Short-term Prediction of User Motion from Live Tracking Data

Divyendu Dutta

April 14, 2024

Technische Universität Dresden

INTERACTIVE
MEDIA LAB
DRESDEN

Interactive Media Lab Dresden
Institute of Software and Multimedia Technology
Faculty of Computer Science

CMS-PRO - Research Project Documentation

Short-term Prediction of User Motion from Live Tracking Data

Divyendu Dutta

Supervisor Dr.-Ing. Wolfgang Büschel
Interactive Media Lab Dresden
Technische Universität Dresden

Examiner Prof. Dr.-Ing. Raimund Dachselt
Interactive Media Lab Dresden
Technische Universität Dresden

April 14, 2024

Divyendu Dutta
Short-term Prediction of User Motion from Live Tracking Data
CMS-PRO - Research Project Documentation, April 14, 2024
Examiner: Prof. Dr.-Ing. Raimund Dachselt
Supervisor: Dr.-Ing. Wolfgang Büschel

Technische Universität Dresden
Faculty of Computer Science
Institute of Software and Multimedia Technology
Interactive Media Lab Dresden
Nöthnitzer Str. 46
D-01062 and Dresden

Abstract

Live human tracking via the Kinect has seen an increasing rise in usage across various domains. However, there exists a delay in terms of when the Kinect tracks human motion and when such data is available to us for use in real-time, hampering user experience and hindering analysis.

This research project aims to address the aforementioned issue by investigating the potential to remedy this problem using human motion prediction. Specifically, it explores the capacity to integrate 3D human motion prediction models with live-tracked data in the short term to reduce the Kinect's latency between tracking and processing.

Starting with an exploration of the literature and the related work, various 3D human motion prediction models were evaluated in terms of their latency and accuracy. This included both autoregressive and non-autoregressive models. The results were then analyzed to make an informed selection of a top-performing model suitable for use in real-time applications.

The STS GCN model, which is a graph convolutional network, stood out amongst all the evaluated models, having the quickest inference time and competitive accuracy. Additionally, a diffusion-based model was evaluated due to its ability for noise correction in data.

Acknowledgement

First and foremost, I express my gratitude to my supervisor and advisor, Dr.-Ing. Wolfgang Büschel, who has spent valuable hours from his workday helping me grow from a novice student to a capable researcher. His focus on detail and accuracy has in turn helped me hone these qualities for myself. When discussing ideas and observations, his advice and comments helped me figure out and correct many of my mistakes, both in the research project and its report. I am forever grateful for his time and knowledge.

I am also profoundly grateful to Prof. Dr.-Ing. Raimund Dachselt for offering me the opportunity to undertake this remarkable research endeavor and allowing me to utilize the resources of the Interactive Media Lab for this research project. Dipl.-Ing. Ricardo Langner's "Clean Thesis" was invaluable to the creation of this research project's report. I highly appreciate Mr. Manuel Kaufmann's clarification about the results of his paper.

Contents

Introduction 1

The Microsoft Kinect has seen increasing usage in a wide range of fields such as Virtual Reality (VR), Augmented Reality (AR), robotics, and Human-Computer Interaction (HCI), due to its capability to track the 3D motion of human beings in real-time at a low cost. In the context of immersive applications such as VR and AR systems, the Kinect enables users to interact with virtual environments through natural movements. This enhances the overall experience by adding an element of realism for the users. For robotics, the Kinect plays a crucial role in human-robot interaction. By tracking the user's movements, robots can mimic and respond to human gestures, allowing for more intuitive and seamless interactions between humans and robots. When it comes to Human-Computer Interaction (HCI), the Kinect offers new and immersive ways to interact with digital media. Applications like gaming, fitness, and entertainment benefit significantly from this technology as users can engage with the media more naturally. However, there exists a nontrivial amount of delay from when the Kinect tracks the user motion and when this data is available to us for use in code. This delay could impede analysis and sabotage the user experience in the context of real-time applications such as visual analysis systems.

Motivation The aspiration to address this issue motivates us to utilize a 3D human motion prediction model to forecast the future poses of live-tracked human data and reduce the latency between tracking and processing. However, the integration of a 3D human motion prediction model with live-tracked data is still mainly uninvestigated despite recent advancements made in motion prediction models. That said, careful consideration and selection of an appropriate model is of utmost importance. This requires various aspects related to the model to be thoughtfully explored and judged.

