EPFL

Thèse n° 8976

Automated Human Motion Analysis and Synthesis

Présentée le 24 février 2023

Faculté informatique et communications
Laboratoire de vision par ordinateur
Programme doctoral en informatique et communications

pour l'obtention du grade de Docteur ès Sciences

par

Sena KICIROGLU

Acceptée sur proposition du jury

Prof. S. Süsstrunk, présidente du jury
Prof. P. Fua, Dr M. Salzmann, directeurs de thèse
Dr A. Kapoor, rapporteur
Prof. C. Sminchisescu, rapporteur
Prof. A. Alahi, rapporteur

École
polytechnique
fédérale
de Lausanne

2023

We have to continually be jumping off cliffs
and developing our wings on the way down.
— Kurt Vonnegut, *If This Isn't Nice, What Is?*

To my parents and my sister

Acknowledgements

I'm very glad to have done my PhD at EPFL. This part of my life was a time of immense growth, both personally and professionally. I would like to thank the people I met along the way.

To my advisors, Pascal Fua and Mathieu Salzmann, thank you for your supervision and guidance throughout the years. I'm immensely grateful to have had the opportunity to work in CVLab. Thank you Pascal for being such a hands-on supervisor, especially early on in my PhD, and for sprinkling copious amounts of your writing magic on our papers. Thank you Mathieu for all the weekly meetings, from the beginning to the end. It was something I actually looked forward to (rare for a meeting with a supervisor, I think), because I knew discussions with you would always be fruitful. Thank you both for all the support, it was a pleasure to work with you.

I am very grateful to my jury president Sabine Süsstrunk, and my jury members Alexandre Alahi, Ashish Kapoor, and Cristian Sminchisescu for reading my thesis and providing valuable comments. I really enjoyed sharing this work with you and the discussion that followed.

Thank you Sudipta Sinha, Sai Vemprala, Helge Rhodin, and Wei Wang for our collaborations. Discussions with you were always much appreciated and very enlightening. Thank you Wei additionally for being my office mate for almost two years, I always enjoyed our chats and the adorable baby pictures you showed me. I can't thank Helge enough for showing me the ropes in my first two years. I had come to the PhD fresh out a bachelor's in a completely different field and I needed a lot of guidance to get on track. It was thanks to you that I got started. You are always one step ahead of every problem, your students are lucky to have you! Additionally, I've supervised many students over the semesters and I would like to thank them all, in particular, Ziyi Zhao, Hugues Vinzant, and Tim Lebailly. They all have major contributions to this thesis and I'm truly glad to have worked with them.

Thank you to the people who powered through the PhD with me; Eric, Kiki, Merlin, Bogdan, Mariam (by proxy), Sahand, Paritosh, Sebastien, TJ, Ceyhun- we made it (and will make it!) Thank you Jakab and Matteo for inviting me to your awesome house parties. Thank you Karen for being the best co-fun events manager, and for the slow running sessions. Thank you Sandra, Bharath, George, for all the delicious food, good company, and long discussions on exactly what is wrong with the world. You have been my family here and always made me feel welcome in your home.

Thank you Barbora and all my yoga/bouldering friends, we've truly suffered through some hard workouts together ("now let's hold this pose for one million breaths"). Thanks for making it fun!

I would like to give a shout out to my Bilkent friends - Safa, Ilkay, Erdem, Rahmi, MFS, Omert, Omer, Kubra, Zubeyr, Fatih, Simsek, Nazli, Merve - for always keeping my chat applications active! To my Q1 buddy Cagdas, thanks for all the postcards! To my Turkish buddies here, Erhan, Ceren, Doga, Asli, Baran, Bahar, Bahar, and Dilan, who are all extremely funny people, thank you for always instantly lifting my mood.

Thank you to Agata, Roger, and Anne for letting me feel like one of the cool kids during the first years of my PhD, and later on, showing me that life after the PhD does exist! Thank you to Mateusz for your guiding wisdom ("stop being lazy and go write your paper!"). And to all the others at CVLab, past and present - Nikita, Curly Benoit, Beardy Benoit, Argentinian Federico, Italian Federico, Louis, Semih (for our insane office chats), Zheng, Zhen, Weizhe (for carrying the LoL games with no complaints), Shuxuan, Kaicheng (for your amazing homecooked dinners), Yinlin, Bugra, Artem, Andrii, Ksenia, Eduard, Udaranga, Krishna, Shuangqi, Soumava, Jiancheng, Leo, Edo, Andrey (for always sharing your snacks), Sina, Saqib, Krzy (whom my office plants also thank), Michal, Victor, Doruk, Okan, Shaifali, Matthias, Patrick, Chen, Terry, Joachim, Matthias, Andrew, Jakub, Yann, Deniz, Malo, Alex, Aoxiang - we're such a huge lab! - and many others, thank you for making the lab environment cozy and welcoming. I will miss the daily lab lunches, with coffee after!

A huge thank you to Martin and Vidit for being the best via ferrata buds and for tolerating my singing (*tell me why?*) I always enjoyed bursting in your office for random chats, especially when you had chocolates. Thank you Nico for sharing my Hollow Knight, anime, Disneyland enthusiasm. I will miss making binocular eyes at you from the opposite metro platform. Thank you to Fayez for never letting me win at Ticket to Ride, to Pei-i for the humbling experience of watercoloring together, to Luca (hey Luca, why did the chicken cross the road?). David, Terka, and Hynek (though he can only take credit for the final year), thank you for the many hikes and board game nights. Losing to you was fun occasionally, but winning - ah, those wins felt good.

A massive thank you to Isinsu, my lab sister, travel buddy, and support system. It's been an amazing five years of running, chatting, working, watercoloring, laughing, and basically holding each other up on every occasion. Hands down, the best part of this PhD was meeting you.

Thank you Jan for bringing me into your world and teaching me everything from snowboarding to via ferratas to making Christmas cookies. It's been a wonderful time exploring Switzerland with you. Thank you for all the research discussions and politics discussions, and just plainly for being my best friend. Here's to someday spotting every constellation in the sky!

Finally, to my father, mother, and my five-years-younger-twin Serra, I love you so much. Thank you for all the emotional support, reassurance, and love throughout these years. Knowing that you were always just a phone call away made all the difficulties easier to face (even those pesky paper rejections) and all the wins even more joyful!

And to anyone else who has a hobby of reading other people's acknowledgments sections - I see you! Thanks for dropping by. Maybe have a look at the rest of the thesis too...

Lausanne, February 1, 2023 Sena Kiciroglu

Abstract

Human motion analysis and synthesis is integral to many computer vision applications, from autonomous driving to sports analysis. In this thesis, we address several problems in this domain. First we consider active viewpoint selection for pose estimation where we choose the next viewpoint of the camera so that we obtain accurate 3D pose estimations across time. Afterwards we consider motion prediction, which is the task of predicting future human motion sequences given past ones. Finally, we address the application-based problem of providing automated physical exercise feedback by analyzing the motion.

For any human motion analysis framework, it is necessary to first obtain the 3D human pose from images. We consider a variant of this problem using a moving camera: within a 3D pose estimation framework, our goal is to choose the next best viewpoint to obtain accurate pose estimation results. We design an active viewpoint selection algorithm that uses uncertainty as a proxy for estimating potential error values. The camera moves to the candidate viewpoint or trajectory that has the least uncertainty of the 3D pose estimation. We compare against naive baselines such as constant rotation and random viewpoint selection and show that our active policy achieves more accurate results.

In order to build such systems reacting to the human motion, one must also have a good estimate of the future state. Therefore, we study the problem of motion prediction from observed past poses, both for time horizons of 1 second and 5 seconds. Our first framework focuses on estimating highly accurate futures of up to 1 second. Existing methods observe past sequences of fixed length to predict the future. We design a framework which aggregates features extracted from subsequences of multiple lengths. This information is extracted via Temporal Inception Modules (TIM), where the convolutional kernel sizes are proportional to the length of the input subsequence. We demonstrate that our architecture outperforms existing methods on mean per joint error metrics up to the future time-horizon of 1 second.

We extend our time horizon to 5 seconds and design a framework to predict into the long term future. Many existing motion prediction works fail to synthesize dynamic and realistic human motions over extended time horizons. Our approach uses the most essential poses in the sequence, which we refer to as “keyposes". Keyposes are extracted automatically from the data as the poses which minimize the reconstruction loss of the original sequence. Designing a Gated Recurrent Unit (GRU)-based sequence prediction framework, we observe past keyposes

and predict future ones. The introduced method is able to outperform existing state-of-the-art motion prediction methods for a future time-horizon of 5 seconds. We demonstrate that our method produces more dynamic and realistic human motion sequences which are plausible continuations of the observed past.

A highly relevant application of human motion analysis and synthesis is for sports. We focus on providing automated feedback to individuals performing physical exercises. Our feedback comes in two forms: we classify the type of mistake the exercise contains, and we provide personalized corrections in the forms of synthesized human motions. Our method achieves 90.9% mistake identification accuracy, and corrects incorrectly performed exercises with 94.2% success.

Keywords: motion analysis, motion synthesis, pose estimation, motion prediction, exercise analysis.

Résumé

L'analyse et la synthèse du mouvement humain font partie intégrante de nombreuses applications de vision par ordinateur, des véhicules autonome à l'analyse sportive. Dans cette thèse, nous abordons plusieurs problèmes dans ce domaine. Tout d'abord, nous considérons la sélection active du point de vue pour l'estimation de la pose, où nous choisissons le prochain point de vue de la caméra afin d'obtenir des estimations de pose 3D précises dans le temps. Ensuite, nous considérons la prédiction de mouvement, qui est la tâche de prédire les futures séquences de mouvement humain en fonction des séquences passées. Enfin, nous abordons une application pratique consistant à fournir un retour d'information automatisé sur les exercices physiques en analysant le mouvement.

Pour tout cadre d'analyse du mouvement humain, il est nécessaire d'obtenir d'abord la pose humaine 3D à partir d'images. Nous considérons la variante de ce problème en utilisant une caméra mobile : dans un cadre d'estimation de pose 3D, notre objectif est de choisir le meilleur point de vue pour obtenir des résultats d'estimation de pose précis. Nous concevons un algorithme de sélection active du point de vue qui utilise l'incertitude pour estimer les valeurs d'erreur potentielles. La caméra se déplace vers le point de vue ou la trajectoire candidate qui présente le moins d'incertitude pour l'estimation de la pose 3D. Nous comparons ces résultats à ceux obtenus par des méthodes naïves telles que la rotation constante et la sélection aléatoire de points de vue et montrons que notre politique active permet d'obtenir des résultats plus précis.

Afin de construire de tels systèmes réagissant au mouvement humain, il faut également disposer d'une bonne estimation de l'état futur. Nous étudions donc le problème de la prédiction du mouvement à partir des poses passées observées, pour des horizons temporels de 1 seconde et de 5 secondes. Notre premier travail se concentre sur l'estimation de futurs très précis jusqu'à 1 seconde. Les méthodes existantes observent des séquences passées de longueur fixe pour prédire le futur. Nous concevons une méthode qui regroupe les caractéristiques extraites de sous-séquences de longueurs multiples. Ces informations sont extraites via des Temporal Inception Modules (TIM), où la taille des noyaux convolutifs est proportionnelle à la longueur de la sous-séquence d'entrée. Nous démontrons que notre architecture surpasse les méthodes existantes sur les mesures d'erreur moyenne par jointure jusqu'à l'horizon temporel futur de 1 seconde.

Nous étendons notre horizon temporel à 5 secondes et concevons une solution permettant de prédire l'avenir à long terme. De nombreux travaux de prédiction de mouvements existants ne parviennent pas à synthétiser des mouvements humains dynamiques et réalistes sur des horizons temporels étendus. Notre approche utilise les poses les plus essentielles de la séquence, que nous appelons "keyposes". Elles sont extraites automatiquement des données, comme les poses qui minimisent la perte de reconstruction de la séquence originale. En concevant une méthode de prédiction de séquence basé sur les Gated Recurrent units (GRU), nous observons les keyposes passées et prédisons les futures. La méthode introduite est capable de surpasser les méthodes existantes de prédiction de mouvement pour un horizon temporel futur de 5 secondes. Nous démontrons que notre méthode produit des séquences de mouvements humains plus dynamiques et réalistes qui sont des continuations plausibles du passé observé.

Le sport est une application très pertinente de l'analyse et de la synthèse du mouvement humain. Nous nous concentrons sur le retour d'information automatisé aux personnes effectuant des exercices physiques. Notre retour se présente sous deux formes : nous classons le type d'erreur que contient l'exercice et nous fournissons des corrections personnalisées sous la forme de mouvements humains synthétisés. Notre méthode atteint une précision d'identification des erreurs de 90,9% et corrige les exercices incorrectement exécutés avec 94,2% de réussite.

Keywords : analyse du mouvement, synthèse du mouvement, estimation de la pose, prédiction du mouvement, analyse de l'exercice.

Contents

List of Figures

List of Tables

1 Introduction

It has become increasingly popular to develop computer vision applications which are either directly or indirectly focused on human motion. In many cases, detecting and analyzing human motion is the main objective, as in sports analysis [20, 49], or surveillance [113]. In many other applications, analyzing human motions is not the main task, but a crucial component of the application. For instance, the main focus of self-driving cars is to draw a safe path for the car to navigate within traffic. However in order to avoid dangerous situations, it is imperative for the vehicle to have a sense of whether there are pedestrians around, what they are doing, and where they will go next [62]. In all of such computer vision applications, careful design of human motion algorithms is key.

Our work in this thesis focuses on human motion analysis and synthesis problems, which are the cornerstones of many computer vision applications. These are two closely related tasks. Human motion analysis focuses on understanding existing sequences. Human motion synthesis focuses on generating plausible human motions.

Computer vision has been concerned with introducing automation to studying human motion, in order to have quick and accurate results without expanding manual effort. Even before the emergence of deep learning methods, there was a plethora of research on automatically detecting the humans in the image [40], reconstructing them in 3D [150, 152, 198], tracking them [24, 121, 155, 154, 169], interpreting their motions in a semantic way [153, 168], and synthesizing new human motions [9, 86]. In more recent years, deep learning architectures have been deployed to learn these tasks using large datasets for training [36, 71, 104].

An application to process human motion information usually begins with the image frames, either first reconstructing the human in 2D [29, 87, 119], followed by 3D [5, 57, 114]; or reconstructing in 3D directly [77, 111, 112, 164]. Many frameworks only consider the pose of the person, which is represented either by the angles or 3D locations of their joints [109]. There also exist frameworks which reconstruct the shape of the person as well, using parametric models such as SMPL [99] or GHUM [176]. Once the human representation is obtained in 3D, one can analyze it for the specific purposes of the application. For instance, there are

scenarios where it is imperative to understand what humans are doing in fine detail, whether it is for entertainment purposes such as an AR game, or for assisting human at the assembly line, or even the operating table [17]. Some applications take this one step further and try to synthesize new sequences, which also requires an understanding of realistic human motion. Examples of such application can be character animation for video games [9].

In this thesis, we specifically consider the problems of active viewpoint selection for human pose estimation, human motion prediction, and physical exercise feedback. We first design an algorithm that reconstructs the 3D human pose and finds the optimal viewpoint for the future time step. To do so, we have several requirements, namely, to find an approximation of the 3D pose estimation accuracy from future viewpoints, and to have an accurate representation of human motion for the future time steps. Initially, we focus on the first requirement and use the uncertainty of the 3D human pose estimation as an approximation to the human reconstruction error.

Our next step is to focus on human motion prediction. We approach the problem in two different ways: the first way is to predict the human pose with very high accuracy for the 1 second time horizon. We achieve this by designing a deep learning architecture that makes use of both short term and long term inputs and finds appropriate convolutional kernel sizes to extract the information in the sequences. Introducing a temporal inception module, we show that we are able to predict high fidelity human motion in the 1 second time horizon.

We extend our motion prediction time horizon to 5 seconds. To achieve successful realistic motions in the long term, it is not enough to retrain existing state-of-the-art works to predict motion for longer time horizons. We have noticed that these methods fail to preserve the realistic motion dynamics of the sequences. Instead, we introduce the concept of keyposes, which can be used to detect essential events in the sequence. Using keyposes, we can learn the patterns of transition from one important pose to another. By modeling human motion as a sequence of keyposes, we train our model to predict likely future keyposes and interpolate them to reconstruct the sequence. Furthermore, because we model future keyposes in a probabilistic manner, we synthesize multiple plausible motions via sampling.

Finally we focus on a real-world application of human motion analysis and synthesis, physical exercise evaluation. In an effort to provide an automated physical trainer to amateur athletes who exercise on their own, we design a system which provides feedback in two forms. First, we identify the type of exercise that is being performed and what type of mistake is being made if the exercise is performed incorrectly. Second, we synthesize a correct version of the exercise. This is completely automated, and specific to the subject that is performing the exercises.

The projects we focus on in this thesis can be contextualized in the form of a hypothetical application: a personal trainer drone. Such an autonomous framework would be able to reconstruct 3D human pose and position itself continuously to obtain the best 3D pose; predict the motion of the person in the short and long term for accurate route planning; and recognize mistakes in the exercises being perfomed and tailor appropriate feedback to the

athlete. On this note, we now continue with introducing motivation and more applications for research in human motion analysis and synthesis.

1.1 Motivation and Applications

Human motion analysis and synthesis plays a key role in many real-world applications. For many developing technologies, such as autonomous driving, accurate analysis of human motion is integral to the feasibility of bringing this futuristic dream to life. In entertainment, having realistic synthesis of human motion has become a crucial component to making a high quality products e.g. in video games or animated movies production. In this section, we detail some of the applications that motivate researchers to further the knowledge in this field. Example images from each applications are shown in Figure 1.1.

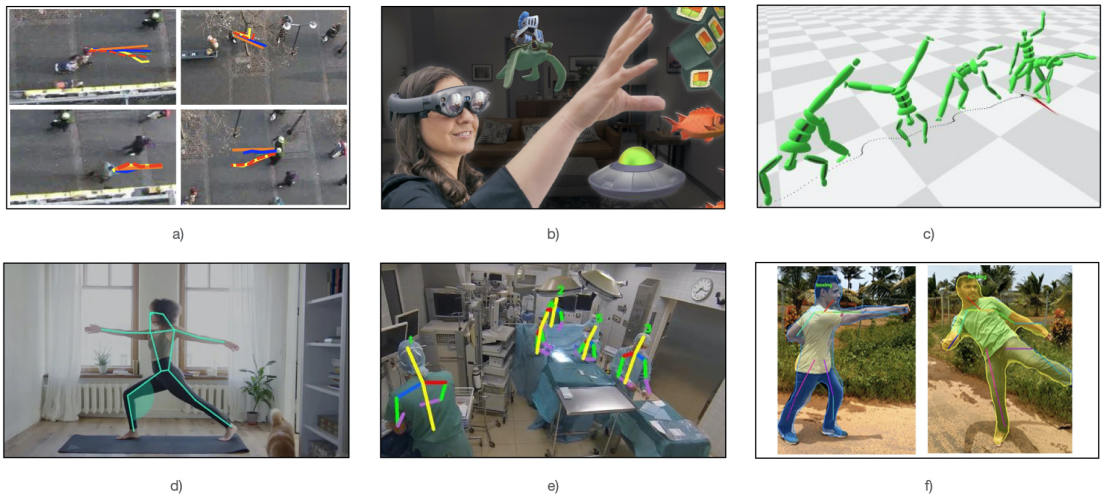


Figure 1.1 – **Example images from applications** of human motion analysis and synthesis. **a) Autonomous driving**: Pedestrian trajectories are predicted using the “Social LSTM" method (shown as the dashed red line) and other methods [8]. **b) AR/VR**: The Magic Leap augmented reality goggles with hand gesture recognition [103]. **c) Character animation**: An animation constructed using parametric motion graphs [64]. **d) Sports analysis and assisted athletic training**: Viso.ai’s demonstration of using pose estimation results for exercise identification and counting repetitions automatically [173]. **e) Healthcare and Assisted Recovery**: 3D poses of surgeons in the operating room can give important insights into surgical workflows [17]. **f) Security and Surveillance**: The “Deep-Violence" method detects violent activity detection using pose sequences [118].

Autonomous Driving. Autonomous driving is a safety-critical application where robust human motion analysis and synthesis is a primary requirement. Self-driving car technologies must ensure that they are capable of detecting and localizing pedestrians [18, 21, 62], and tracking them across the scene [83, 138]. For appropriate planning of future actions, such as adjusting speed and drawing paths that avoid collisions, the vehicle must also accurately

forecast pedestrian trajectories [8, 60, 81, 84, 148]. This is often done probabilistically to synthesize several pedestrian paths, allowing the vehicles to plan according to the different possibilities. Developments in these topics are truly important, as without them it would be impossible to bring widespread autonomous driving technologies to life.

Augmented and Virtual Reality. Augmented and virtual reality (AR/VR) has become an exciting direction for research with many companies developing technologies that will bring AR/VR to everyday life, such as AR/VR glasses [56, 103]. Virtual environments are envisioned as the social medias and workplaces of the future, featuring accurate representations of the users for a realistic experience [107, 115]. In order to make this a reality, it is important to integrate human motion analysis algorithms. For instance, accurate reconstructions of the human bodies and faces can make for immersive conference calls. Another example is to control the virtual environment via gesture recognition of the hands. By improving these methods we come one step closer to creating truly immersive virtual worlds.

Character Animation. Designing how characters move in animations, CGI movies, and video games is a task that demands a lot of artistic skill and time. Therefore, automated motion synthesis techniques are heavily used in character animation. For realistic and smooth animations, motion synthesis techniques such as motion graphs are used to interpolate realistically between key frames. [64, 86, 85].[1] Since then, many deep learning based methods have emerged for character animation [26, 129] which learn natural human motions from large motion capture datasets. These methods all pave the way to lightening the workload of artists and technical staff in entertainment.

Sports Analysis and Assisted Athletic Training. Automation in the world of sports analysis has introduced new experiences for spectators, athletes, referees, and coaches. Spectators can now watch careful breakdowns of recaps thanks to emerging technologies offered by companies for basketball and martial arts [37, 143]. With deep learning methods providing real-time analysis of sports, an interest has formed around this topic once again. However this has been a prominent topic for computer vision researchers before the advent of deep learning as well, with techniques being applied to basketball [7, 20] and golf [168]. Motion analysis for sports can also be used by coaches for enhanced training and injury prevention. Once the 3D poses of the athletes are extracted via pose estimation techniques, further analysis can be done on their pose sequences to determine whether they are performing the exercises correctly and efficiently. All together, these frameworks can be useful for increasing performance and the quality of entertainment for the spectators, and for avoiding injuries.

Healthcare and Assisted Recovery. Using human pose analysis in healthcare can automate processes such as surgery training and assessment. For example, 3D pose estimation of surgeons in the operating room [17] can be combined with action recognition to bring important insights into operations. Similarly, hand motions of surgeons can be tracked and analyzed

[1] In Chapter 5 we discuss how we design a similar method to Motion Graphs [86]. However, our goal is to synthesize future motions by predicting and interpolating “keyposes".

to detect the surgical maneuvers automatically [65, 15], which can subsequently reviewed or used for training. Additionally, healthcare applications go hand in hand with sports rehabilitation. Sports analysis techniques can also be applied to physiotherapy exercises in order to guide patients with automated feedback [2]. These example applications highlight possible innovations which can help practitioners study medical procedures in a safer and more practical way.

Security and Surveillance. It can be time consuming and labor intensive to manually go through security footage. Moreover, by the time the security personnel realizes that a dangerous activity is occurring, it may already be too late. With computer vision techniques, footage from surveillance cameras can be analyzed in an automatized manner [12, 52, 61]. For instance, action recognition can be used to determine whether the actions of an individual are suspicious or not [118]. Using datasets such as Violent-Flows [63], methods can be trained to detect whether individuals are committing acts of violence [51]. It must be stressed that it is very important to also conduct ethical research that is not victim to racial biases [128]. These applications remind us that the surveillance cameras around us can be used for the security of the public, but researchers and policy makers must be extremely careful in how they are developed and what they are used for.

1.2 Problem Definition

Human motion analysis and synthesis encompasses a wide range of challenging problems. In this thesis, we focus on active viewpoint selection for human pose estimation, motion prediction, and physical exercise feedback. We have striven to bring new problems into light, and to advance the state-of-the-art in existing problems. We define our focus in detail below.

1.2.1 Active Viewpoint Selection for Human Pose Estimation

Human pose estimation is the task of extracting the 2D and 3D joint locations of a human from images. We make use of existing research in 2D and 3D human pose estimation in order to study the problem of choosing the next best viewpoint for the more accurate 3D pose reconstruction over time.

Human pose estimation is a challenging topic due to occlusions, motion blur, depth ambiguity, etc. Active viewpoint selection for pose estimation has the added difficulty of estimating the accuracy of a yet unknown pose from candidate viewpoints. In order to estimate the accuracy, we need to know the future pose and what our estimate of the pose will be from candidate viewpoints. Our approach to tackle this problem is to design a proxy for accuracy. With our active viewpoint selection strategy, we select the next viewpoint from which we can reconstruct a more accurate 3D human pose.

1.2.2 Human Motion Prediction

Human motion prediction is the task of predicting future human pose sequences conditioned on the past ones.

Motion prediction offers a unique set of challenges of its own, due to the inherent uncertainty that the future holds. It is not enough to assume constant velocity of human joint positions. Humans can have highly abrupt changes in their poses, which are not always easily anticipated. It is necessary to learn the patterns inherent to natural human motion from the observed data, in order to predict future sequences in high fidelity.

As we try to predict longer term time horizons, we encounter the problem that the future branches into diverse paths. It is therefore often necessary to anticipate multiple likely futures from a single observed sequence. By doing so, we can allow our autonomous systems to plan for several possible outcomes in the future. This is mainly a human motion synthesis problem, where we construct realistic continuations of the observed sequence.

1.2.3 Pose Based Exercise Feedback

We design a framework for providing automated feedback of a person doing physical exercises, based on their 3D pose. The feedback can be given on numerous levels: we focus on identifying the types of mistakes that are potentially being made, and providing corrections of incorrectly performed exercises. This correction is provided as a 3D pose sequence, which to the best of our knowledge has not been previously attempted.

The challenges to face in this topic especially using deep learning approaches are numerous since it is still fairly young and under-explored. Mainly, there are not many annotated datasets where mistakes in sports are explicitly made. Moreover, it can be difficult to determine what is an incorrect performance or not without expert help. Indeed, there can be many correct ways of doing an exercise, and these should be taken into consideration when giving personalized feedback.

1.3 Contributions

ActiveMocap: Active Viewpoint Selection for Human Motion Capture

The accuracy of monocular 3D human pose estimation depends on the viewpoint from which the image is captured. While freely moving cameras, such as on drones, provide control over this viewpoint, automatically positioning them at the location which will yield the highest accuracy remains an open problem. This is the problem that we address in this work. Specifically, given a short video sequence, we introduce an algorithm that predicts which viewpoints should be chosen to capture future frames so as to maximize 3D human pose estimation accuracy. The key idea underlying our approach is a method to estimate

the uncertainty of the 3D body pose estimates. We integrate several sources of uncertainty, originating from deep learning based regressors and temporal smoothness. Our motion planner yields improved 3D body pose estimates and outperforms or matches existing ones that are based on person following and orbiting. This work was published as the conference paper [78].

S. Kiciroglu, H. Rhodin, S. Sinha, M. Salzmann, P. Fua. *ActiveMocap: Optimized Viewpoint Selection For Active Human Motion Capture*. CVPR 2020.

Motion Prediction Using Temporal Inception Module

Human motion prediction is a necessary component for many applications in robotics and autonomous driving. Existing methods do not focus on exploiting different temporal scales for different length inputs. We argue that the diverse temporal scales are important as they allow us to look at the past frames with different receptive fields, which can lead to better predictions. In this work, we propose a Temporal Inception Module (TIM) to encode human motion. Making use of TIM, our framework produces input embeddings using convolutional layers, by using different kernel sizes for different input lengths. The experimental results on standard motion prediction benchmark datasets show that our approach consistently outperforms the evaluated methods. This work was published as the conference paper [88].

T. Lebailly, S. Kiciroglu, M. Salzmann, P. Fua, W. Wang. *Motion Prediction Using Temporal Inception Module*. ACCV 2020.

Long Term Motion Prediction Using Keyposes

Long term human motion prediction is essential in safety-critical applications such as human-robot interaction and autonomous driving. In this paper we show that to achieve long term forecasting, predicting human pose at every time instant is unnecessary. Instead, it is more effective to predict a few keyposes and approximate intermediate ones by interpolating the keyposes. We demonstrate that our approach enables us to predict realistic motions for up to 5 seconds in the future, which is far longer than the typical 1 second encountered in the literature. Furthermore, because we model future keyposes probabilistically, we can generate multiple plausible future motions by sampling at inference time. Over this extended time period, our predictions are more realistic, more diverse and better preserve the motion dynamics than those state-of-the-art methods yield. This work was published as the conference paper [79].

S. Kiciroglu, W. Wang, M. Salzmann, P. Fua. *Long Term Motion Prediction Using Keyposes*. 3DV 2022.

3D Pose Based Feedback for Physical Exercises

Unsupervised self-rehabilitation exercises and physical training can cause serious injuries if performed incorrectly. We introduce a learning-based framework that identifies the mistakes made by a user and proposes corrective measures for easier and safer individual training. Our framework does not rely on hard-coded, heuristic rules. Instead, it learns them from data, which facilitates its adaptation to specific user needs. To this end, we use a Graph Convolutional Network (GCN) architecture acting on the user's pose sequence to model the relationship between the the body joints trajectories. To evaluate our approach, we introduce a dataset with 3 different physical exercises. Our approach yields 90.9% mistake identification accuracy and successfully corrects 94.2% of the mistakes. This work was published as the conference paper [192].

Z. Zhao, S. Kiciroglu, H. Vinzant, Y. Cheng, I. Katircioglu, M. Salzmann, P. Fua. *3D Pose Based Feedback for Physical Exercises.* ACCV 2022.

1.4 Outline

In this section we discuss the outline of this thesis. We have dedicated Chapter 1 as an introduction to the field of human motion analysis, discussed our motivation in studying this field, introduced the specific problems we will tackle and defined the challenges. Chapter 2 is dedicated to the related work on human motion analysis, focusing specifically on the topics of active human pose estimation, motion prediction, action recognition, and physical exercise analysis. In Chapter 3, we present our work on optimized viewpoint selection for 3D human pose estimation. We present our active strategy to choose the next best trajectory for accurate human pose estimation. In Chapter 4, we discuss our first proposed approach to the human motion prediction problem via temporal inception modules. In Chapter 5, we extend our motion prediction horizon to 5 seconds and discuss our work on long-term human motion prediction. We introduce our concept of keyposes and how they can be used to synthesize realistic future motions. In Chapter 6, we focus on a particular application of motion analysis: physical exercises. We introduce our framework, capable of both recognizing the incorrectly performed exercises and giving personalized feedback to the user. In Chapter 7, we summarize our findings and discuss directions for future research.

