ_v in Table 2.1 and has an opposite sign.

2.5.2 Robustness and Generalizations

For a comprehensive view of what our results look like in final render we refer readers to the supplementary video. However, the snapshots in Figure 2.6 give an impression of what a walk motion predicted by our u-net looks like. Note the solid foot positioning as an indicator that the global position is estimated with high precision and minimal visible drift.

Table 2.1: Comparison between different networks and datasets, either ALL, or run, walk and idle (RWI). The error mean μ and standard deviation σ is shown for forward, lateral, and vertical directions as denoted by subscripts. All units are cm per frame except for those where the vertical output is an absolute position estimate, in which case the units are cm.

Model	Data	μ_f	μ_l	μ_v	σ_f	σ_l	σ_v
u-net	ALL	**-0.004**	**0.002**	**-4.062**	**0.191**	**0.185**	**23.123**
CNN	ALL	0.034	0.007	-4.202	0.211	0.219	24.274
u-net	RWI	-0.026	0.034	-5.886	3.331	0.203	29.010
CNN	RWI	-0.017	-0.040	-14.582	3.349	0.303	78.551

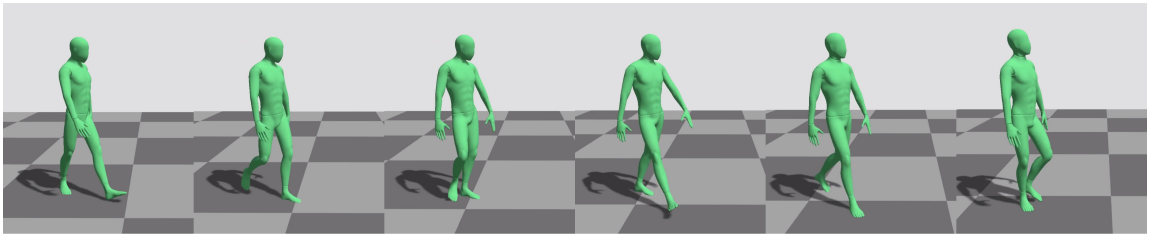


Figure 2.6: This walk motion shows solid foot plants for this walking data that does not need any clean up as it is well represented in our database and the u-net is able to predict the motion with minimal error.

A comparison between the walking results of the u-net model and the CNN model can be seen in Figure 2.7. The trajectory estimated by both models as well as the reference trajectory can be seen. The u-net predicts estimates very close to the original motion while the CNN estimate displays significant drift over the range of motion.

A more complex motion asset of a character's motion is plotted in Figure 2.8 (for the u-net alone). The character is initially standing still, then, at approximately frame 130, starts accelerating to full-speed walking, reached around frame 350. The character maintains a constant cyclic walking motion for a duration, until frame 2050 where the character decelerates and come to full stand still. Our system proves

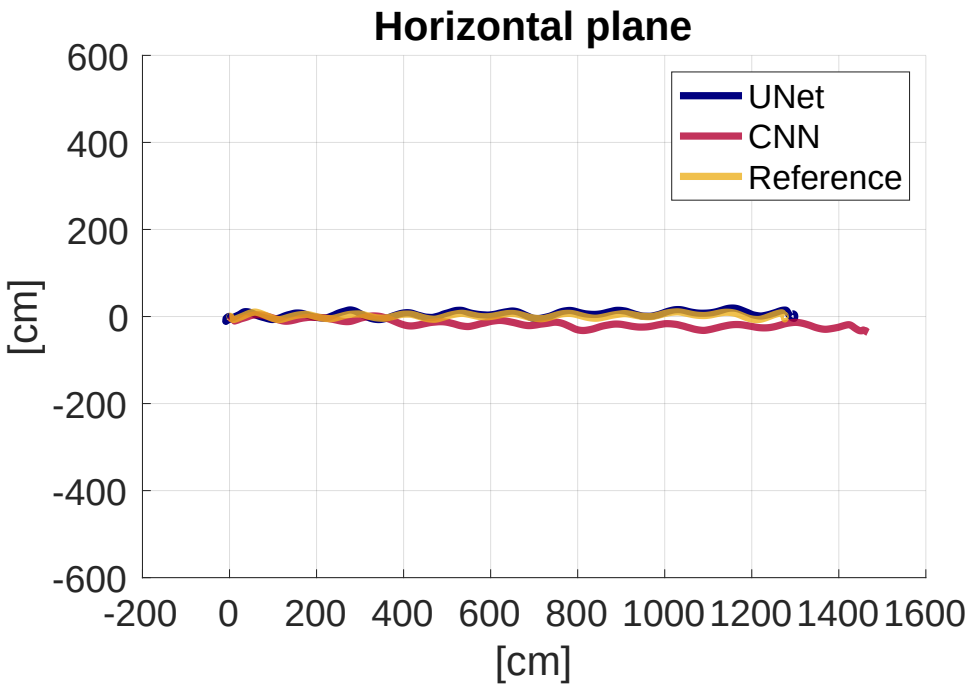


Figure 2.7: Top view of a trajectory of a character walking in a straight line. Notice how the u-net estimate is close to the reference while the CNN shows significant drift.

capable of accurately reconstructing the entire motion sequence. Both displacements in the horizontal plane as well as the position on the vertical axis are estimated to a high precision.

2.5.3 Ablation Studies

Figure 2.9 shows plots of a character's height during a run together with the absolute height estimates and integrated displacement estimates (both u-net). A natural approach to train a network to predict motion would be to make it learn the same type of parameter on all axes. However, the kinematic constraints present in the human morphology and an assumption of ground contact justify attempting to estimate absolute vertical positions instead. We compared the performance in both scenarios. While the absolute height estimate follows the reference trajectory, the

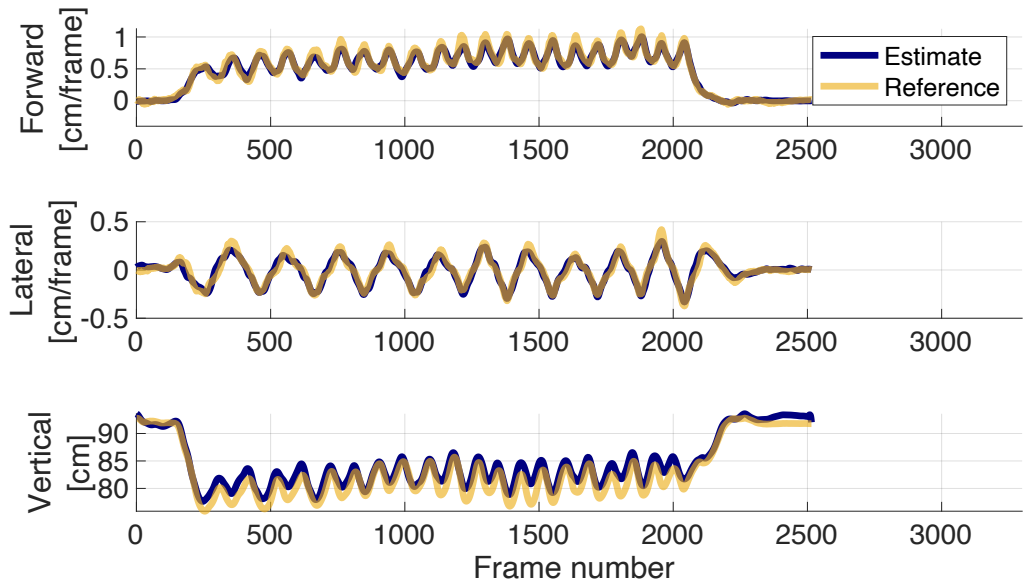


Figure 2.8: Horizontal displacement and vertical position estimates for a compound motion using the u-net architecture. Observe how the system is able to estimate standstill as well as cyclic motion, acceleration and deceleration in all axes.

integrated displacements introduce a downward drift caused by biased estimations of the displacements as the figure reveals.

To support our choice of the CM as the estimation value, we trained two networks: one predicting the global position of the root and one predicting that of the CM. We evaluate the two by reconstructing the root position of the character using the estimate of the CM and compare it with the directly estimated root position. In support of our proposed approach, we found that the root reconstruction from the CM network produces quantitatively better results in general. We show a representative plot in Figure 2.10. It should be noted that even though the root estimation seems to outperform the CM when it comes to the vertical axis prediction, these prediction errors are not integrated; i.e., the vertical *position* error does not grow in time. The forward and lateral predictions are velocities, which must be integrated in order to get a final position.

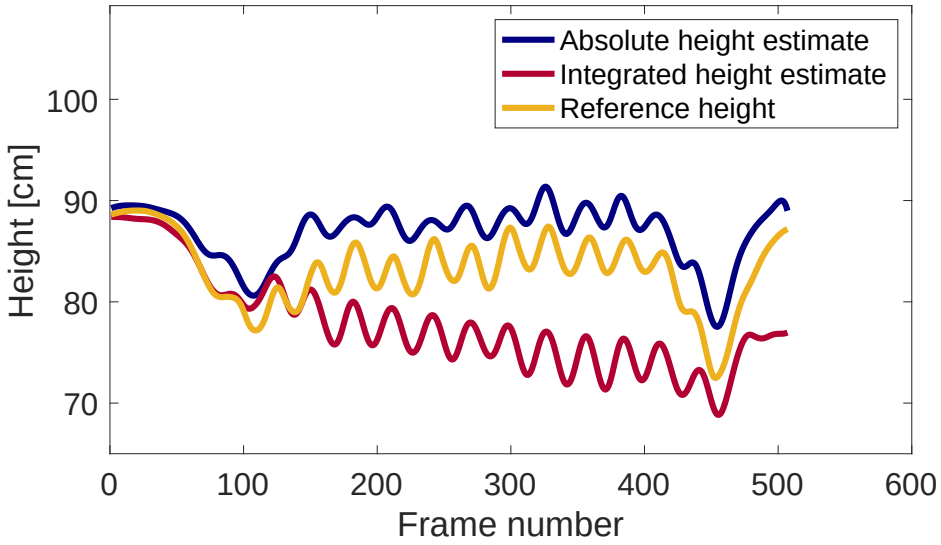


Figure 2.9: Comparison between (u-net) estimates of the absolute height and height by integrating displacements. Note the drift in the integrated result due to the accumulation of the error.

In the case of the animation in Figure 2.10, the center of mass predictions follow the reference more closely in several places in the motion for the lateral and forward displacement while it shows a constant offset with respect to the reference for the vertical direction. When integrating the forward displacements of this animation, the positional error at the end of the motion is 1.5 cm for the CM while it is 58 cm for the root. In comparison, the difference in mean absolute error of the vertical estimation is 3.4 cm for the CM prediction versus 1.6 cm for the root predictions. In light of this, we deem the CM estimation to be a better predictor than the root overall.

We highlight how the system performs differently for the RWI training motion in comparison to training on ALL the data in Figure 2.11. Here we showcase two assets, one a run and the second a dance. The first motion is that of a character accelerating from standstill to a running motion, and then after a period of constant motion slowing down again to end in standstill. We note the specialized model trained with the RWI dataset performs slightly better for running. For the second,

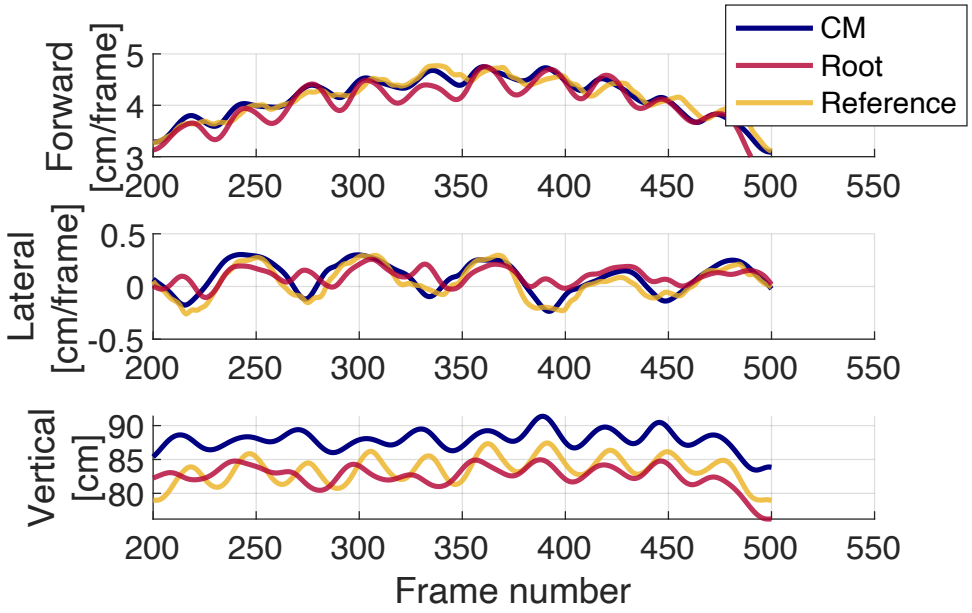


Figure 2.10: Comparison between center of mass and root as the estimation target. The center of mass estimation has been projected back to the root (pelvis). Note that the unit of the forward and lateral predictions is cm/frame while the vertical prediction is in cm.

a dancing motion, a similar comparison is performed. We note that for this motion the model trained with the ALL data performs better. In aggregate, from Table 2.1 we see that the networks trained using the ALL data set outperform the specialized RWI trained networks. From this result, we conclude that training with more data is generally preferable over training with specialized data. More experimentation would likely benefit specific applications.

Running motion is typically a difficult type of motion for heuristic methods due to the highly dynamic nature of the run cycle, which includes a flight phase, making it difficult to predict displacements even if proper foot contacts are determined. In Figure 2.12, snapshots of a running motion are shown where the displacement and height were predicted using our method. Note how the right foot does not move with respect to the tile borders once it is planted, indicating a solid stance phase (see also the supplementary video). Note, no inverse kinematics clean up is performed to obtain this result.

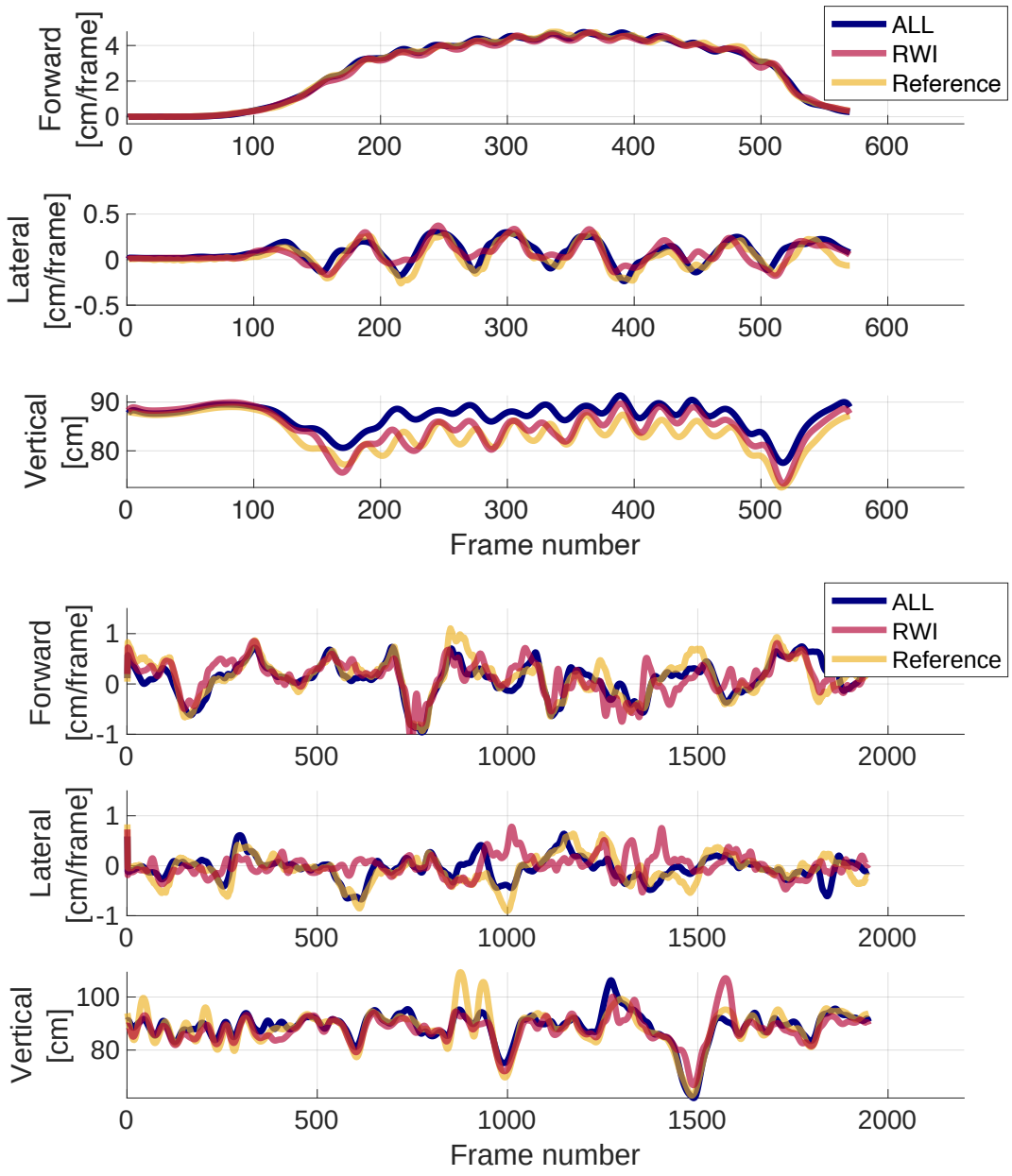


Figure 2.11: Comparison between a u-net trained using the ALL data set and a u-net trained using the more specialized RWI data set. The left plot shows a character running. Notice the larger error in the estimation of the ALL trained model, especially in the vertical prediction. The right plot shows a character dancing, which is a motion type not available in the RWI data set. Observe how the RWI trained model has more difficulty predicting the motion, for example, in the lateral motion around frame 1000.

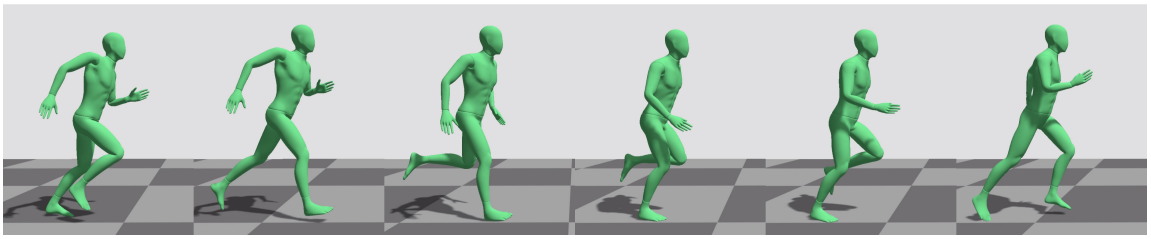


Figure 2.12: Running motion with a flight phase is particularly difficult for heuristic based solutions. Our approach can estimate good lateral motion, as exhibited by the lack of foot skate, and predicts the vertical trajectory that is nearly imperceptible from ground truth. Please refer to the video for this and other comparisons.