1.1 Task Description

The objective of this project is to benchmark various 3D human motion prediction models with the primary goal of selecting a model with the best combination of low latency and high accuracy for use in real-time applications. And as such this work is not a survey of all 3D human motion prediction models. The following outlines the key tasks completed within this project at a high level:

1. Comprehensive exploration of the relevant literature and related prior work.
2. Evaluation and comparison of the latency of specifically selected 3D human motion prediction models
3. Evaluation and comparison of the accuracy of specifically selected 3D human motion prediction models

Please note that all the models were evaluated using standardized human motion datasets rather than using actual live-tracked human motion data from a Kinect. This decision was made due to the substantial effort required for data setup and preprocessing before it can be effectively utilized by the models. Furthermore, not all models were evaluated using a common framework since such a framework exists but does not support many human motion prediction models. However, efforts were made to maintain consistency by ensuring that certain parameters of all evaluated models were kept as similar as possible, where feasible.

1.2 Research Project Report Structure

Chapter 2

This chapter provides relevant background for live human tracking, short-term human motion prediction and how the two can be combined. Furthermore, it discusses the basis for selecting an appropriate human motion prediction model for use in a real-time setting.

Chapter 3

This chapter details the methodology behind the latency and accuracy evaluation of various 3D human motion prediction models.

Chapter 4

This chapter describes and analyzes the latency and accuracy results obtained from the research project.

Chapter 5

This chapter provides a brief conclusion of the project. Furthermore, it describes the potential future work.

2 Background and Related Work

In this chapter, related background information is presented concerning live human tracking and short-term human motion prediction, along with an exploration of their potential integration. Moreover, the context of live human tracking is established through perspectives such as Motion Capture (MoCap), Human-Computer Interaction (HCI), and the application of Microsoft Kinect. Additionally, the chapter delves into the rationale behind the selection of an optimal human motion prediction model suitable for real-time applications.

2.1 Live Human Tracking

2.1.1 Motion Capture (MoCap) in the Context of Live Human Tracking

Full body motion capture (MoCap) is the surveillance and recording of the movement of human beings and objects using different technologies. Such technologies can be classified as either optical or non-optical systems. IMUs (Inertial Measurement Units) are typically used in non-optical systems whereas cameras (including depth cameras) are generally used in optical systems. Moreover, optical systems can be grouped into marker-less or marker-based depending on whether the actor being tracked needs to wear special devices for movement tracking or not [ZL16].

MoCap sees a wide range of use cases such as in robotics [YH09], computer animation [@Rap14], and medicine [AN04] among others. When this is applied specifically to capture the motion of humans and is achieved in real-time, we get live human tracking. For it to be considered "live" or "real-time", such a method or system needs to "respond in a reasonable time or before a deadline, in a reliable manner" [ZL16].

2.1.2 Human Computer Interaction (HCI) and Live Human Tracking

Visual analysis of 3D position data of rigid bodies in-situ ie, directly in the original environment can lead to better analysis of its spatial interaction in Augmented Reality (AR) environments [BLD21]. This can be further extended to an entire humanoid avatar (ie, for all the individual joints) rather than just a single rigid body, the human-tracked data being the raw analysis data for it.

2.1.3 Live Human Tracking using Microsoft Kinect

Historically speaking, the usage of multiple cameras to capture motion in real-time was prevalent [MGB07; YDY03]. But now, improvements in camera technology and better algorithms allow a single camera to be used for tracking motion [ZL16]. Particularly, when a real-time depth camera is used, a single such camera is satisfactory for the task [GWK05].

One point to note is that the accuracy of marker-less camera based tracking systems are lower compared to IMUs or marker-based camera based tracking systems [M+20]. But at the same time, it's important to consider that in some scenarios such as in Human-Computer Interaction, highly accurate tracking data is not always necessary since the focus is more on real-time assessment.

The Kinect was released by Microsoft in November 2010 and can produce color along with the approximate depth of each pixel. Since its release, its affordable price has made it a popular choice in the development of Natural User Interfaces (NUI) and, as a single camera marker-less optical system for tracking human movement in real-time. It supports various camera resolutions and aspect ratios with a maximum frame rate of 30 frames per second (fps) [@Mic22b].