2 Related Work

We focus on several related problems under the domain of human motion analysis and synthesis. In this section, we will consider the related works on these problems. We will first start with active human pose estimation and introduce works which focus on pose estimation while taking the camera placement into account. Afterwards, we discuss works on human motion prediction and introduce the deep learning architectures that are primarily being used. Finally, we discuss the physical exercise analysis field, for which we first introduce the more general field of action recognition, then focus on works that primarily target sports applications.

2.1 Active Human Pose Estimation

Most recent approaches to 3D pose estimation rely on deep networks that regress pose from monocular images [75, 109, 110, 125, 126, 131, 139, 159, 165, 167, 175, 184, 195]. These methods tend to rely on static cameras in the scene and do not consider the effect that actively controlling the camera has on accuracy. We focus on the research direction that also considers the effects of a moving camera. We introduce the works that optimize camera placement in multi-camera setups and those that guide robots in a previously-unknown environment, which are integral to our discussion in Chapter 3.

2.1.1 Optimal Camera Placement for Motion Capture

Optimal camera placement is a well-studied problem in the context of static multi-view setups. Existing solutions rely on maximizing image resolution while minimizing self-occlusion of body parts [4, 32] or target point occlusion and triangulation errors [133]. However, these methods operate offline and on pre-recorded exemplar motions. This makes them unsuitable for motion capture using a single moving camera that films *a priori* unknown motions in a much larger scene where estimation noise can be high. Pirinen *et al.* [130] optimize multiple cameras poses for triangulation of joints in a dome environment using a self-supervised

reinforcement learning approach. On the other hand we consider the monocular problem in Chapter 3, which we approach not with a learning based method, but with optimization using the loss function itself.

2.1.2 View Planning for Static and People Reconstruction

There has been much robotics work on active reconstruction and view planning. This usually involves moving so as to maximize information gain while minimizing motion cost, for example by a discretizing space into a volumetric grid and counting previously unseen voxels [41, 72] or by accumulating estimation uncertainty [123]. When a coarse scene model is available, an optimal trajectory can be found using offline optimization [67, 137]. Gebhardt *et al.* [53] also apply this method to achieve desired aesthetic properties in cinematography. Another approach is to use reinforcement learning to define policies [35] or to learn a metric [66] for later online path planning. These methods deal with rigid unchanging scenes, except the method of Cheng *et al.* [33] that performs volumetric scanning of people during information gain maximization. However, this approach can only deal with very slowly moving people who stay where they are.

2.1.3 Human Motion Capture on Drones

Drones can be viewed as flying cameras and are therefore natural applications for active pose estimation methods. One problem, however, is that the drone must keep the person in its field of view. To achieve this, the algorithm of Zhou *et al.* [196] uses 2D human pose estimation in a monocular video and non-rigid structure from motion to reconstruct the articulated 3D pose of a subject, while that of Naegeli *et al.* [116] reacts online to the subject's motion to keep them in view and to optimize for screen-space framing objectives. AirCap [141] calculates trajectories of multiple drones that aim to keep the person in view while simultaneously performing object avoidance. This was extended by Tallamraju *et al.* [162] so as to optimize multiple MAV trajectories by minimizing the uncertainty of the global human position.

In [117], this was integrated into an autonomous system that actively directs a swarm of drones and simultaneously reconstructs 3D human and drone poses from onboard cameras. This strategy implements a pre-defined policy to stay at constant distance to the subject and uses pre-defined view angles (90° between two drones) to maximize triangulation accuracy. This enables mobile large-scale motion capture, but relies on markers for accurate 2D pose estimation. Xu *et al.* [177] use three drones for markerless motion capture, using an RGBD video input for tracking the subject.

In short, existing methods either optimize for drone placement but for mostly rigid scenes, or estimate 3D human pose but without optimizing the camera placement. Other works such as that of Pirinen *et al.* [130] perform optimal camera placement for multiple cameras. In Chapter 3, we propose an approach that aims to find the best next drone location for monocular view

so as to maximize 3D human pose estimation accuracy.

2.2 Human Motion Prediction

Before the deep learning era, analytical models of human motion have been developed by restricting the human motions to simpler or cyclic trajectories like walking, or a golf swing [122, 170]. However, these models do not generalize well to more complex motions. The availability of large human motion datasets makes deep learning an ideal framework for tackling the task of motion prediction. In this section, we first review the two main classes of deep models that have been used in the field and then discuss approaches that depart from these main trends. These works, in particular the ones using GCNs, have been the building blocks to our motion prediction frameworks in Chapters 4 and 5, and our physical exercise feedback framework in Chapter 6.

2.2.1 Human Motion Prediction using RNNs

Recurrent neural networks (RNN) are widely used architectures for modeling time-series data, for instance for natural-language processing [186] and music generation [157, 151]. Since the work of Fragkiadaki *et al.* [50], these architectures have become highly popular for human motion forecasting. In this context, the S-RNN of Jain *et al.* [73] transforms spatio-temporal graphs to a feedforward mixture of RNNs; the Dropout Autoencoder LSTM (DAE-LSTM) of Ghosh *et al.* [54] synthesizes long-term realistic looking motion sequences; the recent Generative Adversarial Imitation Learning (GAIL) of Wang *et al.* [174] was employed to train an RNN-based policy generator and critic networks. HP-GAN [16] uses an RNN-based GAN architecture to generate diverse future motions of 30 frames. Luo *et al.* [100] build an RNN based video autoencoder framework which produces embeddings used for action recognition and motion prediction.

Despite their success, using RNNs for long-term motion prediction suffers from drawbacks. As shown by Martinez *et al.* [108], they tend to produce discontinuities at the transition between observed and predicted poses, and often yield predictions that converge to the mean pose of the ground-truth data in the long term. In [108], this was circumvented by adding a residual connection so that the network only needs to predict the residual motion. In Chapter 5, we also develop an RNN-based architecture. However, because we treat keypose prediction as a classification task, our approach does not suffer from the accumulated errors that such models tend to generate when employed for regression.

2.2.2 Human Motion Prediction using GCNs

Mao *et al.* [106] proposed to overcome the weaknesses of RNNs by encoding motion in discrete

cosine transform (DCT) space, to model temporal dependencies, and learning the relationships between the different joints via a GCN. In Chapter 4 we build on top of this work by combining a GCN architecture with a temporal inception layer. The temporal inception layer serves to process the input at different subsequence lengths, so as to exploit both short-term and long-term information. Alternatively, [105, 76] combine the GCN architecture with an attention module aiming to learn the repetitive motion patterns. These works were designed for forecasting up to 1 second in the future. As will be shown by our experiments in Chapter 5, for longer timespans, they tend to degenerate to static predictions.

Nevertheless, GCN models have proven to be highly suitable architectures for processing human motions, due to the graph-like connection of human body joints. In Chapter 6 for physical exercise feedback, we make use of GCN architectures. Our motion correction branch is inspired by [106], but instead of forecasting future motion, we synthesize correctly performed exercises.

2.2.3 Other Human Motion Prediction Approaches

Several other architectures have been proposed for human motion prediction. For example, Bütepage *et al.* [26] employ several fully-connected encoder-decoder models to encode different properties of the data. One of the models is a time-scale convolutional encoder, with different filter sizes. In [27], a conditional variational autoencoder (CVAE) is used to probabilistically model, predict and generate future motions. This probabilistic approach is extended in [28] to incorporate hierarchical action labels. Aliakbarian *et al.* [10] also perform motion generation and prediction by encoding their inputs using a CVAE. They are able to generate diverse motions by randomly sampling and perturbing the conditioning variables. Similarly, Yuan *et al.* [182] also use a CVAE based approach to generate multiple futures. Li *et al.* [91] use a convolutional neural network for motion prediction, producing separate short-term and long-term embeddings. In [38, 1], interactions between humans and objects in the scene are learned for context-aware motion prediction. Aksan *et al.*[6] use transformer networks to predict up to 20 seconds in the future, but only for cyclic motions. Zhou *et al.* [197] also target long term predictions, but provide only qualitative results for sequences from walking, dancing, and martial arts, which tend to follow well-structured patterns. Diller *et al.* [43] use characteristic 3D poses resembling our keyposes for long-term motion prediction. However, these poses are manually annotated rather than automatically extracted from sequences. A different related task is to generate realistic motions by conditioning on the action label, rather than the past motion [129, 59].

In Chapter 4, we focus on developing a motion prediction method that gives highly accurate results within the 1 second time horizon. In Chapter 5 which focuses on long term motion prediction we show that truly long-term prediction can be achieved more accurately by focusing the prediction on the essential poses, or keyposes, in a sequence. These poses are extracted automatically from the sequence, without manual annotations.

2.3 Exercise Analysis and Feedback

Our work in Chapter 6 is at the intersection of several sub-fields of computer vision: (i) We draw inspiration from GCN based human motion prediction architectures which have already discussed in Section 2.2.2; (ii) we identify the users' mistakes in an action recognition fashion; and (iii) we address the task of automated physical exercise analysis. We discuss action recognition and automated physical exercise analysis below.

2.3.1 Action Recognition

Although there is a vast literature on video-based action recognition [124, 158], in this thesis we focus on its skeleton-based counterpart, as our approach in Chapter 6 also processes 3D poses. Early deep learning based approaches to skeleton-based action recognition mostly relied on RNNs [46, 95, 96, 145, 156]. Li *et al.* [93] used convolutional neural networks (CNNs) to extract features hierarchically by first finding local point-level features and gradually extracting global spatial and temporal features. Zhang *et al.* [188] designed CNN and RNN networks that are robust to viewpoint changes.

Recently, [90, 163, 189] employed GCNs for action recognition. Specifically, Tang *et al.* [163] designed a reinforcement learning scheme to select the most informative frames and feed them to a GCN. Li *et al.* [90] developed a GCN framework that not only models human joint connections, but also learns to infer "actional-links", which are joint dependencies learned from the data.

Zhang *et al.* [189] designed a two-module network, consisting of a first GCN-based module that extracts joint-level information and a second frame-level module capturing temporal information via convolutional layers and spatial and temporal max-pooling. In Chapter 6, our classification branch borrows ideas from Mao *et al.*'s [106] and Zhang *et al.*'s [189] architectures. It is composed of graph convolutional blocks as proposed by Mao *et al.* [106] combined with the frame-level module architecture proposed by Zhang *et al.* [189].

2.3.2 Physical Exercise Analysis

Physical exercise analysis aims to prevent injuries that may arise when a person performs motions incorrectly. In its simplest form, such an analysis amounts to detecting whether the subject performs the exercise correctly or not. This was achieved several works [31, 44, 134] by exploiting 2D poses extracted from the input images. In particular, Dittakavi *et al.* [44] detected which joints need to be fixed by finding the overall joint angle distribution of the dataset and detecting poses in which a joint angle is an anomaly. This framework operates on single frames, as opposed to our method which operates on entire sequences. In [185], Zell *et al.* represented the human body as a mass-spring model and analyzed the extension torque on certain joints, allowing them to classify whether a motion is performed correctly or not. While

useful, such classification-based approaches offer limited information to the user, as they do not provide them with any feedback about the specific type of mistakes they made. Moreover, most of existing works operate on 2D pose inputs [31, 44, 74, 134, 180]. Similar to [49], we also design our framework to work with 3D poses enabling us to be robust to ambiguities found in 2D poses.

While a few works took some steps toward giving feedback [49, 74, 180], this was achieved in a hard-coded fashion, by thresholding angles between some of the body joints. As such, this approach relies on manually defining such thresholds, and thus does not easily extend to new exercises. Furthermore, it does not provide the user personalized corrective measures in a visual manner, by demonstrating the correct version of their performance. We address this in Chapter 6 by following a data driven approach able to automatically learn the different "correct" forms of an exercise, and that can easily extend to different types of exercises and mistakes. To the best of our knowledge, our framework is the first to both identify mistakes and suggest personalized corrections to the user.

3 Optimized Viewpoint Selection for Active Human Motion Capture

Monocular approaches for 3D human pose estimation has been a popular research topic due to the widespread use of single RGB camera systems. However, obtaining accurate pose reconstructions can be challenging if the viewpoints chosen for image capture are not ideal.

In this chapter, we explore the use of a moving camera whose motion we can control to resolve ambiguities inherent to monocular 3D reconstruction and to increase pose estimation accuracy. This is known as *active vision* and has received surprisingly little attention in the context of using modern approaches to body pose estimation. An active motion capture system, such as one based on a personal drone, would allow one to film themselves performing a physical activity and analyze their motion, for example to get feedback on their performance. When using only one camera, the quality of such feedback will strongly depend on selecting the most beneficial viewpoints for pose estimation. Fig. 3.1 depicts an overview of our approach based on a drone-based monocular camera.

In this paper, we introduce an algorithm designed to continuously position a moving camera at optimal viewpoints to maximize the 3D pose estimation accuracy for a freely moving subject. We achieve this by moving the camera in 6D pose space to viewpoints that maximize a utility function designed to predict reconstruction accuracy. However, the utility function cannot be defined in terms of reconstruction accuracy because doing so would require knowing the true person and camera position, leading to a chicken and egg problem. Instead we use prediction uncertainty as a surrogate for accuracy. This is a common strategy used in robot navigation systems for unknown scenes where the robot explores areas that are most incomplete in its internal map representation [123]. However, in our situation, estimating uncertainty is much more difficult since multiple sources of uncertainty need to be considered. These include uncertainties about what the subject will do next, the reliability of the pose estimation algorithm, and the accuracy of distance estimation along the camera's line of sight.

Our key contribution is therefore a formal model that provides an estimate of the *posterior variance* and probabilistically fuses these sources of uncertainty with appropriate prior distributions. This has enabled us to develop an active motion capture technique that takes raw

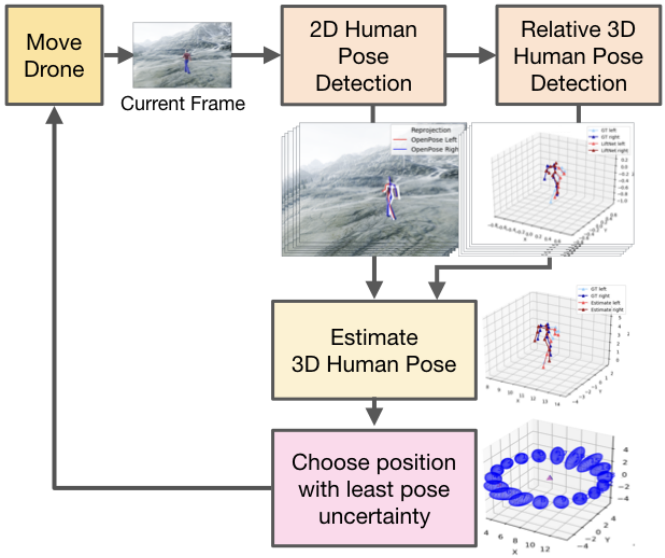


Figure 3.1 – **Method overview.** The 2D and 3D human pose is inferred from the current frame of the drone footage, using off the shelf CNNs. The 2D pose and relative 3D pose of the last k frames is then used to optimize for the global 3D human motion. The next view of the drone is chosen so that the uncertainty of the human pose estimation from that view is minimized, which improves reconstruction accuracy.

video footage as input from a moving aerial camera and continuously computes future target viewpoints for positioning the camera, in a way that is optimized for human motion capture. We demonstrate our algorithm in two different scenarios and compare it against standard heuristics, such as constantly rotating around the subject and maintaining a constant angle with respect to the subject. We find that when allowed to choose the next viewpoint without physical constraints, our algorithm outperforms the baselines consistently. For simulated drone flight, our results are on par with constant rotation, which we conclude is the best trajectory to choose in the case of no obstacles blocking the circular flight path. Our code is available at https://github.com/senakicir/ActiveMoCap.

3.1 Methodology

Our goal is to continuously position the camera in 6D pose space so that the acquired by the camera can be used to achieve the best overall human pose estimation accuracy. What makes this problem challenging is that, when we decide where to send the camera, we do not yet know where the subject will be and in what position exactly. We therefore have to guess. To this end, we propose the following three-step approach depicted by Fig. 3.1:

1. Estimate the 3D pose up to the current time instant.
2. Predict the person's future location and 3D pose at the time the camera acquires the next image, including an uncertainty estimate.

3. Select the optimal camera pose based on the uncertainty estimate and move the camera to that viewpoint.

We will consider two ways the camera can move. In the first case, the camera can teleport from one location to the next without restriction, allowing us to explore the theoretical limits of our approach. Such a teleportation mode can be simulated using a multi-camera setup, enabling us to evaluate our model on both simulated data and real image datasets acquired from multiple viewpoints. In the second, more realistic scenario, the camera is carried by a simulated drone, and we must take into account physical limits about the motion it can undertake.

3.1.1 3D Pose Estimation

The 3D pose estimation step takes as input the video feed from the on-board camera over the past N frames and outputs for each frame, $t \in (1,\dots,N)$, the 3D human pose, represented as 15 3D points $\mathbf{X}^t \in \mathbb{R}^{15\times 3}$, and the drone pose, as 3D position and rotation angles $\mathbf{D}^t \in \mathbb{R}^{2\times 3}$. Our focus is on estimating the 3D human pose using the real-time method proposed by [29], which detects the 2D locations of the human's major joints in the image plane, $\mathbf{M}^t \in \mathbb{R}^{15\times 2}$, and the subsequent use of [165], which lifts these 2D predictions to 3D pose, $\mathbf{L}^t \in \mathbb{R}^{15\times 3}$. However, these per-frame estimates are error prone and relative to the camera. To remedy this, we fuse 2D and 3D predictions with temporal smoothness and bone-length constraints in a space-time optimization. This exploits the fact that the drone is constantly moving so as to disambiguate the individual estimates. The bone lengths, $\mathbf{b}_{\text{calib}}$, of the subject's skeleton are computed during an apriori calibration stage, where the subject has to stand still for 20 seconds. This is performed only once for each subject. Formally, we optimize for the global 3D human pose by minimizing an objective function E_{pose}, which we detail below.

Formulation

Our primary goal is to improve the global 3D human pose estimation of a subject changing position and pose. We optimize the time-varying pose trajectories across the last k frames. Let t be the last observed frame. We capture the trajectory of poses $\mathbf{X}^{t-k}$ to $\mathbf{X}^t$ in the pose matrix $\mathbf{X}$. We then write an energy function

$$E_{\text{pose}} = E_{\text{proj}}(\mathbf{X},\mathbf{M},\mathbf{D}) + E_{\text{lift}}(\mathbf{X},\mathbf{L}) + E_{\text{smooth}}(\mathbf{X}) + E_{\text{bone}}(\mathbf{X},\mathbf{b})\,. \tag{3.1}$$

The individual terms are defined as follows. The lift term, E_{lift}, leverages the 3D pose estimates, $\mathbf{L}$, from LiftNet [165]. Because these are relative to the hip and without absolute scale, we subtract the hip position from our absolute 3D pose, $\mathbf{X}^t$, and apply a scale factor m to $\mathbf{L}$ to match the bone lengths $\mathbf{b}_{\text{calib}}$ in the least-square sense. We write

$$E_{\text{lift}}(\mathbf{X},\mathbf{L}) = \omega_l \sum_{i=t-k}^{t} \| m\cdot \mathbf{L}^i - (\mathbf{X}^i - \mathbf{X}^i_{\text{hip joint}})\|_2^2\,, \tag{3.2}$$

with ω_l its relative weight.

The projection term measures the difference between the detected 2D joint locations and the projection of the estimated 3D pose in the least-square sense. We write it as

$$E_{\text{proj}}(\mathbf{X},\mathbf{M},\mathbf{D}) = \omega_p \sum_{i=t-k}^{t} \|\mathbf{M}^i - \Pi(\mathbf{X}^i,\mathbf{D}^i,\mathbf{K})\|_2^2 , \tag{3.3}$$

where Π is the perspective projection function, $\mathbf{K}$ is the matrix of camera intrinsic parameters, and ω_p is a weight that controls the influence of this term.

The smoothness term exploits that we are using a continuous video feed and that the motion is smooth by penalizing velocity computed by finite differences as

$$E_{\text{smooth}}(\mathbf{X}) = \omega_s \sum_{i=t-k+1}^{t} \|(\mathbf{X}^{i+1} - \mathbf{X}^i)\|_2^2 . \tag{3.4}$$

with ω_s as its weight.

To further constrain the solution space, we use our knowledge of the bone lengths $\mathbf{b}_{\text{calib}}$ found during calibration and penalize deviations in length. The length of each bone b in the set of all bones b_{all} is found as $\mathbf{b}_b^t = \|(\mathbf{X}_{b_1} - \mathbf{X}_{b_2})\|_2$ for frame t. The bone length term is then defined as

$$E_{\text{bone}}(\mathbf{X}) = \omega_b \sum_{i=t-k}^{t} \sum_{b \in b_{\text{all}}} d(\mathbf{b}_b^i, \mathbf{b}_{\text{calib},b}) , \tag{3.5}$$

with ω_b as its weight.

The complete energy E_{pose} is minimized by gradient descent at the beginning of each control cycle, to get a pose estimate for control. The resulting pose estimate $\hat{\mathbf{X}}$ is the maximum a posteriori estimate in a probabilistic view.

Calibration Mode

Calibration mode only has to be run once for each subject to find the bone lengths, $\mathbf{b}_{\text{calib}}$. In this mode, the subject is assumed to be stationary. The situation is equivalent to having the scene observed from multiple stationary cameras, such as in [136]. We find the single static pose $\mathbf{X}^{\text{c}}$ that minimizes

$$E_{\text{calib}} = E_{\text{proj}}(\mathbf{X}^{\text{c}},\mathbf{M},\mathbf{D}) + E_{\text{symmetry}}(\mathbf{X}^{\text{c}}). \tag{3.6}$$

In this objective, the projection term, E_{proj}, is akin to the one in our main formulation but acts on all calibration frames. It can be written as

$$E_{\text{proj}}(\mathbf{X}^{\text{c}},\mathbf{M},\mathbf{D}) = \omega_{\text{p}} \sum_{i=0}^{t} \|\mathbf{M}^i - \Pi(\mathbf{X}^{\text{c}},\mathbf{D}^i,\mathbf{K})\|_2^2 , \tag{3.7}$$

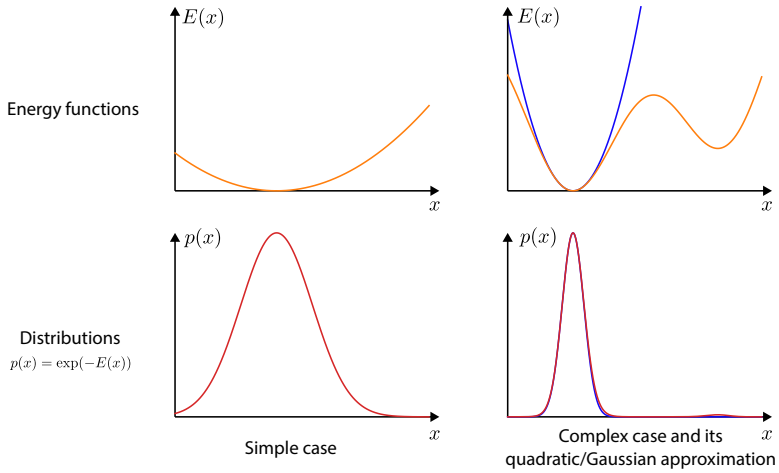


Figure 3.2 – **Probabilistic interpretation.** Left: A quadratic energy function and its associated Gaussian error distribution. Right: A complex energy function, which is locally approximated with a Gaussian (blue) near the minimum. The curvature of the energy function is a measure of the confidence in the estimate and the variance of the associated error distribution. The energy on the right is more constrained and its error distribution has a lower variance.

with ω_{p} controlling its influence. The symmetry term, E_{symmetry}, ensures that the left and right limbs of the estimated skeleton have the same lengths by penalizing the squared difference of their lengths.

3.1.2 Next Best View Selection

Our goal is to find the next best view for the drone at the future time step $t+1$, $\mathbf{D}^{t+1}$. We will model the uncertainty of the pose estimate in a probabilistic setting. Let $p(\mathbf{X}|\mathbf{M},\mathbf{D},\mathbf{L},\mathbf{b})$ be the posterior distribution of poses. Then, E_{pose} is its negative logarithm and its minimization corresponds to maximum a posteriori (MAP) estimation. In this formalism, the sum of the individual terms in E_{pose} models that our posterior distribution is composed of independent likelihood and prior distributions. For a purely quadratic term, $E(x) = \omega(x-\mu)^2$, the corresponding distribution $p_E = \exp(-E)$ is a Gaussian with mean μ and standard deviation $\sigma = \frac{1}{\sqrt{2\omega}}$. Notably, σ is directly linked to the weight ω of the energy. Most of our energy terms involve non-linear operations, such as perspective projection in E_{proj}, and therefore induce non-Gaussian distributions, as visualized in Fig. 3.2. Nevertheless, as for the simple quadratic case, the weights ω_p and ω_l of E_{proj} and E_{lift} can be interpreted as surrogates for the amount of measurement noise in the 2D and 3D pose estimates.

A good measure of uncertainty is the sum of the eigenvalues of the covariance Σ_p of the underlying distribution p. The sum of the eigenvalues captures the spread of a multivariate distribution with a single variable, similarly to the variance in the univariate case. To exploit this uncertainty estimation for our problem, we now extend E_{pose} to model not only the current and past poses but also the future ones and condition it on the choice of the future

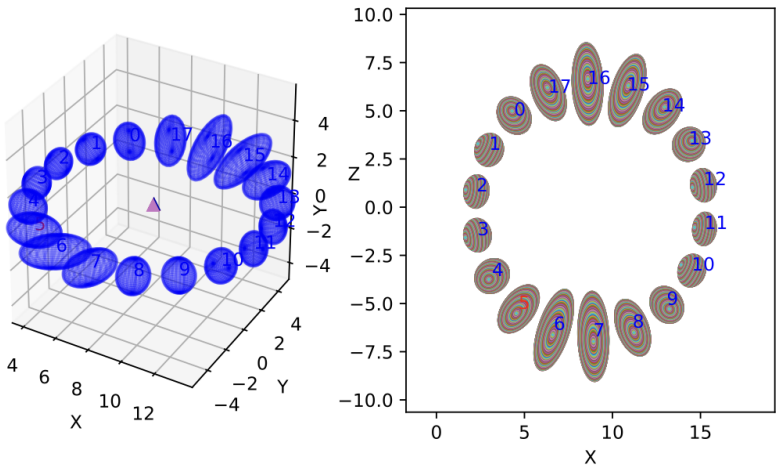


Figure 3.3 – **Uncertainty estimates** for each candidate drone position, visualized on the left as 3D ellipsoids and on the right from a 2D top-down view. Each ellipse visualizes the eigenvalues of the hip location when incorporating an additional view from its displayed position. Here, the previous image was taken from the top (position 16) and uncertainty is minimized by moving to an orthogonal view. The complete distribution has more than three eigenvectors and cannot straightforwardly be visualized in 3D.

drone position. To determine the best next drone pose, we sample candidate positions and chose the one with the lowest uncertainty. This process is illustrated in Figure 3.3.

Future pose forecasting. In our setting, accounting for the dynamic motion of the person is key to successfully positioning the camera. We model the motion of the person from the current frame t to the next M future frames $t+i$, $i \in (1,\dots,M)$ linearly, i.e. we aim to keep the velocity of the joints constant across our window of frames. We also constrain the future poses by the bone length term. The future pose vectors $\mathbf{X}^{t+i}$ are constrained by the smoothness and bone length terms, but for now not by any image-based term since the future images are not yet available at time t. Minimizing this extended E_{pose} for future poses gives the MAP poses $\hat{\mathbf{X}}^{t+i}$. It continues the motion $\hat{\mathbf{X}}^{t-k,\cdots,t+K}$ smoothly while maintaining the bone lengths. As we predict only the near future, we have found this simple extrapolation to be sufficient. We leave as future work the use of more advanced methods [50, 187] to forecast further.

Future measurement forecasting. We aim to find the future drone position, $\mathbf{D}^{t+1}$, that reduces the posterior uncertainty, but we do not have footage from future viewpoints to condition the posterior on. Instead, we use the predicted future human pose $\hat{\mathbf{X}}^{t+i}$, $i \in (1,\dots,M)$, as a proxy for $\mathbf{L}^{t+i}$ and approximate $\mathbf{M}^{t+i}$ with the projection

$$\hat{\mathbf{M}}^{t+1} = \Pi(\hat{\mathbf{X}}^{t+1}, \mathbf{D}^{t+1}, \boldsymbol{K})\,. \tag{3.8}$$

At first glance, constraining the future pose on these virtual estimates in E_{pose} does not add anything since the terms E_{proj} and E_{lift} are zero at $\hat{\mathbf{X}}^{t+1}$ by this construction. However, it

changes the energy landscape and models how strong a future observation would constrain the pose posterior. In particular, the projection term, E_{proj}, narrows down the solution space in the direction of the image plane but cannot constrain it in the depth direction, creating an elliptical uncertainty as visualized in Fig 3.3. The combined influence of all terms is conveniently modeled as the energy landscape of E_{pose} and its corresponding posterior.

In our current implementation we assume that the 2D and 3D detections are affected by pose-independent noise, and their variance is captured by ω_p and ω_l, respectively. These factors could, in principle, be view dependent and in relation to the person's pose. For instance, [30] may be more accurate at reconstructing a front view than a side view. However, while estimating the uncertainty in deep networks is an active research field [132], predicting the expected uncertainty for an unobserved view has not yet been attempted for pose estimation. It is an interesting future work direction.

Variance estimator. E_{pose} and its corresponding posterior has a complex form due to the projection and prior terms. Hence, the sought-after covariance Σ_p cannot be expressed in closed form and approximating it by sampling the space of all possible poses would be expensive. Instead, for the sake of uncertainty estimation, we approximate $p(\mathbf{X}|\mathbf{D},\mathbf{M},\mathbf{L},\mathbf{b})$ locally with a Gaussian distribution q, such that

$$\Sigma_{p(\mathbf{X}|\mathbf{D},\mathbf{M},\mathbf{L})} \approx \Sigma_q \text{ where } q = N(\mathbf{X}|\hat{\mathbf{X}},\Sigma_q)\,, \tag{3.9}$$

with $\hat{\mathbf{X}}$ and Σ_q the Gaussians mean and covariance matrix, respectively. Such an approximation is exemplified in Figure 3.2. For a Gaussian, the covariance of q can be computed in closed form as the inverse of the Hessian of the negative log likelihood[1], $\Sigma_q = H^{-1}_{-\log q}$, where $H_{-\log q} = \frac{\partial^2 -\log q(\mathbf{X})}{\partial \mathbf{X}}\Big|_{\mathbf{X}=\hat{\mathbf{X}}}$. Under the Gaussian assumption, Σ_p is thereby well approximated by the second order gradients, $H^{-1}_{E_{\text{pose}}}$, of E_{pose}. Our experiments show that this simplification holds well for the introduced error terms.

To select the view with minimum uncertainty among a set of K candidate drone trajectories, we therefore

1. Optimize E_{pose} once to forecast M human poses $\hat{\mathbf{X}}^{t+i}$, for $1 \le i \le M$.
2. Use these forecasted poses to set $\hat{\mathbf{L}}^{t+i}$ and $\hat{\mathbf{M}}^{t+i}$ for each $1 \le i \le M$ for each candidate trajectory c.
3. Compute the second order derivatives of E_{pose} for each c, which form H_c.
4. Compute and sum up the respective eigenvalues to select the candidate with the least uncertainty.