2.5.4 IMU Data Evaluation

We use real IMU based motion capture data recorded with Rokoko's Smartsuit Pro, to compare our trained u-net to Rokoko's heuristic approach (in the video). The supplemental video also shows position reconstruction for other motions recorded using an IMU suit. While our network was trained on optical motion capture data, we show we can also reconstruct global positions for motion captured with the IMU suit. We do note, however, that the raw IMU pose data was poorer quality in general and had larger errors, assumably because the IMU error accumulates at all joints from the root to the extremities.

2.6 Discussion and Conclusion

Our experiments show that a trained network can reasonably estimate the position of a humanoid character in a global coordinate frame interactively from local joint angle and orientation data alone. We propose to apply this estimation method for IMU motion capture but feel it is particularly valuable for interactive applications where position is needed, especially as it runs at faster than real-time rates. Within the scope of our investigations, we compare standard CNNs with the u-net architecture, and found that u-nets are capable of estimating motion with a higher level of precision. Notably the u-net both drifted significantly less and produced smaller

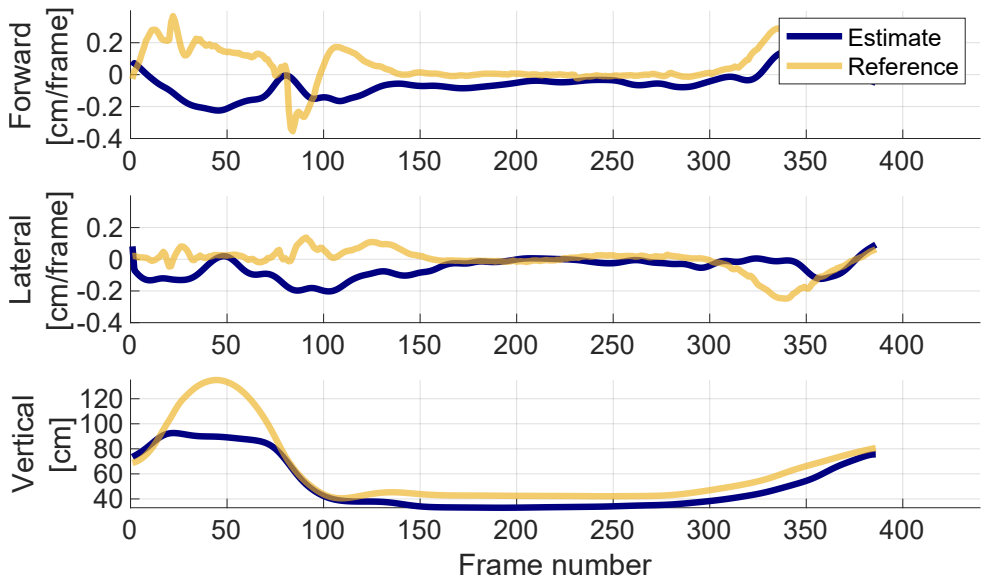


Figure 2.13: Estimation plot of a character jumping at time $t = 0$. Note that the network is unable to track the height as the character lifts off but recovers as soon as the character touches down again.

per-frame errors in comparison to the CNN. In addition, our approach was able to eliminate drift in the vertical dimension by estimating an absolute position, instead of displacement, thereby eliminating the need for integration. While this relies on the assumption that motion appears on a horizontal plane, it could also be reverted to the displacement (with errors as reported) if elevation change is required for an application.

We compare networks trained using test data from a wide span of motion with models trained on a more specialized dataset. We found that the specialized network was slightly better at predicting motions similar to those present in the specialized training data. We see this as useful for in-depth studies where the motion type is known in advance, and the luxury of a dedicated dataset is justified. In contrast, when the trained system is presented with novel data, such as that of a character dancing, the specialized model performed less well than the one trained on the more general dataset. From this we conclude that the model trained on the broader dataset is likely able to estimate new motion types better. This experimentation also spurs

the pursuit of more complex systems where clusters of networks might compete for optimal prediction in future work.

We observe that our method fails in cases where the character is not in physical contact with the world over a prolonged period of time. The results for one such asset is plotted in Figure 2.13. In this asset, a character starts with a jump. The model initially tracks the motion until the feet lift off the ground and the model poorly estimates the motion in the air. Once the characters feet touch down, an accurate prediction is restored. As the capacity to estimate positions depends on the presence of like data in the database, in order to estimate general free fall, the model would need to incorporate additional training data or perhaps a model of the dynamics itself. For example, with parameters for gravity and momentum, we believe reconstructed flight phases could be improved. Incorporating these types of dynamic constraints within this scope is, to the best of our knowledge, unexplored and an additional interesting topic of future work.

Our method is meant to be used in real-time in order to get an on-the-fly estimate of global displacement. To this regard execution time has to be fast enough to run concurrently with a motion capture solution. Our solution is light enough to execute at 200 fps on a modern CPU, far faster than the 100 fps it is designed for. This is without any hardware acceleration such as GPU or code optimisations implemented, leaving room for far higher frame rates. Due to the windowing of data, estimates do include a short delay of less than one second.

Returning to our original motivation, we show that our solution is capable of estimating global placement for data from IMU systems alone but note the output quality is lower than of the optical examples, which is not surprising since the input data to the network is also lower quality. We expect that introducing IMU data to the training will improve the performance for this type of data. This is a key direction to make this approach practical for future commercialization with IMUs. However, as is, our solution still represents a significant step forward in the potential for global positioning from joint angle and orientation data, especially in real-time applications.

Chapter 3

Root Position Reconstruction in 3 Dimensions For IMU Data

This chapter presents the article:

Schreiner et al. (2024a):

Paul Schreiner, Sune Darkner, and Kenny Erleben. Root3D: Root position reconstruction in 3 dimensions for imu based motion capture. *Under review at the Symposium on Interactive 3D Graphics and Games*, 2024a

The full manuscript is contained in this chapter.

3.1 Abstract

Lightweight global trajectory reconstruction in 3 dimensions is a yet to be solved problem for animation data from IMU based motion capture systems. In this work we use a lean U-Net neural network architecture to estimate global displacements in a stable and accurate manner. Our network takes as an input a combination of local pose information and acceleration signals from IMU sensors and estimates short, character centred trajectories of 8 frames. We use a weighted average of the predictions to reduce the effect of estimation bias and noise during integration of the displacement and empirically show its advantage over other methods. We test our trained model on a dataset of unseen data, from actors that were not included in the training set. Our method is capable of accurately reconstructing the ground truth trajectory, without significant drift effects, for both horizontal planar motion as well as motion in the vertical direction. We further show how with declining pose data quality, estimation accuracy deteriorates and how acceleration signals are pivotal to maintain high quality trajectory reconstruction.

3.2 Introduction

The animation industry in 2023 is a multibillion global industry (Statista Research Department, 2023; Knowledge Sourcing, 2023). Within this industry motion capture serves a key role in the creation of artistically free and realistic animation. Apart from the animation industry, motion capture has many applications in other fields such as sports and medicine.

For decades, optical marker based solutions have dominated the field of motion capture due to their high accuracy and the lack of serious competition from other technologies. However, their high startup and operational costs reserves them to a large degree for high end production studios and the like and puts them out of reach of smaller studios or individual creatives.

Because of these limitations, and the wide availability of cameras in everyday life appliances, alternative methods are gaining traction, like monocular motion capture,

where poses and positions are estimated from video from low cost cameras. However quality can still not compare to high end optical systems.

Inertial systems form a more mature alternative for marker based motion capture, at a much lower price point. These systems, where poses are estimated from body mounted inertial measurement units (IMUs), have the additional advantage of being untethered, and are insensitive to daylight. These, however, are advantages that come at the cost of quality and a lack of state information. A problem central to IMU based motion capture, is that of estimating global positions. As the sensor measures derivative signals of the quantities of interest, namely, angular velocity for orientation and acceleration for position, we rely on the integration of these signals to recover orientations and positions. Orientation measurement can be stabilized using knowledge about earth's magnetic and gravitational fields, as measured by a magnetometer and accelerometer respectively. However, there are no passive sensors that can stabilise integration of the acceleration signals. To complicate things more, in order to obtain position from acceleration, we would need to integrate twice, causing hardware related noise and bias errors to accumulate exponentially over time.

To date, a simple, robust solution to this problem, that does not require additional external hardware such as camera sensors, GPS or high frequency magnetic field generators, has not been found.

In this work, we build on top of the work from Schreiner et al. (2021a) and propose a method to robustly estimate global positions from local pose information and acceleration signals.

The main contributions of this work are:

- A robust method based on a feedforward U-Net neural network for estimating global positions from sequences of local pose information and acceleration signals from IMU sensors

- An extension of the method proposed by Schreiner et al. (2021a) to include displacements in the vertical direction, enabling us to track multilevel motions such as stair walking

- The presentation and empirical validation of a method for improving estimations, by making use of all the information from the time series data that is output by the U-Net

3.3 Related Work

Global position estimation A shortcoming of wearable IMUs is their lack of accurate global position information. Positional information could be obtained by double integration of the acceleration signal (Floor-Westerdijk et al., 2012), but this is vulnerable to drift problems, due to the presence of sensor noise and biases. Drift can be limited by incorporating contact detection with the environment (Roetenberg et al., 2009; Yuan et al., 2011; Zheng et al., 2014) but this limits the use of the method to contact rich motions and requires specialised contact detection algorithms. Alternatively, the IMU data can be fused with data from other sources (Roetenberg et al., 2009; Corrales et al., 2008; Schreiner et al., 2021b); however this implies the use of additional hardware, potentially constraining the user to a specific recording volume.

This work seeks to solve the positioning problem by employing machine learning in the form of neural networks. It is an extension of the work from Schreiner et al. (2021a), who successfully employed U-Nets to reconstruct global trajectories in a horizontal plane from sequences of local pose information. We address two topics that were left open, namely improving robustness by including acceleration signals and extending the framework to the 3D space.

Computer vision In recent years, camera based motion capture has become a serious competitor to inertial motion capture. This branch of motion capture suffers from its own difficulties with global position estimation. In this case the positions have to be transformed from a 2D projection (the image) to a 3D representation. A number of works have contributed to solving this problem (Mehta et al., 2017; Zhou et al., 2018; Véges and Lőrincz, 2019; Pavllo et al., 2019; Shimada et al., 2020, 2021; Wang et al., 2020). However, the computer vision problem is fundamentally

different to ours, in that it utilizes the embedding of 3D positions in a 2D encoding. Closer to our approach is the resent work of Yuan et al. (2022a) who had impressive results with a combination of motion infilling and global trajectory estimation. They employed conditional variational auto encoders in combination with an optimisation strategy that aligns the estimated trajectories with information extracted from the images.

Bio-mechanical approaches Using IMUs for capturing information about human motion and the topic of position estimation is also explored in the fields of sports and medicine where precise measurements under particular conditions are crucial. Various methods have been developed to fulfill these domain-specific requirements (Lapinski et al., 2009; Kok et al., 2017; Suh, 2014; Li et al., 2020; Widagdo et al., 2017; Coyte et al., 2013; Fasel et al., 2017).

Deep learning: pose estimation, motion synthesis, and generative AI Over the last decade, deep learning has earned its place within the field of motion capture. It has seen a wide variety of applications, ranging from pose estimation to motion synthesis, and more recently, generative applications. Motion synthesis, correction and de-noising saw early breakthroughs with the works of Holden et al. (2015) and Holden et al. (2016) where motion data was formatted as 2D tensors with pose information on one axis and the temporal dimension on the other, making them suitable for convolutional neural networks (CNN). These would learn motion manifolds, allowing the authors to de-noise motion data, and synthesise transformations by performing inbetweening. These works established the potential of CNNs for motion data and could estimate global trajectories between start and end points, however, they did not deal with the problem of unconstrained global position estimation. In Holden et al. (2017) the authors introduce phase-functioned neural networks (PFNN) as a form of neural motion matching. Their method uses an input from the character's current motion phase to calculate an optimal set of network weights to predict the next step of a motion. This concept is further improved by Zhang et al. (2018) who replace the phase-function with another neural network for quadruped

control and Starke et al. (2020), who exploit joint contacts with the environment to calculate local motion phases.

The past year has seen the rise and the becoming mainstream of large language models (LLM) and generative AI models such as the GPT and DALL-E series (Brown et al., 2020; Ramesh et al., 2021). Inspired by these works, researchers have attempted to employ these techniques in various fields, likewise in the field of character animation. A broad range of works propose generative solutions that are capable of generating motion from text prompts (Tevet et al., 2022; Yuan et al., 2023; Petrovich et al., 2022). However the quality of the motion that is output by these models is still far behind on motion capture quality.

Recurrent neural networks (RNN) are an attractive option for motion data due to their ability to remember. In Ghosh et al. (2017) the authors performed motion synthesis over long time-horizons using RNNs, while Martinez et al. (2017) predict future poses based on past sequences. These types of models are notoriously hard to train, tending to either converge to the mean motion of the dataset or diverge to produce jittery motion. Wang et al. (2019) proposes to mitigate some of these issues by using spatio-temporal RNNs (STRNN). In Huang et al. (2018), full body poses are estimated using sparse IMU sensor configurations in combination with bi-directional RNNs. The authors do not address the positioning problem, but they show how including raw acceleration signals is essential for performance and helps avoiding overfitting. More recently, a number of works have sought to reconstruct full pose information including global translations from sparse IMU sensor configurations (Yi et al., 2021, 2022; Van Wouwe et al., 2023). These methods show impressive results considering the small amount of sensors they operate on, but still depend on constraints such as that the motion is contact rich and over flat ground. In Pan et al. (2023) the authors fuse sparse IMU data with monocular video inputs as they seek to eliminate global drift. Another problem with RNNs is that they are notoriously hard to train (Pascanu et al., 2013). As an alternative to RNNs Perslev et al. (2019) and Perslev et al. (2021) show improved training performance and stability over RNNs, using the U-Net architecture from Ronneberger et al. (2015) to work with time-series data from sleep-cycles. U-Nets are also implemented by Schreiner et al.

(2021b) to predict global position information in a horizontal plane for high quality motion capture and are shown to outperform a standard CNN.

In contrast to previous work, we use a straight forward, easy to train, convolutional neural network. Moreover, we do not rely on additional hardware for constraining inputs or make assumptions about the space that we are moving in.

3.4 Method

In previous work, Schreiner et al. (2021a) reconstructed global trajectories, using local pose information. They demonstrated good results on high quality data, however they struggled with transferring these results to recordings with lower quality motion capture data. To improve performance for this type of motion data, they suggest including acceleration signals from IMU sensors into the training data of the neural network. In this work, we pick up on this idea and show how adding acceleration data is an effective and essential improvement for estimating global trajectories. We further improve on their work, by extending the prediction framework from horizontal planar motion to including the vertical dimension.

3.4.1 Data, Network Architecture, Feature Vector and Targets

Data For this work we recorded a custom dataset using Rokoko's Smartsuit Pro II, an IMU based motion capture suit, and an Optitrack optical motion capture setup. We recorded a total of 434 motion assets, with 17 different actors. The actors were asked to perform three types of motion: walking in the horizontal plane, walking up stairs, and walking down stairs. Each motion was recorded using both the optical setup and the IMU suit, using a trigger to synchronize recording start and stop. All recordings were exported at 100 frames per second.

Network architecture Our model takes a window of W motion data frames and predicts an equally long mini-trajectory of the root position of the character. We

choose a U-Net convolutional neural network as our model architecture for its ability to handle time series data. In Schreiner et al. (2021a) this choice is discussed in detail and shown to be superior over standard CNNs. We refer to that work for a more in-depth analysis of the performance of U-Nets for this type of problem. The main difference with our implementation is that our network adds a small set of fully connected layers right after the input to pre-process the data. The rationale behind this is that the U-Net uses convolutional layers, which are optimised to find patterns between adjacent features in the feature tensor. Human motion data has a hierarchical structure, where there are clear physical relations between the data points. However, these relations are not necessarily represented in the partition of the input tensor. Hence, the fully connected layers serve to extract an abstract representation of the feature tensor, more suitable for use with convolutional layers.The convolutional layers in our network each contain 9 input and output channels except for the first and last layers. These layers serve to increase the channels from 1 to 9 and decrease them from 9 to 1 respectively. Our network architecture is represented in figure 3.1. The result is a very lean network, containing a little under 40k parameters, making it suitable for real time computation and potentially even possible to integrate it on dedicated devices.

Feature vector The feature vector of our network consists of a combination of local pose information and acceleration signals. We define our feature vector as

$$\mathbf{x} \equiv \left\{ \begin{array}{cc} \mathbf{p}_{joints} & \mathbf{a} \end{array} \right\} \in \mathbb{R}^{117}, \tag{3.1}$$

Here $\mathbf{p}_{joints} \in \mathbb{R}^{60}$ is the 20 joint positions, in 3D space, of the character with respect to its root, flattened to form a 1D tensor. The acceleration signals, $\mathbf{a} \in \mathbb{R}^{57}$ are recorded using 19 body-mounted IMU sensors. Each IMU sensor measures acceleration along its own three axes. In order to obtain accelerations in global space, we use the IMU sensor's orientation estimate, to rotate the signals from sensor frame to world frame. We then subtract gravity to obtain the free body accelerations in

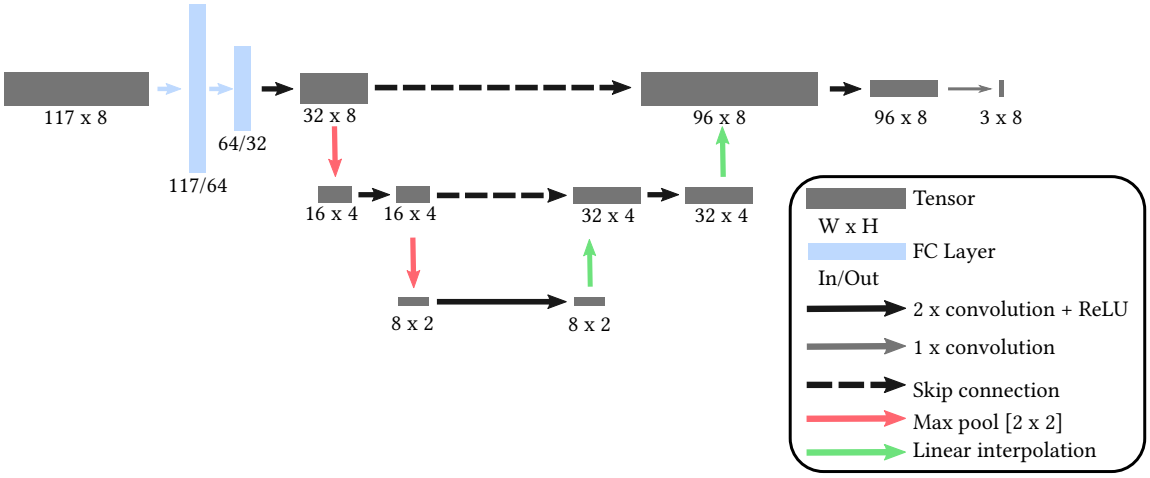


Figure 3.1: Illustration of our U-Net layout. We have added two fully connected layers in the beginning of the network, to optimise the feature partitioning for the convolutional layers. All convolutional layers have 9 in and output channels, apart from the first (1 input channel, 9 output) and the last (9 input channels, 1 output).

the world frame. For each body n we have

$$\mathbf{a}_n = \mathbf{q}_{IMU,n} \otimes \mathbf{a}_{IMU,n} \otimes \mathbf{q}^*_{IMU,n} - \mathbf{g}\,. \tag{3.2}$$

Here $\mathbf{a}_{IMU,n}$ is the measured acceleration in sensor frame of the n^{th} sensor, $\mathbf{q}_{IMU,b}$ is the rotation of the sensor with respect to the world frame, $\otimes$ indicates a quaternion multiplication, and $\mathbf{q}^*$ is the conjugate of unit-quaternion $\mathbf{q}$.