The low cost of the Microsoft Kinect has resulted in its widespread utilization in several research areas ranging from medicine to workplace efficiency. [AMM; Auv+15] focus on utilizing the Kinect for analyzing human movement such as walking. [Gal+14] test the accuracy of the Kinect to measure motion in Parkinson's disease and conclude that the Kinect has the potential to be a low-cost, home-based sensor to measure movement. [Wan+13] use the Kinect to provide a natural user interface for training and evaluation of emergency evacuations and other real-time applications of crowd simulation. An open-source package utilizing the Kinect for hand pose and gesture recognition was developed by [Ped+14]. With regards to sports and fitness, [Cas+14] propose a prototype platform called Online-Gym which allows users to interact using a Kinect and participate in online gymnastics sessions.

[DA14] assess the use of the Kinect to observe and study human posture. They conclude that the correct classification of postures is significantly dependent on the camera's point of view relative to the position of the tracked subject.

2.2 Short-Term Human Motion Prediction

A crucial ability of human beings that allows us to navigate and interact with our surroundings is the capability to predict what other human beings are likely to do in the upcoming moments. For machines, this capability is encapsulated as 3D human (user) motion prediction which is articulated as predicting the most likely future movement poses of a human while considering their past motion poses as input. Generally, the predicted future poses is called a "target sequence" and the human's past motion poses taken as input is called a "seed sequence".

This task of pose prediction can further be classified into two categories, short-term or long-term, based on the length of the target sequence being aimed for. Furthermore, such a task is typically conceived as a generative modeling task. Generative modeling is a subfield of machine learning that focuses on enabling machines to imagine and synthesize new substances [Lam21]. This in itself is a difficult task, more so for pose prediction since it requires combining spatial and temporal information from the seed sequence to predict the future poses.

Human pose prediction has a wide range of applications such as in robotics [Che+18; Wag+18; Wen+16], human robot interaction [BKK18; LS17; KSS16], autonomous navigation [Man+20], computer graphics [Lev+12; KGP08] and 3D human/object tracking [Gon+11; UFF06].

2.2.1 Overview of Various Human Motion Prediction Models

Traditionally, recurrent neural networks (RNNs) have been widely preferred for the task of human motion prediction, aiming to forecast 3D human movement due to their capability to capture temporal dependencies in sequential data [Tor+19; AKH19; PGA18]. [MBR17] show via empirical evidence that prevailing RNNs have difficulty obtaining good performance for long-term and short-term predictions. They suggest improvements to the existing RNN architectures but conclude that their new architecture is still unable to perform well.

[Mao+19; Sof+21; Dan+21] utilize a graph convolutional network (GCN) to accomplish the task of human motion prediction. In a similar vein, [Ma+22] propose a two-network system comprising of, a spatial dense GCN and a temporal dense GCN which operate alternatively. [Aks+21; Saa+23a; MVO21] utilize the transformer architecture to forecast human poses due to the recent success of transformer models for text generation. These works illustrate the versatility of transformer models, showcasing their ability to be applied in either autoregressive or non-autoregressive modes.

More simplistic methods such as [SFS21; BSA20; SJA22] take as input the velocities and the positions of the joints in the seed sequence and output the predicted velocities of the future joints from which future joint positions can be computed. More recently, [Saa+23b] propose a diffusion model that frames the prediction of future human poses as a denoising problem. The main idea here is to address the challenge of occlusions during 3D human pose prediction. Furthermore, they show that their model can be used as a black box in combination with other pose prediction models.

2.3 Live Human Tracking via Kinect and Human Motion Prediction

2.3.1 Challenges in Real-Time Human Motion Tracking via Kinect

There exists a non-trivial amount of delay from when the Kinect tracks the user motion and when this data is available to us for use in code (for eg, when we can use this tracking data to animate a humanoid avatar and visualize its motion on screen). This data arrives to us in the form of orientations and positions of various joints of the human body [@Mic22a]. The delay is considered "non-trivial" since it is more than $1000/30 = 33.33$ ms ie, the time between each frame tracked by the Kinect.

The consequence of this delay is that, either we drop frames to ensure that the motion we visualize on the screen matches what is being tracked or if we don't choose to drop frames, then the motion being visualized on screen will lag behind what is being tracked. Both these approaches are not ideal since most visual analysis applications would require visualizing motion without any dropped frames and much lag.

2.3.2 Potential Solutions to Mitigate Latency in Kinect Motion Tracking

One solution to reduce this latency could be to extract raw RGBD data from the Kinect sensors, process and extract the orientations and positions on the GPU ourselves [@AA]. However, such custom solutions are quite complex and involve a lot of steps in the pipeline to finally obtain the tracking data from the Kinect sensors. Also, using these custom solutions would mean that we are no longer able to use the Kinect Body Tracking SDK which integrates easily with popular game engines such as Unity.