Discussion. In principle, $p(\mathbf{X}|\mathbf{M},\mathbf{D},\mathbf{L},\mathbf{b})$, i.e. the probability of the most likely pose, could

[1] A derivation can be found at [183].

also act as a measure of certainty, as implicitly used in [133] on a known motion trajectory to minimize triangulation error. However, the term $E_{\text{proj}}(\hat{\mathbf{X}}, \hat{\mathbf{M}})$ of E_{pose} is zero for the future time step $t+i$, because the projection of $\hat{\mathbf{X}}^{t+i}$ is by construction equal to $\hat{\mathbf{M}}^{t+i}$ and therefore uninformative. Another alternative that has been proposed in the literature is to approximate the covariance through first order estimates [166], as a function of the Jakobi matrix. However, as also the first order gradients of E_{proj} vanish at the MAP estimate, this approximation is not possible in our case.

3.1.3 Drone Control Policies and Flight Model

In the experiments where we simulate drone flight, the algorithm decides between 9 candidate trajectories in the directions up, down, left, right, up-right, up-left, down-right, down-left and center. To ensure that the drone stays a fixed distance away from the person, the direction vector is normalized by the fixed-distance value.

In the remainder of this section, we describe how we model the flight of the drone so that we can predict the position of the drone along a potential trajectory in future time steps. By forecasting the future M locations of the drone on a potential trajectory c, we can predict the 2D pose estimations $\hat{\mathbf{M}}^{t+i}$ for each $\{i\}_{i=1}^{M}$ more accurately.

We control the flight of our drone by passing it the desired velocity vector and the desired yaw rotation amount with the maximum speed kept constant at 5 m/s. The drone is sent new commands once every $\Delta t = 0.2$ seconds.

We model the drone flight in the following manner. We assume that the drone moves with constant acceleration during a time step Δt. If the drone has current position x_{current} and velocity V_{current}, then with an current acceleration a_{current}, its next position x_{goal} in Δt time will be

$$x_{\text{goal}} = x_{\text{current}} + V_{\text{current}}\Delta t + 0.5 a_{\text{current}}\Delta t^2 \,. \tag{3.10}$$

The current acceleration at time t is found as a weighted average of the input acceleration a_{input} and the acceleration of the previous step a_{previous}. This can be written as

$$a_{\text{current}} = \alpha a_{\text{input}} + (1-\alpha) a_{\text{previous}}. \tag{3.11}$$

a_{input} is determined according to the candidate trajectory being evaluated. The direction of the acceleration vector is set to the direction of the candidate trajectory. We determine the magnitude of the input acceleration through least-square minimization of the difference between the predicted x_{goal} and the actual drone position. α is found by line search.

By estimating the future positions of the drone, we are able to forecast more accurate future 2D pose estimations, leading to more accurate decision making. Examples of predicted trajectories are shown in Figure 3.4.

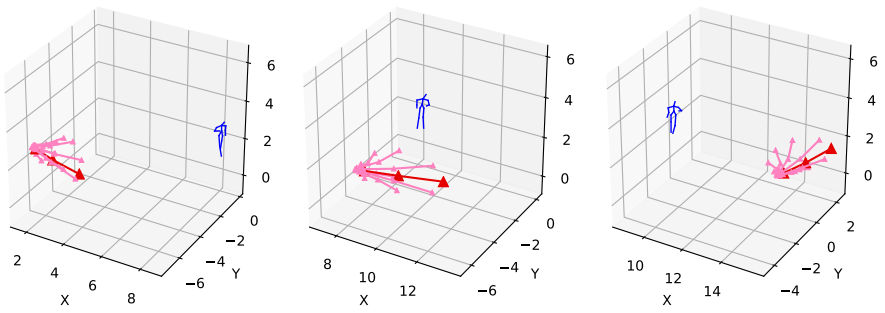


Figure 3.4 – **Predicted trajectories** as the drone is circling the subject. The future drone positions are predicted for the future 3 steps, represented by triangle markers on the trajectories. Red depicts the chosen trajectory.

3.2 Evaluation

In this section we evaluate the improvement on 3D human pose estimation that is achieved through optimization of the drone flight.

Simulation environment. Although [135, 29, 165] run in real time, and online SLAM from a monocular camera [42] is possible, we use a drone simulator since the integration of all components onto constrained drone hardware is difficult and beyond our expertise. We make simulation realistic by driving our characters with real motion capture data from the CMU Graphics Lab Motion Capture Database [36] and using the AirSim [144] drone simulator that builds upon the Unreal game engine and therefore produces realistic images of natural environments. Simulation also has the advantage that the same experiment can be repeated with different parameters and be directly compared to baseline methods and ground-truth motion.

Simulated test set. We test our approach on three CMU motions of increasing difficulty: *Walking* straight (subject 2, trial 1), *Dance* with twirling (subject 5, trial 8), and *Running* in a circle (subject 38, trial 3). Additionally, we use a validation set consisting of *Basketball* dribble (subject 6, trial 13), and *Sitting* on a stool (subject 13, trial 6), to conduct a grid search for hyperparameters.

Real test set. To show that our planner also works outside the simulator, we evaluate our approach on a section of the MPI-INF-3DHP dataset, which includes motions such as running around in a circle and waving arms in the air. The dataset provides 14 fixed viewpoints that are at varying distances from one another and from the subject, as depicted in Figure 3.6. In this case, the best next view is restricted to one of the 14 fixed viewpoints. This dataset lets us evaluate whether the object detector of [135], the 2D pose estimation method of [30], and the 3D pose regression technique of [165] are reliable enough in real environments. Since

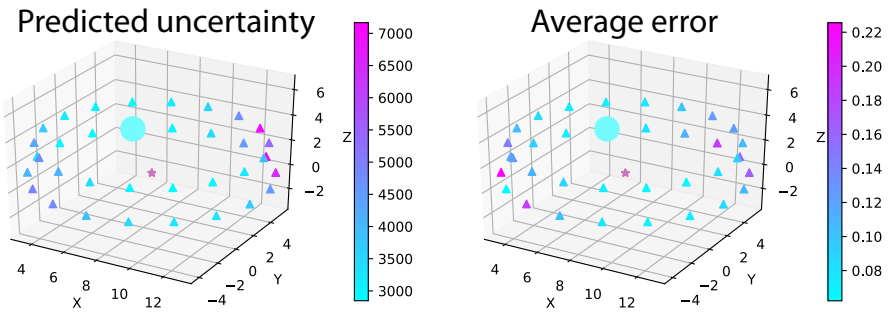


Figure 3.5 – **Uncertainties estimates** across potential viewpoints (left image) compared with the average error we would obtain if we were to visit these locations (right image). The star represents the location of the subject and the large circle depicts the chosen viewpoint according to the lowest uncertainty.

we cannot control the camera in this setting, we remove those cameras from the candidate locations where we predict that the subject will be out of the viewpoint.

	Noisy ground truth				Networks	
	CMU-Walk	CMU-Dance	CMU-Run	MPI-INF-3DHP	MPI-INF-3DHP	Total
Oracle	0.101±0.001	0.101±0.001	0.109±0.001	0.136±0.002	0.17±0.0005	0.142±0.027
Ours (Active)	**0.113**±0.001	**0.116**±0.003	**0.135**±0.002	**0.145**±0.006	**0.21**±0.0008	**0.144**±0.35
Random	0.123±0.002	0.125±0.003	0.159±0.003	0.286±0.027	0.28±0.03	0.195±0.07
Constant Rotation	0.157±0.002	0.146±0.004	0.223±0.003	0.265±0.010	0.29±0.03	0.216±0.06
Constant Angle	0.895±0.54	0.683±0.31	0.985±0.24	1.45±0.63	1.73±0.61	1.15±0.38

Table 3.1 – **3D pose accuracy on the teleportation experiment**, using noisy ground truth to estimate **M** and **L** in the first three columns, and using the networks of [191, 165] in the fourth column. We outperform all predefined baseline trajectories and approach the accuracy of the oracle that has access to the average error of each candidate position.

Baselines. Existing drone-based pose estimation methods use predefined policies to control the drone position relative to the human. Either the human is followed from a constant angle and the angle is set externally by the user [117] or the drone undergoes a constant rotation around the human [196]. As another baseline, we use a random decision policy, where the drone picks uniformly randomly among the proposed viewpoints. Finally, the oracle is obtained by moving the drone to the viewpoint where the reconstruction in the next time step will have the lowest average error, which is achieved by exhaustively trying all viewpoints *with* the corresponding image in the next time frame.

Hyper parameters. We set the weights of the loss term for the reconstruction as follows: $\omega_p = 0.0001$ (projection), $\omega_s = 1$ (smoothness), $\omega_l = 0.1$ (lift term), $\omega_b = 1$ (bone length), which were found by grid search. We set the weights for the decision making as $\omega_p = 0.001$, $\omega_s = 1$, $\omega_l = 0.1$, $\omega_b = 1$. Our reasoning is, we need to set the weights of the projection and lift terms slightly lower because they are estimated with large noise, which is introduced by the neural networks or as additive noise. However, they do not need to be as low for the uncertainty estimation.

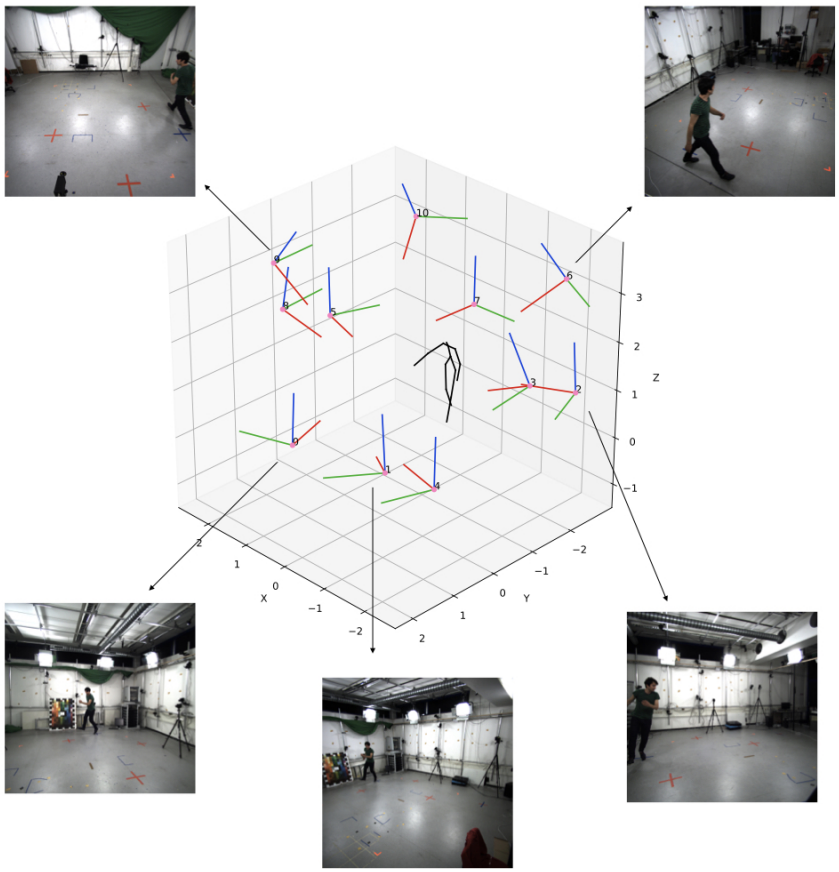


Figure 3.6 – **MPI_INF_3DHP dataset**, which has images taken from 14 viewpoints with various distances to the subject. We use this dataset to evaluate our performance on datasets with realistic camera positioning and real images.

3.2.1 Analyzing Reconstruction Accuracy

We report the mean Euclidean distance per joint in meters in the middle frame of the temporal window we optimize over. For teleportation mode, the size of the temporal window is set to $k = 2$ past frames and 1 future frame, and for the drone flight simulations, to $k = 6$ for past frames and 3 future frames.

Simulation Initialization. The frames are initialized by *back-projecting* the 2D joint locations estimated in the first frame, $\mathbf{M}^{t=0}$, to a distance d from the camera that is chosen such that the back-projected bone lengths match with the average human height. We then refine this initialization by running the optimization without the smoothness term, as there is only one frame. All the sequences are evaluated for 120 frames, with the animation sequences played at 5 Hz.

Teleportation Mode. To understand whether our uncertainty predictions for potential viewpoints coincide with the actual 3D pose errors we will have at these locations, we run the

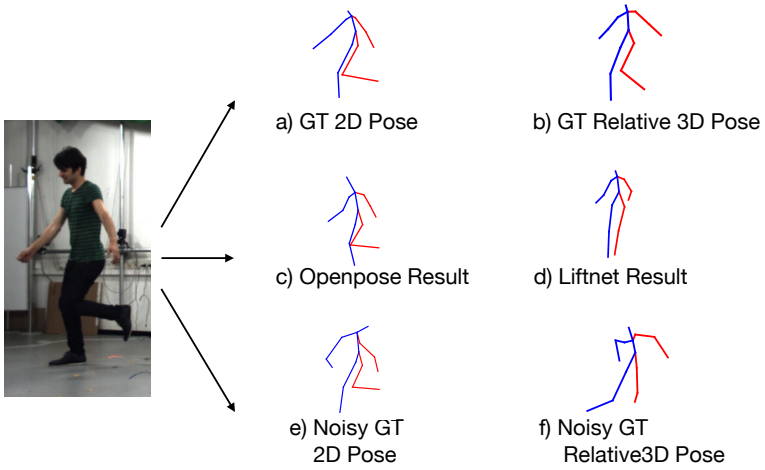


Figure 3.7 – **Example image from the MPI-INF-3DHP dataset** along with the 2D pose detections **M** and 3D relative pose detections **L** obtained using ground truth, noisy ground truth or the networks of [29] and [165]. The noise we add on the ground truth poses is determined according to the statistics of [29] and [165], measured on our validation set.

following simulation: We sample a total of 18 points on a ring around the person, as shown in Fig. 3.5, and allow the drone to teleport to any of these points. We optimize over a total of $k = 2$ past frames and forecast 1 frame into the future. We chose this window size to emphasize the importance of the next choice of frame.

We perform two variants of this experiment. In the first one, we simulate the 2D and 3D pose estimates, $\mathbf{M}, \mathbf{L}$, by adding Gaussian noise to the ground-truth data. The mean and standard deviation of this noise is set as the error of [29] and [165], run on the validation set of animations. Figure 3.7 shows a comparison between the ground truth values, noisy ground truth values and the network results. The results of this experiment are reported in Table 3.1, where we also provide the standard deviations across 5 trials with varying noise and starting from different viewpoints. On the MPI-INF-3DHP dataset, we also provide results using [29] and [165] on the simulator images to obtain the 2D and 3D pose estimates.

Altogether, the results show that our active motion planner achieves consistently lower error values than the baselines and we come the closest to achieving the best possible error for these sequences and viewpoints, despite having no access to the true error. The random baseline also performs quite well in these experiments, as it takes advantage of the drone teleporting to a varied set of viewpoints. The trajectories generated by our active planner and the baselines is depicted in Figure 3.8. Importantly, Figure 3.5 evidences that our predicted uncertainties accurately reflect the true pose errors, thus making them well suited to our goal.

Simulating Drone Flight. To evaluate more realistic cases where the drone is actively controlled and constrained to only move to nearby locations, we simulate the drone flight using the AirSim environment. While simulating drone flight, we target a fixed radius of 7m from the

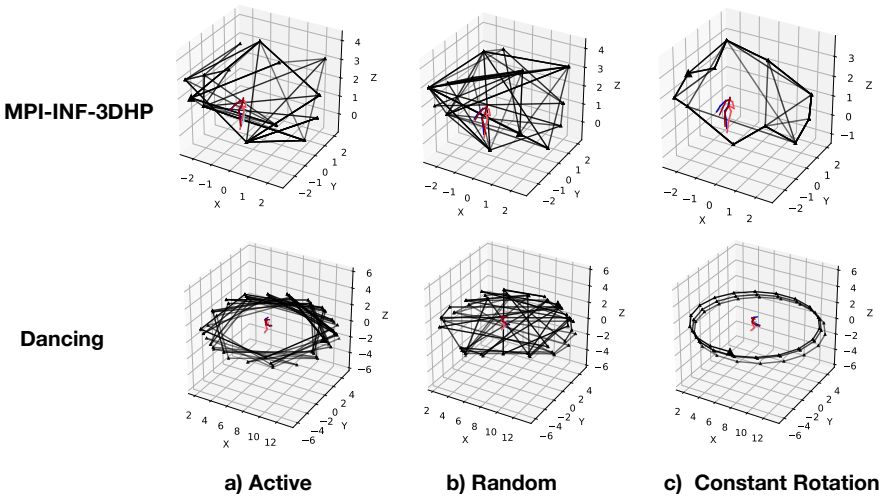


Figure 3.8 – **Trajectories found by our active planner** along with random and constant rotation baselines. The first row depicts the trajectories for the MPI-INF-3DHP dataset, and the second row shows the trajectories for the dancing motion. The trajectories obtained with our algorithm are regular and look different from the random trajectories, especially for the dancing motion. Our algorithm prefers trajectories resulting in large angular variance with respect to the subject between viewpoints.

subject and therefore provide direction candidates that lead to preserving this distance. We do not provide samples at different distances, as moving closer is unsafe and moving farther leads to more concentrated image projections and thus higher 3D errors. We also restrict the drone from flying outside the altitude range 0.25m-3.5m, so as to avoid crashing into the ground and flying above the subject.

In this set of experiments, we *fly* the drone using the simulator's realistic physics engine. To this end, we sample 9 candidate directions towards up, down, left, right, up-right, up-left, down-right, down-left and center. We then predict the 3 consecutive future locations using our simplified (closed form) physics model, to get and estimate where the drone will be at when continuing in each of the 9 directions. We then estimate the uncertainty at these sampled viewpoints and choose the minimum.

We achieve comparable results to constant rotation on simulated drone flight. In fact, except for the first few frames where the drone starts flying, we observe the same trajectory as constant rotation, only the rotation direction varies. Constant rotation being optimal in this setting is not counter-intuitive, as constant rotation is very useful for preserving momentum. This allows the drone to sample viewpoints as far apart from one another as possible, while keeping the subject in view. Figure 3.9 depicts the different baseline trajectories and the active trajectory.

Ablation Study on Our Drone Flight Model. We replace our drone flight model with uniform

	CMU-Walk	CMU-Dance	CMU-Run	Total
Ours (Active)	**0.26**±0.03	0.22±0.04	0.44±0.04	0.31±0.10
Constant Rotation	0.28±0.06	**0.21**±0.04	**0.41**±0.02	**0.30**±0.08
Random	0.60±0.13	0.44±0.19	0.81±0.16	0.62±0.15
Constant Angle	0.41±0.07	0.63± 0.06	1.26±0.17	0.77±0.36

Table 3.2 – **Results of drone full flight simulation**, using noisy ground truth as input to estimate **M** and **L**. The results of constant rotation are the average of 10 runs, with 5 runs rotating clockwise and 5 counter-clockwise. Our approach yields results comparable to those of constant rotation, outperforming the other baselines. The trajectory our algorithm draws also results in a constant rotation, the only difference being the rotation direction.

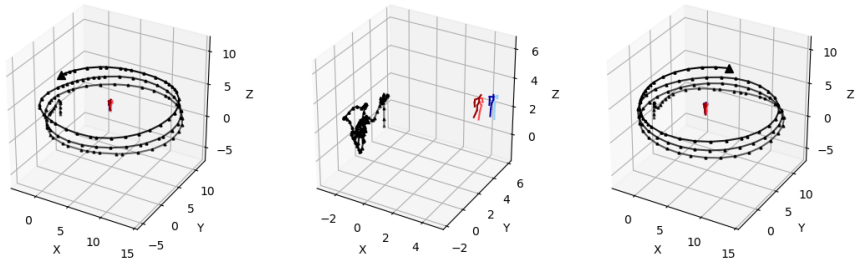


Figure 3.9 – **Trajectories found during flight** by our active planner and the baselines. Our algorithm also chose to perform constant rotation. Because of the drone momentum, the random baseline cannot increase the distance between its camera viewpoints.

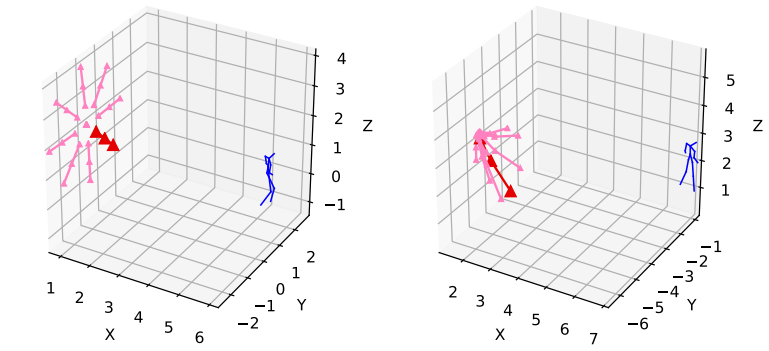
sampling around the drone. This is illustrated in Figure 3.10. We evaluate the performance of our active decision making policy with the uniform sampling in Table 3.3. The trajectories found using this sampling policy is shown in Figure 3.11. We find that the algorithm cannot find the constant rotation policy when we remove the drone flight model and in turn, performs worse.

3.3 Conclusion

We have proposed a theoretical framework for estimating the uncertainty of future measurements from a drone. This permits us to improve 3D human pose estimation by optimizing the drone flight to visit those locations with the lowest expected uncertainty. We have demonstrated with increasingly complex examples, in simulation with synthetic and real footage, that this theory translates to closed-loop drone control and improves pose estimation accuracy. We envision our approach being developed further for improving the performance of athletes and performance artists. It is important to preserve the subjects' privacy in such autonomous systems. We encourage researchers to be sensitive to this issue.

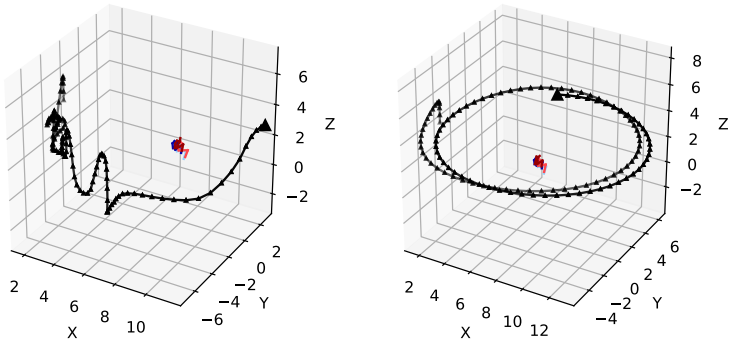
	CMU-Dribble	CMU-Sitting	CMU-Dinosour	Total
Active with Flight Model	**0.28**±0.006	**0.15**±0.007	**0.12**±0.02	**0.18**±0.01
Active w/o Flight Model	0.65±0.09	0.48±0.09	0.22±0.07	0.45±0.08
Constant Rot.	0.30±0.02	**0.15**±0.01	0.15±0.03	0.20±0.02

Table 3.3 – **Ablation study on the importance of having a drone flight model**. We show 3D pose accuracy on simulated drone flight using noisy ground truth for estimating **M** and **L**. We show that we have a large improvement when we use our flight model to predict the future locations of the drone. Using a flight model allows us to find the same trajectories as constant rotation.



a) Future positions w/o flight model b) Future positions with flight model

Figure 3.10 – **The candidate trajectories of the drone** (a) without using our flight model and (b) using our flight model.



a) Active trajectories w/o flight model b) Active trajectories using flight model

Figure 3.11 – **The trajectories drawn by our active decision making policy** (a) without using our flight model and (b) using our flight model. We are able to find the well performing policy of constant rotation when we are using more realistic sampling of future drone positions, found using our drone flight model.

Our framework forecasts human motion by assuming constant velocity. While this is adequate to predict the immediate future, it is not sufficient for longer time horizons. In the next two chapters of this thesis, we address the problem of human motion prediction. In Chapter 4 we consider human motion prediction up to 1 second. Chapter 5 extends this time horizon to 5 seconds.

4 Motion Prediction Using Temporal Inception Module

Human motion prediction is an essential component for a wide variety of applications. For instance, in the field of robotics, robots working closely with humans require an internal representation of the current and future human motion to navigate around them safely [58]. Autonomous driving is another important use case where cars need to forecast pedestrian motion accurately to avoid accidents [60, 48]. Other applications such as sports tracking also heavily use these forecasting methods for better performances, as we have discussed in Chapter 3.

In order to achieve high accuracy motion prediction, we show that the encoding of the body joint trajectories (i.e., sequence of 3D joint locations) is key. In [106] this is achieved by representing each trajectory using its Discrete Cosine Transform (DCT) coefficients [3], a technique previously used to encode human motion for human pose estimation [94, 69]. However, we show that we can gain a large boost in accuracy by using a network to encode the trajectories at multiple temporal scales. In particular, inspired by the Inception Module of [160], we have created a “Temporal Inception Module”, which uses various size convolutional kernels to filter the trajectory at different temporal scales for different input sizes. This allows the network have different receptive fields in the temporal domain.

Following [92, 106], the backbone of our prediction architecture is based on a graph convolutional network (GCN) [25] which is a high capacity feed-forward model. As input to the GCN, Mao *et al.* [106] transform time sequences of joint locations from the 10 past frames into a DCT representation. Moreover, they demonstrate that more frames from past do not help to boost the performance. In our paper we show that by looking at the trajectory at a multiple temporal scale, more frames from the past actually do help to further improve the performance, which is especially true for long-term future motion prediction. Therefore, instead of using the DCT coefficients of the trajectory as the input to the GCN, we use an encoder module to produce the input embeddings at multiple temporal scales.

We would like to make a note here that time horizons longer than 500 ms are generally considered “long-term", and have been referred to as such in existing literature. Therefore, in

this chapter, we will use this term in the same manner. In Chapter 5 we will predict futures even longer-term as we expand our time horizon to 5 seconds.

Our key idea lies in the fact that recently seen frames hold more relevant information for the prediction of the near future frames than older ones that are far away from the current frame. Therefore by having many smaller kernels that look specifically at recent frames we are able to place more emphasis on the recent frames. This is especially useful for short-term prediction. Nevertheless, for long-term future frame prediction, the older frames also become important as they are able to describe the high-level motion patterns. For instance, for a walking motion which contains the pattern of moving left and right legs in turn, the most recently seen frames only contain the motion of one leg, rather than the cyclic motion of both legs. These high-level motion patterns are usually lower frequency signals. Incorporating this prior knowledge in the encoding of the trajectory allows us to keep local features of the recently seen frames while also keeping the high-level motion pattern for older frames. This inductive bias gives us a boost in accuracy.

In summary, our contributions are twofold:

- We introduce the Temporal Inception Module (TIM), which allows the network to view the motion trajectory at different temporal scales which leads to better performance.
- We present our action-agnostic end to end trainable pipeline combining TIM and GCN which can be trained once to handle all actions evaluated.

We demonstrate our results on the Human 3.6M [71] and CMU Motion Capture[1] datasets, where we outperform the existing methods. Our code is publicly available at https://github.com/tileb1/motion-prediction-tim.

Background: Inception Module

The Inception Module was first introduced by Szegedy *et al.* [160] and used for the task of object detection and classification. They showed that the Inception architecture was able to achieve state-of-the-art results on these tasks, due to its design benefits which keep the computational costs relatively low while increasing the depth and width of the network. The main novelty of this architecture is to combine several convolutional filters of different sizes within the same layer, allowing the network to learn features from the most useful scales.

Since the Inception Module was proposed, different designs have emerged [161] and it has been adapted to a large variety of tasks including human pose estimation [98], action recognition [34, 70, 179], road segmentation [45], single image super-resolution [149], and object recognition [11]. To the best of our knowledge, we are the first to attempt to modify inception modules for generating input embeddings for motion prediction.

[1] CMU Motion Capture is available at http://mocap.cs.cmu.edu/

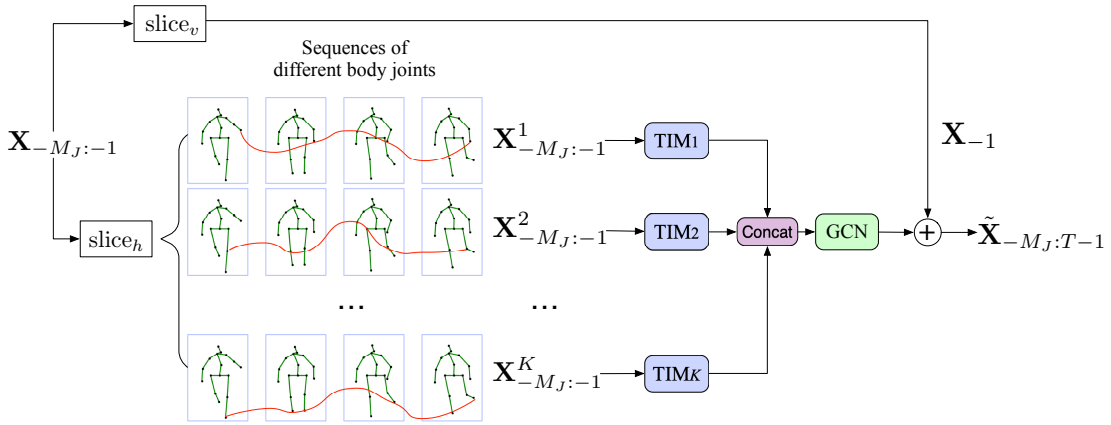


Figure 4.1 – **Overview of the whole framework** making use of multiple TIMs. Using the slice$_h$ operator, we split the input across different joint coordinates. The joint trajectories are fed into the TIMs to produce the embedding, which is then used by the GCN to obtain the residual motion predictions. Using the slice$_v$ operator, separate the most recently seen frame $\mathbf{X}_{-1}$, which is broadcasted to all timestamps and summed with the residual GCN results for the final prediction.

4.1 Methodology

The main encoding methods that have been widely used to represent human motion are 3D joint positions and Euler angle representation. Euler angle representation suffers from ambiguities: two different sets of angles can represent the same pose, which can lead to needlessly over-penalizing predictions. Recent approaches have tried to solve this by changing the encoding to quaternions instead of Euler angles [127]. For the sake of simplicity, our work is solely based on 3D-joint positions. As such, our data consists of time-sequences of skeletons where each skeleton is encoded as a stack of the 3D encoding of its individual body joints.