Finally, we assemble windows of W feature vectors as the input to our model:

$$\mathbf{X}_W \equiv \begin{bmatrix} \mathbf{x}_0 & \mathbf{x}_1 & \cdots & \mathbf{x}_W \end{bmatrix} \tag{3.3}$$

In our particular setup we use $W = 8$ which is a huge improvement over Schreiner et al. (2021a) that used $W = 64$.

Target Our model predicts mini-trajectories of W consecutive frames at every iteration. We define the predictions of the model at the n^{th} step as

$$\mathcal{S}_n \equiv \begin{bmatrix} \mathbf{s}_{n,0} & \mathbf{s}_{n,1} & \cdots & \mathbf{s}_{n,W} \end{bmatrix} \tag{3.4}$$

where W is the window size and $\mathbf{s}_{n,j} \in \mathbb{R}^3$ is the root prediction of the j^{th} time slice in the n^{th} step with respect to the current position of the character.

Pre-processing During training we align each window with the global frame. We take the forward axis of the character root at the first frame of the window, and rotate it in the horizontal plane, such that it aligns with the global forward axis. This is done to reduce the complexity of the task by eliminating global heading as a factor in the reconstruction problem.

3.4.2 Trajectory Reconstruction Using One or More Prediction Steps

Reconstruction of the full animation trajectory is then done by using one or more of these predictions to increment one step of the full trajectory. We denote a full trajectory with length L as

$$\mathcal{T} \equiv [\mathbf{T}_0,\ \mathbf{T}_1,\ \cdots,\ \mathbf{T}_L], \tag{3.5}$$

where $\mathbf{T}_i \in \mathbb{R}^3$ is the root position in global space.

Now we can define a function,

$$\mathcal{F}(\mathcal{S}_n) \equiv \mathbf{d}_n\,. \tag{3.6}$$

that maps the predictions to a root displacement $\mathbf{d}_n \in \mathbb{R}^3$, so we can increment the full trajectory with one step as

$$\mathbf{T}_{n+1} \leftarrow \mathbf{T}_n + \mathbf{d}_n\,, \tag{3.7}$$

and obtain any point in the trajectory by recursively applying (3.7):

$$\mathbf{T}_n \leftarrow \mathbf{T}_0 + \sum_{j=0}^{n-1} \mathbf{d}_j \,. \tag{3.8}$$

The most straightforward candidate for 3.6 would be

$$\mathbf{d}_n \approx \mathcal{F}_0(\mathcal{S}_n) \equiv \mathbf{s}_{n,0} \,, \tag{3.9}$$

which would simply return the first prediction element of the model. This would be the ideal definition if we had a perfect estimator. Unfortunately, we do not have this. Hence, we wish to utilise more information to make a more effective prediction. Since we are predicting small trajectories of W steps at each iteration of our model, we have a level of redundancy in our data. For example, let us consider the estimation vector, $\mathcal{S}_n$, defined in 3.4. Taking the second term, we know that we can write this as

$$\mathbf{s}_{n,1} = \mathbf{s}_{n,0} + \mathbf{d}_{n,0} \,, \tag{3.10}$$

where $\mathbf{d}_{n,0}$ is the displacement between the first and the second step. Hence, $\mathbf{s}_{n,1}$ includes information about $\mathbf{s}_{n,0}$. Based on this, our idea is to use this relation to reduce noise and bias effects inherently present in our model's estimations, by aggregating multiple predictions in one single prediction. For a basic first-order correction, time-normalising the second step's prediction as $\frac{1}{2}\mathbf{s}_{n,1}$ and averaging leads to

$$\mathcal{F}_1(\mathcal{S}_n) \equiv \frac{\mathbf{s}_{n,0} + \frac{1}{2}\mathbf{s}_{n,1}}{2} \,. \tag{3.11}$$

to estimate the displacement instead of 3.9. The idea can be extended to include higher-order information as well. This leads to the definition of the W^{th}-order pre-

diction,

$$\mathcal{F}_W(\mathcal{S}_n) \equiv \frac{1}{W+1} \sum_{k=0}^{W} \frac{\mathbf{s}_{n,k}}{k+1} \,. \tag{3.12}$$

3.4.3 Further Intuition About The First Order Prediction Scheme

We will now try to write up an expression that allows us to build more intuition about the error of our 1^{st}-order prediction scheme. Let us examine the following prediction sequence. We can write the recurrence relation

$$\begin{aligned} \mathbf{T}_1 &\approx \mathbf{T}_0 + \mathcal{F}_1(\mathcal{S}_0) \,, \\ \mathbf{T}_2 &\approx \mathbf{T}_1 + \mathcal{F}_1(\mathcal{S}_1) \,, \\ &\vdots \\ \mathbf{T}_n &\approx \mathbf{T}_{n-1} + \mathcal{F}_1(\mathcal{S}_{n-1}) \,. \end{aligned} \tag{3.13}$$

To derive an expression for the error of our scheme we will assume we have a perfect model. This means that for the first two terms of the prediction $\mathcal{S}_n$:

$$\mathbf{s}_{n,0} = \mathbf{d}_n \,, \tag{3.14}$$

$$\mathbf{s}_{n,1} - \mathbf{s}_{n,0} = \mathbf{d}_{n+1} \,, \tag{3.15}$$

$$\mathbf{s}_{n,1} = \mathbf{d}_{n+1} + \mathbf{d}_n \,. \tag{3.16}$$

Substituting this in definition in (3.11) we get:

$$\mathcal{F}_1(\mathcal{S}_n) = \frac{3\,\mathbf{d}_n}{4} + \frac{\mathbf{d}_{n+1}}{4} \,, \tag{3.17}$$

Now we can rewrite the recurrence relation from (3.13) as

$$\begin{aligned} \mathbf{T}_1 &\approx \mathbf{T}_0 + \frac{3\,\mathbf{d}_0}{4} + \frac{\mathbf{d}_1}{4}, \\ \mathbf{T}_2 &\approx \mathbf{T}_1 + \frac{3\,\mathbf{d}_1}{4} + \frac{\mathbf{d}_2}{4}, \\ &\vdots \\ \mathbf{T}_n &\approx \mathbf{T}_{n-1} + \frac{3\,\mathbf{d}_{n-1}}{4} + \frac{\mathbf{d}_n}{4}. \end{aligned} \tag{3.18}$$

Back substitution gives us

$$\mathbf{T}_n \approx \overline{\mathbf{T}}_n \equiv \mathbf{T}_0 + \frac{3\,\mathbf{d}_0}{4} + \left(\sum_{j=1}^{n-1} \mathbf{d}_j\right) + \frac{\mathbf{d}_n}{4}\,. \tag{3.19}$$

Using (3.8), we can define the error of the first order scheme in step n with respect to the real trajectory as,

$$\epsilon_{1,n} \equiv \mathbf{T}_n - \overline{\mathbf{T}}_n = \frac{\mathbf{d}_0 - \mathbf{d}_n}{4}\,. \tag{3.20}$$

We conclude this is a shift of the trajectory with respect to the ground truth. More importantly, the error at each position in the trajectory remains stable, bound to the first and last displacement, and it does not accumulate. Especially for longer trajectories, as $\mathbf{T}_n$ grows large, $\mathbf{d}_0 \ll \mathbf{T}_n$ and $\mathbf{d}_n \ll \mathbf{T}_n$ so $\epsilon_{1,n}$ becomes negligible. This example was meant to create some intuition behind this form of integration of the predictions, therefore we worked out the simple 'first order' example, to reduce the mathematical complexity. However, this approach can be extended to the W^{th} order. In Section 3.5, we show empirically that using the full prediction order is beneficial for both the precision and accuracy of the trajectory estimation.

3.5 Results

In this section we show that global displacements in 3D space can be estimated by a simple convolutional neural network, using local pose information and acceleration signals from body mounted IMUs.

All the models discussed in this section were implemented in PyTorch/Lightning AI and trained using a Nvidia RTX A5500 GPU.

All experiments are performed using the same dataset, discussed in section 3.4.1. We split the dataset into three parts: a training set on which the model is fitted, a validation set used to monitor training performance as training progresses, and a test set to evaluate the fitted model's performance. The test set was about 10 % of the total dataset size and exclusively contained motions from actors that were not present in the training data. The remaining data was split 80/20 between training and validation sets.

3.5.1 Assessing Model Performance in 3D Trajectory Estimation and the Choice of Estimation Function

We found that the batch-wise L1 loss, while effective for training, falls short in measuring full trajectory estimation accuracy. This is because it overlooks the effect of biases in error estimation, which, unlike larger unbiased errors, can accumulate significantly over a trajectory. To address this, we incorporate Mean Squared Error (MSE) across the entire trajectory as a complementary metric:

$$e_{MSE} \equiv \frac{1}{3T} \sum_{t=0}^{T} \parallel p_t - \bar{p}_t \parallel^2 . \tag{3.21}$$

This accounts for drift effects which accumulate in the estimated trajectory. Additionally, analysing the distribution of estimation errors provides insight into potential biases and under or overestimation tendencies of our model. Finally we use visual inspections of the results, which are available as a supplementary video to this work.

In section 3.4.1 we discussed different methods of reconstructing a motion trajectory given the short mini-trajectories that our model estimates. To test these methods we reconstructed the root trajectories of the assets in our test dataset using equation 3.12, with

$$\mathcal{F}_w(\mathcal{S}_n), \quad w \in \{0, 1, 2, 3, 4, 5, 6, 7\}. \tag{3.22}$$

Figure 3.2 shows the effect on the e_{MSE} metric . A larger W has a positive impact on the model's prediction power, reducing the average e_{MSE} over all trajectories in the test dataset. Apart from a lower error, the predictions become more reliable, as witnessed by the lower variance, visualised with the blue shaded area in the figure. We conclude that this is because the final prediction used to increment the trajectory becomes a weighted average of several prediction values, helping to reduce bias and variance that would be present in the individual terms. In the remainder of this work we therefore use equation 3.12 with $W = 8$ to reconstruct root trajectories.

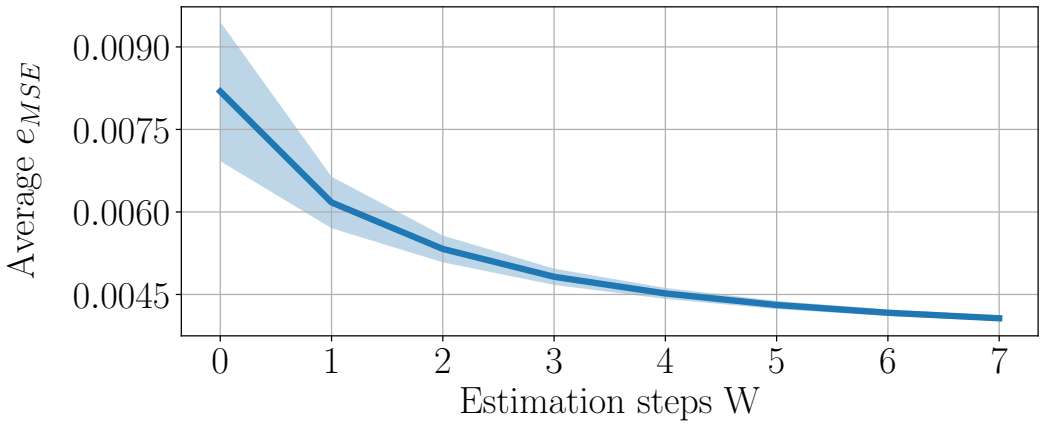


Figure 3.2: Average Errors in Trajectory Reconstruction: the figure shows the relation between the number of prediction steps, W, used to reconstruct the trajectories and the average value of e_{MSE} over all assets in the test dataset. The shaded area gives the variance. Note how both the average and variance decrease significantly for higher values of W.

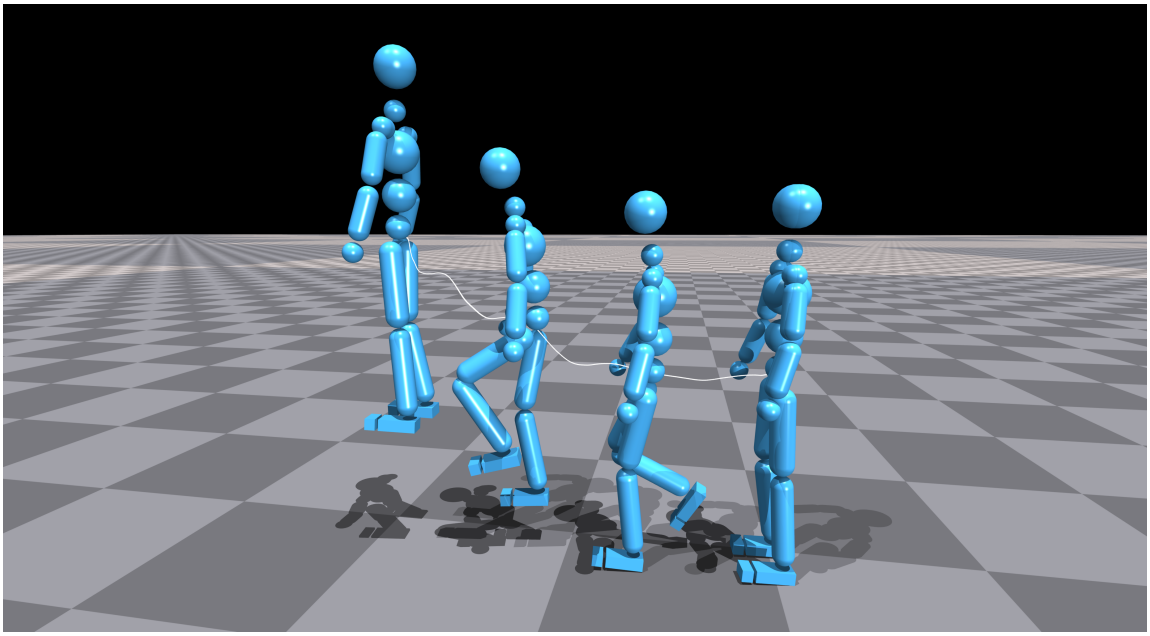


Figure 3.3: Reconstructed root trajectory: still frames from an animation of a character walking up stairs. The white line shows the reconstructed trajectory.

3.5.2 Evaluating Root Trajectory Reconstruction: Accuracy and Error Analysis

Figure 3.3 shows still frames of an animation where our method reconstructed the global root trajectory of a character walking up stairs. The full rendered animation is included in the supplementary video. The white line in the plot shows the root trajectory of the character. The character first moves forward in the horizontal plane, before ascending the stairs, indicating our model can distinct and switch between these modes of the motion.

In figure 3.4 we have plotted the trajectories for three types of motions. The corresponding animations are included in the supplementary video for a more profound and visual inspection of these results. The first two plots show a side view of a character walking up and down the stairs respectively. Our method effectively reconstructs the trajectory within an error margin of a few centimetres, faithfully

following the motion dynamics of the ground truth.

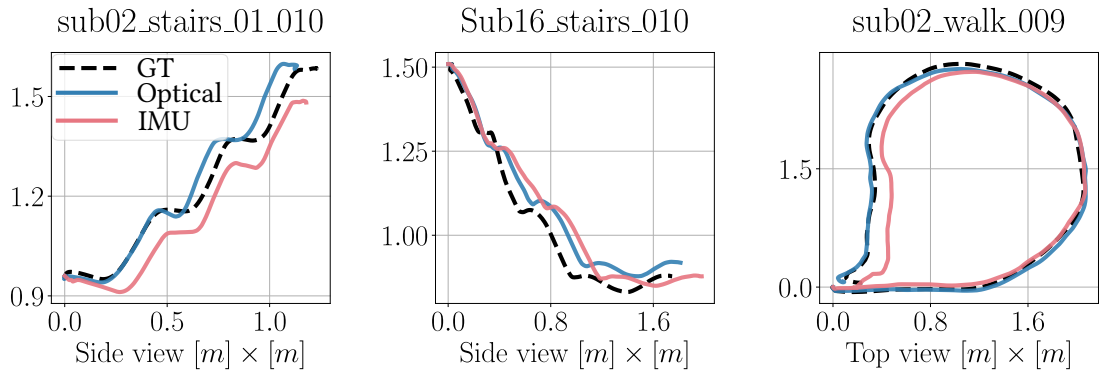


Figure 3.4: Plots of three reconstructed trajectories compared to the ground truth (dashed black). The blue line is estimated using high quality optical pose data, while the red line uses poses estimated from IMU signals. From left to right, a character walks up the stairs, down the stairs and in a counter-clockwise circle in the horizontal plane. The models both effectively track the ground truth trajectory, however slight drift effects are visible, especially when using IMU poses.

The last plot shows a top view of a character walking in a counter clockwise circle in the horizontal plane, while the predicted and ground truth trajectories are nearly perfectly matched. The entire trajectory is several meters long while the final misalignment error is at most a few centimeters. For more examples of trajectory plots, we refer to appendix 3.7.

We have not found any persistent tendencies of the model, such as structural over or underestimation, bias, etc. However, we do notice these tendencies on a per-actor basis. An example is subject 2, whose planar walks tend to show a slight downward drift, as exhibited by figure 3.5. This figure plots the height of the character per frame against the ground truth. We think that these actor specific effects are an indication that our training data lacks some diversity in terms of actors. This is important because there can be a large variation in sensor placement configuration from actor to actor.

The trajectory plots from figure 3.4 are only a selection of our model's performance on the test data. To give a better insight into the prediction errors,

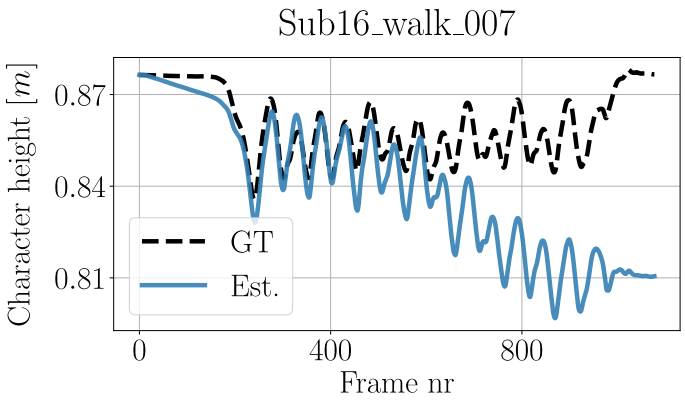


Figure 3.5: Height plot of a character walking in the horizontal plane. The model is capable of reconstructing the higher frequency dynamics of the motion, however it also introduces a low frequency downward motion, not present in the ground truth.

$e = estimate - target$, we plot their probability density functions (PDF) in forward, lateral and vertical direction in figure 3.6 (blue). The error distributions are narrow and centred around zero, indicating stable and accurate predictions of our model. The vertical direction shows a slight offset from zero, we argue that this is caused by the downward trend in a portion of the motions.