Another solution is to utilize the previously mentioned 3D human motion prediction models, given that these models can predict future poses at a faster rate than the delay introduced by the Kinect sensor. Furthermore, along with the previously mentioned point, the models also need to predict the future frames much faster than 33.3 ms so that we're able to make up for the original delay introduced by the Kinect sensor since we would still require tracking data from the Kinect to be provided as input to these models.

2.4 Selection of Appropriate Human Motion Prediction Model

While a lot of work has been done regarding 3D human motion prediction, its integration and use with live tracked data (such as from a Kinect) remains largely unexplored. As per my knowledge, there exists a gap in utilizing real-time live tracked data in the realm of 3D human pose prediction, in spite of improvements in this particular predictive modeling technique.

2.4.1 Latency and Accuracy

Before utilizing the results of a motion prediction model, there are various factors of the model to be considered but the two major ones are latency and accuracy.

Latency is the amount of time required by the motion prediction model to predict one frame of the motion. This gives us valuable insight into selecting a suitable model depending on the amount of latency introduced by the Kinect that we are trying to combat. On the other hand, accuracy is one or more error metrics that inform us about how close the predicted frame is to the ground truth. Latency is measured in the millisecond range whereas different error metrics can have differing units. An appropriate model for this task should exhibit acceptable values of both latency and accuracy, since a model which predicts a single frame quickly but is not accurate is not of much use in a real-world scenario.

2.4.2 Autoregressive vs Non Autoregressive Models

Like most models that work on temporal data and predict future data from past data, 3D human motion prediction models can be either autoregressive or non-autoregressive in nature.

Autoregressive models break down the prediction of future frames into steps, where each step depends on the previous predictions [Aks+21]. Due to this, such models tend to accumulate errors over time until eventually, the predictions crash to an unlikely result. The implication is that these models are not very good at long-term predictions. Another disadvantage of these models is that due to their step-like prediction behavior, it is not possible to parallelize the prediction of the next N steps since they are generated sequentially one at a time.

Non-autoregressive models predict a specific number of frames in one go, ie. in parallel [Hig+21; MVO21]. But with these models, we have less control over the upper limit of the number of frames being predicted. For example, if such a model is trained to predict 25 frames, we can always use just 5 frames from the prediction and throw away the rest but we would be unable to predict 30 frames using this model unless we train another variant of the model which is specifically tailored to predict 30 frames.

3 Benchmarking 3D Human Motion Prediction Models

This chapter explores the approach used to select and evaluate the latency and accuracy of a range of 3D human motion prediction models spanning autoregressive and non-autoregressive models. The aim was to identify one or more suitable models for real-time application. Additionally, the feasibility of a diffusion-based model is evaluated from a latency perspective, considering its potential utility in refining occluded poses as a preprocessing and postprocessing step in the pose prediction pipeline.

3.1 Initial Requirements to Evaluate 3D Human Motion Prediction Models

Initial exploration of the relevant literature pointed out that there's a substantial amount of work that has been done in the field of 3D human motion prediction. However, conducting evaluations on every such model would not only be impractical but also infeasible for a variety of reasons. The following details the specific criteria that had to be met for a particular model from an existing research work to be considered for evaluation.

1. **Availability of Code**: The authors of the research paper made their model's training and evaluation code publicly available. This was necessary to get a better understanding of the model's functioning and to be able to make necessary code changes to measure the model's latency and accuracy.[@] was utilized in part to determine such research work.

2. **Open Source Compliance**: The research work followed open source policies to ensure their availability for use in future research endeavors.

3. **Availability of Pretrained Models**: Training deep learning models from scratch can require a lot of computing power. Hence to ensure that this work could be completed in the given time, research papers were prioritized which

provided pre-trained models. However, as detailed in subsequent sections, this approach was not always viable. Hence research work offering pre-trained models was used and where necessary models were trained from the ground up.

3.2 Hardware Specification Used for Benchmarking

All 3D human motion prediction models were trained and evaluated using the below hardware and software.