Let us now define our task. We are given input sequence of K joint trajectories across time, $\mathbf{X}_{-M:-1} = [\mathbf{X}^0_{-M:-1}, \cdots, \mathbf{X}^k_{-M:-1}, \cdots, \mathbf{X}^{K-1}_{-M:-1}]$, where $k \in \{0, 1, \cdots, K-1\}$ represents a Cartesian coordinate value of a joint. Moreover, each joint trajectory $\mathbf{X}^k_{-M:-1} = [\mathbf{X}^k_{-M}, \mathbf{X}^k_{-M+1}, \cdots, \mathbf{X}^k_{-1}]$ is a series of M past joint positions which have already been observed, where $\mathbf{X}^k$ represents a joint coordinate at time index i. We aim to predict the poses in the next T frames, $\mathbf{X}_{0:T-1}$. Negative time indices therefore belong to the observed sequence and positive time indices belong to the prediction. For simplicity, we refer to the trajectory of a joint coordinate as "joint trajectory" throughout this paper.

The overall framework converts the input human motion $\mathbf{X}_{-M:-1}$ into embeddings using our temporal inception module (TIM). These embeddings are then fed to the graph convolutional network (GCN) in order to produce the residual motion. The framework is depicted in detail in Fig. 4.1. The details of the TIM and GCN are introduced below.

4.1.1 Temporal Inception Module

Our main contribution, the Temporal Inception Module (TIM) is illustrated in Fig. 4.2. This module is used to obtain embeddings $\mathbf{E}^k$ of the input motion $\mathbf{X}_{-M:-1}$ for each $k \in \{0, 1, \cdots, K-1\}$ joint coordinate.

TIM takes as input a single joint trajectory $\mathbf{X}^k_{-M_J:-1}$ with the length M_J. Then the subsequence sampling block nested in TIM samples the long motion sequence into multiple sequences with different lengths M_j ($M_j > M_i$ if $j > i$).

For example, in our implementation, we consider two different input sizes $M_1 = 5$ and $M_2 = 10$ where the past motion the inception module sees are $\mathbf{X}_{-M_1:-1}$ and $\mathbf{X}_{-M_2:-1}$ respectively. Each input goes through several 1D-convolutions with different sized kernels. The inception module is used to adaptively determine the weights corresponding to these convolution operations.

Each subsequence $\mathbf{X}^k_{-M_j:-1}$ has its unique convolutional kernels whose sizes are proportional to the length M_j. In other words, we have smaller kernel size for shorter subsequences and larger kernel size for longer subsequences. The intuition is as follows. Using a smaller kernel size allows us to effectively preserve the detailed local information. Meanwhile, for a longer subsequence, a larger kernel is capable of extracting higher-level patterns which depend on multiple time indices. This allows us to process the motion at different temporal scales.

All convolution outputs are then concatenated into one embedding $\mathbf{E}^k$ which has the desired features matching our inductive bias i.e. local details for recently seen frames and a low-frequency information for older frames.

More formally, we have

$$\mathbf{E}^k_j = \text{concat}(C_{S^j_1}(\mathbf{X}^k_{-M_j:-1}), C_{S^j_2}(\mathbf{X}^k_{-M_j:-1}), \cdots, C_{S^j_L}(\mathbf{X}^k_{-M_j:-1})) \tag{4.1}$$

followed by

$$\mathbf{E}^k = \text{concat}(\mathbf{E}^k_1, \mathbf{E}^k_2, \cdots, \mathbf{E}^k_J) \tag{4.2}$$

where $C_{S^j_l}$ is a 1D-convolution with filter size S^j_l. The embedings for each joint trajectory $\mathbf{E}^k$ are then used as input feature vector for the GCN. An overview of the global framework is illustrated in Fig. 4.1.

4.1.2 Graph Convolutional Network

For the high capacity feed-forward network, we make use a graph convolutional neural network as proposed by Mao et al. [106]. This network is currently a state-of-the-art network for human motion prediction from separate time embeddings of each body joint. This makes

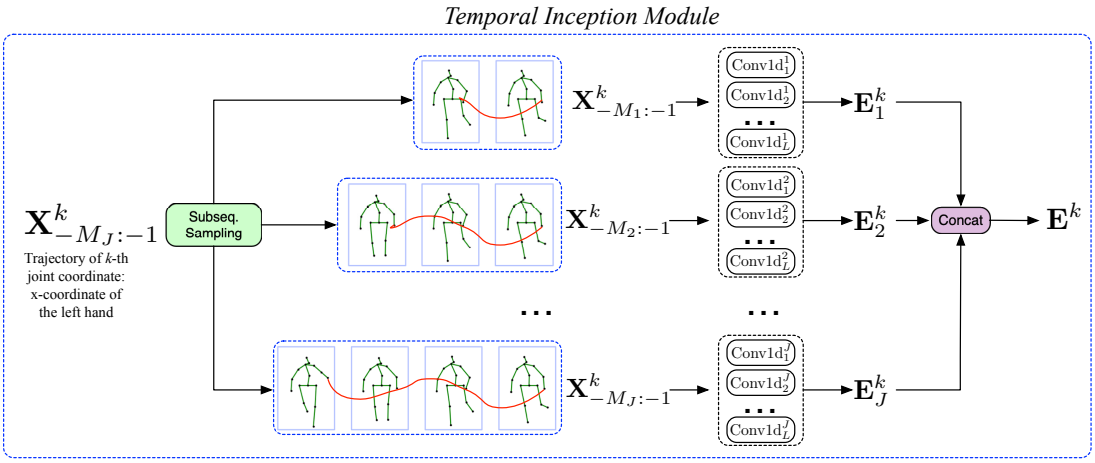


Figure 4.2 – **Overview of the Temporal Inception Module (TIM).** TIM processes each joint coordinate k separately, expressed as a superscript in this figure. The subseq sampling block splits a 1D input sequence into J subsequences, each of length M_j. The Conv1d^j_l block corresponds to a 1D convolution operator with kernel size S^j_l. The results of the convolutions are concatenated to form the embeddings of each subsequence $\mathbf{E}^k_j$, which are concatenated again to form the input embeddings $\mathbf{E}^k$ to the GCN.

it very well suited for our task. As shown in their previous work, using the kinematic tree of the skeleton as predefined weight adgency matrix is not optimal. Instead, a separate adgency matrix is learned for each layer.

Following the notation of [106], we model the skeleton as a fully connected set of K nodes, represented by the trainable weighted adjacency matrix $\mathbf{A}^{K\times K}$. The GCN consists of several stacked graph convolutional layers, each performing the operation

$$\mathbf{H}^{(p+1)} = \sigma(\mathbf{A}^{(p)}\mathbf{H}^{(p)}\mathbf{W}^{(p)}) \tag{4.3}$$

where $\mathbf{W}^{(p)}$ is the set of trainable weights of layer p, $\mathbf{A}^{(p)}$ is the learnable adgency matrix of layer p, $\mathbf{H}^{(p)}$ is the input to layer p, $\mathbf{H}^{(p+1)}$ is the output of layer p (and input to layer $p+1$) and $\sigma(\cdot)$ is an activation function.

The GCN receives as input the embeddings $\mathbf{E}$ produced by the multiple TIMs and regresses the residual motion which is later summed up with the most recently seen human pose $\mathbf{X}_{-1}$ to produce the entire motion sequence,

$$\tilde{\mathbf{X}}_{-M_J:T-1} = G(\mathbf{E}) + \mathbf{X}_{-1} \tag{4.4}$$

where the GCN is denoted as G. Since $\tilde{\mathbf{X}}_{0:T-1}$ is a subset of $\tilde{\mathbf{X}}_{-M_J:T-1}$, we thus predict the future

motion. This is depicted in Fig. 4.1.

4.1.3 Implementation and Training Details

The Temporal Inception Module used for comparison with other baselines uses 2 input subsequences with lengths $M_1 = 5$ and $M_2 = 10$. Both are convolved with different kernels whose sizes are proportional to the subsequence input length. A detailed view of these kernels can be found in Table 4.1. The kernel sizes are indeed chosen to be proportional to the input length. The number of kernels are decreased as the kernel size increases to avoid putting too much weight on older frames. We have also added a special kernel of size 1 which acts as a pass-through. This leaves us with an embedding $\mathbf{E}^k$ of size 223 ($12{\cdot}4+9{\cdot}3+9{\cdot}8+7{\cdot}6+6{\cdot}4+1{\cdot}10$) for each joint coordinate $k \in \{0, 1, \cdots, K-1\}$ which are fed to the GCN. For more details on the GCN architecture, we refer the reader to [106].

Table 4.1 – Detailed architecture of Temporal Inception Module used to compare with baselines.

Subsequence input length (M_j)	Number of kernels	Kernel size
5	12	2
5	9	3
10	9	3
10	7	5
10	6	7
10	1	1

The whole network (TIM + GCN) is trained end to end by minimizing the Mean Per Joint Position Error (MPJPE) as proposed in [71]. This loss is defined as

$$\frac{1}{K(M_J+T)} \sum_{t=-M_J}^{T-1} \sum_{i=1}^{I} ||\mathbf{p}_{i,t} - \hat{\mathbf{p}}_{i,t}||^2 \tag{4.5}$$

where $\hat{\mathbf{p}}_{i,t} \in \mathbb{R}^3$ is the prediction of the i-th joint at time index t, $\mathbf{p}_{i,t}$ is the corresponding ground-truth at the same indices and I is the number of joints in the skeleton ($3 \times I = K$ as the skeletons are 3D). Note that the loss sums over negative time indices which belong to the observed sequence as it adds an additional training signal.

It is trained for 50 epochs with a learning-rate decay of 0.96 every 2 epochs as in [106]. One pass takes about 75ms on an NVIDIA Titan X (Pascal) with a batch-size of 16.

4.2 Evaluation

We evaluate our results on two benchmark human motion prediction datasets: Human3.6M [71] and CMU motion capture dataset [36]. The details of the training/testing split of the datasets are shown below, followed by the experimental result analysis and ablation study.

4.2.1 Datasets

Human3.6M. Following previous works on motion prediction [108, 73], we use 15 actions performed by 7 subjects for training and testing. These actions are *walking, eating, smoking, discussion, directions, greeting, phoning, posing, purchases, sitting, sitting down, taking photo, waiting, walking dog and walking together*. We also report the average performance across all actions. The 3D human pose is represented using 32 joints. Similar to previous work, we remove global rotation and translation and testing is performed on the same subset of 8 sequences belonging to Subject 5.

CMU Motion Capture. The CMU Motion Capture dataset contains challenging motions performed by 144 subjects. Following previous related work's training/testing splits and evaluation subset [89], we report our results across eight actions: *basketball, basketball signal, directing traffic, jumping, running, soccer, walking, and washwindow*, as well as the average performance. We implement the same preprocessing as the Human3.6M dataset, *i.e.*, removing global rotation and translation.

4.2.2 Baselines

We select the following baselines for comparison: Martinez *et al.* (Residual sup.) in order to compare against the well known method using RNNs [108], Li *et al.* (convSeq2Seq) as they also encode their inputs using convolution operations [89] and Mao *et al.* (DCT+GCN) [106] to demonstrate the gains of using TIM over DCT for encoding inputs.

4.2.3 Results

In our results (*e.g.*, Tables 4.5, 4.3, 4.2 and 4.1), for the sake of robustness we report the average error over 5 runs for our own method. We denote our method by "Ours $(5-10)$" since our final model takes as input subsequences of lengths $M_1 = 5$ and $M_2 = 10$.

We report our short-term prediction results on Human3.6M in Table 4.2. For the majority of the actions and on average we achieve a lower error than the existing methods. Our qualitative results are shown in Figure 4.3.

Our long-term predictions on Human3.6M are reported in Table 4.3. Here we achieve an even larger boost in accuracy, especially for case of 1000ms. We attribute this to the large kernel sizes we have set for input length 10, which allows the network to pick up the underlying

Table 4.2 – **Short-term prediction test error of 3D joint positions on H3.6M.** We outperform the baselines on average and for most actions.

	Walking [ms]				Eating [ms]				Smoking [ms]				Discussion [ms]			
Name	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
Residual sup. [108]	23.8	40.4	62.9	70.9	17.6	34.7	71.9	87.7	19.7	36.6	61.8	73.9	31.7	61.3	96.0	103.5
convSeq2Seq [89]	17.1	31.2	53.8	61.5	13.7	25.9	52.5	63.3	11.1	21.0	33.4	38.3	18.9	39.3	67.7	75.7
DCT + GCN [106]	**8.9**	**15.7**	**29.2**	**33.4**	8.8	18.9	39.4	47.2	7.8	14.9	25.3	**28.7**	9.8	22.1	**39.6**	**44.1**
Ours (5 – 10)	9.3	15.9	30.1	34.1	**8.4**	**18.5**	**38.1**	**46.6**	**6.9**	**13.8**	**24.6**	29.1	**8.8**	**21.3**	40.2	45.5

Directions [ms]				Greeting [ms]				Phoning [ms]				Posing [ms]			
80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
36.5	56.4	81.5	97.3	37.9	74.1	139.0	158.8	25.6	44.4	74.0	84.2	27.9	54.7	131.3	160.8
22.0	37.2	59.6	73.4	24.5	46.2	90.0	103.1	17.2	29.7	53.4	61.3	16.1	35.6	86.2	105.6
12.6	24.4	**48.2**	**58.4**	14.5	30.5	74.2	89.0	**11.5**	20.2	**37.9**	**43.2**	9.4	23.9	66.2	82.9
11.0	**22.3**	48.4	59.3	**13.7**	**29.1**	**72.6**	**88.9**	**11.5**	**19.8**	38.5	44.4	**7.5**	**22.3**	**64.8**	**80.8**

Purchases [ms]				Sitting [ms]				Sitting Down [ms]				Taking Photo [ms]			
80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
40.8	71.8	104.2	109.8	34.5	69.9	126.3	141.6	28.6	55.3	101.6	118.9	23.6	47.4	94.0	112.7
29.4	54.9	82.2	93.0	19.8	42.4	77.0	88.4	17.1	34.9	66.3	77.7	14.0	27.2	53.8	66.2
19.6	**38.5**	**64.4**	**72.2**	10.7	24.6	50.6	62.0	11.4	**27.6**	56.4	67.6	6.8	**15.2**	**38.2**	**49.6**
19.0	39.2	65.9	74.6	**9.3**	**22.3**	**45.3**	**56.0**	**11.3**	28.0	**54.8**	**64.8**	**6.4**	15.6	41.4	53.5

Waiting [ms]				Walking Dog [ms]				Walking Together [ms]				Average [ms]			
80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
29.5	60.5	119.9	140.6	60.5	101.9	160.8	188.3	23.5	45.0	71.3	82.8	30.8	57.0	99.8	115.5
17.9	36.5	74.9	90.7	40.6	74.7	116.6	138.7	15.0	29.9	54.3	65.8	19.6	37.8	68.1	80.2
9.5	22.0	**57.5**	73.9	32.2	58.0	102.2	122.7	**8.9**	**18.4**	**35.3**	**44.3**	12.1	25.0	51.0	61.3
9.2	**21.7**	55.9	**72.1**	**29.3**	**56.4**	**99.6**	**119.4**	**8.9**	18.6	35.5	**44.3**	**11.4**	**24.3**	**50.4**	**60.9**

higher-level patterns in the motion. We validate this further in our ablation study. We present our qualitative results in Figures 4.3 and 4.4.

Table 4.3 – **Long-term prediction test error of 3D joint positions on H3.6M.** We outperform the baselines on average and on almost every action. We have also found that we can have an even higher accuracy for 1000ms in our ablation study, where we show the effect of adding another input subsequence of length $M_j = 15$.

	Walking [ms]		Eating [ms]		Smoking [ms]		Discussion [ms]		Average [ms]	
Name	560	1000	560	1000	560	1000	560	1000	560	1000
Residual sup. [108]	73.8	86.7	101.3	119.7	85.0	118.5	120.7	147.6	95.2	118.1
convSeq2Seq [89]	59.2	71.3	66.5	85.4	42.0	67.9	84.1	116.9	62.9	85.4
DCT + GCN [106]	42.3	51.3	**56.5**	**68.6**	**32.3**	**60.5**	70.5	103.5	50.4	71.0
Ours (5 – 10)	**39.6**	**46.9**	56.9	**68.6**	33.5	61.7	**68.5**	**97.0**	**49.6**	**68.6**

Our predictions on the CMU motion capture dataset are reported in Table 4.4. Similar to our results on Human3.6M, we observe that we outperform the state-of-the-art. For all timestamps except for 1000 ms, we show better performance than the baselines. We observe that both our and Mao *et al.*'s [106] high capacity GCN based models are outperformed by convSeq2Seq [89], a CNN based approach. Since the training dataset of CMU-Mocap is much smaller compared to H36M, this leads to overfitting for high-capacity networks such as ours. However, this is not problematic for short-term predictions, as in that case it is not as crucial for the model to be generalizable. We do however outperform Mao *et al.*'s results for the 1000ms prediction which makes use of the same backbone GCN as us. We observe that on average and for many actions, we outperform the baselines for the 80, 160, 320 and 400 ms.

4.2.4 Ablation Study

The objective of this section is twofold.

- First, we inquire the effect of choosing a kernel size proportional to the input size M_j;
- Second, we inquire the effect of the varying length input subsequences .

Both results are shown in Fig. 4.5, where the version name represents the set $\{M_j : j \in \{1,2,\cdots,J\}\}$ of varying length subsequences.

Proportional filter size. In our design of TIM , we chose filter sizes proportional to the subsequence input length M_j. In Table 4.5, we observe the effects of setting a "constant kernel size" of 2 and 3 for all input subsequences. Note that we also adjust the number of filters such that the size of the embedding is the more or less the same for both cases, for fair comparison. We can observe that for both versions $5-10$ and $5-10-15$, having a proportional kernel size to the subsequence input length increases the accuracy for the majority of the

Table 4.4 – **Prediction test error of 3D joint positions on CMU-Mocap.** For all timestamps except for 1000ms, we demonstrate better performance than the baselines. Our model performs better in this case for short term predictions. We observe that on average and for many actions, we surpass the baselines for the 80, 160, 320 and 400 ms.

	Basketball [ms]					Basketball Signal [ms]					Directing Traffic [ms]				
Name	80	160	320	400	1000	80	160	320	400	1000	80	160	320	400	1000
Residual sup [108].	18.4	33.8	59.5	70.5	106.7	12.7	23.8	40.3	46.7	77.5	15.2	29.6	55.1	66.1	127.1
convSeq2Seq [89]	16.7	30.5	53.8	64.3	**91.5**	8.4	16.2	30.8	37.8	76.5	10.6	20.3	38.7	48.4	**115.5**
DCT+GCN [106]	14.0	25.4	49.6	61.4	106.1	3.5	6.1	11.7	**15.2**	**53.9**	7.4	15.1	31.7	42.2	152.4
Ours (5 – 10)	**12.7**	**22.6**	**44.6**	**55.6**	102.0	**3.0**	**5.6**	**11.6**	15.5	57.0	**7.1**	**14.1**	**31.1**	**41.4**	138.3

Jumping [ms]					Running [ms]					Soccer [ms]				
80	160	320	400	1000	80	160	320	400	1000	80	160	320	400	1000
36.0	68.7	125.0	145.5	195.5	15.6	19.4	31.2	36.2	43.3	20.3	39.5	71.3	84	129.6
22.4	44.0	87.5	106.3	**162.6**	**14.3**	**16.3**	**18.0**	**20.2**	**27.5**	12.1	21.8	**41.9**	**52.9**	**94.6**
16.9	34.4	76.3	96.8	164.6	25.5	36.7	39.3	39.9	58.2	11.3	**21.5**	44.2	55.8	117.5
14.8	**31.1**	**71.2**	**91.3**	163.5	24.5	37.0	39.9	41.9	62.6	**11.2**	22.1	45.1	58.1	122.1

Walking [ms]					Washwindow [ms]					Average [ms]				
80	160	320	400	1000	80	160	320	400	1000	80	160	320	400	1000
8.2	13.7	21.9	24.5	**32.2**	8.4	15.8	**29.3**	**35.4**	61.1	16.8	30.5	54.2	63.6	96.6
7.6	12.5	23.0	27.5	49.8	8.2	15.9	32.1	39.9	**58.9**	12.5	22.2	40.7	49.7	**84.6**
7.7	11.8	**19.4**	23.1	40.2	**5.9**	**11.9**	30.3	40.0	79.3	11.5	20.4	37.8	46.8	96.5
7.1	**11.1**	19.9	**22.8**	39.3	**5.9**	12.3	32.1	42.6	80.4	**10.8**	**19.5**	**36.9**	**46.2**	95.7

actions and this brings better performance on average. Therefore, our empirical results match our intuition that using larger filters for longer length inputs that look back further into the past helps by capturing higher-level motion patterns which yield embeddings of better quality.

Varying Length Input Subsequences. The goal of having the Temporal Inception Module is to sample subsequences of different length M_j which, once processed, yield embeddings with different properties. Embeddings of longer input sequences contain higher level information of the motion (lower frequencies), whereas embeddings of shorter input sequences would contain higher spatial resolution and higher frequency information of the short-term future motion. We expect our model to perform better on very long term prediction of 1000ms prediction the bigger M_J is. As can be seen from Table 4.5, we also observe that there is unfortunately a trade-off to be made between aiming for very long term predictions (1000ms) or shorter term predictions (560ms). The 5−10−15 model yields higher accuracy than the 5−10 model on 1000ms and performs worse on 560ms predictions. This matches our intuition since the 5−10−15 model is trained to place more emphasis on the high-level motion pattern and is therefore tuned for very long term predictions at 1000ms.

Note that we obtain even better performance for very long-term prediction with the 5−10−15 model compared with the 5−10 model which has already outperformed the baselines in Table 4.3.

Table 4.5 – **Effect of the kernel size and subsequence lengths M_j** on the framework performance for long-term prediction on H3.6M. We observe that proportional kernel sizes on average yield better performance. We also observe that including the input subsequence with length $M_j = 15$ allows us to look back further into the past, boosting the predictions of the furthest timestamp evaluated, 1000ms.

	Walking [ms]		Eating [ms]		Smoking [ms]		Discussion [ms]		Average [ms]	
Version	560	1000	560	1000	560	1000	560	1000	560	1000
5-10 (proportional kernel size)	39.6	46.9	56.9	68.6	**33.5**	**61.7**	**68.5**	97.0	**49.6**	68.6
5-10 (constant kernel size)	**38.4**	45.6	56.9	68.5	34.9	63.8	73.2	100.1	50.8	69.5
5-10-15 (proportional kernel size)	43.3	43.1	**45.8**	**65.2**	36.4	62.9	97.1	**94.6**	55.7	**66.5**
5-10-15 (constant kernel size)	42.8	**41.6**	47.1	66.0	36.6	63.2	98.3	96.6	56.2	66.9

4.3 Conclusion

The task of human motion prediction has gained more attention with the rising popularity of autonomous driving and human-robot interaction. Currently, deep learning methods have made much progress, however, none has focused on utilizing different length input sequences seen at different temporal scales to learn more powerful input embeddings which can benefit the prediction. Our Temporal Inception Module allows us to encode various length input subsequences at different temporal scales and achieves state-of-the-art performance.

Our method gives state-of-the-art performance for a time-horizon of up to 1 second. However, when it is trained for a longer time horizon of 5 seconds, the performance degrades. This is the issue we will address in the next chapter of this thesis. By designing a framework to take into consideration only the most essential poses of the sequence, we will place even more emphasis on extracting information at a more broad and semantic level.

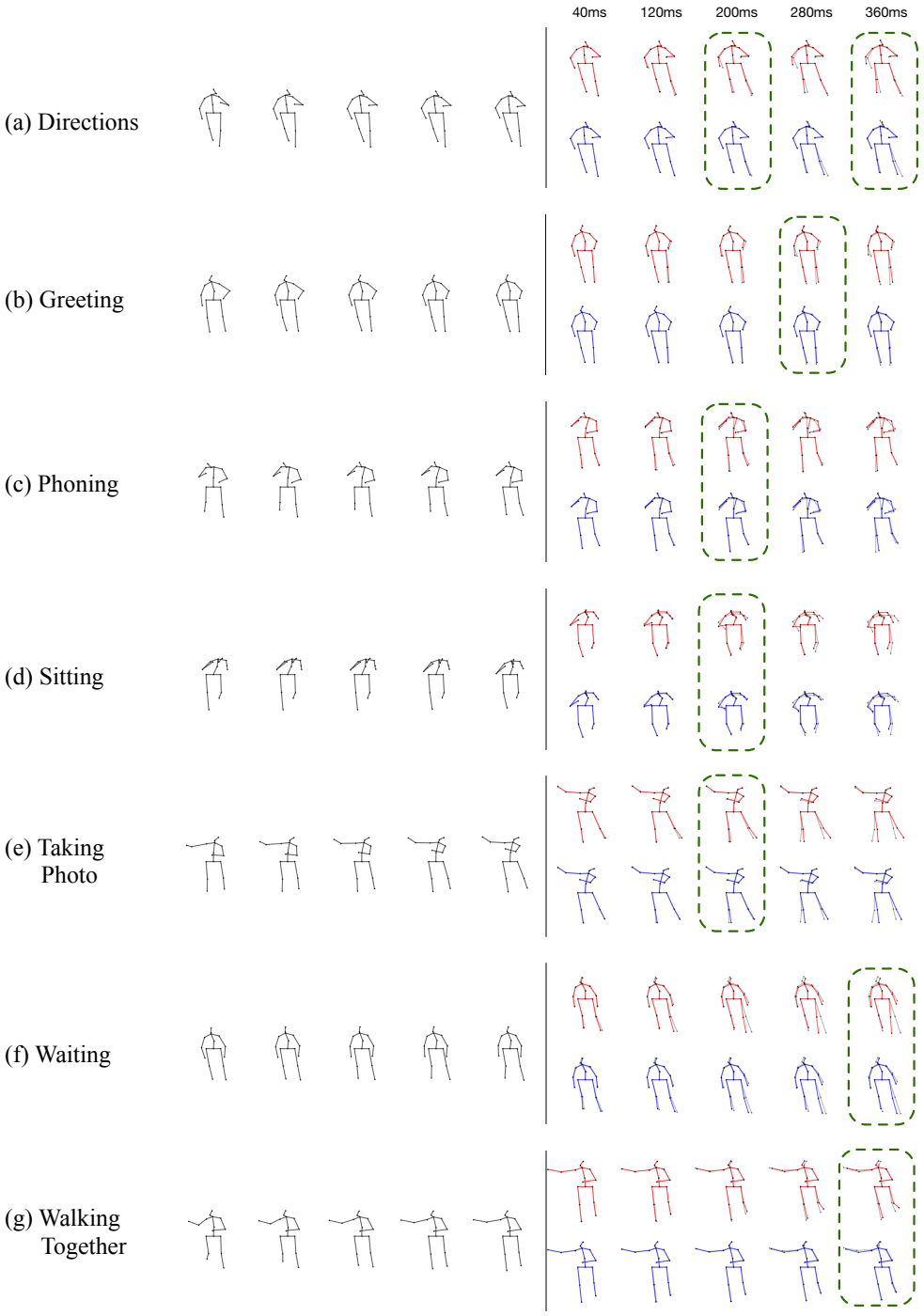


Figure 4.3 – **Qualitative comparison** between (DCT+GCN)[106] (red) and ours (blue) on H3.6M predicting up to 400ms. The ground truth is superimposed faintly in black on top of both methods. Poses on the left are the conditioning ground truth and the rest are predictions. We observe that our predictions closely match the ground truth poses. We have highlighted some of our best predictions with green bounding boxes.

Figure 4.4 – **Long-term qualitative comparison** between ground truth (top row)(DCT+GCN)[106] (middle) and ours (bottom row) on H3.6M predicting up to 1000ms. The ground truth is superimposed faintly on top of both methods. Poses on the left are the conditioning ground truth and the rest are predictions. We observe that our predictions closely match the ground truth poses, though as expected, the error increases as the time index increases. We have highlighted some of our best predictions with green bounding boxes.

5 Long Term Motion Prediction Using Keyposes

This chapter focuses on longer-term prediction, which is critical in many areas, such as providing an autonomous system sufficient time to react to human motions. Most approaches formulate this task as one of regressing a person's pose at every future time instant given the past poses. While recurrent neural networks [54, 108] and graph convolutional networks [106, 105] are effective for short-term predictions, typically up to one second in the future, their prediction accuracy degrades quickly beyond that, and addressing this shortcoming remains an open problem.

Our key insight is that, for this task, predicting the pose in *every* future frame is unnecessary. For example, consider a boxing jab motion. The most significant poses are the ones where the hand is closest to the chest and where the arm is the most extended. The in-between poses are transition ones that can be interpolated from these two. Therefore instead of treating a motion as a sequence of consecutive poses, we downsample it to a set of *keyposes* from which all other poses can be interpolated up to a given precision. We then use these keyposes for long-term motion prediction.

The simplest way to so would be to replace the poses in existing frameworks by our keyposes. However, while all keyposes are unique, some tend to be similar to each other. We therefore cluster those we extract from a training set and develop a framework that treats keypose prediction as a classification problem. This has two main advantages. First, it overcomes the tendency of regression-based prediction methods to converge to the mean pose in the long term. Second, it allows us not only to predict the most likely future motion by selecting the most probable clusters but also to generate multiple plausible predictions by sampling the relevant probability distributions. This is useful because people are not entirely predictable, as in the case of a pedestrian standing on the curb who may, or may not, cross the street.

In summary, our contributions are threefold. (i) We introduce a keypose extraction algorithm to represent human motion in a compact way. (ii) We formulate motion prediction as a classification problem and design a framework to predict keypose labels and durations. (iii) We demonstrate that our approach enables us to predict multiple realistic motions for up

to 5 seconds in the future, which is far longer than the typical 1 second encountered in the literature. The motions we generate preserve the dynamic nature of the observations, whereas the methods designed for shorter timespans tend to degenerate to static poses. Our code and an overview video can be accessed via our project website, https://senakicir.github.io/projects/keyposes.

Background: Keyposes

The concept of detecting “keyposes" in sequences and building algorithms around them has previously been discussed in numermous works for different tasks. One such task is action recognition. For example, in [101], 2D keyposes are used for single view action recognition. In [97], Adaboost is used to select keyposes that are discriminative for each action. In [22], linear latent low-dimensional features extracted from sequences for action recognition and action prediction. Furthermore, [86] focus on generating realistic transitions between nodes in a motion graph, which resembles our notion of keyposes, to synthesize short animated sequences. However, none of these works predict future keyposes given past ones.

5.1 Methodology

Classically, the task of motion prediction is defined as producing the sequence of 3D poses from $t = 1$ to $t = N$, denoted as $\mathbf{X}_{1:N}$, given the sequence of poses from $t = -M$ to $t = 0$, denoted as $\mathbf{X}_{-M:0}$. Each pose value $\mathbf{X}_t$ is of dimension $3 \times J$, where J is the total number of joints. Therefore, motion prediction is written as

$$\mathbf{X}_{1:N} = F(\mathbf{X}_{-M:0}) \,,$$

where F is the prediction function.

Our approach departs from this classical formalism by predicting keyposes from keyposes. As will be discussed in more detail in Section 5.1.1, keyposes encode the important poses in a sequence $\mathbf{X}_{1:T}$, such that the remaining poses can be obtained by linear interpolation between subsequent keyposes. Therefore, our keypose-to-keypose framework takes as input a motion $\mathbf{X}_{-M:0}$ defined by its keyposes $\mathbf{K}_{-I_1:0}$, where $I_1 \ll M$ is the number of keyposes in the past sequence. We then predict $\mathbf{K}_{1:I_2}$, where $I_2 \ll N$ is the number of keyposes in the future sequence. We write this as

$$\mathbf{K}_{1:I_2} = G(\mathbf{K}_{-I_1:0}) \,,$$

where G is the keypose-to-keypose prediction function.