In conclusion, we observe that prediction errors, however small, can have great visual impact when they cause for example foot-skating. These artefacts are easily fixed in post-processing using a simple anti foot-skate filter which sets foot velocities to zero when they fall under a small threshold. An example of this is shown in the supplementary video.

3.5.3 IMU Only Data

In the previous section, we used real acceleration signals from body-mounted IMU sensors, while the pose data was taken from recordings with a high quality optical motion capture system. In this section, we will evaluate the effects on performance when training with poses estimated from the IMU sensor data. Our dataset of IMU

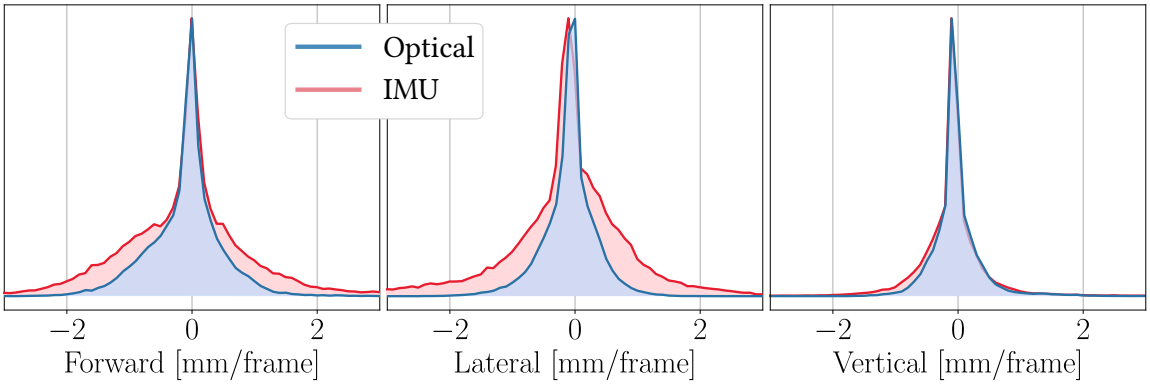


Figure 3.6: Estimation Error Probability Density Functions: The plot shows the PDFs of two models. Blue was trained using high quality pose data from an optical motion capture setup. Red uses poses estimated from IMU signals. We observe a larger spread of the forward and lateral errors when using the IMU poses, indicating a negative impact on the estimation accuracy.

poses was recorded synchronously with the previous dataset, so that the underlying motions contained in the training, validation and test sets are identical to those used in the previous section.

Upon evaluation of the test dataset, we report an average e_{MSE} of $\mu_{e,IMU} = 0.0266$ with a standard deviation $\sigma_{e,IMU} = 0.0768$. This is significantly higher than the values reported when using the optical pose data, which were $\mu_{e,opt} = 0.0041$ and $\sigma_{e,opt} = 0.0069$.

The red plots in figure 3.4 show ground truth trajectories estimated by this model. We observe that the displacements are accurate and close to those of the optical trained model, however, with some loss of accuracy. This is most apparent in the circular walk in the horizontal plane, where the model trained using IMU poses slightly undershoots the ground truth. This reduction of quality was to be expected due to the lower quality pose information. The effects are visible in the final motion, in the form of foot skating. Post processing this with our anti foot-skate filter fixes this in most of the cases. For an extensive qualitative review of this model's performance, including comparison with and without foot skating filter, we refer to the supplementary video.

These results are further confirmed by inspecting the (red) PDFs in figure 3.6, which shows the PDF of the estimation errors for the model trained on IMU poses and acceleration signals, together with the same distributions from the model trained with optical poses and IMU acceleration signals. Also here we see an increase in errors, as witnessed by the broader distributions.

3.5.4 Confirming the Importance of Acceleration Signals: Training on IMU Poses

We started section 3.4 by mentioning difficulties in past work with transferring learning from high quality pose data to lower quality IMU based pose data. In order to confirm the pivotal role that acceleration information plays in effective trajectory reconstruction for IMU based motion capture data, we trained a model exclusively on IMU pose information. The training data, targets and model configuration were all identical to the previous experiments, with the exception of the feature vector, which now is defined as exclusively the pose information

$$\mathbf{x} \equiv \left\{ \ \mathbf{p}_{joints} \ \right\} \in \mathbb{R}^{117}, \tag{3.23}$$

dropping the acceleration signals. Even though training converges, the resulting trajectory estimates massively undershoot the ground truth, as witnessed by figure 3.7 which shows the trajectory estimate and height plot for the same circular walk from figure 3.4. Note, how apart from underestimating the displacements, the network also struggles with determining the correct direction of motion and drifts significantly downwards.

3.6 Discussion and Conclusion

In this work we have presented a method that accurately and stably predicts motion trajectories from a combination of local pose information, and acceleration signals from body mounted IMU sensors. Our method was trained for two different data

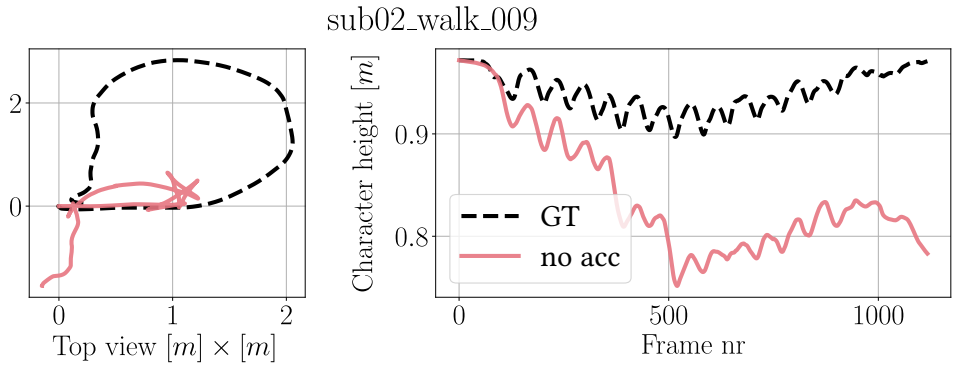


Figure 3.7: Trajectory and height reconstructions using only IMU pose data. The model heavily underestimates the displacements and displays a downward drift, underlining the importance of including acceleration information in the data. We also observe that at several points, the model struggles with finding the direction of motion altogether.

modalities: high quality pose data stemming from an optical motion capture setup, and lower quality pose data estimated from IMU sensors. Both models used acceleration signals from IMU sensors. Our models both accurately reconstructed the ground truth trajectories, although higher quality poses from optical motion capture equipment improved the model's estimation power.

The model predicts mini-trajectories of the character's root, consisting of 8 positions with respect to the character's current position. It makes it predictions from a window of pose and acceleration information of the same length. To improve estimation accuracy and optimally exploit the information present in the model's output, we presented an integration method that uses a weighted average displacement, estimated from these 8 positions.

Finally, we showed how using acceleration information is essential to ensure satisfactory performance when training on poses estimated from the IMU signals. Leaving this information out had a detrimental effect on the predictions of the model, rendering it incapable of reconstructing the root motion trajectories.

Our model showed some actor specific drift tendencies, which we suspect are

the result of a lack of variety in actors included in the data set. Furthermore, we note that the data used in this work was somewhat homogeneous in terms of the motions contained. Future work should focus on training on larger datasets, containing a bigger variety of actors and motion types. Additionally, future work could investigate normalising the sensor signals in such a way that they become less susceptible to differences in placement.

The type of data needed for the datasets used in this work is scarce and recording is tedious and expensive, as it requires the use of both optical and IMU based motion capture simultaneously. Another interesting direction for future work would be to investigate realistic synthesis of IMU like acceleration signals and pose information. This would enable the use of more widely available optical motion capture datasets.

3.7 Appendix A: Trajectory Plots

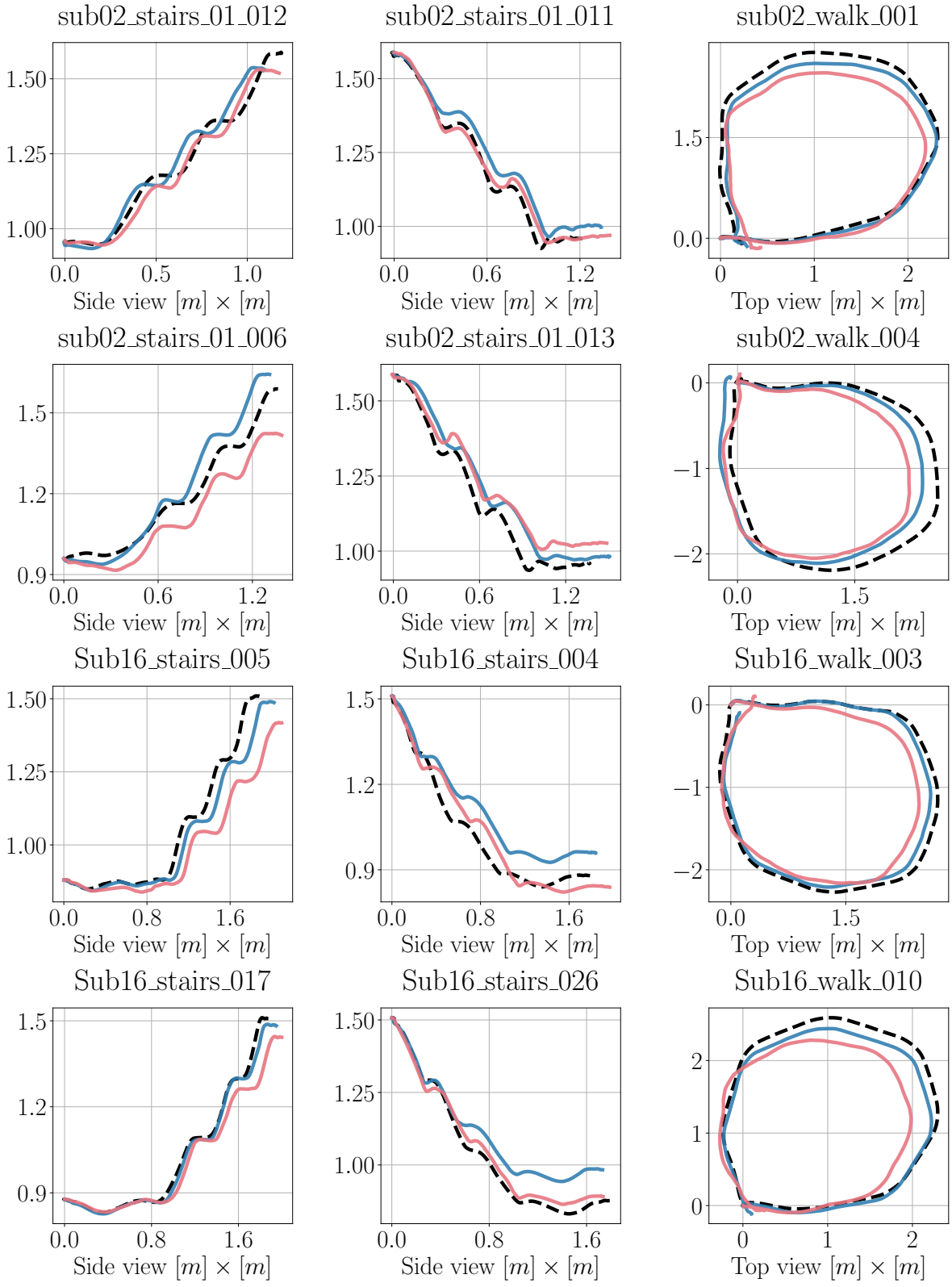


Figure 3.8: Additional trajectory plots as predicted by our model trained on optical pose data (blue lines), IMU pose data (red lines), and compared to the ground truth.

Chapter 4

ADAPT: Ai-Driven Artefact Purging Technique for IMU Based Motion Capture

---

This chapter presents the article:

Schreiner et al. (2024b)

Paul Schreiner, Rasmus Netterstrøm, Sune Darkner, Hang Yin, and Kenny Erleben. ADAPT: Ai-driven artefact purging technique for imu based motion capture. *Under review at Computer Graphics Forum*, 2024b,

The full manuscript is contained in this chapter.

4.1 Abstract

While IMU based motion capture offers a cost-effective alternative to premium camera-based systems, it often falls short in matching the latter's realism. Common distortions, such as self-penetrating body parts, foot skating, and floating, limit the usability of these systems, particularly for high-end users. To address this, we employed reinforcement learning to train an AI agent that mimics erroneous sample motion. Since our agent operates within a simulated environment, it inherently avoids generating these distortions. Impressively, the agent manages to mimic the sample motions while preserving their distinctive characteristics. We assessed our method's efficacy across various types of input data, showcasing an ideal blend of artefact-laden IMU-based data with high-grade optical motion capture data. Furthermore, we compared the configuration of observation and action spaces with other implementations, pinpointing the most suitable configuration for our purposes. All our models underwent rigorous evaluation using a spectrum of quantitative metrics complemented by a qualitative review. These evaluations were performed using a benchmark dataset of IMU-based motion data from actors not included in the training data.

4.2 Introduction

IMU-based motion capture is affordable and untethered, however, it lacks the quality of higher end marker based solutions. In this work, we present a physics-based, AI-enhanced method for improving motion data quality. The last decade has seen a rise in the popularity of IMU-based motion capture systems, making this technology common ground. It has proven itself as a reasonable alternative to marker based optical motion capture systems, in scenarios dictated by limited budgets or a need to capture 'in the wild'. Nevertheless, these systems lack the quality of high end marker based solutions. Physically implausible artefacts are common ground in the raw capture data. Examples of artefacts are self-collisions, foot skating, and, floating. For visual examples, see Figure 4.1.

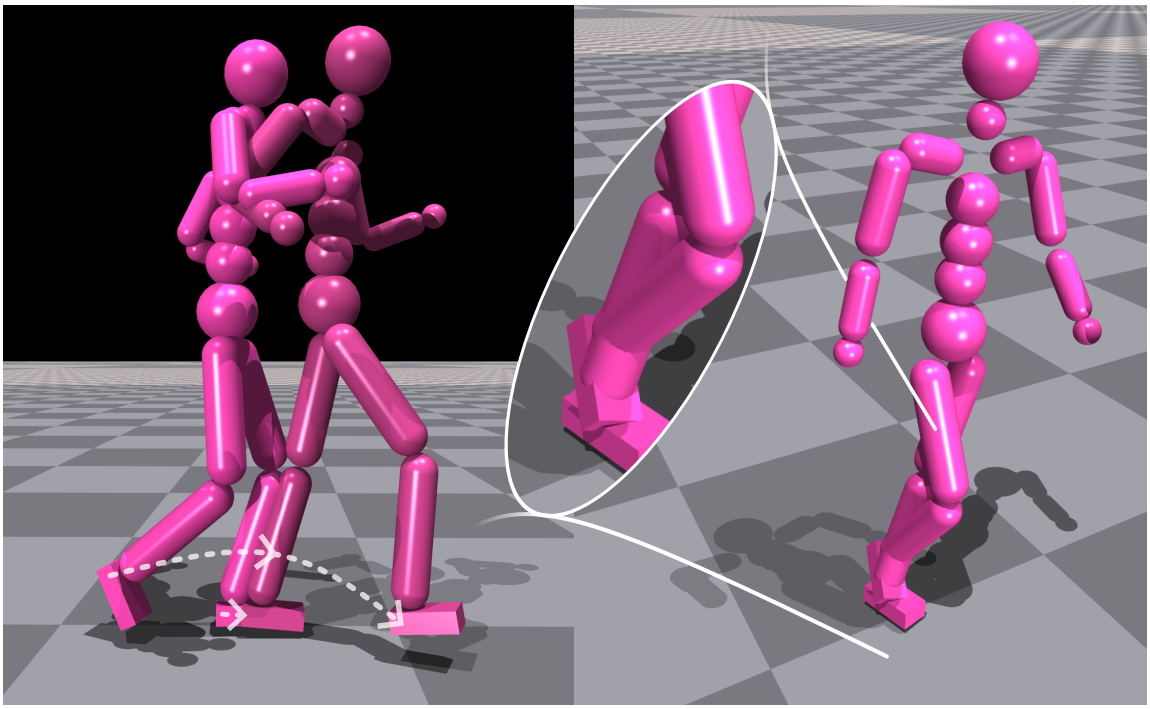


Figure 4.1: Examples of typical artefacts as seen in IMU-based motion capture recordings. On the left is an example of two frames from an animation clip where we see foot skating: both the foot in motion (rear foot), and the foot from the supporting leg, are displaced. On the right is an example of self-collision, where one leg is penetrating the other near the ankle.

In this work, we present a method for cleaning motion data while maintaining natural human-like motion qualities. We use a combination of physics simulation and learned AI behavior, through reinforcement learning. Our agent learns to mimic faulty sample motions and generalise this learning to motion data that was not seen during training. The core idea of our approach is that since our agent 'lives' in a physically accurate simulated environment it is not capable of reproducing errors that are in contest with the laws of physics, resulting in visually pleasing and physically plausible output motion. Our method builds on top of recent developments in the state of the art in this field. However, it distinguishes itself by its capability to operate on unseen motion data, stemming from unseen characters. One of our major

findings is the importance of the choice of data source when training the agent. We show that the key to a robust and versatile agent is training on a mix of high quality and faulty motion capture data from different sources. Agents trained solely on either high quality or faulty data lack the robustness to generalise to unseen data. Finally, we compare fundamental differences in the configuration of the observation and actuation spaces of our agent for our use case. We demonstrate an optimal configuration and its impact on the agent's capabilities to produce high quality output motion. In summary, the contributions presented in this work are:

- A physics-based framework for cleaning artefacts from motion capture data.
- A method that focuses on generalising to unseen IMU data, from unseen actors. This is in contrast to most other motion-mimicking methods, that draw their sample motions from known distributions.
- We show that mixing faulty and high quality training data is pivotal in generalising our method to unseen animation data.
- We show that the choice of observation and actuation configuration greatly impacts the agent's ability to learn and produce quality output.

4.3 Related Work

Imperfections are a common trait in real-world data, motion capture data being no exception. Inherent hardware limitations can cause missing or faulty sensor information. In the case of optical motion capture, common sources of error are missing information due to occluded markers or markers going out of view of the cameras. For IMU based motion capture, errors are mostly due to signal noise and bias, limitations in sensor precision, and resolution. These imperfections can cause a range of faults in the resulting animations, such as jittering limbs, self-collisions, foot-skating, and others.

Recognizing the challenges posed by these imperfections in motion capture data, both researchers and industry professionals have continuously sought efficient and

effective solutions to assist animators in the data cleaning process. While a significant portion of this cleaning is still manually performed, aided by a range of commercial software suites the quest for full automation remains an unsolved challenge. The following sections delve into some of the innovative approaches in this domain.

4.3.1 Non-Physics Based Methods

Existing techniques often focus on eliminating these artefacts using straightforward kinematic and/or geometric techniques (Pejsa and Pandzic, 2010). A shared trait by many of these methods is that they don't take the physics of the character into account, leading to unnatural poses and dynamic behaviour and they rely on post-clean-up techniques using IK solvers (Girard and Maciejewski, 1985; Bodenheimer et al., 1997; Choi and Ko, 1999; Lee and Shin, 1999; Kovar et al., 2002; Glardon et al., 2006; Lu and Liu, 2014). Recent deep learning approaches focus on robust detection of foot planting or global root position and apply IK as a post-clean-up process (Schreiner et al., 2021a; Mourot et al., 2022a). It is known that the IK clean-up is somewhat inferior in quality and that humans are extremely sensitive to even small foot sliding errors (Pražák et al., 2011).