1. Operating System: Windows 11 Pro 64-bit
2. CPU: Intel Core i9-14900K (32 CPUs) 3.2GHz
3. RAM: approx. 65GB
4. GPU: NVidia GeForce RTX 4090
5. WSL 2: Ubuntu 22.04 LTS

3.3 UnPOSed: A Framework to Test and Benchmark Various 3D Human Motion Prediction Models

This work extensively utilized UnPOSed which provides an open-source framework for training and evaluating various human motion prediction models [Saa+23a] in Pytorch. It supports various human pose datasets and also leverages the outcomes of prior open-source research efforts. Essentially, UnPOSed streamlines their training and evaluation processes within its framework.

3.3.1 Brief Overview of Models Evaluated via UnPOSed

The following 3D human motion prediction models were tested and evaluated via UnPOSed.

Zero Velocity is a simple baseline model that uses the last pose of the seed sequence as the prediction of all the future poses. It acts as a simple and competitive baseline to compare the accuracy of all the other models.

STS GCN models the human pose dynamics using a graph convolutional network (GCN). It includes the temporal evolution and the spatial joint interaction within a single graph structure. This facilitates the exchange of information between motion and spatial correlations. It is a smaller model in terms of the number of parameters allowing it to train in a short period of time [Sof+21].

ST Trans is a transformer-based model that utilizes the attention mechanism to capture both spatial and temporal dependencies in human poses. This means that it not only considers the positions of joints at different times (temporal aspect) but also how these joints relate to each other spatially [Saa+23a].

PV LSTM is a sequence-to-sequence LSTM model that takes as input the velocities and positions of the joints of the human and outputs the predicted velocities of the joints in future poses. The positions of the joints are then computed from the predicted joint velocities [BSA20; SJA22].

All the above models are non-autoregressive models which means that they are trained to predict the poses of all the future frames in one go rather than in iterative steps.

3.3.2 Training and Evaluation Dataset Used

Unfortunately, UnPOSed does not provide pre-trained models of the supported human pose prediction models and it was not possible to use the pre-trained models provided by the original research work within the framework itself. Hence in this case, the decision was made to train the above mentioned models from the ground up within UnPOSed since it provided a common and easy way to further assess the models. As per the recommendation of the original research work, all the models were trained for 10 frames of seed sequence and 25 frames of target sequence. Furthermore, due to hardware and time constraints, each model was trained for 15 epochs and the best model was selected.

All the models were trained using the Archive of Motion Capture As Surface Shapes (AMASS) dataset [Mah+19] which unifies multiple MoCap datasets. In AMASS, a human pose is represented by 52 joints, including 22 body joints and 30 hand joints. But since the focus is on predicting human body movement, UnPOSed discarded the hand and static joints, leading to an 18-joint human pose. They also downsample the AMASS dataset to 25 fps. Each of the individual MoCap datasets was downloaded from the AMASS website [@Mah+] and compiled together for further preprocessing by UnPOSed. After preprocessing, the motion sequences contained the positions of each joint converted from meters to millimeters.

3.4 Spatio-Temporal Autoregressive Transformer Model

All the models evaluated via UnPOSed were non-autoregressive. In contrast, [Aks+21] propose a spatio-temporal model which predicts the future poses in an autoregressive manner. Their model learns high-dimensional embeddings for human joints and composes a temporally coherent pose via a decoupled temporal and spatial self-attention mechanism. Their provided pre-trained model, trained on a seed sequence of 120 frames, was evaluated. The length of the target sequence doesn't matter since autoregressive models can iteratively keep predicting future poses but even so, up to 25 future frames were predicted to compare the accuracy of this model with the previously mentioned non-autoregressive models. This model takes as input the joint orientations which are specified in rotation matrix format and the model outputs the future poses as joint orientations as well. Their pretrained model was trained on the AMASS dataset as well although their training, validation and test splits differ from UnPOSed.

3.4.1 Challenges Faced

The CUDA version necessary to run these models on an Nvidia RTX 4090 is 11.0 and above [@; @]. This model was written using Tensorflow 1.12.0 which is compatible with CUDA version 9.0 and the highest version of Tensorflow 1.x ie, 1.15.0 is compatible with CUDA version 10.0 only[@]. Hence it was not possible to run this model on the GPU without migrating the code to Tensorflow 2.x. But such a task required significant code changes and testing which was out of the scope of this research work. Another alternative was to use the Nvidia-tensorflow version which is an Nvidia maintained version of Tensorflow 1.15 that is compatible with CUDA 11.0 and above, making it a suitable solution for this problem [@; @]. However, at present Nvidia does not support the installation and usage of Nvidia-tensorflow on Windows but only for the Linux operating systems [@]. Instead of using a separate system with the Linux operating system, it was decided to use Windows Subsystem for Linux (WSL) 2 with Ubuntu 22.04 LTS.