Our overall pipeline, illustrated in Figure 5.1, consists of extracting keyposes from input sequences, feeding them to the keypose prediction network, reconstructing the predicted sequence via linear interpolation, and refining the final result via a refinement network. We describe each of these steps in detail below.

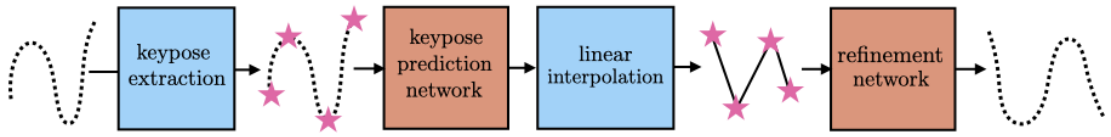


Figure 5.1 – **Our overall pipeline** for predicting future motions via keyposes. It consists of the following steps: keypose extraction, keypose prediction, linear interpolation to reconstruct the sequence, and refining the final sequence.

5.1.1 Keyposes

Let us now discuss how we obtain keyposes $\mathbf{K}_i$, $i \in [1, I]$, given a sequence of poses $\mathbf{X}_t$, $t \in [1, T]$. We define the keyposes as the poses in $\mathbf{X}_{1:T}$ between which linear interpolation can be used to obtain the remaining poses. We therefore employ an optimization-based strategy to identify the poses from which the L2 error between the original sequence $\mathbf{X}$ and the sequence reconstructed by linear interpolation is minimized. Our method proceeds as follows:

- We set $\mathbf{X}_1$ and $\mathbf{X}_T$ to be the initial keyposes.
- We reconstruct the sequence by linearly interpolating the set of keyposes. We denote the reconstruction as $\hat{\mathbf{X}}_t$, $t \in [1, T]$.
- We select the pose $\mathbf{X}_t$ at position t which has the highest L2 error with respect to $\hat{\mathbf{X}}_t$, the pose reconstructed by linear interpolation at the same time index. We add $\mathbf{X}_t$ to our set of keyposes.
- The algorithm continues recursively, selecting keyposes from the sequences between $[1, t]$ and $[t, T]$. The recursion ends once the average reconstruction error of the linear interpolation is below a threshold, yielding a set of keyposes.

An example distribution of keyposes in a sequence is shown in Figure 5.2.

5.1.2 Motion Prediction with Keyposes

In principle, we could directly use the above-mentioned keyposes for prediction, by simply learning to regress keypose values. However, for long-term prediction, this would exhibit the same tendency as existing frameworks to converge to a static pose. To overcome this, we propose to cluster the training keyposes and treat keypose prediction as a classification task, where the clusters act as categories.

To this end, we extract the keyposes for every training motion individually, and cluster all the resulting training keyposes into K clusters via k-means. Each keypose is then given a label determined by the cluster it is assigned to. Finally, we prune the keyposes by removing the unnecessary intermediate ones that have the same label as their preceding and succeeding keypose. We show a sample of 500 keypose cluster centers in Figure 5.3. It is necessary for them to be varied in order to be able to express the wide range of poses seen across different

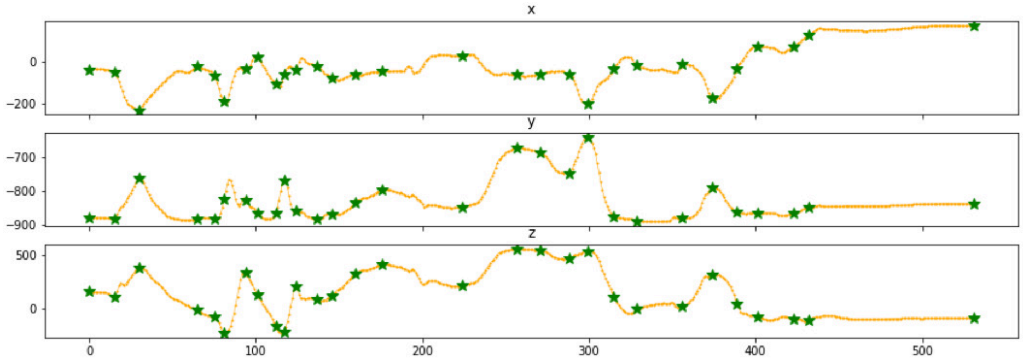


Figure 5.2 – **Distribution of keyposes** in a sequence from the Human3.6M dataset. The plots depict the x, y, and z coordinates of the right foot of Subject 1 during the purchases action. We show the locations of the keyposes as green stars, and the transition poses as orange dots. The downsampling effect is prominent. The keyposes are distributed more densely where the motion is the most varied, and these keyposes have shorter duration. Note that these plots only correspond to one joint, whereas the optimization takes into account the average error of all the joints.

action categories. We find that the keypose cluster centers include poses from many categories, such as sitting, crouching, squatting, standing, walking, and making different arm gestures.

Clustering the keyposes allows us to cast keypose prediction as a classification problem. Specifically, instead of predicting the future keypose values, we predict their labels. Given the labels, l_i and l_{i+1}, of two subsequent keyposes, $\mathbf{K}_i$ and $\mathbf{K}_{i+1}$, we can simply estimate the intermediate poses via linear interpolation between the corresponding cluster centers. However, this requires the duration d_{i+1} between the two keyposes, indicating the number of intermediate poses, which we therefore also predict.

Network Design and Training

We have designed an RNN based neural network as our keypose-to-keypose prediction framework, as shown in Figure 5.4. At each time step, in addition to the hidden representation of the previous time step, our recurrent unit takes as input the previous keypose label l_i and duration d_i. Specifically, we represent the label as a distribution L_i computed as follows.

1. If we know the true keypose value (i.e., for observed past keyposes): We compute the proximity between the keypose value $\mathbf{V}_i$ and every cluster center C_j, $j \in [1, K]$ as the negative average Euclidean distance between the corresponding joints in $\mathbf{V}_i$ and $\mathbf{C}_j$. These values form a K-dimensional proximity vector for each keypose i.

2. If we do not know the keypose value (i.e., for inferred future keyposes): We compute the

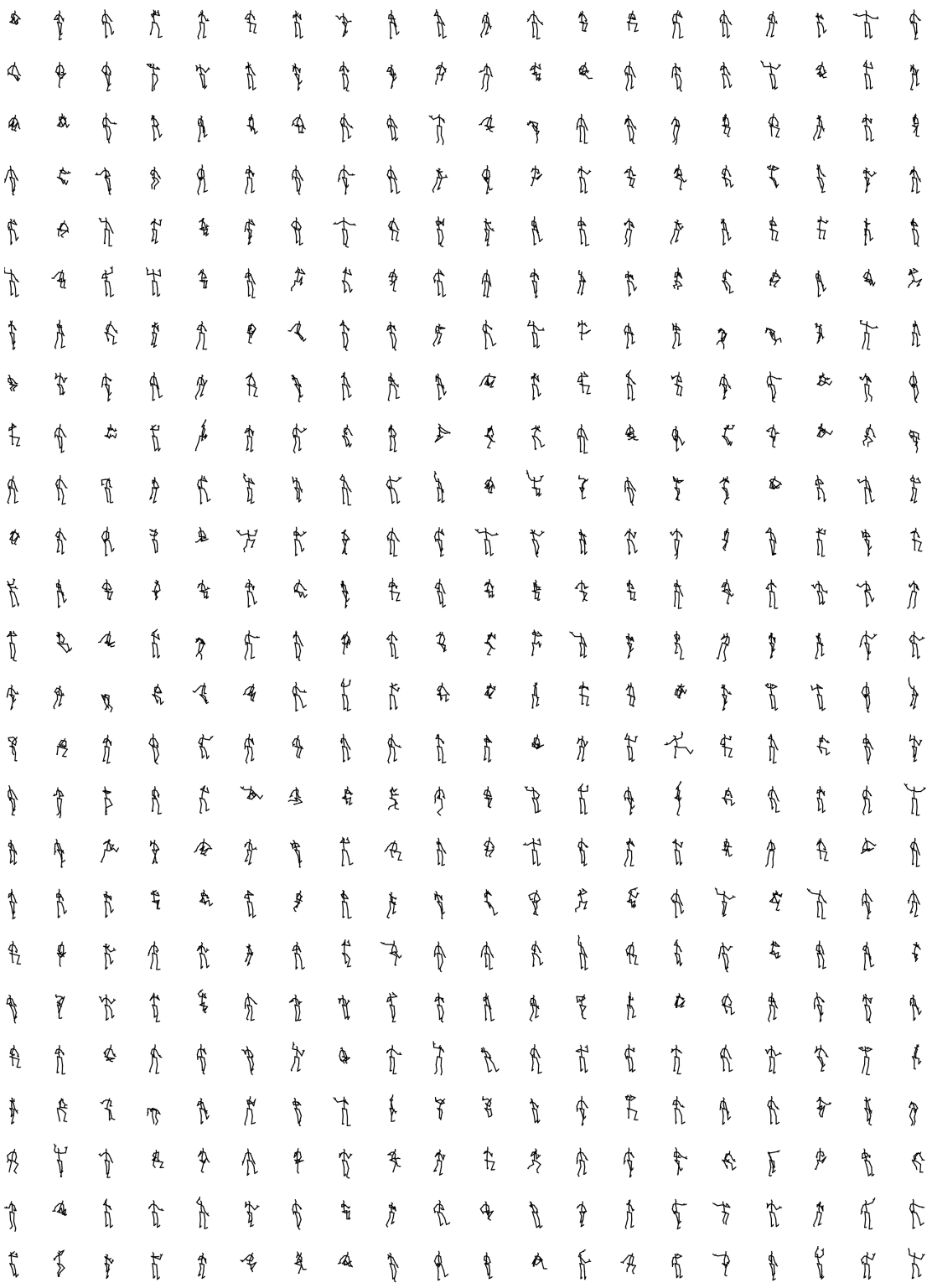


Figure 5.3 – **Visualization of keypose cluster centers**. The sampled 500 keypose cluster centers here show that the cluster centers are quite varied and are able to represent the keyposes throughout the different categories of motions.

proximities between the cluster center corresponding to the predicted label l_i, $\mathbf{C}_{l_i}$, and all cluster centers C_j, $j \in [1, K]$.

3. We pass the resulting proximity vector through a softmax operation with a temperature of 0.03 to obtain a distribution L_i over the labels.

To also treat duration prediction as a classification task, we categorize the durations into very short (less than 4 frames), short (between 5 and 10 frames), medium (between 10 and 14 frames), long (between 14 and 25 frames), and very long (more than 25 frames). We then encode the duration d_i of a keypose as a one-hot encoding D_i over these categories and output a distribution for the future keyposes.

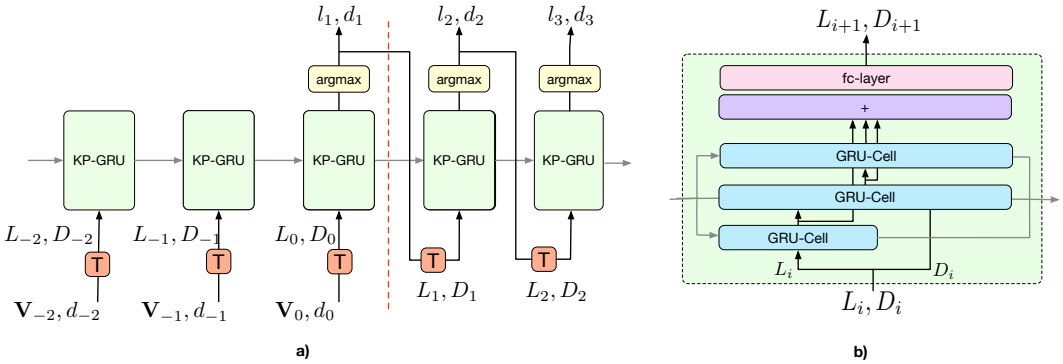


Figure 5.4 – **Keypose-to-keypose network structure**.(a) Overall architecture. At each time step i, a keypose GRU (KP-GRU) unit predicts the keypose labels and durations of the next step $i+1$. The time of the last observation is denoted by $i = 0$. Before this time-step, the network is given ground-truth keyposes as conditioning signal. The label distribution L_i for past keyposes is found using the keypose value $\mathbf{V}_i$. After time-step $i = 0$, the network is given its own predictions as input rather than the ground truth. The label distribution L_i, in this case, is found using the predicted label l_i. The orange T blocks represent the transformation to compute the distributions. (b) Inner structure of the KP-GRU unit, which consists of a three layer GRU network followed by a fully connected layer.

Therefore, our network predicts a pair of distributions: over the labels and over the duration categories. We train the network using two loss functions:

- E_{labels}: The cross-entropy loss between the ground-truth cluster label and the predicted label distribution;
- E_{dur}: The cross-entropy loss between the predicted duration distribution and ground-truth duration category.

The overall loss of our network therefore is

$$E = w_{\text{labels}} E_{\text{labels}} + w_{\text{dur}} E_{\text{dur}} \,, \tag{5.1}$$

where w_{labels} and w_{dur} weigh the different loss terms.

During training, the label of the next keypose l_{i+1} is determined as the one with the highest predicted probability. We then compute a distribution L_{i+1} from this label as described above. This procedure prevents error accumulation as the prediction progresses and guarantees that the network will never see anything very different from what it was trained on. The duration of the next keypose d_{i+1} is determined similarly: According to the category with the highest probability, the duration is set to 3 for very-short, 6 for short, 12 for medium, 16 for long, and 25 for very long. Using the predicted label and duration of each time-step, we can reconstruct the sequence via linear interpolation between the corresponding cluster centers, as described previously.

During training, we observe 7 past keyposes and predict 12 future keyposes. At test time, we predict until we reach 5 seconds. Weights of the loss terms are set to $w_{\text{labels}} = 1.0$, $w_{\text{dur}} = 0.1$. Our network is trained for 100 epochs with a batch size of 64. We use an Adam optimizer with a learning rate of 0.0001 and a 0.01 weight decay. We report the results of the model with the highest validation score.

Inference and Interpolation

Our network produces distributions over the keypose clusters. Hence, at inference time, for each iteration of the recurrent network, we can sample the future label and duration from the predicted distributions. In practice, before sampling, we smooth the predicted distributions via a softmax with a temperature of 0.3. This sampling scheme allows us to produce multiple future sequences given a single observation.

Once we have predicted a set of keypose labels and their durations, we can interpolate the intermediate poses and reconstruct the future sequence. Denoting by t the time index of keypose $\mathbf{K}_i$ in the sequence, the intermediate pose at time $t_1 > t$ is computed as

$$\mathbf{X}_{t_1} = \mathbf{C}_{l_i} + (t_1 - t)\frac{\mathbf{C}_{l_{i+1}} - \mathbf{C}_{l_i}}{d_{i+1}} ,$$

where $\mathbf{C}_{l_i}$ and $\mathbf{C}_{l_{i+1}}$ are the cluster centers corresponding to labels l_i and l_{i+1}.

The sequences obtained by linear interpolation can then be refined using a pretrained refinement network trained to produce sequences that preserve the poses of the original sequence. Formally this operation can be written as

$$\mathbf{X}^{\text{ref}}_{1:N} = R(\mathbf{X}_{1:N}) ,$$

where R denotes the refinement function and $\mathbf{X}^{\text{ref}}_{1:N}$ denotes the refined pose sequence. We describe this network in more detail in Appendix A.

e use a 3 layer GRU, depicted in Fig. 5.4. The GRU cells all have hidden states of size 512. A Gaussian noise of magnitude 0.1 is added to the proximities during training to increase

robustness. Furthermore, to gradually teach the network to process its own samples, we use scheduled sampling for teacher forcing, as proposed in [19]. During training, we observe 7 past keyposes and predict 12 future keyposes. At test time, we predict until we reach 5 seconds. Weights of the loss terms are set to $w_{\text{labels}} = 1.0$, $w_{\text{dur}} = 0.1$. Our network is trained for 100 epochs with a batch size of 64. We use an Adam optimizer with a learning rate of 0.0001 and a 0.01 weight decay. We report the results of the model with the highest validation score.

5.2 Evaluation

5.2.1 Datasets

Human3.6M [71] is a standard 3D human pose dataset and has been widely used in the motion prediction literature [108, 73, 106]. It contains 15 actions performed by 7 subjects. Human pose is represented using the 3D coordinates of 32 joints. As previous work [106, 88, 105], we load the exponential map representation of the dataset, remove global rotation and translation, and generate the Cartesian 3D coordinates of each joint mapped onto a uniform skeleton. Following the implementation of existing works [106, 88, 105], subject 5 is reserved for testing, subject 11 for validation and the remaining subjects are used for training. We test each method on the same 64 sequences formed using indices randomly selected from Subject 5's sequences. Note that the observed keyposes are extracted using the sequence only up to the present time index as opposed to the entire sequence. The threshold used for keypose extraction is 500mm, and we cluster the keyposes into 1000 clusters.

CMU-Mocap [36] is another standard benchmark dataset for motion prediction and was used in [91, 106, 88]. As explained in [91], the eight action categories with enough trials are used for motion prediction. We used six out of eight actions, *basketball, basketball signal, directing traffic, jumping, soccer, and wash window*, as the sequences for *running and walking* were too short to provide enough input keyposes for our method. One sequence of each action is reserved for testing, one for validation and the rest are used for training. The dataset is loaded and processed in the same manner as Human3.6M. The threshold used for keypose extraction is 250mm, as some sequences are quite short, and we found that extracting more keyposes increases validation accuracy. We cluster the keyposes into 100 clusters, as this dataset is much smaller than Human3.6M and contains only 6 action classes as opposed to 15.

5.2.2 Baselines

We selected the following baselines for comparison purposes: HisRep [105] and TIM-GCN [88] constitute the SOTA among the methods designed for long-term prediction. For HisRep, we evaluate two versions. The first one, HisRep10, was presented as the best model in [105]. It is trained to output 10 frames and iteratively use the predicted frames as input for longer term prediction. We also evaluate HisRep125, which directly predicts 125 frames by taking 150 past frames as input. For TIM-GCN, we trained a model that observes subsequences of

lengths 10, 50 and 100 and predicts 125 frames, hence tailoring the architecture to longer-term predictions of 5 seconds. Finally we compare against Mix&Match [10] and DLow [182], the SOTA methods for multiple long-term motion prediction, trained to predict 125 future frames using 100 past frames. For all the baselines, we used the model that gave the best validation accuracy, to be consistent with our model selection strategy.

5.2.3 Metrics

As in [54], we evaluate the quality and plausibility of the generated motions by passing them through an action classifier trained to predict the action category of a given motion. If the predicted motion is plausible, such a classifier should output the correct class. To focus our evaluation on the quality of the predicted *motions*, we designed a Motion-Only Action Classifier (MOAC) based on the architecture of [93], with the pose stream removed and only the motion stream remaining. It takes as input motions encoded as the difference between poses in consecutive time-steps. This eliminates the scenario of a static prediction scoring very high under this metric. We have trained it on the training sequences of Human3.6M and CMU-Mocap separately. We report the top-K action recognition accuracy in percentages obtained with this classifier. For our method, Mix&Match, and DLow, which can output multiple future predictions, we report the average accuracy over 100 predictions.

We also report the PSKL metric [140], which is the KL divergence between the power spectrums of the ground-truth future motions and the predictions. As the KL divergence is asymmetric, we evaluate it in both directions and denote the results as 'gt-pred' and 'pred-gt' respectively. These values being close indicates that the ground truth and predicted motions are similarly complex.

The mean per-joint position error (MPJPE) is the most commonly used metric to evaluate motion prediction. We report the MPJPE errors at 1 second, which is the conventional long-term timestamp, and at 5 seconds. For multiple-prediction methods, we report the MPJPE results of the closest predicted sequences. We present two results: the MPJPE calculated by finding the sequence with the minimum *average* MPJPE (denoted as "ave") and the sequence with the minimum MPJPE at the second being evaluated (denoted as "best").

Finally, for multiple-prediction methods we report the results of a diversity metric [10, 182] for 100 predictions, calculated by finding the average pairwise $L2$ distances between all pairs of generated sequences.

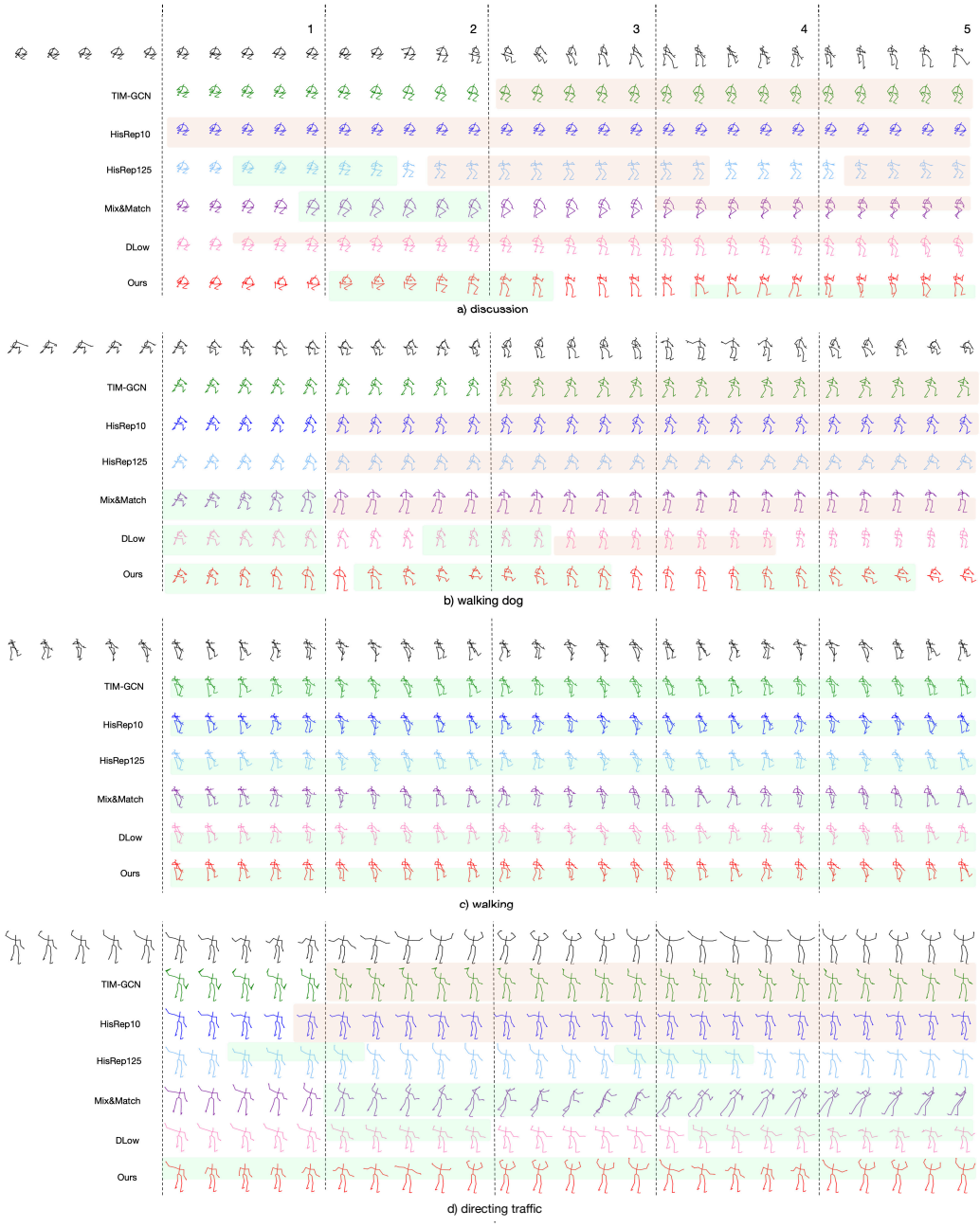


Figure 5.5 – **Qualitative evaluation of our results** on the Human3.6M (Figures a, b, c) and CMU-Mocap (Figure d) datasets. We present the results of: TIM-GCN (green), HisRep10 (dark blue), HisRep125 (light blue), Mix&Match (violet), DLow (pink), Ours (red). For the multiple prediction methods, we display the prediction that has the lowest average MPJPE error with respect to the ground truth. The top black row depicts the ground truth, and the first 5 poses are the conditioning ones. The numbers at the top indicate the future timestamp in seconds. We highlight the segments and body parts that undergo significant motion in green, and the areas that are static for long stretches in red. Our approach yields more dynamic poses for discussion, walking dog and directing traffic, which are acyclic motions. For cyclic motions such as walking, the other methods are also able to produce dynamic poses.

5.2.4 Comparative Results

	top-1	top-2	top-3	top-5
oracle	51	70	79	91
TIM-GCN [88]	16	26	36	55
HisRep10 [105]	21	32	39	53
HisRep125 [105]	20	32	44	60
Mix&Match [10]	18	32	45	61
DLow [182]	16	26	39	56
Ours	**32**	**44**	**54**	**69**

Table 5.1 – **Results of the motion-only action classifier (MOAC) on the Human3.6M dataset**. We compare the classification accuracies for the motions predicted with our method and with the SOTA ones. We also report the accuracies of the oracle, which evaluates the ground-truth future motions, as an upper bound. We report the top-1, top-2, top-3 and top-5 accuracies. The results indicate that the motions predicted by our keypose network are more realistic than those produced by the competing methods.

	top-1	top-2	top-3	top-5
oracle	86	88	90	100
TIM-GCN	44	69	85	95
HisRep10	42	54	62	88
HisRep125	34	48	57	82
Mix&Match	30	39	58	85
DLow	36	49	60	79
Ours	**74**	**81**	**88**	**99**

Table 5.2 – **Results of the motion-only action classifier (MOAC) on the CMU-Mocap dataset**. We compare the MOAC accuracies of our method to SOTA methods. We observe that the trend is similar to Human3.6M, as we achieve higher accuracies than SOTA methods.

We compare our approach to the baselines on Human3.6M and CMU-Mocap in Tables 5.1 and 5.2 on the MOAC metric. In both cases, our method outperforms the others by a large margin. In Tables 5.3 and 5.4, we report the results for the PSKL metric on each dataset, and show that we outperform the other methods by having both lower PSKL values and very close ‘gt-pred‘ and ‘pred-gt‘ values.

In Table 5.5, we evaluate the diversity and MPJPE losses of the predicted sequences. We observe that the diversity value of our method increases as we increase the softmax temperature used for sampling during inference. Increased diversity allows us to achieve lower MPJPE values since we now have a higher chance of sampling the correct future motion. However this also leads to a drop in average MOAC accuracy. This clearly shows the tradeoff between predicting diverse motions and motions that represent the action of interest, or are close to the ground truth. Therefore, in our evaluations we choose to set the temparature to 0.3, trading a bit of accuracy for more diverse predictions. Our method outperforms the others in having both

	gt-pred	pred-gt	average	difference
TIM-GCN [88]	0.0069	0.0098	0.0083	0.0029
HisRep10 [105]	0.0076	0.0129	0.0103	0.0053
HisRep125 [105]	0.0070	0.0097	0.0083	0.0027
Mix&Match [10]	0.0067	0.0075	0.0071	0.0008
DLow [182]	0.0062	0.0080	0.0071	0.0018
Ours	**0.0059**	**0.0061**	**0.0060**	**0.0002**

Table 5.3 – **PSKL results on the Human3.6M dataset.** Lower numbers indicate better results. We report the PSKL values between ground truth and predictions ('gt-pred') and vice-versa ('pred-gt'), their average, and their absolute difference. For the multiple prediction methods, Mix&Match, DLow and Ours, we report the best PSKL value, obtained from the predictions that have the most similar power spectrum to the ground truth future motion. We observe that the trend is similar to the MOAC results, with our method outperforming the SOTA.

	gt-pred	pred-gt	average	difference
TIM-GCN [88]	0.0073	0.0101	0.0087	0.0028
HisRep10 [105]	0.0061	0.0081	0.0071	0.0020
HisRep125 [105]	0.0065	0.0093	0.0079	0.0028
Mix&Match [10]	0.0090	0.0104	0.0097	0.0014
DLow [182]	0.0069	0.0073	0.0071	0.0008
Ours	**0.0057**	**0.0062**	**0.0059**	**0.0005**

Table 5.4 – **PSKL results on the CMU-Mocap dataset.** Lower numbers indicate better results. We report the PSKL values between ground truth and predictions ('gt-pred') and vice-versa ('pred-gt'), their average, and their absolute difference. For the multiple prediction methods, Mix&Match, DLow and Ours, we report the best PSKL value, obtained from the predictions that have the most similar power spectrum to the ground truth future motion. The trend is similar to the results on the Human3.6M dataset, we outperform the SOTA methods.

high diversity, the best average MOAC accuracies, and low MPJPE. For MPJPE, at 1 second we are comparable to the other methods, but at 5 seconds, especially for the "best" MPJPE, our performance is noticeably better.

Note that we present the MPJPE results to give a complete picture but do not believe it to be the best metric for evaluating long term prediction methods, especially for acyclic motions. Consider, for example, the walking dog action, where the subject, in the middle of the walk, kneels down to pet the dog and stands back up. Our method, in contrast to others, is able to predict the order of these motions, as reflected by our high MOAC score. By contrast, the MPJPE is highly sensitive to the timing of the motions and can be thrown off by slight shifts in timing. For instance, the MPJPE error between two phase shifted sinusoidals, or sinusoidals of slightly different frequencies, would be high. For such cases, the MPJPE between a flat signal and a sinusoidal might even be lower, but the flat signal would be completely incorrect.

	diversity ↑	accuracy ↑	1s ave ↓	1s best ↓	5s ave ↓	5s best ↓
TIM-GCN [88]	-	16	143	143	196	196
HisRep10 [105]	-	21	**116**	**116**	197	197
HisRep125 [105]	-	20	136	136	191	191
Mix&Match [10]	1002	18	161	156	244	237
DLow [182]	3501	16	136	131	**189**	171
Ours (0.1)	6936	**34**	177	168	208	173
Ours (0.3)	10328	32	157	138	196	151
Ours (0.5)	12362	30	154	125	191	137
Ours (0.7)	13491	27	144	118	**190**	130
Ours (1.0)	**14995**	20	145	**116**	194	**127**

Table 5.5 – **Results on the diversity metric, top-1 MOAC accuracy and MPJPE errors** on the Human3.6 dataset. Higher diversity values indicate more variation in the multiple future predictions and lower MPJPE values indicate closer predictions to ground truth future motion. We have highlighted close best results in bold. We provide several results of our method with varying sampling softmax temperature, indicated in parentheses. As the temperature increases, the diversity values of the predictions increase and MPJPE values decrease, however the average top-1 MOAC accuracy begins to decrease as well.