Deep learning approaches have also been adopted to train generative models to directly produce believable dynamic behaviours, possibly taking a pre-clean-up sample in an encoding step for reconstruction. Care is often taken to the model design for effectively capturing the motion data statistics. Examples include MoGlow using latent flow models (Henter et al., 2020) and its variant encoding skeleton data with graph neural networks (Yin et al., 2021). Generative models demonstrate high performance in synthesizing complex human movements of various styles and rich contextual input (Valle-Pérez et al., 2021; Dong et al., 2020; Yin et al., 2023b). Physical constraints, however, tend to not be explicitly enforced but with an expectation of extracting them solely from the motion data. As a result, state-of-the-art such as diffusion models are often exploited to represent the behaviours to imitate (Tevet et al., 2023; Yin et al., 2023a), in combination with physically-grounded approaches reviewed below.

4.3.2 Deep Reinforcement Learning (DRL)

Ever since the 1980's physics simulation has been a topic within (interactive) character animation, steadily gaining traction over the past decades. A detailed overview of the early stages of this research was made by Geijtenbeek and Pronost (2012). More recently DRL for character animation has become an active field of research, as shown by Mourot et al. (2022b). The ability to incorporate physics makes DRL an attractive approach for motion data cleaning. DRL with physics simulations give these methods an inherent sense of realism: they allow for interactivity, and they can be employed online and in real-time. In this class of methods, an agent is trained to reproduce sample motions in a physically simulated environment. These methods can be subdivided into two sub-classes: methods that employ a phase variable to give the agent a notion of timing of the sample motion and those that directly supply the agent with a sample of motion data. Where the former can be used to learn single motions and general behaviour, the latter is more suitable to mimic diverse sets of sample motions closely.

4.3.2.1 Phase Variable Methods

A number of works successfully mimicked reference motion by relying on the use of a phase variable, providing the policy with information about the timing of the reference motion. With their groundbreaking work, DeepMimic, Peng et al. (2018) showed that an agent could learn to mimic complex behaviours while a user could set high-level task objectives, such as hitting a target when performing a spin kick. In follow-up work, Peng et al. (2021) used GANs to have an agent learn similar behaviour from unstructured data sets. Ma et al. (2021) improved sample efficiency using a set of constraints called spacetime bounds, effectively limiting the action search space. Inspired by NLP models, Peng et al. (2022) map a set of motions into latent motion embeddings. The authors then train a low-level policy to generate motions from these embeddings, using adversarial imitation learning. Subsequently, they train a high-level policy to complete new tasks by passing latent embeddings to the low-level policy.

In Xu et al. (2022) a differentiable physics simulator is used in combination with a new policy learning algorithm, short horizon actor-critic (SHAC), to improve training time. They showcase a speed up in training humanoid agents compared to the state of the art, reducing training time by a factor of up to 17x. Ren et al. (2023) proposes a method to directly learn a policy by back-propagating through a differentiable physics simulation, thereby speeding up the learning process and eliminating the need for reinforcement learning altogether.

While phase-variable methods are successful in training an agent to imitate single motions, general motion behaviours and styles, they are less suitable for use in our case where the requirement is to mimic a large variety of specific sample motions. This is due to the lack of information available to the agent during inference about the sample motions.

4.3.2.2 Increasing the Variety of Learned Skills

A more suitable approach for close reference tracking of diverse motion capture data is to replace the phase variable with a sample of the reference motion in the state information of the agent. The first to implement this idea were Chentanez et al. (2018), whose agent was trained on a large body of unstructured motion capture data. It was able to mimic a large variety of human motion tasks, even some from unseen data. While impressive in their versatility, the output motion quality does not compare to the phase variable methods earlier discussed. Different works have since tried to improve diversity in the agent's skill set and quality of the output motion. Multiple works focused on making their agent robust for in game use and interactivity. Park et al. (2019) and Bergamin et al. (2019) concurrently came with a two layer pipeline. They first generated kinematic motion samples from a user's control input and subsequently mimicked the motion through a RL agent. The latter step improves the generated motion by making it physically plausible, while the first allows for flexible high-level user control, perfect for in-game use. On top of that they enable interaction between the character and their environment. Similarly, Luo et al. (2020) proposes a framework for user control of quadrupeds, using GANs to

map high-level user control inputs, such as direction and velocity, to influence factors for primitive motor actions. The controller is then fine-tuned using deep RL to improve robustness. In Merel et al. (2020) the authors encoded latent task intentions from motion trajectories to learn interactive tasks such as catching a ball or carrying an object to a specified target. These works focused on real-time performance, for example for in-game use. Unlike our approach, they rely on a fixed body of underlying data during inference, which stems from the same data source and has been seen during training. This makes them less suitable for tracking arbitrary reference motions, which is a hard requirement for our case.

Other works focused on anatomical features of agents. Won and Lee (2019) and Lee et al. (2021) used parameterised controllers to accommodate for body shape variations on the fly or changing environmental and motion characteristics. Lee et al. (2019) do motion mimicking with a musculoskeletal model to study the effects of anatomical symptoms and prosthetics on locomotion. While they were effective for their purpose, the complex anatomical model makes the simulations and subsequent policy training computationally heavy and slow. Each of these contributions serves a clear purpose for each of their individual use cases, however they do not intend to solve the problem of generalising to unseen motion data while maintaining output motion quality and they do not address the topic of diverse data modality for training, which we found to be essential for generalisation to unseen IMU data.

More interesting for our objective are those works that explicitly focus on improving skill diversity. To this end, Won et al. (2020) designed a mixture of experts policy but they relied on high quality motion data. One of the few works that do report diverse data sources is Wang et al. (2020). They introduced constrained multi-objective reward optimisation to avoid domination of individual reward terms, a motion balancer to ensure a uniform distribution of motion classes during training, and adaptive policy variance control to avoid local minima. They demonstrate spectacular results in terms of versatility, robustness, and generalisability, even to unseen motion samples. However, their robustness seems to come at too high of a cost of motion quality for our purpose, specifically for unseen motion data.

Possibly most relevant to our work, Yuan et al. (2022b) used motion mimicking to

clean up physically implausible artefacts from motion data generated by a diffusion model. The core difference here is that the diffusion model and mimicking policy are both trained and evaluated on the same dataset. The policy subsequently never sees real unseen data during training, but rather samples drawn from the known distribution. In our work we focus on making our method robust towards unseen data, by combining different data modalities and a large observation space.

4.4 Method

Unique to our approach is the way we focus on generalising to unseen data, from unseen actors. We achieve this by mixing lower-quality and faulty IMU-based motion capture data with high-quality optical motion capture data. This is in contrast to other methods that usually rely on high-quality motion capture data, often from a single source.

Our method follows three general stages. During **data preparation** we first record a dataset of animations using IMU-based motion capture suits. These recordings are post-processed with industry-standard clean-up filters. We then select recordings such that the resulting data set contains a mixture of faulty and clean recordings (add a figure showing examples of faulty and non faulty assets?). Finally, we mix in a set of high-quality recordings from an optical motion capture source. We have compared this approach to using only IMU-based data and found this last step crucial in getting optimal performance on unseen motion data. Section 4.4.1 describes the data in more detail.

Next, during the **training stage**, our policy is optimised to mimic the presented sample motions. In a large parallel physically simulated environment, we collect a number of rollouts from our policy. For each rollout, we select a random sample asset and initialise it at a random frame in the animation. Rollouts have a maximum length of 300 frames. After collecting a fixed number of rollouts, we optimise our policy using the RL-games implementation of PPO from Makoviichuk and Makoviychuk (2021), as PPO is the de facto standard for policy gradient optimisation for this type of problem. We repeat this until the policy converges in terms of reward collection

and episode length. A general overview of the motion-mimicking formulation is given in Section 4.4.2. Choices for our agent's state observation and action spaces are described in Sections 4.4.2.2 and 4.4.2.1 respectively. Section 4.4.2.3 discusses the way we calculate the agent's rewards. Details about our policy and value network configuration are given in Section 4.4.2.4 and the remaining details on our training and simulation setup are discussed in Section 4.4.2.5.

In the last stage, our policy is **evaluated on unseen data** using a set of performance metrics that we found descriptive for our context. These metrics are the episode length ratio compared to the ground truth, the mean squared error between the sample and ground truth root trajectories, the survival rate of the agent and the ratio between achieved and maximum achievable reward. To this end, we collect 20 rollouts for each asset in our test data set and evaluate our model in terms of the performance metrics. We repeat this every 400 epochs to find the best policy from a training run. The performance metrics are described in detail in Section 4.4.2.6.

4.4.1 A Mixed IMU and Optical Mocap Data Set

For this project, we used both high-quality motion capture data from optical sources and IMU-based motion capture recorded using Rokoko's Smartsuit Pro II. This section describes the details concerning data collection, hardware used, and total numbers. For details about specific training datasets, we refer to Section 4.5.2.

Optical Data The optical data is sourced from a large, commercially available motion asset library. These assets have been recorded at various motion capture studios, using optical systems such as Optitrack and Vicon. The exact hardware is not known by the authors. All assets are recorded at 120 frames per second (fps) and are down-sampled to 60 fps. We used a total of 107 assets, containing a total of 80102 frames, in this project, which were exclusively used during training and not for testing.

IMU Data The IMU-based motion data was recorded on different occasions using Rokoko's Glove Ready Smartsuit Pro II. These suits contain 17 IMU sensors distributed over the body. The sensors record bone orientations, which are fitted to a body model through Rokoko's studio software to produce animations of a humanoid character. Since these sensors exclusively record orientations, the animations initially lack a sense of global placement. Through post-processing filters provided by the studio software, the animations are cleaned and augmented with global position estimates. The resulting animations still contain artefacts, such as self-collisions, foot-skating, and jittering limbs.

In this work, we use a total of 118 such recordings, recorded using 13 actors. The actors had various body shapes, dimensions, and genders. Of these recordings, 13 assets were exclusively used as test data. To ensure that the test result is representative of both cleaning abilities as well as the agent's ability to mimic motion, the test assets were carefully selected to contain both fairly clean and vivid animations as well as animations with typical flaws. The actors used to record these 11 test assets were kept out of the training data.

The IMU-based animations were initially recorded at 200 fps and subsequently exported to FBX files, sampled at 60 fps during training and inference.

4.4.2 Motion Mimicking for Artefact Clean Up

We aim to fix physical implausibilities in the mocap data, such as self-collisions, foot skating, floating, ground penetrations, and jitter. To this end, we train an AI-based agent that is capable of mimicking a given sample motion. The agent constitutes a physically plausible model of a humanoid and 'lives' in an environment, which is a physically simulated approximation of the real world. Hence the agent must adhere to common laws of physics while mimicking sample motions, prohibiting it from copying the above artefacts from sample motions.

The agent's joints are actuated by joint torques computed from PD controllers within the simulation environment. A policy $\pi(\mathbf{a}|\mathbf{s})$, constitutes the probability of an action $\mathbf{a}$ given the agent's current state $\mathbf{s}$. The actions in this setting are joint angle

targets for the PD controllers. After applying actions $\mathbf{a}$, the environment computes the agent's new state and a reward, r, based on how well the agent performed. See Figure 4.2 for a schematic overview of this process.

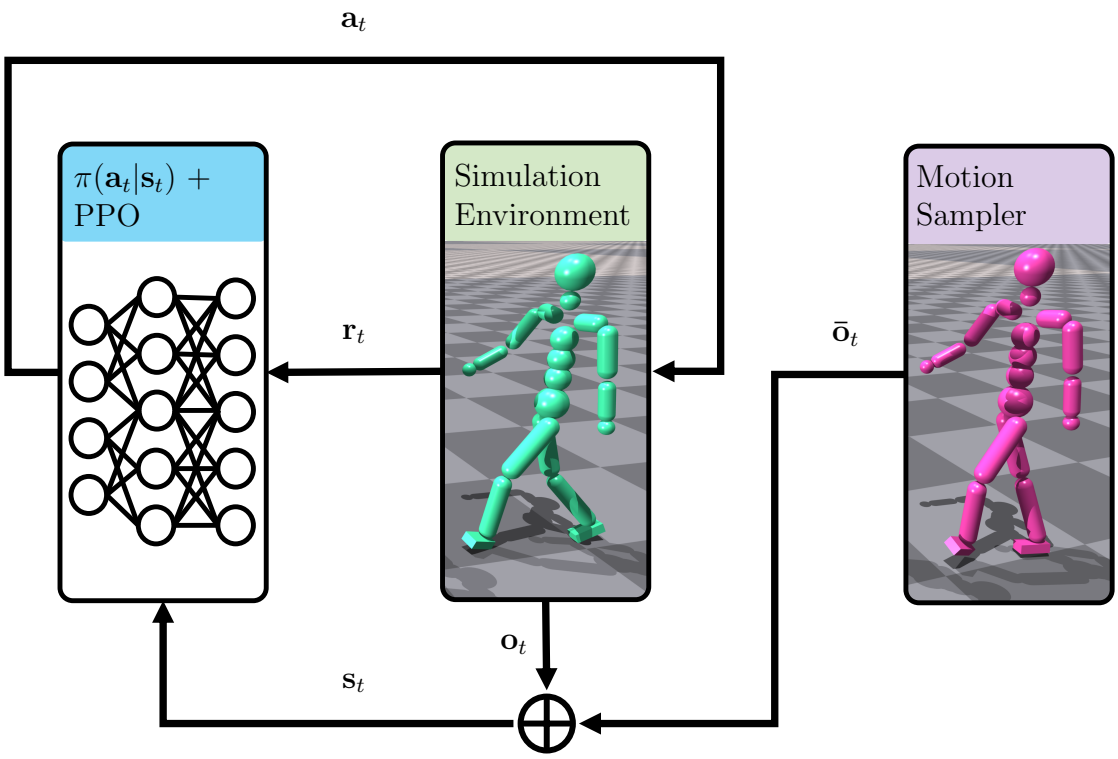


Figure 4.2: Schematic of the policy evaluation loop used to create rollouts from policy $\pi(\mathbf{a}_t|\mathbf{s}_t)$. The simulation and motion sampler both generate parts the state vector $\mathbf{s}_t$. The policy computes the most likely action given the state, which is applied to the character by the simulation environment. This results in a new state and reward. After a batch of rollouts is collected, the policy is updated using the collected rewards. We use PPO to optimise the policy.

4.4.2.1 State Observation

We use a skeleton containing 20 bodies, connected by 19 joints. The elbows and knees are modeled as 1-DoF hinge joints. All other joints are modeled as 3 hinge joints connected in series to achieve a 3-DoF rotational joint. The total number of

DoF is $13 \times 3\text{DoF} + 4 \times 1\text{DoF} = 43\text{DoF}$. Hand motion is not actuated, and therefore their joints are not counted for the total number of DoF. The pelvic bone is assigned to be the character's root.

The agent's policy gets information about its own state in the form of a state vector. Based on this it computes the actions for the next step. As discussed in Section 4.3, one key feature is that we want the policy to use the information on the desired state, as well as its own state, in order to enhance its ability to mimic arbitrary reference motions. Therefore we include information about the simulated character's motion and the reference motion to the state vector. In Table 4.1 we present our state vector $\mathbf{s}_t \in \mathbb{R}^{273}$ at time step t and its components.

Table 4.1: The agent's state observation vector. The state observation vector is composed of a number of key observations on the agent's state, stacked as a 1D vector of 273 elements.

Symbol	Dim.	Description
$\mathbf{v}_{com}$	$\mathbb{R}^3$	Center of Mass (CoM) velocity of the agent
$\bar{\mathbf{v}}_{com}$	$\mathbb{R}^3$	CoM velocity of the reference motion
$\boldsymbol{\Delta}\mathbf{v}_{com}$	$\mathbb{R}^3$	$\mathbf{v}_{com} - \bar{\mathbf{v}}_{com}$
$\bar{\mathbf{v}}_{hor}$	$\mathbb{R}^2$	The reference motion's velocity in the horizontal plane
$\boldsymbol{\Delta}\mathbf{v}_{hor}$	$\mathbb{R}^2$	$\mathbf{v}_{hor} - \bar{\mathbf{v}}_{hor}$
$\mathbf{p}_a$	$\mathbb{R}^{60}$	Positions of all 20 rigid bodies of the agent with respect to its CoM
$\mathbf{v}_a$	$\mathbb{R}^{60}$	Global velocities of all 20 rigid bodies of the agent
$\Delta\mathbf{p}_a$	$\mathbb{R}^{60}$	$\mathbf{p}_a - \bar{\mathbf{p}}_a$
$\Delta\mathbf{v}_a$	$\mathbb{R}^{60}$	$\mathbf{v}_a - \bar{\mathbf{v}}_a$
$\mathbf{a}_{t-1}$	$\mathbb{R}^{20}$	Smoothed actions from the previous time step (see section 4.4.2.2)
$\mathbf{s}_t$	$\mathbb{R}^{273}$	$\{\mathbf{v}_{com}, \bar{\mathbf{v}}_{com}, \Delta\mathbf{v}_{com}, \bar{\mathbf{v}}_{hor}, \Delta\mathbf{v}_{hor}, \mathbf{p}_a, \mathbf{v}_a, \Delta\mathbf{p}_a, \Delta\mathbf{v}_a, \mathbf{a}_{t-1}\}$

All velocity quantities are expressed in the global reference frame. The rigid body origin positions, $\mathbf{p}_a$, are expressed in the frame whose origin is attached at the agent's centre of mass (CoM) and whose axes are parallel to the global frame. To compute the difference between the agent's and the reference motion's rigid body positions, we compute the reference motion's rigid body positions $\bar{\mathbf{p}}_a$ in a similar frame attached to the reference motion's CoM. We then simply calculate the difference between the

two vectors: $\Delta \mathbf{p}_a = \mathbf{p}_a - \bar{\mathbf{p}}_a$.

The last entry of the state vector is the smoothed actions from the previous time step. This is done in accordance with Bergamin et al. (2019) to give the policy information about the smoothing process.

4.4.2.2 Action Space

In this section, we give a brief overview of our action space. For our implementation, we took inspiration from the work of Bergamin et al. (2019). We follow their implementation of the action space closely but with a few tweaks.

Our agent's joints are actuated by torques computed by PD controllers. All computations are on a per-frame basis, but in our notation, we omit the frame number for clarity.

$$\tau_d = k_p e_d + k_d \dot{e}_d \,, \tag{4.1}$$

$$e_d \equiv \theta_d - \tilde{\theta}_d \,. \tag{4.2}$$

Here θ_d is the current angle of degree of freedom d and $\tilde{\theta}_d$ is the target angle for that degree of freedom. The target angles are computed using the reference motion and a correction term from our policy:

$$\tilde{\theta}_d \equiv \bar{\theta}_d + \alpha_d a_d \,. \tag{4.3}$$

Here $\bar{\theta}_d$ is the angle of degree of freedom d from the reference motion at the current frame, which serves as a feed-forward open-loop action. Our policy computes the closed-loop action a_d based on the current state of the agent. Finally, α_d is a fixed binary operator that can be either 0 or 1, thus excluding certain degrees of freedom from closed-loop actuation. We set $\alpha_d = 1$ for the degrees of freedom of a set of key joints:

$$\begin{aligned} \{&\textit{Right shoulder, Right hip, Right knee (1D), Right ankle,} \\ &\textit{Left shoulder, Left hip, Left knee (1D), Left Ankle}\} \,. \end{aligned} \tag{4.4}$$

In contrast to Bergamin et al. (2019), we found that actuating any of the spinal elements in a closed-loop occasionally causes unnatural spinal wobbling and therefore we excluded all spinal elements from the closed-loop actuation.

In order to avoid high-frequency oscillations in the control signal, we further follow the example from Bergamin et al. (2019) and perform a smoothing operation on our closed-loop action signals:

$$\mathbf{a}_t \leftarrow \beta\ \mathbf{a}_{raw,t} + (1-\beta)\ \mathbf{a}_{t-1}\,. \tag{4.5}$$

Here $\mathbf{a}_t$ are the smoothed actions at time t, $\mathbf{a}_{raw,t}$ are the actions generated by the policy, and β is a smoothing factor. For details and a justification for the value of this parameter, we refer to the original work. We use $\beta = 0.2$ as reported by the authors for all experiments in this work.

4.4.2.3 Rewards

Our agent is rewarded based on the similarity between the simulated character state and the sample motion state. We use a compound reward based on the similarity between the simulated and the sample's joint angles, positions of bone landmarks, and bone landmark velocities. The reward scaling and threshold parameters discussed further in this section are listed in Table 4.2.

Table 4.2: Reward scaling parameters and threshold parameters. The weights were empirically chosen to ensure different reward terms contribute equally to the total reward.

Parameter	α_{local}	α_p	α_v	$\epsilon_{discount}$
Value	2.5	1.0	0.1	0.025

Local Pose Reward We calculate a local pose reward based on the sum of the local joint angles for the J joints of the skeleton, using:

$$r_{local} \equiv \exp\left(-\alpha_{local}\frac{1}{J}\sum_{j=0}^{J} \parallel \mathbf{q}_j \ominus \bar{\mathbf{q}}_j \parallel_q\right). \tag{4.6}$$

Here $\mathbf{q_j}$ represents the rotation of the j-th joint in a local frame attached to the joint's origin and whose axes are fixed to the joint's parent body. The operator $\ominus$ indicates the difference between two quaternions and $\parallel \cdot \parallel_q$ indicates the angle of the enclosed quaternion. J is the number of joints. The $\bar{}$ indicates ground truth. The parameter α_{local} is an empirically chosen weighing factor for the local joint angle reward.

Position Reward The position and velocity rewards are calculated using the positions and velocities of landmarks on the simulated character and the sample motion skeleton, much like Bergamin et al. (2019). These landmarks are the 6 face centres of unit cubes mounted at each bone's origin. This is done to account for both positional and rotational errors of the respective bodies. The position error is calculated as:

$$r_p \equiv \exp\left(-\alpha_p\frac{1}{J}\sum_{j=1}^{J}\sum_{k=1}^{6} \parallel \mathbf{p}_{jk} - \bar{\mathbf{p}}_{jk} \parallel_2\right). \tag{4.7}$$

Here $\mathbf{p}_{jk}$ and $\bar{\mathbf{p}}_{jk}$ are the positions of face centre k of the j^{th} body with respect to the root bone of the simulation and sample skeletons respectively. The weighing factor α_p is empirically chosen.

Velocity Reward The velocity reward is computed analogously to the position reward, except that we now use the norm of the difference between the face centre velocities:

$$r_v \equiv \exp\left(-\alpha_v\frac{1}{J}\sum_{j=1}^{J}\sum_{k=1}^{6} \parallel \mathbf{v}_{jk} - \bar{\mathbf{v}}_{jk} \parallel_2\right). \tag{4.8}$$

Finally, the entire reward is discounted based on a deviation discount, again similar to Bergamin et al. (2019). This discount is calculated as a difference between the head position of the sample motion and the head position of the simulated character, both in a global reference frame. The discount factor serves as a correction on the reward, in case the simulated character falls behind on the sample motion.

$$e_{\text{discount}} \equiv \text{clamp}(1.3 - 1.4 \parallel p_{head} - \bar{p}_{head} \parallel_2, 0, 1) \,. \tag{4.9}$$

The resulting reward function is:

$$r \equiv e_{\text{discount}}(r_{local} + r_p + r_v) \,. \tag{4.10}$$

Finally, we terminate the episode in case the agent either falls behind beyond a given threshold, $e_{discount} < \epsilon_{discount}$ or when the agent has fallen. Falling is detected as 3 or more spinal elements being in contact with the ground plane at the same time. The termination threshold is set to $\epsilon_{discount} = 0.025$.

4.4.2.4 Policy and Value Networks

Our policy and value functions are approximated using neural networks, both with an identical architecture. We follow the approach of Yuan et al. (2022b) and use multi-layer perceptrons (MLPs) with 3 fully connected layers of size [1024, 1024, 512]. At each layer output, we use *tanh* activation functions as suggested by Bergamin et al. (2019). The policy network estimates the mean of a normal distribution. During training, the agent's actions are sampled from this distribution, using a fixed variance. This variance determines to a high degree the amount of exploration an agent performs. We found empirically that a value of $\sigma^2 = 0.03$ gave a good balance between exploration and stable training. Table 4.3 gives an overview of these and other hyperparameters and simulation parameters.

4.4.2.5 Training & Simulation

We trained all our agents using NVIDIA's Isaac Gym. This choice was driven by Isaac Gym's ability to do massive, GPU accelerated parallel simulations, reducing training times to a minimum. As a reinforcement learning algorithm, we use PPO (Schulman et al. (2017)), as it is the de facto standard for these types of tasks as can be seen in Bergamin et al. (2019); Peng et al. (2018, 2021). We use the PPO implementation from RL-games by Makoviichuk and Makoviychuk (2021), which comes shipped with Isaac Gym. The training was performed on an NVIDIA RTX A5500 with 24GB of memory. For more information about our training and simulation configuration, we refer to Table 4.3.

Table 4.3: Training and simulation parameters for our motion mimicking agent's policy optimisation.

Parameter	Value	Parameter	Value
Discount factor γ	0.99	PPO clipping ϵ_{clip}	0.2
GAE τ	0.95	Batch size	65536
Episode horizon	16	Minibatch size	16384
Number of environments	4096	Mini epochs	6
Network layers	[1024, 1024, 512]	Simulation time step	0.0166 s (60 Hz)
Activation	tanh	# of substeps	2
Learning rate	$5 \cdot 10^{-5}$	Control frequency	30 Hz
Distribution variance σ^2	0.03	Episode length	300

4.4.2.6 Performance Metrics

To determine performance on unseen motion data, we use four metrics discussed in this section. The metrics are effective on the condition that the ground truth assets all have the same length, as only then does it make sense to calculate statistics on episode length ratio, root displacement errors, survival rate, and maximum achievable rewards.

We discuss the metrics here as they are computed per rollout. For the evaluation

of policies, we then calculate mean and standard deviations for each metric, after evaluating them on a test dataset of unseen motion assets.

Episode Length Ratio The episode length ratio is defined as the length of rollout i, divided by the length of the ground truth motion asset (gt):

$$ELR_i \equiv \frac{\text{length}(rollout_i)}{\text{length}(gt_i)} \,. \tag{4.11}$$

This metric provides information about the ability of a policy to mimic sample motions well enough to not terminate early. If the metric is one, then the rollout was not terminated early; if the metric is zero, it does not mimic a single pose.

Root Displacement Error This metric measures the mean squared error (MSE) between the root trajectories of rollout i and its ground truth motion asset:

$$RDE_i \equiv \frac{1}{N} \sum_{n=0}^{N-1} \parallel \mathbf{p}_{i,n} - \bar{\mathbf{p}}_{i,n} \parallel_2^2 \,. \tag{4.12}$$

We use N for the number of frames of the rollout. The RDE provides information about the accuracy with which the policy tracks the ground truth motion trajectory.

Survival Rate .9 The survival rate measures the probability of a policy successfully mimicking a given asset from the start until a given percentage of the ground truth length. The ".9" means the probability of an agent surviving 90% of the given ground truth asset. If the metric is one, the agent survives 90% or more of the given asset.

$$SR_i \equiv \begin{cases} 1 & \text{if } len(rollout_i) \geq len(gt) \cdot 0.9 \\ 0 & \text{else} \end{cases} \,. \tag{4.13}$$

Max Reward Ratio The max reward ratio measures the fraction of the maximum cumulative reward an agent achieves over rollout i:

$$MRR_i \equiv \frac{R_i}{R_{max,i}}\,. \tag{4.14}$$

The cumulative reward is calculated as:

$$R_i = \sum_{n=0}^{N-1} r_n\,. \tag{4.15}$$

For a rollout with length N and using Equation 4.10. For the maximum cumulative reward, $R_{max,i}$ we use $N = length(gt_i)$ and $e_{discount} = r_{local} = r_p = r_v = 1.0$. Note that due to using exponential rewards and the choice of weights, achieving the maximum reward is rather unlikely and in practice the score for this metric falls well below 1.0.

4.5 Results

In this work, we hypothesise that a humanoid agent, trained using reinforcement learning, is capable of fixing common physically implausible motion artefacts in unseen IMU-based motion capture data. We seek to find answers to questions, such as what data mixture is optimal for training, and which configurations of the state and action spaces result in the most robust agent. In this section, we discuss the experiments and their results that provide answers to these questions.

4.5.1 Unseen Data Study

To determine how well our method generalises to unseen data, we tested our agent's performance on a set of motion data that was not present during training. The test data consists of 15 IMU-based recordings from four different actors, with different body types and genders. The recordings were split into 27 segments of 300 frames long. This was done to mitigate the effects of failure due to specifically difficult

Table 4.4: The results of our method, in bold, compared for models trained on different training data and configurations of the observation and action spaces. The models are compared based on the ratio of the episode lengths and the original asset's lengths, the fraction of episodes that reached at least 90% of the original asset length, the MSE of the root trajectories per frame in m, and the ratio of the obtained and maximum obtainable reward.

Experiment	Episode length ratio μ_{el}	Episode length ratio σ_{el}	SR_{90}	Root MSE / frame μ_{mse}	Root MSE / frame σ_{mse}	Max reward ratio μ_r	Max reward ratio σ_r
IMU + Optical (50 assets, 39270 frames)	**0.969**	**0.081**	0.889	$\mathbf{5.118 \cdot 10^{-05}}$	$\mathbf{4.789 \cdot 10^{-05}}$	**0.400**	**0.114**
IMU large data (107 assets, 80102 frames)	0.961	0.120	**0.907**	$6.386 \cdot 10^{-05}$	$5.639 \cdot 10^{-05}$	0.371	0.111
IMU (33 assets, 28925 frames)	0.889	0.191	0.757	$6.949 \cdot 10^{-05}$	$5.882 \cdot 10^{-05}$	0.386	0.121
Optical, large data (103 assets, 68392 frames)	0.782	0.248	0.469	$1.424 \cdot 10^{-04}$	$1.107 \cdot 10^{-04}$	0.257	0.070
Optical (17 assets, 10345 frames)	0.468	0.212	0.017	$2.132 \cdot 10^{-04}$	$1.325 \cdot 10^{-04}$	0.329	0.095
Configuration Drecon Bergamin et al. (2019)	0.942	0.141	0.885	$5.863 \cdot 10^{-05}$	$5.598 \cdot 10^{-05}$	0.372	0.110
Conf. full state & action space	0.924	0.168	0.839	$6.251 \cdot 10^{-05}$	$7.980 \cdot 10^{-05}$	0.273	0.099

movements in long recordings, which would otherwise render large parts of the test data inaccessible. This was justified by the fact that we wanted to test the agent for overall capabilities and not so much for surviving for the longest possible amount of time.

The first row of Table 4.4 shows the performance in terms of our metrics. For each of the 27 segments, 20 rollouts were collected all starting at slightly different timings. The metrics were calculated on a total of 540 rollouts. The standard deviations in the table give a measure of how much the mean values differed between rollouts.

Figure 4.3 is a visual comparison of a ground truth asset with heavy self-collisions, and the same asset reproduced by our agent. The agent manages to faithfully keep key features and details of the motion intact, while not exhibiting any self-collisions. Our method also improves the CoM position of the character: in the last frame of the sequence, the ground truth character can be seen to lean unnaturally to a side, causing the projection of the CoM on the floor to lie outside of the base of support of the feet. This is unusual for human motion and would cause the character to risk losing balance in a real-world setting Hof et al. (2005). In contrast, our character has a more natural, upright stance, causing the CoM projection to be within the base of support. Our agent was also successful in improving the root motion of

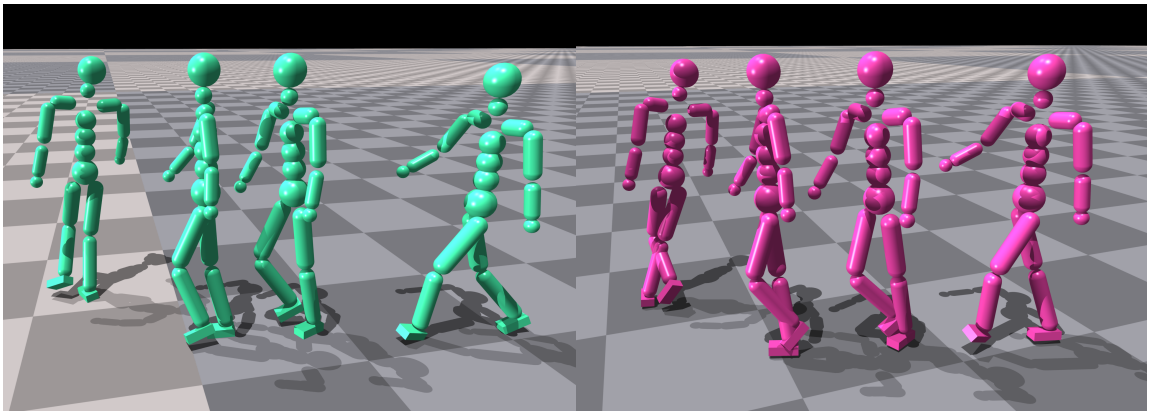


Figure 4.3: Collision fixing on a flawed animation. The image shows still frames from an animation before and after processing by our method. The ground truth (pink) displays self-collisions at the legs in 3 of the still frames while the agent (green) does not. The agent also maintains a more natural pose throughout the animation, while the ground truth leans in unnaturally in the last still frame, placing its center of mass far outside of its foot base.

unseen ground truth animations that were flawed. In figure 4.4 we demonstrate an example, where the ground truth contained heavy foot skating and a generally unstable CoM trajectory. Our agent maintained a smoother CoM trajectory while eliminating foot skate. For full visualisations of these and other examples, we refer to the supplementary video which contains full side-by-side recordings of our agent with the ground truth.

4.5.2 Data Ablation Study

A question we sought to answer was to what extent the type and mixture of the training data impact the performance of the agent. Generally speaking, optical motion capture data is high quality and low on noise while it is confined in recording space. IMU data, on the other hand, is easy to record, untethered, and low cost, but this comes at the expense of data quality and high noise. We trained models on different mixtures and sizes of datasets and evaluated them on our performance metrics. The following 5 datasets were compared:

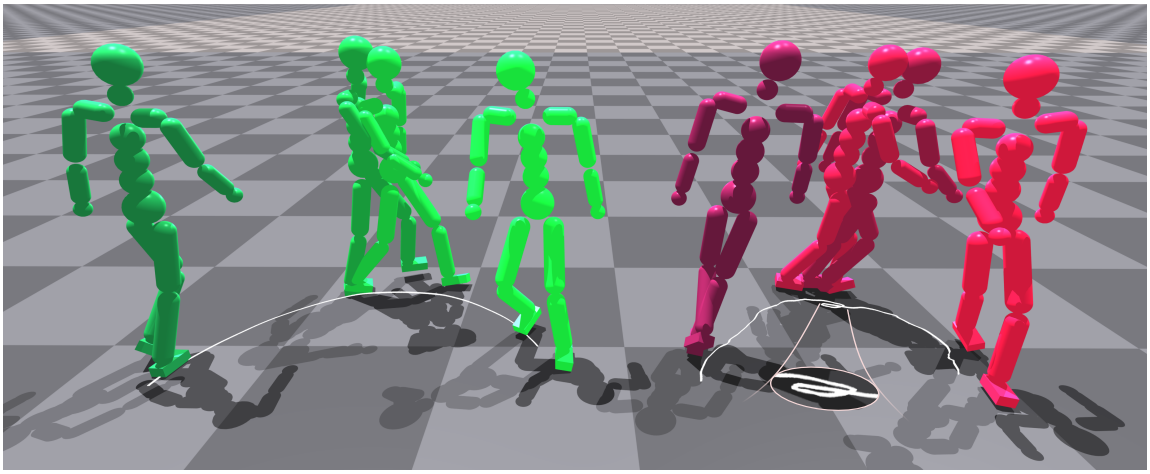


Figure 4.4: A visualisation of our agent (green) applied to an animation containing foot skating (pink/red). The color gradient represents the flow of time, where brighter is later in the sequence. The white lines are the character's centers of mass (CoM) projected onto the horizontal plane. Note how in the ground truth the character floats backward between frames 2 and 3, while our agent follows a much smoother CoM trajectory.

1. **IMU:** This dataset contained 33 motions recorded with an IMU based motion capture suit, totaling 28925 frames at 60 frames per second. Some assets contained artefacts common to IMU setups, such as jittery limbs, self-penetration and foot skate.

2. **Optical** This dataset contains 17 high-quality motion assets recorded with high-end optical motion capture hardware. It contained a total of 10345 frames at 60 frames per second.

3. **IMU + Optical:** This dataset is the combination of the above two datasets (50 motion assets, 39270 frames at 60 frames per second).

4. **IMU Large:** To rule that the IMU+Optical agent outperformed the other agents on sheer data size, we also trained the single source agents on larger datasets. The IMU Large dataset contained 107 assets, with a total of 80102 frames of motion data.

5. **Optical Large:** This dataset contained 107 high-quality optical based motion assets with a total of 68392 frames.

Each agent was trained for approximately 35000 epochs. To get an indication of the agent's general performance the agents were evaluated with our test metrics on a dataset of unseen IMU-based motion assets. The actors recorded in this dataset were not present in the training data. The results for the best scoring epoch for each model can be found in table 4.4. The results for all epochs are plotted in figure 4.5, except for the agent trained on small optical data, as it did not manage to reproduce most of the motions in the test dataset.