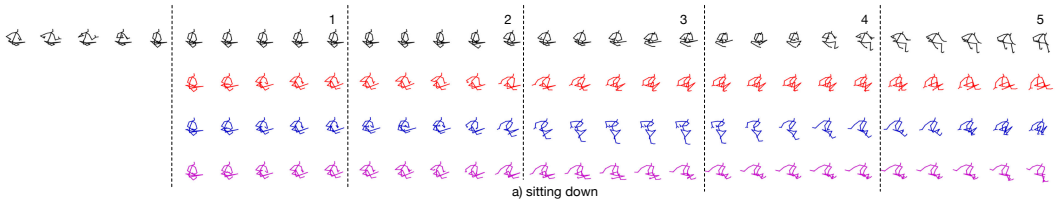


Figure 5.6 – **Qualitative results of our multiple motion prediction** obtained by sampling the predicted label distribution. The numbers at the top indicate the future timestamp in seconds. The top row in black depicts the ground truth, and the remaining rows in color are our multiple generated motions. The sampled motions are diverse, yet can all still be classified as “sitting down".

Fig. 5.5 depicts qualitative results for the discussion, walking dog and walking actions of Human3.6M, and for the directing traffic action of CMU-Mocap. Close visual inspection reveals that, while all methods work reasonably well on cyclic motions such as walking, ours does better on the acyclic ones, such as walking dog. It produces wider motion ranges than the others that tend to predict less dynamic motions. Fig. 5.6 depicts qualitative results for multiple predictions. Our method is capable of generating diverse, yet still plausible motions.

5.2.5 Ablation Study on Keypose Retrieval Methods

We evaluate the effect of using keyposes obtained via different strategies: sampling, clustering and ours. The naive-sampling method evenly samples the motion every 15 frames, which is the average keypose duration from our method. The keyposes are then clustered without any keypose pruning. This method also doesn't require predicting durations, as the duration will

	top-1	top-2	top-3	top-5
Naive-sampling	28	38	51	67
Naive-sampling-pruned	30	42	52	63
Clustering	24	37	48	66
Ours	**32**	**44**	**54**	**69**

Table 5.6 – **Analysis of the method to obtain keyposes.** We compare the MOAC accuracies of different keypose methods. Our method achieves higher classification accuracies than the other ones, indicating that the quality of the keyposes affects the performance.

always be 15. We also evaluate naive-sampling-pruned, where the keyposes are found through naive sampling, and then pruned. The clustering method performs clustering on *every* pose in the sequence, rather than only on the poses found via our optimization strategy and pruned afterwards.

As shown in Table 5.6, our keypose method achieves the highest MOAC accuracies. The comparison with the naive-sampling method emphasizes the importance of having variable-duration keyposes, as opposed to evenly sampling the motion. The comparison with the clustering method emphasizes the importance of optimizing for the keyposes.

5.2.6 Limitations and Failure Modes

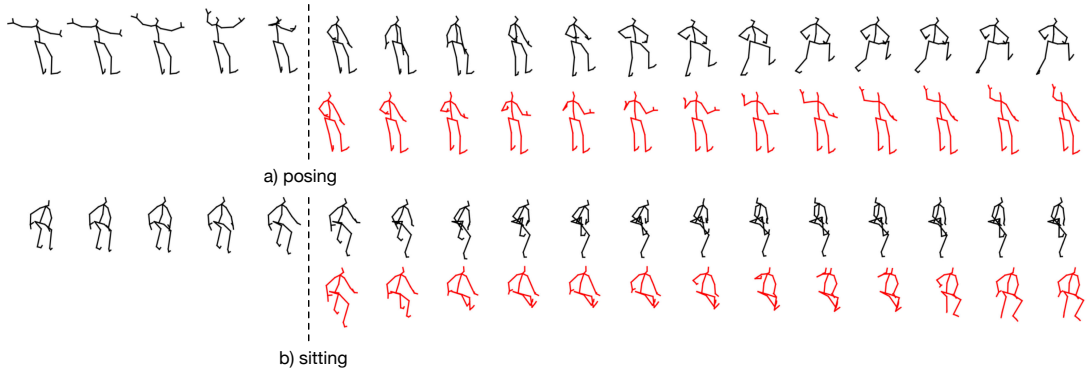


Figure 5.7 – **Examples of failure** to predict the correct keypose labels. Top: Example from the “posing" category. Once our model detects the extended arms, it switches to the waving motion resembling the poses from the “greeting" category. Bottom: Example from the “sitting" category. Although the leg motion is plausible, our prediction lifts a hand to its head, resembling a motion from the “phoning" category.

The main failure mode of our approach arises from incorrect cluster label prediction, as illustrated in Figure 5.7, and from the fact that, while powerful, cluster centers cannot perfectly model all poses. To overcome this, we will study in future work the use of other clustering strategies such as the deep-learning based one of [171] that can be incorporated into our architecture for end-to-end training.

5.3 Conclusion

We have presented an approach to long-term motion prediction. To the best of our knowledge, our work constitutes the first attempt at long-term prediction out to 5 seconds in the future. To this end, we have reformulated motion prediction as a classification problem that guesses in which one of a set of keypose clusters the subject will be. To validate our approach, we have introduced a new action classifier, MOAC, that specifically focuses on the transitions between poses, thus placing an emphasis on the correctness of *motion*, rather than that of *poses*. Our experiments show that our method yields more dynamic and realistic poses than state-of-the-art techniques, even when they are tailored to learn patterns for long-term prediction. Furthermore, our approach lets us easily propose multiple possible outcomes.

Altogether, we believe that our approach could be highly beneficial for autonomous systems, such as an autonomous car that needs more than 1 second window into the future to react to pedestrian motions. Furthermore, the ability to sample many alternative future situations can be exploited to aid the motion planning of autonomous systems. Ultimately, long-term and short-term predictions should be used in a complementary manner, the former to produce long-term probabilistic scenarios for better action planning, and the latter to predict fine details in the immediate future.

In the following chapter, we will discuss an application of human motion analysis and synthesis, which is for the sports domain. We will apply some of the techniques we have studied so far in this thesis; such as using graph convolutional architectures for motion analysis, and training pose-based action recognition networks. By focusing on an application, we understand the use-cases and benefits of studying this field.

6 3D Pose Based Feedback For Physical Exercises

Being able to perform exercises without requiring the supervision of a physical trainer is a convenience many people enjoy. However, the lack of effective supervision and feedback can end up doing more harm than good, which may include causing serious injuries. There is therefore a growing need for computer-aided exercise feedback strategies.

A few recent works have addressed this problem [31, 44, 49, 74, 134, 180]. However, they focus only on identifying whether an exercise is performed correctly or not [31, 134], or they rely on hard-coded rules or thresholds based on joint angles that cannot easily be extended to new exercises [49, 74, 180]. In this chapter, we therefore leverage recent advances in the fields of pose estimation [55, 82, 194], action recognition [93, 189] and motion prediction [6, 105] to design a framework that provides automated and personalized feedback to supervise physical exercises.

Specifically, we developed a method that not only points out mistakes but also offers suggestions on how to fix them without relying on hard-coded, heuristic rules to define what a successful exercise sequence should be. Instead, it learns from data. To this end, we use a two-branch deep network. One branch is an action classifier that tells users what kind of errors they are making. The other proposes corrective measures. They both rely on Graph Convolutional Networks (GCNs) that can learn to exploit the relationships between the trajectories of individual joints. Fig. 6.1 depicts the kind of output our network produces.

To showcase our framework's performance, we recorded a physical exercise dataset with 3D poses and instruction label annotations. Our dataset features 3 types of exercises; squats, lunges and planks. Each exercise type is performed correctly and with mistakes following specific instructions by 4 different subjects. Our approach achieves 90.9% mistake recognition accuracy on a test set. Furthermore, we use the classification branch of our framework to evaluate the performance of the correction branch, considering the correction to be successful if the corrected motion is classified as “correct". Under this metric, our approach successfully corrects 94.2% of users' mistakes. We will make our code and dataset publicly available upon acceptance.

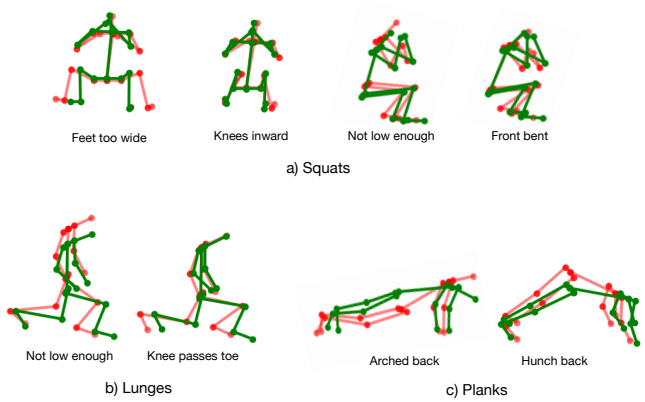


Figure 6.1 – **Example results from our framework** depicting frames from the a) squat, b) lunge, and c) plank classes. The red poses correspond to the exercises performed incorrectly while the green poses correspond to our corrections. Note that although we display a single pose from each mistake type, our framework operates on entire sequences.

6.1 Methodology

Before we introduce our framework in detail, let us formally define the tasks of motion classification and correction. Motion classification seeks to predict the action class c of a sequence of 3D poses from $t = 1$ to $t = N$, denoted as $\mathbf{X}_{1:N}$. We can write this as

$$c = F_{\text{class}}(\mathbf{X}_{1:N}) \,,$$

where F_{class} is the classification function.

We define motion correction as the task of finding the “correct" version of a sequence, which can be written as

$$\hat{\mathbf{X}}_{1:N} = F_{\text{corr}}(\mathbf{X}_{1:N}) \,,$$

where F_{corr} is the correction function and $\hat{\mathbf{X}}_{1:N}$ is the corrected sequence. Ideally, the corrected sequence should be of class “correct". We can use the classification function to verify that this is the case, i.e.,

$$c_{\text{correct}} = F_{\text{class}}(\hat{\mathbf{X}}_{1:N}) \,,$$

where c_{correct} is the label corresponding to a correctly performed exercise.

Given these definitions, we now describe the framework we designed to address these tasks and discuss our training and implementation details.

6.1.1 Exercise Analysis Framework

Our framework for providing exercise feedback relies on GCNs and consists of two branches: One that predicts whether the input motion is correct or incorrect, specifying the mistake being made in the latter case, and one that outputs a corrected 3D pose sequence, providing a detailed feedack to the user. We refer to these two branches as the “classifier" and “corrector" models, respectively.

Inspired by Mao *et al.* [106], we use the DCT coefficients of joint trajectories, rather than the 3D joint positions, as input to our model. This allows us to easily process sequences of different lengths. The corrector model outputs DCT coefficient residuals, which are then summed with the input coefficients and undergo an inverse-DCT transform to be converted back to a series of 3D poses.

To reduce the time and space complexity of training the classifier and the corrector separately and to improve the accuracy of the model, we combine the classification and correction branches into a single end-to-end trainable model. Figure 6.2 depicts our overall framework. It takes the DCT coefficients of each joint trajectory as input. The first layers are shared by the two models, and the framework then splits into the classification and correction branches.

Furthermore, we feed the predicted action labels coming from the classification branch to the correction branch. We depict this in Figure 6.2 as the “Feedback Module". Specifically, we first find the label with the maximum score predicted by the classification branch, convert this label into a one-hot encoding, and feed it to a fully-connected layer. The resulting tensor is concatenated to the output of the first graph convolutional blocks (GCB) of the correction branch. This process allows us to explicitly provide label information to the correction module, enabling us to further improve the accuracy of the corrected motion.

Implementation and Training Details.

We primarily use GCB similar to those presented in [106] in our network architecture, depicted in Figure 6.3. These modules allow us to learn the connectivity between different joint trajectories. Each graph convolutional layer is set to have 256 hidden features. Additionally, our classification branch borrows ideas from Zhang *et al.*'s [189] action recognition model. It is a combination of GCB modules and the frame-level module architecture of [189] consisting of convolutional layers and spatial max-pooling layers.

We train our network in a supervised manner by using pairs of incorrectly performed and correctly performed actions. However, it is not straightforward to find these pairs of motions. The motion sequences are often of different lengths, and we face the task of matching incorrectly performed actions to the closest correctly performed action from the same actor. To do so, we make use of Dynamic Time Warping (DTW) [142], which enables us to find the minimal alignment cost between two time series of different lengths, using dynamic programming. We compute the DTW loss between each incorrect and correct action pair candidate and select

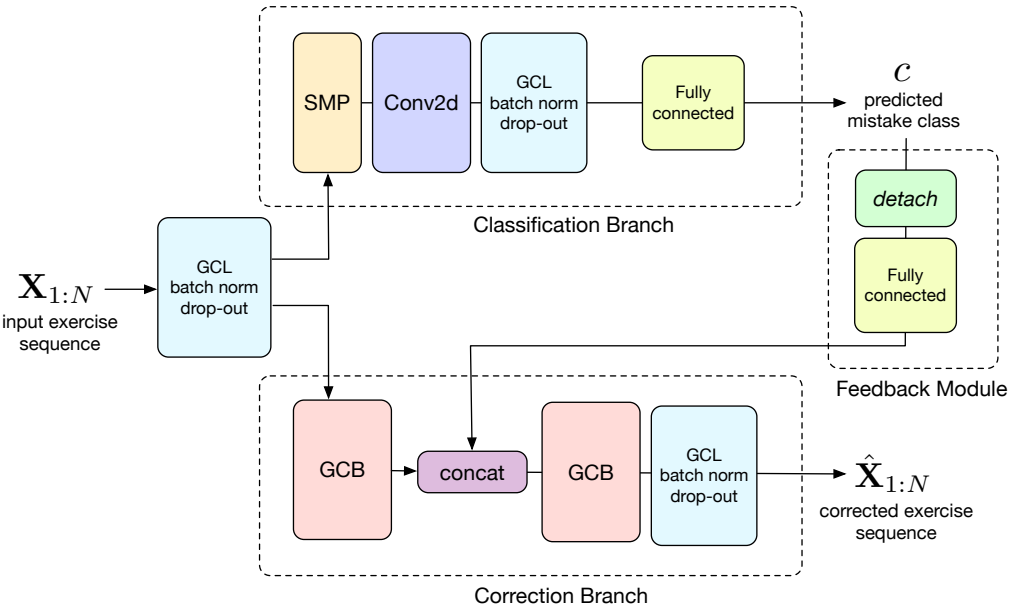


Figure 6.2 – **Our framework** consists of a classification and a correction branch. The input is first fed to a shared graph convolutional layer, which is afterwards split into classification and correction branches. The classification branch identifies the type of mistakes made by the user and the correction branch outputs a corrected pose sequence. The result of the classification branch is fed to the correction branch via a feedback module.

the pair with the smallest loss value.

We use the following loss functions to train our model.

- E_{corr}: The loss of the correction branch, which aims to minimize the soft-DTW [39] loss between the corrected output sequence and the closest correct motion sequence, determined as described previously. The soft-DTW loss is a differentiable version of the DTW loss, implemented by replacing the minimum operation by a soft minimum.

- E_{smooth}: The smoothness loss on the output of the correction branch, to ensure the produced motion is smooth and realistic. It penalizes the velocities of the output motion by imposing an L2 loss on them.

- E_{class}: The loss of the classification branch, which aims to minimize the cross entropy loss between the predicted logits and the ground-truth instruction label.

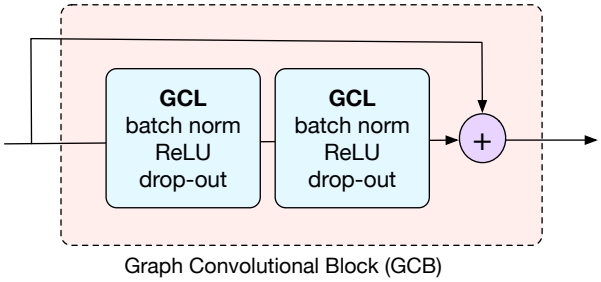


Figure 6.3 – **Graph Convolutional Block (GCB)** consisting of graph convolutional layers, batch normalization layers, ReLUs and drop-outs, as introduced in [106].

We combine these losses into

$$E_{\text{loss}} = w_{\text{corr}} E_{\text{corr}} + w_{\text{class}} E_{\text{class}} + w_{\text{smooth}} E_{\text{smooth}}, \tag{6.1}$$

where E_{loss} is the overall loss and w_{corr}, w_{class}, w_{smooth} are the weights of the correction, classification, and smoothness losses, respectively. For our experiments we set $w_{\text{corr}} = 1$, $w_{\text{class}} = 1$, and $w_{\text{smooth}} = 1e-3$.

During training, we use curriculum learning in the feedback module: Initially the ground-truth instruction labels are given to the correction branch. We then use a scheduled sampling strategy similar to [19], where the probability of using the ground-truth labels instead of the predicted ones decreases from 1 to 0 linearly as the epochs increase. In other words, the ground-truth labels are progressively substituted with the labels predicted by the classification branch, until only the predicted labels are used. During inference, only the predicted labels are given to the correction branch.

We use Adam [80] as our optimizer. The learning rate is initially set to 0.01 and decays according to the equation $\text{lr} = 0.01 \cdot 0.9^{i/s}$, where lr is the learning rate, i is the epoch and s is the decay step, which is set to 5. To increase robustness and avoid overfitting, we also use drop-out layers with probability 0.5. We use a batch size of 32 and train for 50 epochs.

6.2 EC3D Dataset

Existing sports datasets such as Yoga-82 [172], FineGym [146], FSD-1O[147], and Diving48 [181] often include correct performances of exercises but do not include incorrect sequences. They are also not annotated with 3D poses. Therefore to evaluate our approach, we recorded and processed a dataset of physical exercises performed both correctly and incorrectly, and named the "EC3D" (**E**xercise **C**orrection in **3D**) dataset.

Specifically, this dataset contains 3 types of actions, each with 4 subjects who repeatedly

Exercise	Instruction Label	Subject 1	Subject 2	Subject 3	Subject 4	Total (per action)
Squats	Correct	10	10	11	10	132
	Feet too wide	5	8	5	5	
	Knees inward	6	7	5	5	
	Not low enough	5	7	5	4	
	Front bent	5	6	6	7	
Lunges	Correct	12	11	11	12	127
	Not low enough	10	10	10	10	
	Knee passes toe	10	10	11	10	
Planks	Correct	7	8	11	7	103
	Arched back	5	5	11	9	
	Hunch back	10	10	11	9	

Table 6.1 – **The EC3D dataset** with the number of sequences per instruction of each subject, the total number of sequences per instruction and the total number of sequences per action. We reserve Subjects 1, 2, and 3 for training and 4 for testing.

performed a particular correct or incorrect motion as instructed. We show the number of sequences per action and the instructions for each subject in Table 6.1. The dataset contains a total of 132 squat, 127 lunge, and 103 plank action sequences, split across 11 instruction labels.

The videos were captured by 4 GoPro cameras placed in a ring around the subject, using a frame rate of 30 fps and a 1920×1080 image resolution. Figure 6.4 depicts example images taken from the dataset with their corresponding 2D and 3D skeleton representation. The cameras' intrinsics were obtained by recording a chessboard pattern and using standard calibration methods implemented in OpenCV [120].

We annotated the 3D poses in an automated manner, whereas the action and instruction labels were annotated manually. Specifically, the 3D pose annotation was performed as follows: First, the 2D joint positions were extracted from the images captured by each camera using OpenPose [29], an off-the-shelf 2D pose estimation network. We then used bundle adjustment to determine the cameras' extrinsics. For the bundle adjustment algorithm to converge quickly and successfully, additional annotations were made on static landmarks in 5 frames. Since the cameras were static during recording, for each camera, we averaged the extrinsics optimized for each of these frames. Afterwards, these values were kept constant, and we triangulated the 2D poses to compute the 3D poses.

During the triangulation process, we detected whenever any joint had a high reprojection error to catch mistakes in the 2D pose estimates. Such 2D pose annotations were discarded to prevent mistakes in the 3D pose optimization. The obtained 3D pose values were afterwards smoothed using a Hamming filter to avoid jittery motion. Finally, we manually went through the extracted 3D pose sequences in order to ensure that there are no mistakes and that they

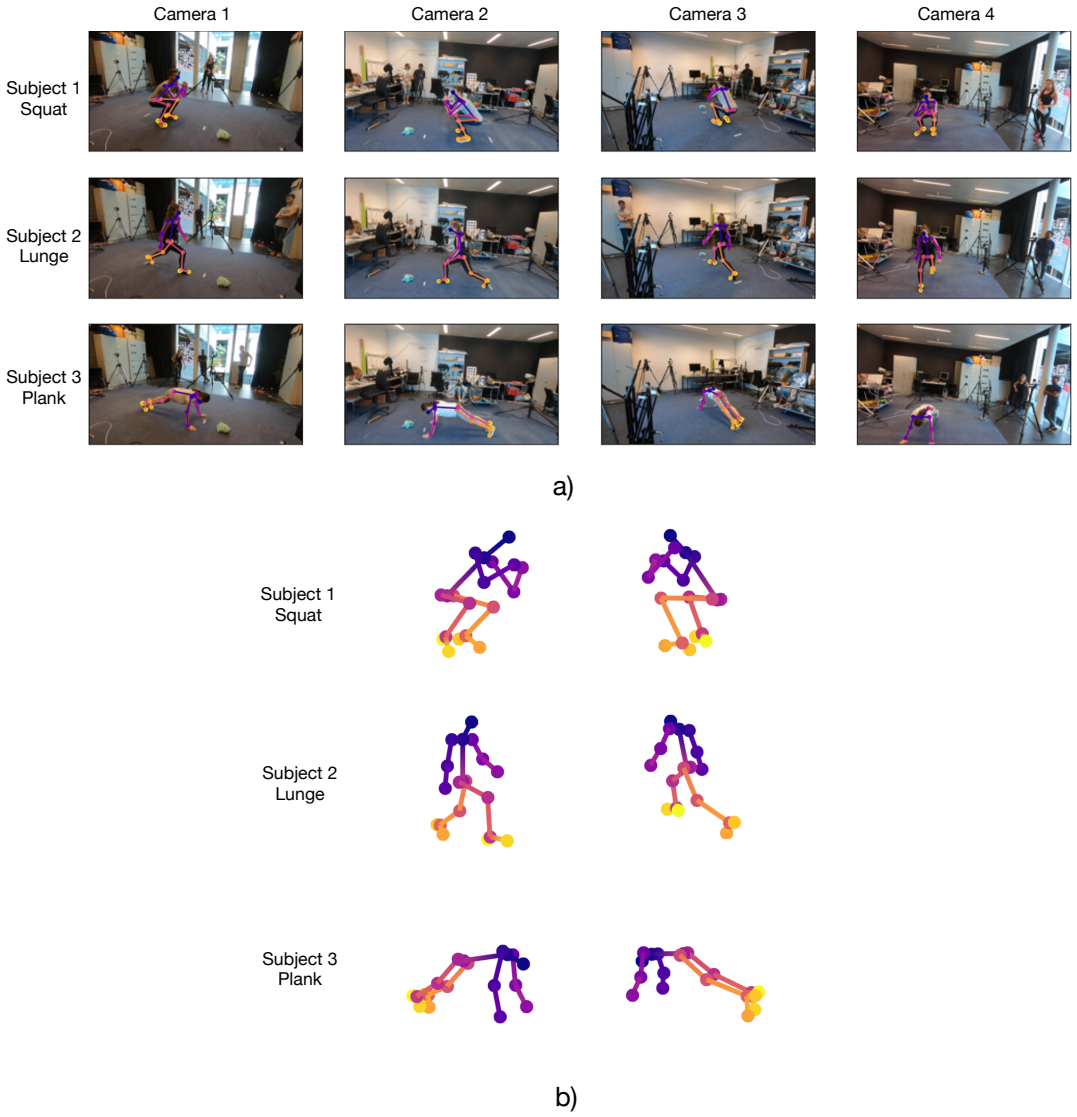


Figure 6.4 – **Examples images from the EC3D dataset,** depicting images from the SQUAT, lunge, and plank classes with their corresponding 3D pose visualizations. a) Images for each exercise type from the dataset from each camera viewpoint, with the 2D poses overlayed. b) The corresponding 3D poses, visualized from two different viewpoints.

are consistent with the desired motion.

To make the resulting 3D poses uniform, we further normalized, centred and rotated them. As the different heights and body sizes of the different subjects cause differences in skeletal lengths, a random benchmark was selected to normalize the skeletal lengths while maintaining the connections between joints. Furthermore, we centered all poses on their hip joint and rotated them so that the spine was perpendicular to the ground and all movements performed in the same direction.

Exercise	Mistake Label	Classification Accuracy (%)	Correction Success (%)
Squats	Correct	90.0	100
	Feet too wide	100	100
	Knees inward	100	100
	Not low enough	100	100
	Front bent	57.1	85.7
Lunges	Correct	66.7	100
	Not low enough	100	60.0
	Knee passes toe	100	90.0
Planks	Correct	85.7	100
	Arched back	100	100
	Hunch back	100	100
Average		90.9	94.2

Table 6.2 – **Results of our classification and correction branches on the EC3D dataset.** We achieve 90.9% recognition accuracy on average and successfully correct 94.2% of the mistakes.

6.3 Evaluation

6.3.1 Dataset and Metrics

We use the EC3D dataset to evaluate our model performance both quantitatively and qualitatively. We use subjects 1, 2, and 3 for training and subject 4 for evaluation.

We use top-1 classification accuracy to evaluate the results of the instruction classification task, as used by other action classification works [93, 189]. For the motion correction task, we make use of the action classifier branch: If the corrected motion is classified as “correct" by our classification branch, we count the correction as successful. We report the percentage of successfully corrected motions as the evaluation metric for this task.

6.3.2 Quantitative Results

We achieve an average mistake recognition accuracy of 90.9% when classifying sequences in EC3D, as shown by the detailed results for each specific exercise instruction in Table 6.2. In the same table, we also show that 94.2% of the corrected results are classified as “correct" by our classification model. The high classification accuracy and correction success show that our framework is indeed capable of analyzing physical exercises and giving useful feedback.

As no existing works have proposed detailed correction strategies, we compare our framework to a simple correction baseline consisting of retrieving the closest “correct" sequence from the training data. The closest sequence is determined as the sequence with the lowest DTW loss value to the input sequence. In Table 6.3, we provide the DTW values between the incorrectly performed input and the corrected output. The DTW loss acts as an evaluation of the accuracy

Exercise	Mistake Label	Retrieval Baseline	Our Framework
Squats	Correct	1.28	**0.56**
	Feet too wide	4.23	**1.46**
	Knees inward	1.61	**0.66**
	Not low enough	1.83	**0.61**
	Front bent	4.74	**2.53**
Lunges	Correct	1.94	**1.82**
	Not low enough	1.86	**1.31**
	Knee passes toe	2.27	**1.48**
Planks	Correct	2.41	**1.79**
	Arched back	12.20	**1.53**
	Hunch back	4.10	**1.09**
Average		3.49	**1.35**

Table 6.3 – **DTW results of the correction branch.** We compare our framework to a simple baseline retrieving the best matching “correct" sequence from the training dataset depending on the classification label. We report the DTW loss between the input and the output sequences (lower is better). Our framework successfully corrects the subject’s mistakes, while not changing the input so drastically that the subject would not be able to recognize their own performance.

of joint positions, as it is an L2 loss on the time aligned sequences. For this metric, the lower, the better, i.e., the output motion should be as close as possible to the original one while being corrected as necessary. Our framework yields a high success rate of correction together with a lower DTW loss than the baseline, thus supporting our claims. Note that we do not evaluate the baseline’s correction success percentage because it retrieves the same sequences that were used to train the network, to which the classification branch might have already overfit.

6.3.3 Qualitative Results

In Figure 6.5, we provide qualitative results corresponding to all the incorrect motion examples from each action category. Note that the incorrect motions are successfully corrected, yet still close to the original sequence. This makes it possible for the user to easily recognize their own motion and mistakes.

6.3.4 Ablation Studies

We have tried various versions of our framework and recorded our results in Table 6.4. In this section, we present the different experiments, also depicted in Figure 6.6, and the discussions around these experiments.

Separated models. We first analyze the results of separated classification networks. According to Table 6.4, our separated classification branch architecture is denoted as “separated classification." We have also evaluated a simpler, fully GCN based separated action classifier branch,

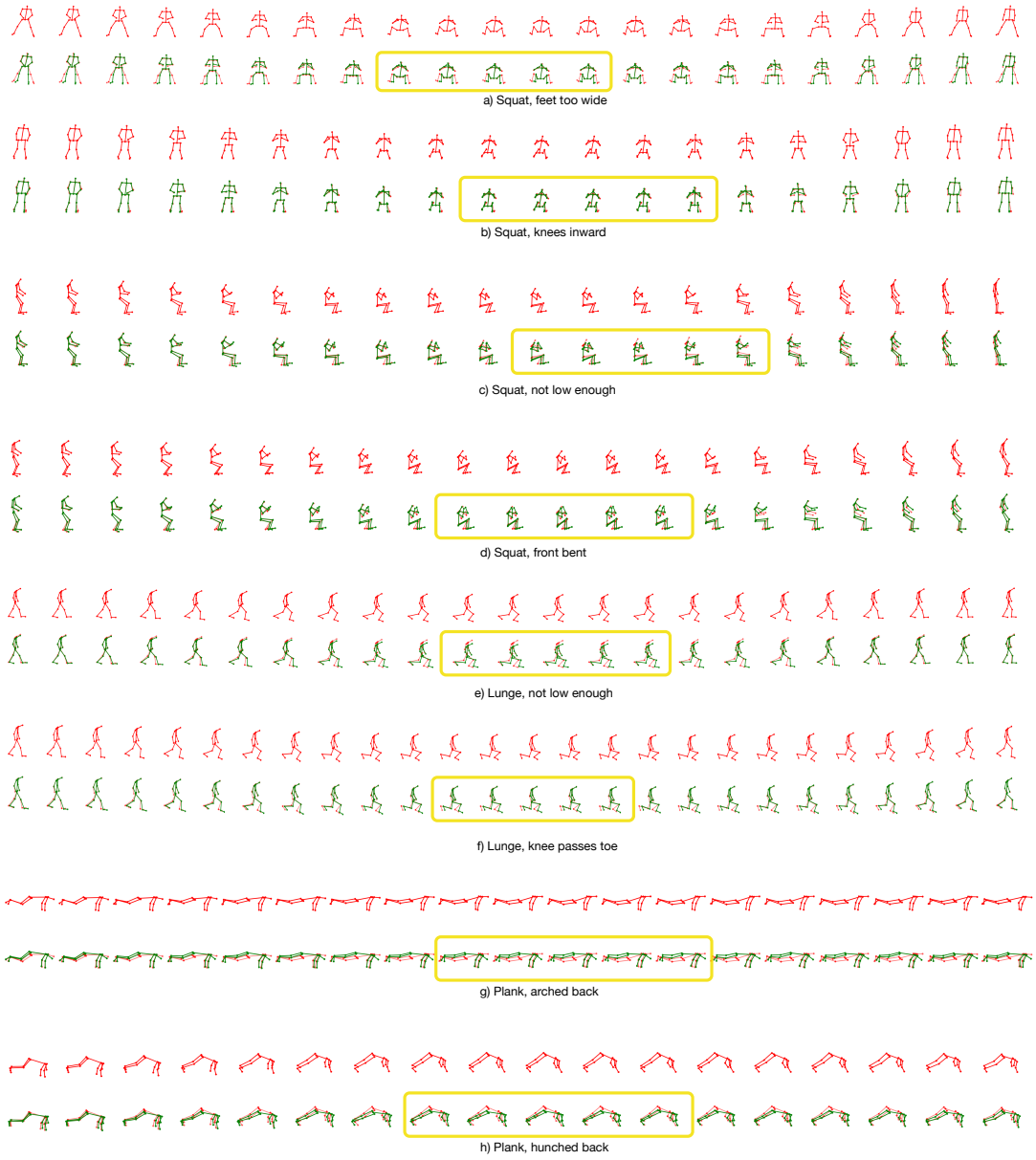


Figure 6.5 – **Qualitative results from our framework** depicting incorrect input motions and corrected output motions from categories a-d) squats, e-f) lunges, g-h) planks. We present the incorrect input sequences (red) in the top row. The corrected sequences (green) overlaid on top of the incorrect input sequences (red) are presented in the bottom row. The most significant corrections are highlighted with a yellow bounding box. We find that our proposals are successful in correcting the incorrect sequences. This figure is best viewed in color and zoomed in on a screen.

denoted as “separated classification (simple)". We show that the results of the classification branch degrade slightly when separated from the correction branch. This indicates that the

classification branch also sees a minor benefit from being part of a combined model. The simpler classification network performs worse than our architecture inspired by [189], showing that the pooling module improves the classification accuracy.