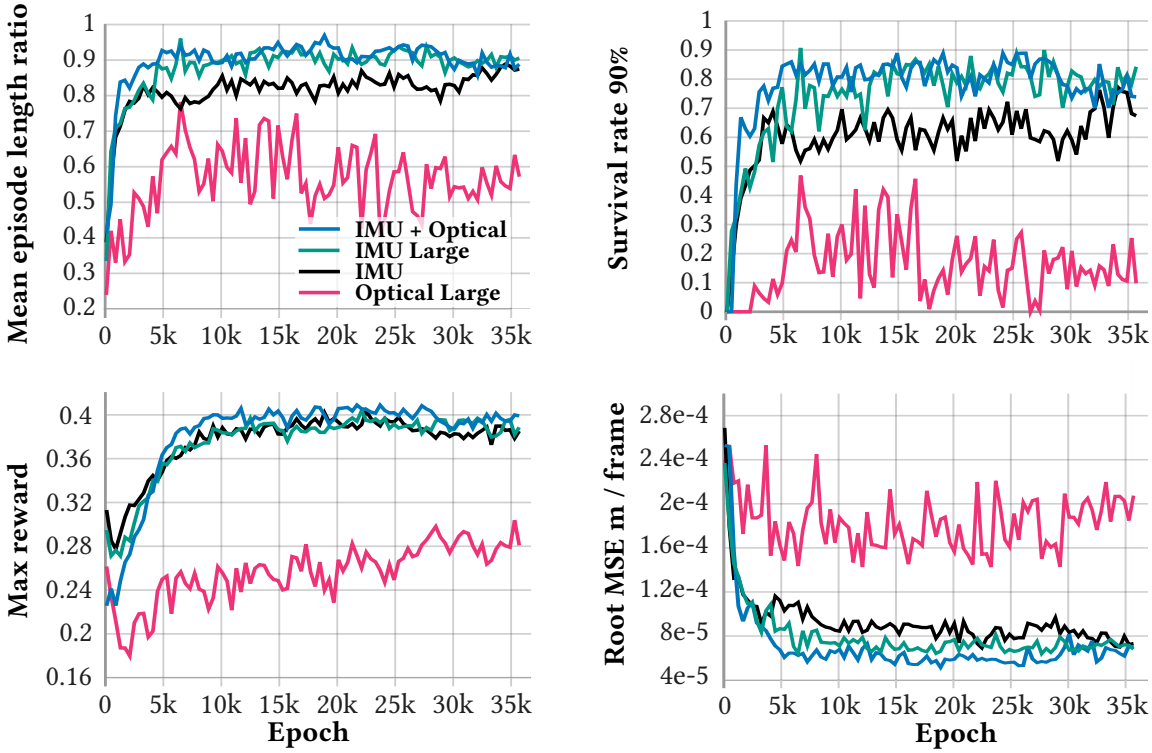


Figure 4.5: Effects of training data modality in terms of performance metrics. The model trained using the IMU+Optical dataset outperforms other dataset configurations on all metrics, except the 90% survival rate. The model trained exclusively on optical data scores lowest on all metrics.

The agent trained on a mixture of IMU and optical-based data outperformed the

other agents, including those trained on larger bodies of motion data. It held the highest score for episode length ratio, max reward ratio, and root MSE per frame, while the IMU Large agent scored slightly better on the 90% survival rate metric.

This led us to conclude that introducing different types of data to the training dataset is beneficial and preferable over simply increasing the dataset size. Our combined IMU and optical motion capture dataset was about half the size of the IMU Large dataset, yet still performed better. This superior performance is rooted in two key factors: On the one hand, introducing different data types makes the agent more robust to different types of input data, which is crucial when evaluating unseen data. On the other hand, the high-quality motion data aids the agent in compensating for flaws in the IMU data.

The agent trained solely on optical motion data exhibited poor performance. The likely reason is a substantial change in the input data distribution across different motion capture sources. Consequently, training on one source and inferring on another is ineffective.

4.5.3 Configuration Ablation Study

In order to find the most robust agent for inference on unseen data, we compared different state and action configurations for our agent. In Bergamin et al. (2019) the authors showed how a reduced state and action space was beneficial for their case (for simplicity called the Drecon configuration here). However, we argue that a configuration with a similar action space but with state information on all bones/joints is beneficial for robustness against unseen data. We therefore tested their proposed configuration against the configuration described in the method section of this work. Finally, we tested whether using the full action space would be beneficial for our case. Figure 4.6 shows six frames of an animation comparing all three configurations as well as the ground truth. While our agent (the green character) faithfully follows the reference motion, the Drecon configuration falls (from frame 4) after a sharp turn. The agent with full state and action space manages to perform the full motion, but introduces blandness to the motion, as witnessed, for example, by the

lesser pronounced arm motion in frames 2, 4, 5 and 6.

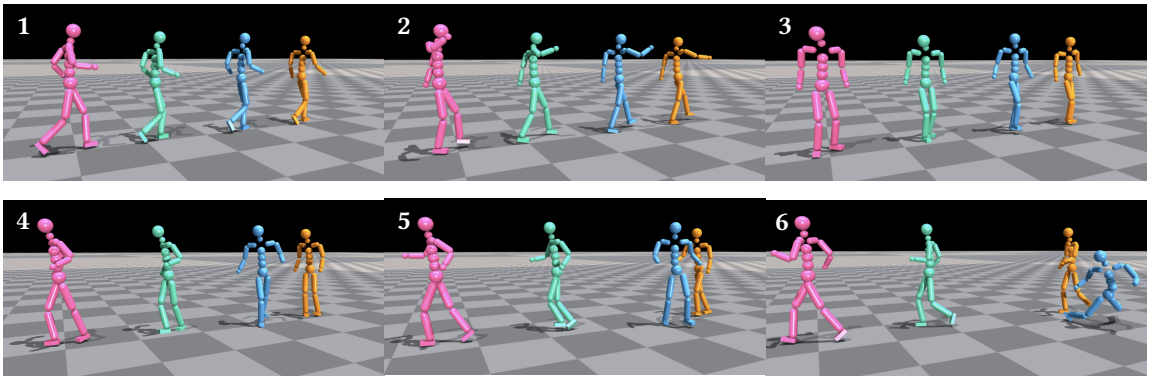


Figure 4.6: A comparison of performance between configurations: The pink character is the ground truth, green is our configuration, blue has reduced state and action spaces, and orange has full state and action spaces. Note how the blue character falls in frames 5 and 6, while the orange character adds an increased blandness to the motion (frames 4 and 5).

Both the Drecon configuration and the full action space configuration were tested against our performance metrics, using the same test dataset as in the previous comparisons. The metric scores of the best epochs for this comparison can be found in the lower section of Table 4.4. The metrics over all training epochs are plotted in Figure 4.7.

Our agent trained with full state information and reduced action space (top row, IMU + Optical) performed best on all metrics, closely followed by the Drecon configuration. We attribute this advantage to the agent having more information to go on than the Drecon agent, adding to its robustness against unseen data.

We confirmed the findings from Bergamin et al. (2019) that using the full action space is not beneficial and that it does not contribute to the case of unseen data. This is due to the problem becoming unnecessarily complex for the policy, leading to reduced motion quality, as witnessed by the lower max reward ratio. This is further confirmed visually in the supplementary video.

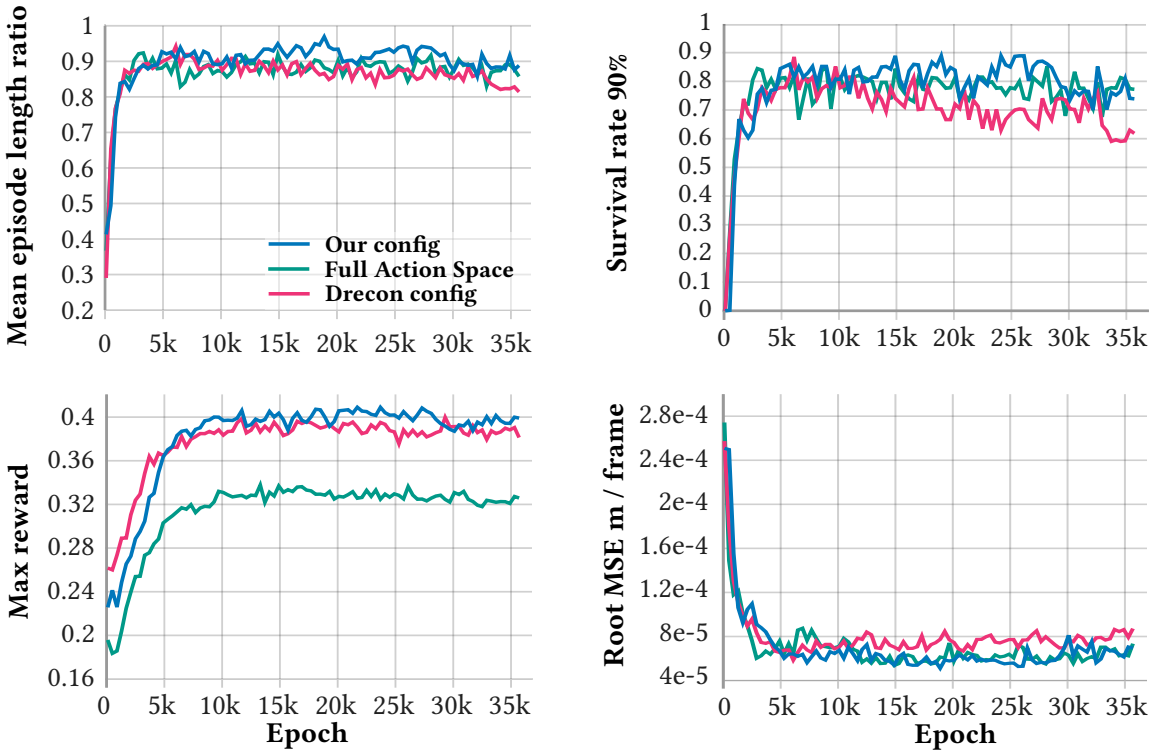


Figure 4.7: Effects of state and action space configurations in terms of performance metrics. The model trained using our full state and reduced action space outperforms other configurations on all metrics, closely followed by the Drecon configuration from Bergamin et al. (2019).

4.6 Discussion and Conclusion

We demonstrated how motion mimicking using reinforcement learning can be effectively deployed to purge faulty motion data from physically implausible artefacts. Our agent was evaluated on a test dataset of IMU based motion capture recording from multiple actors, that were not included in the training data. It could reproduce sample motions faithfully and in their entirety in almost 90% of the cases while fixing artefacts such as self-collisions, foot-skating, and jittery limbs. Moreover, it produced more realistic centre of mass trajectories.

For training of our agent we used a mixture of high and low quality motion data

from different sources. This mixture benefited the performance of our agent, making it more robust to unseen data. Even when the homogeneous dataset was larger then the mixed dataset, training with the mixed dataset yielded better performance. Training exclusively on high quality motion data did not produce an agent that was capable of generalising to unseen IMU based data.

Finally, we proposed a configuration of the observation and action spaces that improved our agent's performance on unseen data. The agent had access to information about all its joints, while it could only actuate a set of key joints. We compared this configuration to other configurations, one where both action and observation spaces were reduced, and one where the agent had access to the full observation and action space. Our configuration showed the best results during validation on unseen data.

As a final note, we observe that we expect our agent to solve two contradicting problems: on the one hand the agent is incentivised to mimic motion closely through joint angle and position rewards, while on the other we want it to deviate from close mimicking when there is unnatural phenomena such as self-collisions. This nuance is not explicitly represented in the reward setup, occasionally causing our agent to try and mimic artefacts as close as possible without breaking the laws of physics. This causes undesirable behaviours in the output motion, such as stumbling or intermediate steps. As a direction for future work, one proposed path to solving this contradiction is combining conventional rewards with a reward from a discriminator network, like the one proposed in Peng et al. (2021), to reward the agent on a global style, rather than exactly mimicking the reference motion.

Chapter 5

Summary and Discussion

This chapter summarises the contents of this thesis. It starts with a summary of the articles from chapter 2 and 3, includes their main conclusions and proposes suggestions for future directions. This chapter concludes with a similar reflection on the work discussed in 4.

5.1 Global Root Trajectory Reconstruction

As discussed in section 1.2, global root trajectory reconstruction is necessary for IMU based motion capture. It is not feasible to measure it directly from the sensors without accumulation of errors that would result in large and unbounded drift of the estimates. In chapter 2 a method was presented to recover global position trajectories using sequences of pose information. The intuition was that a sequence of poses contains enough information about the global displacement of a character to make an accurate prediction about this displacement. A U-Net neural network was used to estimate short global trajectories of 64 frames in a character centred reference frame. This network could run predictions at rates well over the frame rate of the motions, making it suitable for real time inference in cases where this is of interest. It was shown that the U-Net architecture had better estimation accuracy compared to a more standard convolution neural network architecture. Furthermore, the model was

able to estimate absolute height rather than vertical displacements, thus eliminating vertical drift. This ability indicates that the model was able to exploit kinematic relations of the skeleton, thus inferring height from pose information. A consequence of this, however, was that the model failed to estimate correct height for motions where the character was not in contact with the surrounding environment for a longer period of time, as would be the case for example in jumping motion.

Finally, it was shown that the selection of motion assets in the training dataset had an effect on the model's accuracy for known and unknown motion types. A model trained on a dataset specialised for specific types of motion would perform better on those types of motion compared to a model trained on a dataset with a broad variety of motions. The latter, however, turned out to perform better for unseen motion types.

Although the method had some success on IMU pose data, the reduced quality of the poses greatly impacted the performance of the model, indicating a high level of sensitivity towards the quality and modality of the motion data.

In chapter 3 the above method was extended by including acceleration signals from IMU sensors in the network's feature vector. The aim of this change was to improve performance, especially on low quality data from IMU based motion capture. Furthermore, the prediction targets and training data were adapted to accommodate for displacement predictions in the vertical direction, thus enabling trajectory reconstruction in all three dimensions rather than just the horizontal plane. Finally, the estimation window size was reduced from 64 to only 8 frames, making the solution more lean and with lower prediction latency.

The presented method was capable of accurately reconstructing global trajectories in all three dimensions, as witnessed by reconstructions of walking up and down stairs, as well as walking in the horizontal plane. A decrease in accuracy was still visible when using pose information from IMU based motion capture instead of poses from an optical setup. However, the predictions were still highly accurate compared to the ground truth trajectories. This robustness was shown to be due to using acceleration signals, as removing them from the network's feature vector made the

estimations unusable.

The model trained in chapter 3 was trained and tested using a dataset that included a wide variety of actors with different body shapes and proportions. However, sensor placements can vary widely from user to user and even from recording session to recording session. Moreover, the motion types contained in the data were somewhat homogeneous, potentially raising questions about the model's capability to generalise.

As a direction for future work, this should be investigated more by collecting larger and more diverse datasets in terms of actors and motions contained and training on them. A problem that arises here is that collecting this type of data is rather tedious and expensive, due to the need of both optical and IMU based motion capture setups. Alternatively, future work could focus on generating realistic synthetic IMU-like acceleration signals, which would enable the use of existing large optical motion capture datasets.

5.2 Artefact Cleaning for IMU Based Motion Data

Chapter 4 addressed the orthogonal problem of fixing physically implausible artefacts common to IMU based motion capture hardware, such as self penetration, floating and foot-skate.

A method was presented using deep reinforcement learning to train a humanoid agent that was capable of mimicking sample motions containing these types of artefacts, without reproducing them. The agent's policy used information about the agent's own state as well as the sample motion's state, to compute actions to transform the agent to its next state. These actions represented joint target angles for PD-controllers located at the agent's joint. The method focused on optimising performance on unseen motion data. Using a mixture of data modalities in the training data was found to be beneficial for this purpose and more so than simply increasing the amount of training data.

The resulting agent was able to mimic a test set from entirely unseen actors in

almost 90% of the cases, without reproducing artefacts present in the data. Additionally, it produced more realistic centre of mass trajectories when compared to the ground truth motion.

Finally, a configuration of the observation and action spaces was proposed that improved the agent's performance on unseen data. The agent had access to information about all joints, while it was only able to actuate a selection of them. The other joints were actuated directly by feeding the joint angles of the sample motion as targets for the agent. This was found to perform better on unseen data when compared to configurations that either had reduced state and action configurations or had full state and action configurations.

One problem that became evident during this work, was that the agent's task contains an inherent contradiction, in that it must reproduce the sample motion closely, while not reproducing certain specific behaviours. This contradiction should be mitigated through the reward system, and an interesting way to approach this could be the use of discriminator networks, to more abstractly assess the motion on similarity while also taking into account a certain desired distribution of motion data. This could also be a way to implement style transfer on arbitrary motion assets.

Bibliography

Kevin Bergamin, Simon Clavet, Daniel Holden, and James Richard Forbes. Drecon: data-driven responsive control of physics-based characters. *ACM Transactions On Graphics (TOG)*, 38(6):1–11, 2019.

Siddharth Biswal, Joshua Kulas, Haoqi Sun, Balaji Goparaju, M Brandon Westover, Matt T Bianchi, and Jimeng Sun. Sleepnet: automated sleep staging system via deep learning. *arXiv preprint arXiv:1707.08262*, 2017.

Bobby Bodenheimer, Chuck Rose, Seth Rosenthal, and John Pella. The process of motion capture: Dealing with the data. In Daniel Thalmann and Michiel van de Panne, editors, *Computer Animation and Simulation '97*, pages 3–18, Vienna, 1997. Springer Vienna. ISBN 978-3-7091-6874-5.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Nuttapong Chentanez, Matthias Müller, Miles Macklin, Viktor Makoviychuk, and Stefan Jeschke. Physics-based motion capture imitation with deep reinforcement learning. In *Proceedings of the 11th ACM SIGGRAPH Conference on Motion, Interaction and Games*, pages 1–10, New York, NY, USA, 2018. Association for Computing Machinery.

Jason PC Chiu and Eric Nichols. Named entity recognition with bidirectional lstm-cnns. *Transactions of the Association for Computational Linguistics*, 4:357–370, 2016.

Kwang-Jin Choi and Hyeong-Seok Ko. On-line motion retargetting. In *Proceedings. Seventh Pacific Conference on Computer Graphics and Applications (Cat. No.PR00293)*, pages 32–42, 1999. doi: 10.1109/PCCGA.1999.803346.

Juan Antonio Corrales, FA Candelas, and Fernando Torres. Hybrid tracking of human operators using imu/uwb data fusion by a kalman filter. In *2008 3rd ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 193–200. IEEE, 2008.

James L Coyte, David Stirling, Montserrat Ros, Haiping Du, and Andrew Gray. Displacement profile estimation using low cost inertial motion sensors with applications to sporting and rehabilitation exercises. In *2013 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 1290–1295. IEEE, 2013.

Yuzhu Dong, Andreas Aristidou, Ariel Shamir, Moshe Mahler, and Eakta Jain. Adult2child: Motion style transfer using cyclegans. In *Proceedings of the 13th ACM SIGGRAPH Conference on Motion, Interaction and Games*, MIG '20, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450381710. doi: 10.1145/3424636.3426909. URL `https://doi.org/10.1145/3424636.3426909`.

Raphael Dumas and Janis Wojtusch. *Estimation of the Body Segment Inertial Parameters for the Rigid Body Biomechanical Models Used in Motion Analysis*, pages 1–31. 01 2017. doi: 10.1007/978-3-319-30808-1_147-1.

Benedikt Fasel, Jörg Spörri, Julien Chardonnens, Josef Kröll, Erich Müller, and Kamiar Aminian. Joint inertial sensor orientation drift reduction for highly dynamic movements. *IEEE journal of biomedical and health informatics*, 22(1):77–86, 2017.

Marianne J Floor-Westerdijk, H Martin Schepers, Peter H Veltink, Edwin HF van Asseldonk, and Jaap H Buurke. Use of inertial sensors for ambulatory assessment

of center-of-mass displacements during walking. *IEEE transactions on biomedical engineering*, 59(7):2080–2084, 2012.

Eric Foxlin. Pedestrian tracking with shoe-mounted inertial sensors. *IEEE Computer graphics and applications*, 25(6):38–46, 2005.

Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4346–4354, 2015.

Thomas Geijtenbeek and Nicolas Pronost. Interactive character animation using simulated physics: A state-of-the-art review. In *Computer graphics forum*, volume 31, pages 2492–2515. Wiley Online Library, 2012.

Partha Ghosh, Jie Song, Emre Aksan, and Otmar Hilliges. Learning human motion models for long-term predictions. In *2017 International Conference on 3D Vision (3DV)*, pages 458–466. IEEE, 2017.

Michael Girard and A. A. Maciejewski. Computational modeling for the computer animation of legged figures. In *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '85, page 263–270, New York, NY, USA, 1985. Association for Computing Machinery. ISBN 0897911660. doi: 10.1145/325334.325244. URL `https://doi.org/10.1145/325334.325244`.

Pascal Glardon, Ronan Boulic, and Daniel Thalmann. Robust on-line adaptive footplant detection and enforcement for locomotion. *Vis. Comput.*, 22(3): 194–209, mar 2006. ISSN 0178-2789. doi: 10.1007/s00371-006-0376-9. URL `https://doi.org/10.1007/s00371-006-0376-9`.

Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. Hybrid speech recognition with deep bidirectional lstm. In *2013 IEEE workshop on automatic speech recognition and understanding*, pages 273–278. IEEE, 2013.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.

Gustav Eje Henter, Simon Alexanderson, and Jonas Beskow. Moglow: Probabilistic and controllable motion synthesis using normalising flows. *ACM Trans. Graph.*, 39(6), nov 2020. ISSN 0730-0301. doi: 10.1145/3414685.3417836. URL `https://doi.org/10.1145/3414685.3417836`.

At L Hof, MGJ Gazendam, and WE Sinke. The condition for dynamic stability. *Journal of biomechanics*, 38(1):1–8, 2005.

Daniel Holden, Jun Saito, Taku Komura, and Thomas Joyce. Learning motion manifolds with convolutional autoencoders. In *SIGGRAPH Asia 2015 Technical Briefs*, pages 1–4. 2015.

Daniel Holden, Jun Saito, and Taku Komura. A deep learning framework for character motion synthesis and editing. *ACM Transactions on Graphics (TOG)*, 35 (4):1–11, 2016.

Daniel Holden, Taku Komura, and Jun Saito. Phase-functioned neural networks for character control. *ACM Transactions on Graphics (TOG)*, 36(4):1–13, 2017.

Yinghao Huang, Manuel Kaufmann, Emre Aksan, Michael J Black, Otmar Hilliges, and Gerard Pons-Moll. Deep inertial poser: Learning to reconstruct human pose from sparse inertial measurements in real time. *ACM Transactions on Graphics (TOG)*, 37(6):1–15, 2018.

Hojin Ju, Min Su Lee, So Young Park, Jin Woo Song, and Chan Gook Park. A pedestrian dead-reckoning system that considers the heel-strike and toe-off phases when using a foot-mounted imu. *Measurement Science and Technology*, 27(1): 015702, 2015.

Knowledge Sourcing. Global animation market. https://www.knowledge-sourcing.com/report/global-animation-market, 2023. Accessed: November 30, 2023.

Manon Kok, Jeroen D. Hol, and Thomas B. Schön. Using inertial sensors for position and orientation estimation. *CoRR*, abs/1704.06053, 2017.

Lucas Kovar, John Schreiner, and Michael Gleicher. Footskate cleanup for motion capture editing. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '02, page 97–104, New York, NY, USA, 2002. Association for Computing Machinery. ISBN 1581135734. doi: 10.1145/545261.545277. URL https://doi.org/10.1145/545261.545277.

Michael Lapinski, Eric Berkson, Thomas Gill, Mike Reinold, and Joseph A Paradiso. A distributed wearable, wireless sensor system for evaluating professional baseball pitchers and batters. In *2009 International Symposium on Wearable Computers*, pages 131–138. IEEE, 2009.

Jehee Lee and Sung Yong Shin. A hierarchical approach to interactive motion editing for human-like figures. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, page 39–48, USA, 1999. ACM Press/Addison-Wesley Publishing Co. ISBN 0201485605. doi: 10.1145/311535.311539. URL https://doi.org/10.1145/311535.311539.

Jehee Lee, Jinxiang Chai, Paul SA Reitsma, Jessica K Hodgins, and Nancy S Pollard. Interactive control of avatars animated with human motion data. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 491–500, 2002.

Seunghwan Lee, Moonseok Park, Kyoungmin Lee, and Jehee Lee. Scalable muscle-actuated human simulation and control. *ACM Transactions On Graphics (TOG)*, 38(4):1–13, 2019.

Seyoung Lee, Sunmin Lee, Yongwoo Lee, and Jehee Lee. Learning a family of motor skills from a single motion clip. *ACM Transactions on Graphics (TOG)*, 40(4): 1–13, 2021.

Tong Li, Lei Wang, Qingguo Li, and Tao Liu. Lower-body walking motion estimation using only two shank-mounted inertial measurement units (imus). In *2020 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 1143–1148. IEEE, 2020.

Jingting Lu and Xiaoping Liu. Foot plant detection for motion capture data by curve saliency. In *Fifth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, pages 1–6, 2014. doi: 10.1109/ICCCNT.2014.6963001.

Ying-Sheng Luo, Jonathan Hans Soeseno, Trista Pei-Chun Chen, and Wei-Chao Chen. Carl: Controllable agent with reinforcement learning for quadruped locomotion. *ACM Transactions on Graphics (TOG)*, 39(4):38–1, 2020.

Li-Ke Ma, Zeshi Yang, Xin Tong, Baining Guo, and KangKang Yin. Learning and exploring motor skills with spacetime bounds. In *Computer Graphics Forum*, volume 40, pages 251–263. Wiley Online Library, 2021.

Denys Makoviichuk and Viktor Makoviychuk. rl-games: A high-performance framework for reinforcement learning. `https://github.com/Denys88/rl_games`, May 2021.

Julieta Martinez, Michael J Black, and Javier Romero. On human motion prediction using recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2891–2900, 2017.

Dushyant Mehta, Helge Rhodin, Dan Casas, Pascal Fua, Oleksandr Sotnychenko, Weipeng Xu, and Christian Theobalt. Monocular 3d human pose estimation in the wild using improved cnn supervision. In *2017 international conference on 3D vision (3DV)*, pages 506–516. IEEE, 2017.

Alberto Menache. *Understanding motion capture for computer animation*. Elsevier, 2011.

Josh Merel, Saran Tunyasuvunakool, Arun Ahuja, Yuval Tassa, Leonard Hasenclever, Vu Pham, Tom Erez, Greg Wayne, and Nicolas Heess. Catch & carry: reusable neural controllers for vision-guided whole-body tasks. *ACM Transactions on Graphics (TOG)*, 39(4):39–1, 2020.

Lucas Mourot, Ludovic Hoyet, François Le Clerc, and Pierre Hellier. Underpressure: Deep learning for foot contact detection, ground reaction force estimation and footskate cleanup. *Computer Graphics Forum*, 41(8):195–206, 2022a. doi: https://doi.org/10.1111/cgf.14635. URL `https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14635`.

Lucas Mourot, Ludovic Hoyet, François Le Clerc, François Schnitzler, and Pierre Hellier. A survey on deep learning for skeleton-based human animation. In *Computer Graphics Forum*, volume 41, pages 122–157. Wiley Online Library, 2022b.

Shaohua Pan, Qi Ma, Xinyu Yi, Weifeng Hu, Xiong Wang, Xingkang Zhou, Jijunnan Li, and Feng Xu. Fusing monocular images and sparse imu signals for real-time human motion capture. *arXiv preprint arXiv:2309.00310*, 2023.

Soohwan Park, Hoseok Ryu, Seyoung Lee, Sunmin Lee, and Jehee Lee. Learning predict-and-simulate policies from unorganized human motion data. *ACM Transactions on Graphics (TOG)*, 38(6):1–11, 2019.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. PMLR, 2013.

Dario Pavllo, Christoph Feichtenhofer, David Grangier, and Michael Auli. 3d human pose estimation in video with temporal convolutions and semi-supervised training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7753–7762, 2019.

T. Pejsa and I.S. Pandzic. State of the art in example-based motion synthesis for virtual characters in interactive applications. *Computer Graphics Forum*, 29(1):202–226, 2010. doi: https://doi.org/10.1111/j.1467-8659.2009.01591.x. URL `https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2009.01591.x`.

Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel Van de Panne. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions On Graphics (TOG)*, 37(4):1–14, 2018.

Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, and Angjoo Kanazawa. Amp: Adversarial motion priors for stylized physics-based character control. *ACM Transactions on Graphics (TOG)*, 40(4):1–20, 2021.

Xue Bin Peng, Yunrong Guo, Lina Halper, Sergey Levine, and Sanja Fidler. Ase: Large-scale reusable adversarial skill embeddings for physically simulated characters. *ACM Transactions On Graphics (TOG)*, 41(4):1–17, 2022.

Mathias Perslev, Michael Jensen, Sune Darkner, Poul Jø rgen Jennum, and Christian Igel. U-time: A fully convolutional network for time series segmentation applied to sleep staging. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 4415–4426. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper/2019/file/57bafb2c2dfeefba931bb03a835b1fa9-Paper.pdf`.

Mathias Perslev, Sune Darkner, Lykke Kempfner, Miki Nikolic, Poul Jørgen Jennum, and Christian Igel. U-sleep: resilient high-frequency sleep staging. *npj Digital Medicine*, 4(1):1–12, 2021.

Mathis Petrovich, Michael J Black, and Gül Varol. Temos: Generating diverse human motions from textual descriptions. In *European Conference on Computer Vision*, pages 480–497. Springer, 2022.

Martin Pražák, Ludovic Hoyet, and Carol O'Sullivan. Perceptual evaluation of footskate cleanup. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 287–294, 2011.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021.

Jiawei Ren, Cunjun Yu, Siwei Chen, Xiao Ma, Liang Pan, and Ziwei Liu. Diffmimic: Efficient motion mimicking with differentiable physics. *arXiv preprint arXiv:2304.03274*, 2023.

Daniel Roetenberg, Henk Luinge, and Per Slycke. Xsens mvn: Full 6dof human motion tracking using miniature inertial sensors. *Xsens Motion Technologies BV, Tech. Rep*, 1, 2009.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

Simone Sabatelli, Marco Galgani, Luca Fanucci, and Alessandro Rocchi. A double stage kalman filter for sensor fusion and orientation tracking in 9d imu. In *2012 IEEE Sensors Applications Symposium Proceedings*, pages 1–5, 2012. doi: 10.1109/SAS.2012.6166315.

Paul Schreiner, Maksym Perepichka, Hayden Lewis, Sune Darkner, Paul G. Kry, Kenny Erleben, and Victor B. Zordan. Global position prediction for interactive motion capture. *Proc. ACM Comput. Graph. Interact. Tech.*, 4(3), sep 2021a. doi: 10.1145/3479985. URL `https://doi.org/10.1145/3479985`.

Paul Schreiner, Anders Klok Sander, Matias Søndergaard, Maziar Taghiyar-Zamani, and Jakob Balslev. Methods and systems of a hybrid motion sensing framework, August 19 2021b. US Patent App. 17/106,172.

Paul Schreiner, Sune Darkner, and Kenny Erleben. Root3D: Root position reconstruction in 3 dimensions for imu based motion capture. *Under review at the Symposium on Interactive 3D Graphics and Games*, 2024a.

Paul Schreiner, Rasmus Netterstrøm, Sune Darkner, Hang Yin, and Kenny Erleben. ADAPT: Ai-driven artefact purging technique for imu based motion capture. *Under review at Computer Graphics Forum*, 2024b.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Loren Arthur Schwarz, Diana Mateus, and Nassir Navab. Recognizing multiple human activities and tracking full-body pose in unconstrained environments. *Pattern Recognition*, 45(1):11–23, 2012.

Soshi Shimada, Vladislav Golyanik, Weipeng Xu, and Christian Theobalt. Physcap: Physically plausible monocular 3d motion capture in real time. *ACM Transactions on Graphics (TOG)*, 39(6):1–16, 2020.

Soshi Shimada, Vladislav Golyanik, Weipeng Xu, Patrick Pérez, and Christian Theobalt. Neural monocular 3d human motion capture with physical awareness. *ACM Transactions on Graphics (ToG)*, 40(4):1–15, 2021.

Takaaki Shiratori, Hyun Soo Park, Leonid Sigal, Yaser Sheikh, and Jessica K. Hodgins. Motion capture from body-mounted cameras. *ACM Trans. Graph.*, 30(4), July 2011. ISSN 0730-0301.

Ronit Slyper and Jessica K Hodgins. Action capture with accelerometers. In *Symposium on Computer Animation*, pages 193–199, 2008.

Sebastian Starke, Yiwei Zhao, Taku Komura, and Kazi Zaman. Local motion phases for learning multi-contact character movements. *ACM Transactions on Graphics (TOG)*, 39(4):54–1, 2020.

Statista Research Department. Worldwide animation market size. https://www.statista.com/statistics/817601/worldwide-animation-market-size/, 2023. Accessed: November 30, 2023.

Young Soo Suh. Inertial sensor-based smoother for gait analysis. *Sensors*, 14(12): 24338–24357, 2014.

Guy Tevet, Sigal Raab, Brian Gordon, Yonatan Shafir, Daniel Cohen-Or, and Amit H Bermano. Human motion diffusion model. *arXiv preprint arXiv:2209.14916*, 2022.

Guy Tevet, Sigal Raab, Brian Gordon, Yoni Shafir, Daniel Cohen-or, and Amit Haim Bermano. Human motion diffusion model. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=SJ1kSyO2jwu.

Dominic Thewlis, Chris Bishop, Nathan Daniell, and Gunther Paul. Next-generation low-cost motion capture systems can provide comparable spatial accuracy to high-end systems. *Journal of applied biomechanics*, 29(1):112–117, 2013.

Guillermo Valle-Pérez, Gustav Eje Henter, Jonas Beskow, Andre Holzapfel, Pierre-Yves Oudeyer, and Simon Alexanderson. Transflower: Probabilistic autoregressive dance generation with multimodal attention. *ACM Trans. Graph.*, 40(6), dec 2021. ISSN 0730-0301. doi: 10.1145/3478513.3480570. URL https://doi.org/10.1145/3478513.3480570.

Tom Van Wouwe, Seunghwan Lee, Antoine Falisse, Scott Delp, and C Karen Liu. Diffusion inertial poser: Human motion reconstruction from arbitrary sparse imu configurations. *arXiv preprint arXiv:2308.16682*, 2023.

Márton Véges and András Lőrincz. Absolute human pose estimation with depth prediction network. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, 2019.

Daniel Vlasic, Rolf Adelsberger, Giovanni Vannucci, John Barnwell, Markus Gross, Wojciech Matusik, and Jovan Popović. Practical motion capture in everyday surroundings. *ACM transactions on graphics (TOG)*, 26(3):35–es, 2007.

He Wang, Edmond SL Ho, Hubert PH Shum, and Zhanxing Zhu. Spatio-temporal manifold learning for human motions via long-horizon modeling. *IEEE transactions on visualization and computer graphics*, 27(1):216–227, 2019.

Tingwu Wang, Yunrong Guo, Maria Shugrina, and Sanja Fidler. Unicon: Universal neural controller for physics-based character motion. *arXiv preprint arXiv:2011.15119*, 2020.

Prabancoro Adhi Catur Widagdo, Hsin-Huang Lee, and Chung-Hsien Kuo. Limb motion tracking with inertial measurement units. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 582–587. IEEE, 2017.

Jungdam Won and Jehee Lee. Learning body shape variation in physics-based characters. *ACM Transactions on Graphics (TOG)*, 38(6):1–12, 2019.

Jungdam Won, Deepak Gopinath, and Jessica Hodgins. A scalable approach to control diverse behaviors for physically simulated characters. *ACM Transactions on Graphics (TOG)*, 39(4):33–1, 2020.

Xuesu Xiao and Shuayb Zarar. Machine learning for placement-insensitive inertial motion capture. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6716–6721. IEEE, 2018.

Jie Xu, Viktor Makoviychuk, Yashraj Narang, Fabio Ramos, Wojciech Matusik, Animesh Garg, and Miles Macklin. Accelerated policy learning with parallel differentiable simulation. *arXiv preprint arXiv:2204.07137*, 2022.

Xinyu Yi, Yuxiao Zhou, and Feng Xu. Transpose: Real-time 3d human translation and pose estimation with six inertial sensors. *ACM Transactions on Graphics (TOG)*, 40(4):1–13, 2021.

Xinyu Yi, Yuxiao Zhou, Marc Habermann, Soshi Shimada, Vladislav Golyanik, Christian Theobalt, and Feng Xu. Physical inertial poser (pip): Physics-aware real-time human motion tracking from sparse inertial sensors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13167–13178, 2022.

Wenjie Yin, Hang Yin, Danica Kragic, and Mårten Björkman. Graph-based normalizing flow for human motion generation and reconstruction. In *2021 30th IEEE International Conference on Robot & Human Interactive Communication (RO-MAN)*, pages 641–648, 2021. doi: 10.1109/RO-MAN50785.2021.9515316.

Wenjie Yin, Ruibo Tu, Hang Yin, Danica Kragic, Hedvig Kjellström, and Mårten Björkman. Controllable motion synthesis and reconstruction with autoregressive diffusion models. In *2023 32th IEEE International Conference on Robot & Human Interactive Communication (RO-MAN)*, 2023a.

Wenjie Yin, Hang Yin, Kim Baraka, Danica Kragic, and Mårten Björkman. Dance style transfer with cross-modal transformer. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 5058–5067, January 2023b.

Qilong Yuan, I-Ming Chen, and Shang Ping Lee. Slac: 3d localization of human based on kinetic human movement capture. In *2011 IEEE International Conference on Robotics and Automation*, pages 848–853. IEEE, 2011.

Ye Yuan, Umar Iqbal, Pavlo Molchanov, Kris Kitani, and Jan Kautz. Glamr: Global occlusion-aware human mesh recovery with dynamic cameras. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11038–11049, 2022a.

Ye Yuan, Jiaming Song, Umar Iqbal, Arash Vahdat, and Jan Kautz. Physdiff: Physics-guided human motion diffusion model. *arXiv preprint arXiv:2212.02500*, 2022b.

Ye Yuan, Jiaming Song, Umar Iqbal, Arash Vahdat, and Jan Kautz. Physdiff: Physics-guided human motion diffusion model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16010–16021, 2023.

He Zhang, Sebastian Starke, Taku Komura, and Jun Saito. Mode-adaptive neural networks for quadruped motion control. *ACM Transactions on Graphics (TOG)*, 37(4):1–11, 2018.

Yang Zheng, Ka-Chun Chan, and Charlie CL Wang. Pedalvatar: An imu-based real-time body motion capture system using foot rooted kinematic model. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4130–4135. IEEE, 2014.

Xiaowei Zhou, Menglong Zhu, Georgios Pavlakos, Spyridon Leonardos, Konstantinos G Derpanis, and Kostas Daniilidis. Monocap: Monocular human motion capture using a cnn coupled with a geometric prior. *IEEE transactions on pattern analysis and machine intelligence*, 41(4):901–914, 2018.

Yi Zhou, Jingwan Lu, Connelly Barnes, Jimei Yang, Sitao Xiang, et al. Generative tweening: Long-term inbetweening of 3d human motions. *arXiv preprint arXiv:2005.08891*, 2020.