Afterwards, we analyze the results of a separated correction network, denoted as “separated correction". Here the difference is quite profound; we see that separating the correction model from the classification model degrades correction success significantly. We note that 50 epochs was not enough for the separated corrector framework to converge, therefore we trained it for a total of 150 epochs.

Combined models. We train our framework without the feedback module (“combined w/o feedback"), and without the smoothness loss (“combined w/o smoothness"). We find that these perform worse than our model with the feedback module and with the smoothness loss in terms of correction success. This shows that using the classification results as feedback as well as the smoothness loss for training is useful for more successful corrections. We notice that our framework trained without smoothness loss has higher classification accuracy, despite having a lower correction success. We believe this is due to the fact that the smoothness loss acts as a regularizer on the framework, therefore causing slight performance losses to the classification branch. However, the results of the correction success are significantly higher with the smoothness loss. We also evaluate our trained model by passing random incorrect instruction labels to the correction branch instead of the labels predicted by the classification branch (“combined with random incorrect feedback"). The correction success drops significantly, showing that the classification results are indeed very useful for the correction branch.

	Evaluated Variation	Classification Accuracy (%)	Correction Success (%)
Separated	Separated classification (simple)	88.6	-
	Separated classification	89.8	-
	Separated correction	-	83.5
Combined	Combined w/o feedback	82.3	85.3
	Combined with random incorrect feedback	90.9	87.3
	Combined w/o smoothness	**93.4**	87.5
	Ours	90.9	**94.2**

Table 6.4 – **Results of the ablation studies** with several variations of our framework. We report the classification accuracy (%) and the correction success rate (%), where higher is better for both metrics. Our framework benefits greatly from combining the two tasks in a end-to-end learning fashion, from using a feedback module, and from using a pooling layer in the classification branch. The smoothness loss causes slight degradation in classification accuracy but is greatly beneficial for the correction success.

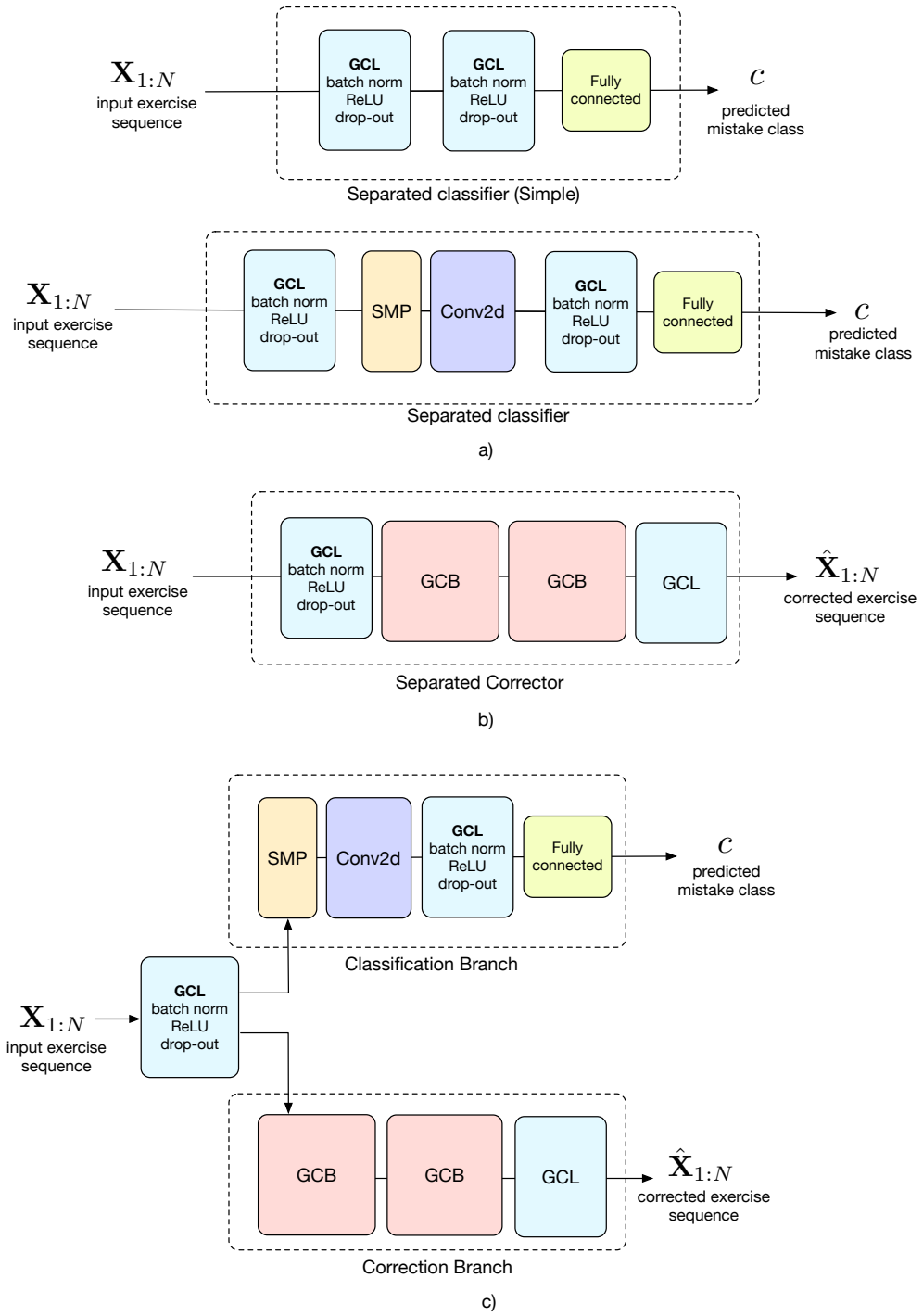


Figure 6.6 – **Ablation study frameworks.** We depict the different architectures we evaluated for the ablation studies: a) Separated classifier (simple) and separated classifier. b) Separated corrector. c) Combined model without feedback. This model does not include a “Feedback Module", the classification branch’s results are not explicitly fed to the correction branch.

6.4 Conclusion

We have presented a 3D pose based feedback framework for physical exercises. We have designed this framework to output feedback in two branches; a classification branch to identify a potential mistake and a correction branch to output a corrected sequence. Through ablation studies, we have validated our network architectural choices and presented detailed experimental results, making a strong case for the soundness of our framework design. We have also introduced a dataset of physical exercises, on which we have achieved 90.9% classification accuracy and 94.2% correction success.

7 Conclusion

We have addressed several problems in the field of human motion analysis and synthesis, namely active viewpoint selection for human pose estimation, human motion prediction, and physical exercise feedback and analysis. We summarize our findings in this chapter, and discuss directions for future research.

7.1 Summary

Human motion analysis and synthesis is an important research direction for many computer vision applications in fields such as autonomous driving, healthcare, and entertainment. We tackle several problems in this domain.

In Chapter 3, we address viewpoint selection for human pose estimation. In a system with a moving camera, we build a framework in which we choose the next best viewpoint in order to have accurate human pose. To do so, we propose using the uncertainty of the human pose from the future viewpoints as a proxy for the potential error of the pose estimation. By selecting the viewpoint with the lowest uncertainty, we create an active viewpoint selection system. Using our active selection method, we achieve lower human pose estimation errors than naive baselines, such as constant rotation, or random viewpoint selection.

In Chapter 4 we consider the human motion prediction problem. Our first approach considers the observed past sequence and studies subsequences of different lengths derived from it: the shortest subsequence is of the immediate past, and the longest looks back furthest into the past. We extract features from the subsequences using different sizes of convolutional kernels, via a temporal inception module (TIM). Among all the compared methods, our method yields the lowest per joint motion prediction error.

In Chapter 5, we address the problem of long-term human motion prediction. Existing work on motion prediction tend to produce static results when trained to predict long term motions, including our own work presented in Chapter 4. To counter this, we use the “keyposes" in the sequence, which are the essential poses from which the other poses can be interpolated. We

design a GRU-based network which given observed past keyposes, predicts future ones. Using the keyposes, we reconstruct the future sequence via interpolation. Our method produces more dynamic and realistic future sequences when compared to existing methods.

In Chapter 6, we consider an application-based problem, providing physical exercise feedback based on the observed human motions. Our framework provides feedback in two forms. The first branch is an action recognition architecture which classifies the user's exercise category and the type of mistake they are making. The second branch outputs the correct version of their incorrectly performed exercise. We achieve 90.9% mistake identification accuracy, and 94.2% success in correcting incorrectly performed exercises.

7.2 Limitations and Future Work

There are many exciting directions for future work concerning the different problems we have studied. In this section, we discuss the limitations of our work and ideas for future research.

7.2.1 Viewpoint Selection for Pose Estimation

In Chapter 3, we have discussed how to use uncertainty as a proxy for error in 3D human pose estimation. Key to the success of our approach is the integration of several sources of uncertainty to form this proxy. Currently, we assume a constant error for the 2D and 3D pose estimates obtained via neural networks. A major improvement to our framework would investigate how to integrate view direction-dependent uncertainty models of deep neural networks, e.g. the network of Prokudin *et al.* [132].

Our framework considers a fairly simple simulation scenario for drone fight. Though we have implemented a physically plausible drone model, we have neglected the possibility of physical obstacles in the environment. In the case of complex scenes with static and dynamic obstacles, we expect our algorithm to outperform any simple, predefined policy. However the flight controller should also be modified to consider virtual no-go areas and restrict the possible flight trajectories accordingly.

Our approach focuses on 3D human pose reconstruction. Using parametric models such as SMPL [99] or GHUM [176], we can also optimize the 3D human shapes [23]. Therefore, our viewpoint selection algorithm can also search for the viewpoint which provides better shape reconstruction. We can add other terms in our loss function specific to human shape estimation, such as a silhouette loss, which corresponds to the difference between the contour rendered using the parametric model and the segmentation of the person in the image [68].

Another interesting future direction could be to use a reinforcement learning based strategy instead of using the uncertainty from the future viewpoint as an approximation. The task can be formulated as a reinforcement learning problem: the “policy" is the active viewpoint selection model, the “state" is the current drone pose and estimated 3D human motion up to

the current frame, and the “reward" is the 3D human pose accuracy obtained from the chosen viewpoint. Using reinforcement learning, the system could learn to recognize several motion cues specific to certain actions and position itself accordingly. Several works have already taken steps in this direction for triangulating human pose [47, 130], and monocular 3D pose estimation [13].

7.2.2 Motion Prediction

We have introduced the Temporal Inception Module for extracting the temporal information of the joint trajectories at different scales. Currently, the subsequence lengths and their corresponding convolutional filter sizes are hard-coded for the datasets used in evaluation. However, human motion sequences can have varying frame rates and sequence lengths, especially if the video stream comes from a source with unreliable throughput. An improvement to the framework could be to implement a system where the subsequence lengths and filter sizes are adjusted automatically according to the input stream.

Our keyposes are automatically extracted from the data as the poses which are the most essential for reconstructing the sequence via interpolation. They can also be used in other related tasks, such as motion synthesis from action labels or verbal descriptions, and action recognition. Moreover, keyposes do not have to be limited to human poses, for instance, Zhao *et al.* recently used similar discrete representations for hand motion analysis [193].

Attention based models are being used increasingly to process complex time-series signals. Recently, they are also being applied to human motion prediction [6, 76, 105] and have shown improved results. However, attention based models, in particular transformers [14], show benefits with large training datasets. Xu *et al.* have tried addressing this limitation with careful initialization and optimization [178]. Maeda *et al.* propose data augmentations which can be used to increase the size of human motion datasets [102]. Using such techniques, transformers can be further applied to human pose estimation, especially for long time horizons, to obtain even more precise results.

7.2.3 Physical Exercise Feedback

We provide a framework for automated feedback for physical exercises. Our framework is trained in a supervised manner, meaning that it requires a dataset with annotations of the exercise that is being performed, as well as a specific mistake that is being made. In our EC3D dataset, the number of exercises and subjects is quite limited. A good way to study the framework further would be to simply use larger datasets and annotate them as necessary, such as that of AIFit [49].

Similar to the motion prediction works, a spatio-temporal attention implementation could also be a good future direction for exercise feedback. Using attention, the framework can learn to relate joints from frames that are far apart in time, leading to an improved recognition of

mistakes in exercises and how to do them correctly in the style of the athlete.

One of the most important challenges of automated physical exercise feedback stems from how different athletes can perform the same exercise in varied ways, all of which can be considered correct. In our EC3D dataset, the subjects performed the exercises in a unified way. Taking inspiration from [190], using a dataset with variations in the exercises, we can learn to decouple the performance style of the athlete and the main content of the exercise.

We hope researchers can make use of these insights and advance motion analysis and synthesis research. We believe there is still much to be discovered and each new advancement leads the way to exciting new applications.

A Refinement Network

In this Appendix, we discuss the details of the refinement network introduced in Chapter 5.

The refinement network is used to improve the results qualitatively. It takes as input the sequences obtained by linear interpolation of the keypose cluster centers and their durations. We recall that the linear interpolation operation to find the pose $\mathbf{X}_{t_1}$ can be expressed as

$$\mathbf{X}_{t_1} = \mathbf{C}_{l_i} + (t_1 - t)\frac{\mathbf{C}_{l_{i+1}} - \mathbf{C}_{l_i}}{d_{i+1}}, \tag{A.1}$$

where $\mathbf{C}_{l_i}$ and $\mathbf{C}_{l_{i+1}}$ are the cluster centers corresponding to the predicted labels l_i and l_{i+1}. d_{i_+1} corresponds to the duration between the two keyposes at i and $i+1$. We use this operation to reconstruct the entire predicted sequence $\mathbf{X}_{1:N}$.

The refinement network is a pretrained network which takes $\mathbf{X}_{1:N}$ as input and refines this result during inference. The resulting sequences are qualitatively much improved: they appear a lot more natural as they are smoother and contain less abrupt motions. We present results in the supplementary video. We have also compared the MOAC accuracies of the predicted sequences compiled using only linear interpolation versus additionally using the refinement network and found them to be very similar. The quantitative results reported in the main paper are obtained using linear interpolation.

The architecture of the refinement network is similar to that of the graph convolutional network (GCN) architecture proposed by Mao *et al.* [106] for short term human motion prediction. We have found that a GCN architecture fits well for capturing the relationship between joint trajectories for the refinement of fine-details. Figure A.1 depicts our network architecture. We use 5 graph convolutional blocks, with 512 output channels. This model is trained with an Adam optimizer, using a learning rate of $5e-2$.

We train this network using input the sequences formed via linear interpolation of the keypose values found within 125 frames. These sequences are compared to their corresponding ground truth sequences. We train the network using three loss functions.

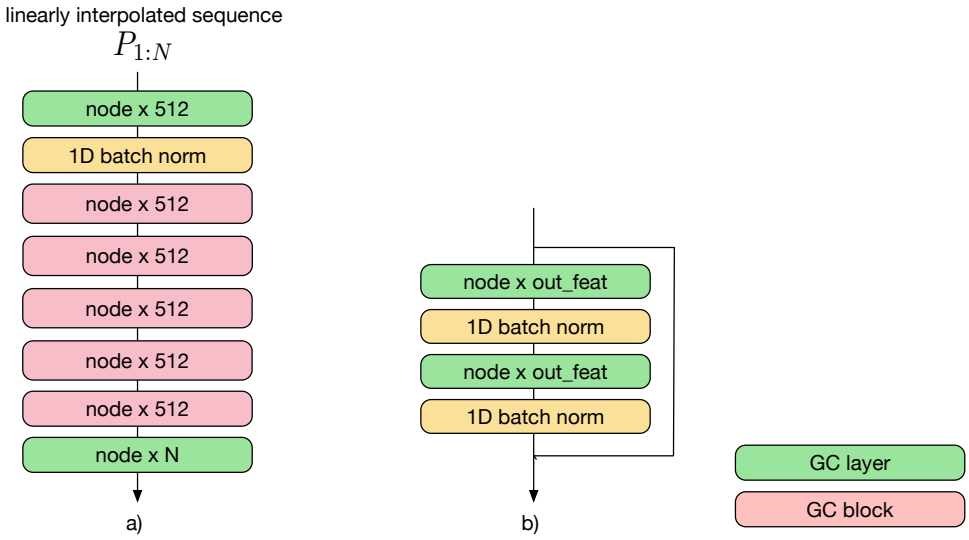


Figure A.1 – **Refinement network model** based on [106]. a) The architecture of the refinement network, where the green blocks represent graph convolution layers (GC layer) and pink blocks represent graph convolution blocks (GC block). b) The contents of a GC block. The dimensions written inside the blocks are the nodes and the number of output features, respectively, of each GC layer and GC block. We have 66 nodes for H36m and 96 nodes for CMU-MoCap datasets, equal to the number of joint trajectories for each dataset. The final number of output features N is set to 125, the number of frames in a 5 second sequence.

- E_{pose}: The MSE loss between the poses of the predicted sequence and the ground truth sequence.
- E_{vel}: The MSE loss between the velocities of the predicted sequence and the ground truth sequence.
- E_{bone}: The MSE loss between the bone lengths of the predicted sequence and the ground truth sequence.

The losses are combined as,

$$E = w_{\text{pose}} E_{\text{pose}} + w_{\text{vel}} E_{\text{vel}} + w_{\text{bone}} E_{\text{bone}} \,, \tag{A.2}$$

where w_{pose}, w_{vel}, and w_{bone} weigh the different loss terms. We set these as $0.1, 100, 1e-6$ respectively, as these values give us the lowest validation E_{pose} loss. We primarily consider this loss for validation as the other terms are more used as regularizers.

Bibliography

[1] V. Adeli, E. Adeli, I. Reid, J.C. Niebles, and H. Rezatofighi. Socially and Contextually Aware Human Motion and Pose Forecasting. In *International Conference on Intelligent Robots and Systems*, 2020.

[2] a-gO, 2020. www.a-go.ai/.

[3] N. Ahmed, T. Natarajan, and K. R. Rao. Discrete Cosine Transform. *IEEE Transactions on Computers*, C-23(1):90–93, 1974.

[4] A. Aissaoui, A. Ouafi, P. Pudlo, C. Gillet, Z.-E. Baarir, and A. Taleb-Ahmed. Designing a Camera Placement Assistance System for Human Motion Capture Based on a Guided Genetic Algorithm. *Virtual reality*, 22(1):13–23, 2018.

[5] I. Akhter and M. J. Black. Pose-Conditioned Joint Angle Limits for 3D Human Pose Reconstruction. In *Conference on Computer Vision and Pattern Recognition*, 2015.

[6] E. Aksan, M. Kaufmann, P. Cao, and O. Hilliges. A Spatio-Temporal Transformer for 3D Human Motion Prediction. In *2021 International Conference on 3D Vision (3DV)*, 2021.

[7] A. Alahi, Y. Boursier, L. Jacques, and P. Vandergheynst. Sport Players Detection and Tracking with a Mixed Network of Planar and Omnidirectional Cameras. In *2009 Third ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC)*, pages 1–8, 2009.

[8] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese. Social LSTM: Human Trajectory Prediction in Crowded Spaces. In *Conference on Computer Vision and Pattern Recognition*, 2014.

[9] M. Alexa and W. Mueller. Representing Animations by Principal Components. In *Eurographics*, 2000.

[10] S. Aliakbarian, F. S. Saleh, M. Salzmann, L. Petersson, and S. Gould. A Stochastic Conditioning Scheme for Diverse Human Motion Prediction. In *Conference on Computer Vision and Pattern Recognition*, 2020.

[11] M. Z. Alom, M. Hasan, C. Yakopcic, T. M. Taha, and V. K. Asari. Improved Inception-Residual Convolutional Neural Network for Object Recognition. *Neural Computing and Applications*, 2018.

[12] J. Arunnehru, G. Chamundeeswari, and S. Prasanna Bharathi. Human Action Recognition using 3D Convolutional Neural Networks with 3D Motion Cuboids in Surveillance Videos. *Procedia Computer Science*, 133:471–477, 2018.

[13] M. A. Arzati and S. Arzanpour. Viewpoint Selection for DermDrone using Deep Reinforcement Learning. In *2021 21st International Conference on Control, Automation and*

Systems (ICCAS), pages 544–553, 2021.

[14] A.Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is All You Need. In *Advances in Neural Information Processing Systems*, 2017.

[15] D. P. Azari, Y. H. Hu, B. L. Miller, B. V. Le, and R. G. Radwin. Using Surgeon Hand Motions to Predict Surgical Maneuvers. *Human Factors*, 2019.

[16] E. Barsoum, J. Kender, and Z. Liu. HP-GAN: Probabilistic 3D Human Motion Prediction via GAN. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[17] V. Belagiannis, X. Wang, H. Ben Shitrit, K. Hashimoto, R. Stauder, Y. Aoki, M. Kranzfelder, A. Schneider, P. Fua, S. Ilic, H. Feussner, and N. Navab. Parsing Human Skeletons in an Operating Room. *Machine Vision and Applications*, 27(7):1035–1046, 2016.

[18] R. Benenson, O. Mohamed, J. Hosang, and B. Schiele. Ten Years of Pedestrian Detection, What Have We Learned? In *European Conference on Computer Vision*, pages 613–627, 2014.

[19] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer. Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks. In *Advances in Neural Information Processing Systems*, 2015.

[20] H. BenShitrit, M. Raca, F. Fleuret, and P. Fua. Tracking Multiple Players Using a Single Camera. Technical report, EPFL, 2013.

[21] L. Bertoni, S. Kreiss, and A. Alahi. MonoLoco: Monocular 3D Pedestrian Localization and Uncertainty Estimation. In *International Conference on Computer Vision*, 2019.

[22] V. Bloom, V. Argyriou, and D. Makris. Linear Latent Low Dimensional Space for Online Early Action Recognition and Prediction. *Pattern Recognition*, 72:532–547, 2017.

[23] F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black. Keep It SMPL: Automatic Estimation of 3D Human Pose and Shape from a Single Image. In *European Conference on Computer Vision*, 2016.

[24] C. Bregler and J. Malik. Tracking People with Twists and Exponential Maps. In *Conference on Computer Vision and Pattern Recognition*, June 1998.

[25] J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun. Spectral Networks and Locally Connected Networks on Graphs. In *International Conference on Learning Representations*, 2014.

[26] J. Butepage, M.J. Black, D. Kragic, and H. Kjellstrom. Deep Representation Learning for Human Motion Prediction and Classification. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[27] Judith Bütepage, Hedvig Kjellström, and Danica Kragic. Anticipating Many Futures: Online Human Motion Prediction and Generation for Human-Robot Interaction. In *International Conference on Robotics and Automation*, 2018.

[28] J. Bütepage, H. Kjellström, and D. Kragic. Predicting the What and How - A Probabilistic Semi-Supervised Approach to Multi-Task Human Activity Modeling. In *Conference on Computer Vision and Pattern Recognition*, pages 2923–2926, 2019.

[29] Z. Cao, T. Simon, S. Wei, and Y. Sheikh. Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields. In *Conference on Computer Vision and Pattern Recognition*, pages 1302–1310, 2017.

[30] Y.W. Chao, J. Yang, B.L. Price, S. Cohen, and J. Deng. Forecasting Human Dynamics from

Static Images. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[31] Steven Chen and Richard R. Yang. Pose Trainer: Correcting Exercise Posture Using Pose Estimation. In *arXiv Preprint*, 2020.

[32] X. Chen and J. Davis. Camera Placement Considering Occlusion for Robust Motion Capture. *Computer Graphics Laboratory, Stanford University, Tech. Rep*, 2(2.2):2, 2000.

[33] Wei Cheng, Lan Xu, Lei Han, Yuanfang Guo, and Lu Fang. ihuman3d: Intelligent human body 3d reconstruction using a single flying camera. In *2018 ACM Multimedia Conference on Multimedia Conference*, pages 1733–1741. ACM, 2018.

[34] S. Cho and H. Foroosh. Spatio-Temporal Fusion Networks for Action Recognition. *Asian Conference on Computer Vision*, 2018.

[35] S. Choudhury, A. Kapoor G., Ranade, and D. Dey. Learning to Gather Information via Imitation. In *ICRA*, 2017.

[36] CMU Graphics Lab Motion Capture Database, 2010. http://mocap.cs.cmu.edu/.

[37] Combat IQ, 2021. http://www.combatiq.io/.

[38] E. Corona, A. Pumarola, G. Alenyà, and F. Moreno-Noguer. Context-Aware Human Motion Prediction. In *Conference on Computer Vision and Pattern Recognition*, 2020.

[39] M. Cuturi and M. Blondel. Soft-DTW: a Differentiable Loss Function for Time-Series. In *International Conference on Machine Learning*, 2017.

[40] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *Conference on Computer Vision and Pattern Recognition*, pages 886–893, 2005.

[41] J. Daudelin and M. Campbell. An adaptable, probabilistic, next-best view algorithm for reconstruction of unknown 3-d objects. *IEEE Robotics and Automation Letters*, 2(3):1540–1547, 2017.

[42] A. J. Davison, I. Reid, N. Molton, and O. Stasse. Monoslam: Real-Time Single Camera Slam. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1052–1067, June 2007.

[43] C. Diller, T. Funkhouser, and A. Dai. Forecasting Characteristic 3D Poses of Human Actions. In *Conference on Computer Vision and Pattern Recognition*, 2022.

[44] B. Dittakavi, D. Bavikadi, S.V. Desai, S. Chakraborty, N. Reddy, V.N. Balasubramanian, B. Callepalli, and A. Sharma. Pose Tutor: An Explainable System for Pose Correction in the Wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2022.

[45] J. Doshi. Residual Inception Skip Network for Binary Segmentation. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 206–2063, 2018.

[46] Y. Du, W. Wang, and L. Wang. Hierarchical Recurrent Neural Network for Skeleton Based Action Recognition. In *Conference on Computer Vision and Pattern Recognition*, 2015.

[47] Z. Fan, X. Li, and Y. Li. Multi-Agent Deep Reinforcement Learning for Online 3D Human Poses Estimation. *Remote Sensing*, 13(19), 2021.

[48] Z. Fan, Z. Wang, J. Cui, F. Davoine, H. Zhao, and H. Zha. Monocular Pedestrian Tracking from a Moving Vehicle. In *Asian Conference on Computer Vision*, pages 335–346, 2012.

[49] Mihai Fieraru, Mihai Zanfir, Silviu Cristian Pirlea, Vlad Olaru, and Cristian Sminchisescu. AIFit: Automatic 3D Human-Interpretable Feedback Models for Fitness Training. In

Conference on Computer Vision and Pattern Recognition, 2021.

[50] K. Fragkiadaki, S. Levine, P. Felsen, and J. Malik. Recurrent Network Models for Human Dynamics. In *International Conference on Computer Vision*, 2015.

[51] Y. Gao, H. Liu, X. Sun, C. Wang, and Y. Liu. Violence detection using Oriented VIolent Flows. *Image and Vision Computing*, 48-49:37–41, 2016.

[52] S. Garg. Review on suspicious human action recognition. Blog Post.

[53] C. Gebhardt, S. Stevsic, and O. Hilliges. Optimizing for Aesthetically Pleasing Quadrotor Camera Motion. In *ACM SIGGRAPH*, 2018.

[54] P. Ghosh, J. Song, E. Aksan, and O. Hilliges. Learning Human Motion Models for Long-Term Predictions. In *International Conference on 3D Vision*, 2017.

[55] K. Gong, B. Li, J. Zhang, T. Wang, J. Huang, M. Mi, J. Feng, and X. Wang. PoseTriplet: Co-Evolving 3D Human Pose Estimation, Imitation, and Hallucination Under Self-Supervision. In *Conference on Computer Vision and Pattern Recognition*, 2022.

[56] Google Glasses, 2014. www.google.com/glass/start.

[57] A. Gosztolai, S. Gunel, V. Lobato-Rios, M. Abrate, D. Morales, H. Rhodin, P. Fua, and P. Ramdya. LiftPose3D, a Deep Learning-Based Approach for Transforming Two-Dimensional to Three-Dimensional Poses in Laboratory Animals. *Nature Methods*, 18:975–981, 2021.

[58] L.-Y. Gui, Y.-X. Wang, X. L., and J. M. F. Moura. Adversarial Geometry-Aware Human Motion Prediction. In *European Conference on Computer Vision*, 2018.

[59] C. Guo, X. Zuo, S. Wang, S. Zou, Q. Sun, A. Deng, M. Gong, and L. Cheng. Action2motion: Conditioned Generation of 3D Human Motions. In *Proceedings of the 28th ACM International Conference on Multimedia (MM '20)*, 2020.

[60] G. Habibi, N. Jaipuria, and J. P. How. Context-Aware Pedestrian Motion Prediction in Urban Intersections. In *arXiv Preprint*, 2018.

[61] Y. Han, P. Zhang, T. Zhuo, W. Huang, and Y.Zhang. Going Deeper With Two-Stream ConvNets for Action Recognition in Video Surveillance. *Pattern Recognition Letters*, 107:83–90, 2018.

[62] I. Hasan, S. Liao, J. Li, S. U. Akram, and L. Shao. Pedestrian Detection: Domain Generalization, CNNs, Transformers and Beyond. In *Conference on Computer Vision and Pattern Recognition*, 2021.

[63] T. Hassner, Y. Itcher, and O. Kliper-Gross. Violent Flows: Real-Time Detection of Violent Crowd Behavior. In *3rd IEEE International Workshop on Socially Intelligent Surveillance and Monitoring (SISM) at the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[64] R. Heck and M. Gleicher. Parametric Motion Graphs. In *Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games*, 2007.

[65] J. Hein, M. Seibold, F. Bogo, M. Farshad, M. Pollefeys, P. Fürnstahl, and N. Navab. Towards Markerless Surgical Tool and Hand Pose Estimation. *International Journal of Computer Assisted Radiology and Surgery*, 16:799 – 808, 2021.

[66] B. Hepp, D. Dey, S.N. Sinha, A. Kapoor, N. Joshi, and O. Hilliges. Learn-To-Score: Efficient 3D Scene Exploration by Predicting View Utility. In *European Conference on Computer Vision*, 2018.

[67] B. Hepp, M. Nießner, and O. Hilliges. Plan3D: Viewpoint and Trajectory Optimization for Aerial Multi-View Stereo Reconstruction. *ACM Transactions on Graphics*, 38(1):4, 2018.

[68] Y. Huang, F. Bogo, C. Lassner, A. Kanazawa, P. V. Gehler, J. Romero, I. Akhter, and M. J. Black. Towards Accurate Marker-less Human Shape and Pose Estimation over Time. In *International Conference on 3D Vision (3DV)*, 2017.

[69] Z. Huang, X. Wang, J. Wang, W. Liu, and J. Wang. Weakly-Supervised Semantic Segmentation Network with Deep Seeded Region Growing. In *Conference on Computer Vision and Pattern Recognition*, 2018.

[70] N. Hussein, E. Gavves, and A. Smeulders. Timeception for Complex Action Recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 254–263, 06 2019.

[71] C. Ionescu, I. Papava, V. Olaru, and C. Sminchisescu. Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014.

[72] S. Isler, R. Sabzevari, J. Delmerico, and D. Scaramuzza. An Information Gain Formulation for Active Volumetric 3D Reconstruction. In *International Conference on Robotics and Automation*, 2016.

[73] A. Jain, A.R. Zamir, and S. Savaresea A. adn Saxena. Structural-Rnn: Deep Learning on Spatio-Temporal Graphs. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[74] R. Kanase, A. Kumavat, R. Sinalkar, and S. Somani. Pose Estimation and Correcting Exercise Posture. In *ITM Web of Conferences*, 2021.

[75] A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik. End-To-End Recovery of Human Shape and Pose. In *Conference on Computer Vision and Pattern Recognition*, 2018.

[76] I. Katircioglu, H. Rhodin, J. Spörri, M. Salzmann, and P. Fua. Dyadic Human Motion Prediction. In *arXiv Preprint*, 2022.

[77] I. Katircioglu, B. Tekin, M. Salzmann, V. Lepetit, and P. Fua. Learning Latent Representations of 3D Human Pose with Deep Neural Networks. *International Journal of Computer Vision*, 126(12):1326–1341, 2018.

[78] S. Kiciroglu, H. Rhodin, S. Sinha, M. Salzmann, and P. Fua. Activemocap: Optimized Viewpoint Selection for Active Human Motion Capture. In *Conference on Computer Vision and Pattern Recognition*, 2020.

[79] S. Kiciroglu, W. Wang, M. Salzmann, and P. Fua. Long term motion prediction using keyposes. In *3DV*, 2022.

[80] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimisation. In *International Conference on Learning Representations*, 2015.

[81] K. M. Kitani, B. D. Ziebart, J. A. Bagnell, and M. Hebert. Activity forecasting. In *European Conference on Computer Vision*, 2012.

[82] M. Kocabas, N. Athanasiou, and M. J. Black. VIBE: Video Inference for Human Body Pose and Shape Estimation. In *Conference on Computer Vision and Pattern Recognition*, pages 5252–5262, June 2020.

[83] R. Korbmacher and A. Tordeux. Review of Pedestrian Trajectory Prediction Methods: Comparing Deep Learning and Knowledge-Based Approaches. *IEEE Transactions on*

Intelligent Transportation Systems, 2022.

[84] P. Kothari, S. Kreiss, and A. Alahi. Human Trajectory Forecasting in Crowds: A Deep Learning Perspective. *IEEE Transactions on Intelligent Transportation Systems*, 23(7):7386–7400, 2022.

[85] L. Kovar and M. Gleicher. Flexible Automatic Motion Blending with Registration Curves. In *ACM Symposium on Computer Animation*, pages 214–224, July 2003.

[86] L. Kovar, M. Gleicher, and F. Pighin. Motion Graphs. In *ACM SIGGRAPH*, pages 473–482, July 2002.

[87] S. Kreiss, L. Bertoni, and A. Alahi. PifPaf: Composite Fields for Human Pose Estimation. In *Conference on Computer Vision and Pattern Recognition*, 2019.

[88] T. Lebailly, S. Kiciroglu, M. Salzmann, P. Fua, and W. Wang. Motion Prediction Using Temporal Inception Module. In *Asian Conference on Computer Vision*, 2020.

[89] C. Li, Z. Zhang, W. S. Lee, and G. H. Lee. Convolutional Sequence to Sequence Model for Human Dynamics. In *Conference on Computer Vision and Pattern Recognition*, 2018.

[90] M. Li, S. Chen, X. Chen, Y. Zhang, Y. Wang, and Q. Tian. Actional-Structural Graph Convolutional Networks for Skeleton-Based Action Recognition. In *Conference on Computer Vision and Pattern Recognition*, 2019.

[91] M. Li, S. Chen, Y. Zhao, Y. Zhang, Y. Wang, and Q. Tian. Dynamic Multiscale Graph Neural Networks for 3D Skeleton Based Human Motion Prediction. In *Conference on Computer Vision and Pattern Recognition*, June 2020.

[92] M. Li, S. Chen, Y. Zhao, Y. Zhang, Y. Wang, and Q. Tian. Dynamic Multiscale Graph Neural Networks for 3D Skeleton Based Human Motion Prediction. In *Conference on Computer Vision and Pattern Recognition*, 2020.

[93] Y. Li, L. Yuan, and N. Vasconcelos. Co-Occurrence Feature Learning from Skeleton Data for Action Recognition and Detection with Hierarchical Aggregation. In *International Joint Conference on Artificial Intelligence*, 2018.

[94] Jiahao Lin and Gim Hee Lee. Trajectory Space Factorization for Deep Video-Based 3D Human Pose Estimation. In *British Machine Vision Conference*, 2019.

[95] J. Liu, a. Shahroudy, D. Xu, and G. Wang. Spatio-Temporal LSTM with Trust Gates for 3D Human Action Recognition. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 9907, 10 2016.

[96] J. Liu, G. Wang, P. Hu, L. Duan, and A. Kot. Global Context-Aware Attention LSTM Networks for 3D Action Recognition. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[97] L. Liu, L. Shao, X. Zhen, and X. Li. Learning Discriminative Key Poses for Action Recognition. *IEEE Transactions on Cybernetics*, 43(6):1860–1870, 2013.

[98] W. Liu, J. J. Chen, C. Li, C. Qian, X. Chu, and X. Hu. A Cascaded Inception of Inception Network With Attention Modulated Feature Fusion for Human Pose Estimation. In *AAAI Conference on Artificial Intelligence*, 2018.

[99] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M.J. Black. SMPL: A Skinned Multi-Person Linear Model. *ACM SIGGRAPH Asia*, 34(6), 2015.

[100] Z. Luo, B. Peng, D.-A. Huang, A. Alahi, and L. Fei-Fei. Unsupervised Learning of Long-Term Motion Dynamics for Videos. In *Conference on Computer Vision and Pattern*

Recognition, 2017.

[101] F. Lv and R. Nevatia. Single View Human Action Recognition Using Key Pose Matching and Viterbi Path Searching. In *Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[102] Takahiro Maeda and Norimichi Ukita. MotionAug: Augmentation With Physical Correction for Human Motion Prediction. In *Conference on Computer Vision and Pattern Recognition*, pages 6427–6436, June 2022.

[103] Magic Leap, 2010. http://www.magicleap.com/.

[104] N. Mahmood, N.Ghorbani, N. F. Troje, G. Pons-Moll, and M. J. Black. AMASS: Archive of Motion Capture as Surface Shapes. In *International Conference on Computer Vision*, 2019.

[105] W. Mao, M. Liu, and M. Salzmann. History Repeats Itself: Human Motion Prediction via Motion Attention. In *European Conference on Computer Vision*, 2020.

[106] W. Mao, M. Liu, M. Salzmann, and H. Li. Learning Trajectory Dependencies for Human Motion Prediction. In *International Conference on Computer Vision*, 2019.

[107] E. Marchand, H. Uchiyama, and F. Spindler. Pose Estimation for Augmented Reality: A Hands-On Survey. *IEEE Transactions on Visualization and Computer Graphics*, 22, 2016.

[108] J. Martinez, M.J. Black, and J. Romero. On Human Motion Prediction Using Recurrent Neural Networks. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[109] J. Martinez, R. Hossain, J. Romero, and J.J. Little. A Simple Yet Effective Baseline for 3D Human Pose Estimation. In *International Conference on Computer Vision*, 2017.

[110] D. Mehta, H. Rhodin, D. Casas, P. Fua, O. Sotnychenko, W. Xu, and C. Theobalt. Monocular 3D Human Pose Estimation in the Wild Using Improved CNN Supervision. In *International Conference on 3D Vision*, 2017.

[111] D. Mehta, S. S., O. Sotnychenko, H. Rhodin, M. Shafiei, H. Seidel, W. Xu, D. Casas, and C. Theobalt. Vnect: Real-Time 3D Human Pose Estimation with a Single RGB Camera. In *ACM SIGGRAPH*, 2017.

[112] D. Mehta, O. Sotnychenko, F. Mueller, W. Xu, M. Elgharib, P. Fua, H.P. Seidel, H. Rhodin, G. Pons-Moll, and C. Theobalt. XNect: Real-Time Multi-Person 3D Motion Capture with a Single RGB Camera. *ACM SIGGRAPH*, 39(4), July 2020.

[113] R. Morais, V. Le, T. Tran, B. Saha, M. Mansour, and S. Venkatesh. Learning Regularity in Skeleton Trajectories for Anomaly Detection in Videos. In *Conference on Computer Vision and Pattern Recognition*, 2019.

[114] G. Mori and J. Malik. Estimating Human Body Configurations Using Shape Context Matching. In *European Conference on Computer Vision*, 2002.

[115] S. Mystakidis. Metaverse. *Encyclopedia*, 2(1):486–497, 2022.

[116] T. Nägeli, L. Meier, A. Domahidi, J. Alonso-Mora, and O. Hilliges. Real-Time Planning for Automated Multi-View Drone Cinematography. *ACM SIGGRAPH*, 2017.

[117] T. Nägeli, S. Oberholzer, S. Plüss, J. Alonso-Mora, and O. Hilliges. Real-Time Environment-Independent Multi-View Human Pose Estimation with Aerial Vehicles. *ACM Transactions on Graphics*, 37(6), 2018.

[118] A.J. Naik and M.T. Gopalakrishna. Deep-Violence: Individual Person Violent Activity Detection in Video. *Multimedia Tools and Applications*, 2021.

[119] A. Newell, K. Yang, and J. Deng. Stacked Hourglass Networks for Human Pose Estimation. In *European Conference on Computer Vision*, 2016.

[120] Open Source Computer Vision Library. http://opencv.org.

[121] M. Oren, C.P. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio. Pedestrian Detection Using Wavelet Templates. In *Conference on Computer Vision and Pattern Recognition*, pages 193–199, 1997.

[122] D. Ormoneit, H. Sidenbladh, M.J. Black, and T. Hastie. Learning and Tracking Cyclic Human Motion. In *Advances in Neural Information Processing Systems*, pages 894–900, 2001.

[123] E. Palazzolo and C. Stachniss. Information-driven autonomous exploration for a vision-based mav. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 4:59, 2017.

[124] P. Pareek and A. Thakkar. A Survey on Video-Based Human Action Recognition: Recent Updates, Datasets, Challenges, and Applications. *Artificial Intelligence Review*, 54:2259–2322, 2021.

[125] G. Pavlakos, X. Zhou, K. Derpanis, G. Konstantinos, and K. Daniilidis. Coarse-To-Fine Volumetric Prediction for Single-Image 3D Human Pose. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[126] G. Pavlakos, X. Zhou, K. Derpanis. G. Konstantinos, and D. Kostas. Harvesting Multiple Views for Marker-Less 3D Human Pose Annotations. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[127] D. Pavllo, D. Grangier, and M. Auli. Quaternet: A Quaternion-Based Recurrent Model for Human Motion. In *British Machine Vision Conference*, 2018.

[128] S. Perkowitz. The Bias in the Machine: Facial Recognition Technology and Racial Disparities.

[129] M. Petrovich, M. J. Black, and G. Varol. Action-Conditioned 3D Human Motion Synthesis with Transformer VAE. In *International Conference on Computer Vision*, 2021.

[130] A. Pirinen, E. Gärtner, and C. Sminchisescu. Domes to Drones: Self-Supervised Active Triangulation for 3D Human Pose Reconstruction. In *Advances in Neural Information Processing Systems*, pages 3907–3917, 2019.

[131] A-.I. Popa, M. Zanfir, and C. Sminchisescu. Deep Multitask Architecture for Integrated 2D and 3D Human Sensing. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[132] S. Prokudin, P. Gehler, and S. Nowozin. Deep Directional Statistics: Pose Estimation with Uncertainty Quantification. In *European Conference on Computer Vision*, pages 534–551, 2018.

[133] P. Rahimian and J. K. Kearney. Optimal Camera Placement for Motion Capture Systems. *IEEE Transactions on Visualization and Computer Graphics*, 23(3):1209–1221, 2016.

[134] T. Rangari, S. Kumar, P. Roy, D. Dogra, and B. Kim. Video Based Exercise Recognition and Correct Pose Detection. *Multimedia Tools and Applications*, 04 2022.

[135] J. Redmon and A. Farhadi. YOLOv3: An Incremental Improvement. In *arXiv Preprint*, 2018.

[136] H. Rhodin, C. Richardt, D. Casas, E. Insafutdinov, M. Shafiei, H.-P. Seidel, B. Schiele,

and C. Theobalt. Egocap: Egocentric Marker-Less Motion Capture with Two Fisheye Cameras. *ACM SIGGRAPH Asia*, 35(6), 2016.

[137] M. Roberts, D. Dey, A. Truong, S.N. Sinha, S. Shah, A. Kapoor, P. Hanrahan, and N. Joshi. Submodular Trajectory Optimization for Aerial 3D Scanning. In *International Conference on Computer Vision*, 2017.

[138] A. Robicquet, A. Sadeghian, A. Alahi, and S. Savarese. Learning Social Etiquette: Human Trajectory Understanding in Crowded Scenes. In *European Conference on Computer Vision*, pages 549–565, 2016.

[139] G. Rogez, P. Weinzaepfel, and C. Schmid. Lcr-Net: Localization-Classification-Regression for Human Pose. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[140] A. H. Ruiz, J. Gall, and F. Moreno-Noguer. Human Motion Prediction via Spatio-Temporal Inpainting. In *International Conference on Computer Vision*, 2019.

[141] N. Saini, E. Price, R. Tallamraju, R. Enficiaud, R. Ludwig, I. Martinović, A. Ahmad, and M. Black. Markerless Outdoor Human Motion Capture Using Multiple Autonomous Micro Aerial Vehicles. In *International Conference on Computer Vision*, October 2019.

[142] H. Sakoe and S. Chiba. Dynamic Programming Algorithm Optimization for Spoken Word Recognition. In *IEEE Trans. on Acoustics, Speech, and Sig. Proc*, 1978.

[143] SecondSpectrum, 2015. http://www.secondspectrum.com/.

[144] S. Shah, D. Dey, C. Lovett, and A. Kapoor. AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles. In *Field and Service Robotics*, 2017.

[145] A. Shahroudy, J. Liu, T. Ng, and G. Wang. NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[146] D. Shao, Y. Zhao, B. Dai, and D. Lin. FineGym: A Hierarchical Video Dataset for Fine-grained Action Understanding. In *Conference on Computer Vision and Pattern Recognition*, 2020.

[147] L. Shenglan, L. Xiang, H. Gao, Q. Hong, H. Lianyu, J. Dong, Z. Aibin, L. Yang, and G. Ge. FSD-10: A Fine-Grained Classification Dataset For Figure Skating. *Neurocomputing*, 413:360–367, 2020.

[148] L. Shi, L. Wang, C. Long, S. Zhou, M. Zhou, Z. Niu, and G. Hua. Sparse Graph Convolution Network for Pedestrian Trajectory Prediction. In *Conference on Computer Vision and Pattern Recognition*, 2021.

[149] W. Shi, F. Jiang, and D. Zhao. Single Image Super-Resolution with Dilated Convolution Based Multi-scale Information Learning Inception Module. *International Conference on Image Processing*, pages 977–981, 2017.

[150] L. Sigal. Human Pose Estimation. *Encyclopedia of Computer Vision*, 2011.

[151] I. Simon and S. Oore. Performance RNN: Generating Music with Expressive Timing and Dynamics, 2017.

[152] C. Sminchisescu, A. Kanaujia, Z. Li, and D. Metaxas. Discriminative Density Propagation for 3D Human Motion Estimation. In *Conference on Computer Vision and Pattern Recognition*, 2005.

[153] C. Sminchisescu, A. Kanaujia, and D. Metaxas. Conditional ,odels for Contextual Human Motion Recognition. *Computer Vision and Image Understanding*, 104(2):210–220, 2006.

[154] C. Sminchisescu and B. Triggs. Estimating Articulated Human Motion with Covariance Scaled Sampling. *International Journal of Robotics Research*, 22(6):371–391, June 2003.

[155] C. Sminchisescu and B. Triggs. Kinematic Jump Processes for Monocular 3D Human Tracking. In *Conference on Computer Vision and Pattern Recognition*, 2003.

[156] S. Song, C. Lan, J. Xing, W. Zeng, and J. Liu. An End-to-End Spatio-Temporal Attention Model for Human Action Recognition from Skeleton Data. In *AAAI Conference on Artificial Intelligence*, 2017.

[157] B. Sturm, J. Santos, O. Ben-Tal, and I. Korshunova. Music Transcription Modelling and Composition Using Deep Learning. *Conference on Computer Simulation of Musical Creativity*, 04 2016.

[158] L. Sun, K. Jia, D. Yeung, and B. E. Shi. Human Action Recognition Using Factorized Spatio-Temporal Convolutional Networks. In *Conference on Computer Vision and Pattern Recognition*, 2015.

[159] X. Sun, J. Shang, S. Liang, and Y. Wei. Compositional Human Pose Regression. In *International Conference on Computer Vision*, 2017.

[160] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going Deeper with Convolutions. In *Conference on Computer Vision and Pattern Recognition*, pages 1–9, June 2015.

[161] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the Inception Architecture for Computer Vision. In *Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[162] R. Tallamraju, E. Price, R. Ludwig, K. Karlapalem, H. Bülthoff, M. Black, and A. Ahmad. Active Perception Based Formation Control for Multiple Aerial Vehicles. *IEEE Robotics and Automation Letters*, PP:1–1, 08 2019.

[163] Y. Tang, Y. Tian, J. Lu, P. Li, and J. Zhou. Deep Progressive Reinforcement Learning for Skeleton-Based Action Recognition. In *Conference on Computer Vision and Pattern Recognition*, 2018.

[164] B. Tekin, I. Katircioglu, M. Salzmann, V. Lepetit, and P. Fua. Structured Prediction of 3D Human Pose with Deep Neural Networks. In *British Machine Vision Conference*, 2016.

[165] B. Tekin, P. Marquez-Neila, M. Salzmann, and P. Fua. Learning to Fuse 2D and 3D Image Cues for Monocular Body Pose Estimation. In *International Conference on Computer Vision*, 2017.

[166] A. Tkach, A. Tagliasacchi, E. Remelli, M. Pauly, and A. Fitzgibbon. Online Generative Model Personalization for Hand Tracking. *ACM Transactions on Graphics*, 36(6):243, 2017.

[167] D. Tome, C. Russell, and L. Agapito. Lifting from the Deep: Convolutional 3D Pose Estimation from a Single Image. In *arXiv Preprint*, 2017.

[168] R. Urtasun, D. Fleet, and P. Fua. Monocular 3D Tracking of the Golf Swing. In *Conference on Computer Vision and Pattern Recognition*, June 2005.

[169] R. Urtasun, D. Fleet, and P. Fua. Temporal Motion Models for Monocular and Multiview 3D Human Body Tracking. *Computer Vision and Image Understanding*, 104(2-3):157–177, 2006.

[170] R. Urtasun and P. Fua. 3D Human Body Tracking Using Deterministic Temporal Motion

Models. In *European Conference on Computer Vision*, May 2004.

[171] A. van den Oord, O. Vinyals, and K. Kavukcuoglu. Neural Discrete Representation Learning. In *Advances in Neural Information Processing Systems*, 2017.

[172] M. Verma, S. Kumawat, Y. Nakashima, and S. Raman. Yoga-82: A New Dataset For Fine-Grained Classification of Human Poses. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2020.

[173] Viso.ai, 2017. http://www.viso.ai/.

[174] B. Wang, E. Adeli, H.-K. Chiu, D.-A. Huang, and J. C. Niebles. Imitation Learning for Human Pose Prediction. In *International Conference on Computer Vision*, pages 7123–7132, 2019.

[175] D. Xiang, H. Joo, and Y. Sheikh. Monocular Total Capture: Posing Face, Body, and Hands in the Wild. In *Conference on Computer Vision and Pattern Recognition*, 2019.

[176] H. Xu, E. G. Bazavan, A. Zanfir, W. T. Freeman, R. Sukthankar, and C. Sminchisescu. GHUM & GHUML: Generative 3D Human Shape and Articulated Pose Models. In *Conference on Computer Vision and Pattern Recognition*, June 2020.

[177] L. Xu, L. Fang, W. Cheng, K. Guo, G. Zhou, Q. Dai, and Y. Liu. FlyCap: Markerless Motion Capture Using Multiple Autonomous Flying Cameras. *IEEE Transactions on Visualization and Computer Graphics*, PP, 2016.

[178] P. Xu, D. Kumar, W. Yang, W. Zi, K. Tang, C. Huang, J. C. K. Cheung, S. J. D. Prince, and Y. Cao. Optimizing Deeper Transformers on Small Datasets. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, 2021.

[179] C. Yang, Y. Xu, J. Shi, B. Dai, and B. Zhou. Temporal Pyramid Network for Action Recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 588–597, 06 2020.

[180] Lei Yang, Yingxiang Li, Degui Zeng, and Dong Wang. Human Exercise Posture Analysis based on Pose Estimation. In *2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, 2021.

[181] L. Yingwei, L. Yi, and V. Nuno. RESOUND: Towards Action Recognition without Representation Bias. In *European Conference on Computer Vision*, 2018.

[182] Y. Yuan and K. Kitani. Dlow: Diversifying Latent Flows for Diverse Human Motion Prediction. In *European Conference on Computer Vision*, 2020.

[183] K. Yuen. *Appendix A: Relationship between the Hessian and Covariance Matrix for Gaussian Random Variables*, pages 257–262. John Wiley & Sons, Ltd, 2010.

[184] A. Zanfir, E. Marinoiu, and C. Sminchisescu. Monocular 3D Pose and Shape Estimation of Multiple People in Natural Scenes - the Importance of Multiple Scene Constraints. In *Conference on Computer Vision and Pattern Recognition*, June 2018.

[185] P. Zell, B. Wandt, and B. Rosenhahn. Joint 3D Human Motion Capture and Physical Analysis from Monocular Videos. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017.

[186] J. Zhang and C. Zong. Deep Neural Networks in Machine Translation: An Overview. *IEEE Intelligent Systems*, 30(5), 2015.

[187] J. Y. Zhang, P. Felsen, A. Kanazawa, and J. Malik. Predicting 3D Human Dynamics from Video. In *International Conference on Computer Vision*, 2019.

[188] P. Zhang, C. Lan, J. Xing, W. Zeng, J. Xue, and N. Zheng. View Adaptive Neural Networks for High Performance Skeleton-Based Human Action Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[189] P. Zhang, C. Lan, W. Zeng, J. Xing, J. Xue, and N. Zheng. Semantics-Guided Neural Networks for Efficient Skeleton-Based Human Action Recognition. In *Conference on Computer Vision and Pattern Recognition*, 2020.

[190] Y. Zhang, Y. Zhang, and W. Cai. Separating Style and Content for Generalized Style Transfer. In *Conference on Computer Vision and Pattern Recognition*, 2018.

[191] B. Zhao, X. Wu, Z.-Q. Cheng, H. Liu, and J. Feng. Multi-View Image Generation from a Single-View. In *arXiv Preprint*, 2017.

[192] Z. Zhao, S. Kiciroglu, H. Vinzant, Y. Cheng, I. Katircioglu, M. Salzmann, and P. Fua. 3D Pose Based Feedback for Physical Exercises. In *arXiv Preprint*, 2022.

[193] Z. Zhao, X. Zhao, and Y. Wang. TravelNet: Self-Supervised Physically Plausible Hand Motion Learning From Monocular Color Images. In *International Conference on Computer Vision*, 2021.

[194] C. Zheng, S. Zhu, M. Mendieta, T. Yang, C. Chen, and Z. Ding. 3D Human Pose Estimation With Spatial and Temporal Transformers. In *International Conference on Computer Vision*, pages 11656–11665, October 2021.

[195] X. Zhou, Q. Huang, X. Sun, X. Xue, and Y. We. Weakly-Supervised Transfer for 3D Human Pose Estimation in the Wild. In *arXiv Preprint*, 2017.

[196] X. Zhou, A. S. Liu, A. G. Pavlakos, A. V. Kumar, and K. Daniilidis. Human Motion Capture Using a Drone. In *International Conference on Robotics and Automation*, 2018.

[197] Y. Zhou, Z. Li, S. Xiao, C. He, Z. Huang, and H. Li. Auto-Conditioned Recurrent Networks for Extended Complex Human Motion Synthesis. In *International Conference on Learning Representations*, 2018.

[198] S. Zuffi, J. Romero, C. Schmid, and M. J. Black. Estimating Human Pose with Flowing Puppets. In *International Conference on Computer Vision*, 2013.

SENA KICIROGLU

senakicir.github.io ⋄ senakicir@gmail.com ⋄ +41779455849

PhD candidate working in machine learning and computer vision,
with a focus on human motion analysis.

EDUCATION

EPFL - Ecole Polytechnique Federale de Lausanne 2017 - Continuing (Expected February 2023)
PhD Candidate in Computer Science with the EDIC Fellowship
Computer Vision Laboratory (CVLAB), Supervisors: Prof. Pascal Fua and Dr. Mathieu Salzmann
Selected courses: Deep learning, machine learning, computer vision, advanced computer graphics

Bilkent University 2013 - 2017
Undergraduate In Electrical and Electronics Engineering CGPA: 3.96/4.0, High Honor Student
Comprehensive Scholarship

National University Of Singapore 2015
Undergraduate Exchange Student Program

AWARDS

- EPFL IC (CS department) teaching assistant award (2021).
- EPFL IC service award (2019), for serving as a part of EPIC (the CS graduate student association).
- First place in Microsoft AI Summer School Poster Competition (2018).
- Fulbright Scholarship: Selected candidate (2016), turned down.
- Placed **41'st** in the Turkish Nationwide University Entrance Exams among 1.8 million students (2013).
- KYK Scholarship (2013-2017) A monthly stipend given by the Turkish government for outstanding performance in the Turkish Nationwide University Entrance Exams.
- Is Bank's "Golden Youths" Award (2013) - For outstanding performance in the Turkish Nationwide University Entrance Exams.
- Comprehensive scholarship from Bilkent University (2013-2017) - Covers 100% tuition, plus monthly stipend for the entire undergraduate study.

PUBLICATIONS

- Z Zhao, **S Kiciroglu**, H Vinzant, Y Cheng, I Katircioglu, M Salzmann, P Fua. 3D Pose Based Feedback for Physical Exercises. *ACCV 2022*.
- **S Kiciroglu**, W Wang, M Salzmann, P Fua. Long Term Motion Prediction Using Keyposes. *3DV 2022* - **oral**.
- T Lebailly, **S Kiciroglu**, M Salzmann, P Fua, W Wang. Motion Prediction Using Temporal Inception Module. *ACCV 2020*.
- **S Kiciroglu**, H Rhodin, S Sinha, M Salzmann, P Fua. ActiveMocap: Optimized Viewpoint Selection For Active Human Motion Capture. *CVPR 2020* - **oral**.

INTERNSHIPS

Microsoft - Autonomous Systems Group June 2021 - August 2021
Worked on visual odometry using pretrained models trained on large amounts of synthetic data.

EPFL - Processor Architecture Lab (LAP) June 2016 - July 2016
Was accepted into the Summer@EPFL program at EPFL, Switzerland. Implemented a string sorting algorithm for Intel HARP using VHDL and C++.

National Magnetic Resonance Research Center (UMRAM) June 2015 - July 2015
Worked a project on the applications of nonlinear gradient fields in MRI, increasing the number of channels using VHDL and C.

TEACHING AND SERVICES

- Teaching assistant for "Introduction to Machine Learning" course, 2018-2022. Head TA for last four semesters.
- Reviewer for the conferences ICCV, CVR, ICPR, ACCV, TPAMI.
- Supervised master's thesis of Hugues Vinzant.
- Supervised semester projects of Ziyi Zhao, Yuan Cheng, Arda Alpay, Dhruti Shah, Tim Lebailly, and Daniel Suter.

SELECTED PROJECTS

Long Term Motion Prediction Using Keyposes *PhD Project (2019 - Ongoing)*
We predict diverse and realistic long term (5 second) human motion by making use of "keyposes", the set of poses which we can use to accurately reconstruct the complete sequence. Oral presentation in 3DV 2022. [pdf].

Motion Prediction Using Temporal Inception Module *PhD Project (2017-Ongoing)*
We predict human motion accurately in the short and long term future by making use of a Temporal Inception Module (TIM). Using TIM, we produce input embeddings using convolutional layers, by using different kernel sizes for different input lengths. Published in ACCV 2020 [pdf].

3D Pose Based Motion Correction for Physical Exercises *PhD Project (2019-Ongoing)*
We recognize incorrectly performed physical exercise sequences and provide feedback as a corrected version of the performed exercise. Published in ACCV 2022 [pdf].

Active Human Motion Capture *PhD Project (2017-Ongoing)*
Given a short video sequence, we introduce an algorithm that predicts which viewpoints should be chosen by a moving camera to capture future frames so as to maximize 3D human pose estimation accuracy. Oral presentation in CVPR 2020 [pdf].

Overlapping Lasers On Moving Target *Undergraduate Senior Project (2016 - 2017)*
Worked in a team of five to overlap two laser pointers on a moving object using image processing and control theory. (A miniaturized version of LaWS developed by the US Navy.) Implemented in C++.

4x4x4 LED Cube *Undergraduate Digital Circuit Design Project (2014)*
A mini light show using a 4X4X4 LED Cube, implemented using VHDL [video]. **Won a best project award among my cohort.**

SKILLS

Programming	Python, PyTorch
Languages	English (Very fluent, TOEFL score: 116/120), Turkish (Native Fluency), French (Beginner)

MISCELLANEOUS

- EPIC (CS PhD Student Association) Committee Member (2018): Took part in organizing EDIC Open House '19, board game events, lunch talks, etc.
- IEEE Bilkent Student Branch: Public Relations Coordinator (2014- 2015) and active member (2013-2017).
- Hobbies: Cross-stitching and embroidery, writing book reviews, bouldering, drawing portraits.