So in essence, the latency of this model was evaluated non-natively on the GPU via WSL 2 and was tested natively on the CPU on Windows. To determine whether running the model via WSL 2 caused any performance impact, the model was also tested non-natively on the CPU via WSL 2 and its latency was compared with it being run natively on the CPU on Windows.

3.5 Diffusion-Based Model

A source of noise in live-tracked human data arises from occlusions of various body joints. A method to combat such a problem would be to fill in the missing data with a specific value such as 1's or via linear interpolation. However, these methods have limitations. A more effective strategy would be to utilize the human pose as a whole to fill in the missing data. Saadatnejad et al. propose a diffusion model that is capable of predicting future human poses while also denoising the combined seed and target sequence [Saa+23b]. More specifically, their main model is a temporal cascaded diffusion model which consists of a short-term and a long-term diffusion block. They mention that improvement of the long-term prediction was the reason behind the cascaded diffusion model architecture. They further specify that their model can be used as a preprocessing or postprocessing step to refine the inputs and outputs of any other existing 3D human motion prediction model.

Their supplied pre-trained model, trained on a seed sequence of 50 frames and a target sequence of 25 frames, was evaluated. They trained their model on the Human3.6M dataset which consists of 3.6 million body poses with each human pose consisting of 32 joints [Ion+14]. They downsampled the dataset to 25 fps and utilized a subset of 22 joints to represent the human pose. Furthermore, for each frame, their model takes as input the joint positions and outputs the same.

3.6 Code Modifications for Latency and Accuracy Measurement

The following subsections outline the code modifications made for measuring the latency and accuracy of the evaluated models, and the reasoning behind certain adjustments.

3.6.1 Latency Measurement

For each of the models evaluated, changes were made in their inference script to determine the time required by the model to predict 1 frame or 25 frames for autoregressive and non-autoregressive models respectively. For non-autoregressive models, the decision was made to not consider the time for 1 frame but rather for all 25 frames together because in practice inferring from such models would require the user to wait for all 25 frames.

The measured time is in the range of milliseconds. Only a single motion sequence was considered for the latency measurement ie, the number of batches and the batch size was explicitly set to 1. This decision was made to keep the conditions as similar as possible to the model's use in practice. In a realistic use of the model, the user would generally pass in a single seed sequence and expect back the predicted target sequence and hence the latency in such a scenario was measured. Finally, the measured latency also includes the time needed to move the data back and forth between the CPU and GPU.

For each model, the following three values are reported as an average over 5 inference runs of the model:

1. Average time excluding first run
2. Average time including first run
3. Time for first run

These three metrics will be further explored in the next section.

Furthermore, specifically for the Spatio-Temporal autoregressive transformer model, the latency was measured on the GPU and the CPU which led to interesting results. This will also be explored in the next section.

3.6.2 Accuracy Measurement

The existing evaluation scripts of the models were utilized and modified to measure and report the displacement error for each frame, up to 25 frames. It is in the range of millimeters. Displacement error is the distance between the predicted position and the actual position averaged over all joints at a particular frame.

It was not feasible to evaluate the Spatio-Temporal autoregressive transformer model due to the size of the test dataset and hardware limitations. Hence, for this model, the accuracy measurements were taken from the original research paper's results. The paper reports the average displacement error over 6, 12, 18 and 24 frames. Hence, specifically when comparing this model with the other models, the accuracy measurements of the other models were modified appropriately.

3.7 Overview of Model Comparisons

The following table summarizes certain key aspects of the different models tested and evaluated in this work. In the table below, the abbreviations used are as defined.

- AR: Autoregressive, NAR: Non autoregressive
- ST Trans(*): Spatio Temporal Autoregressive Transformer
- Input: Seed sequence length, Output: Target sequence length

Name	**Type**	**Input**	**Output**	**Dataset**
Zero Velocity	NAR	10	25	AMASS
STS GCN	NAR	10	25	AMASS
ST Trans	NAR	10	25	AMASS
PV LSTM	NAR	10	25	AMASS
ST Trans(*)	AR	120	NA	AMASS
Diffusion Model	NAR	50	25	Human3.6M

Tab. 3.1: Summary of different aspects of evaluated models.

4 Results

This chapter analyzes and discusses the latency and accuracy results of the benchmarking efforts accomplished in this research project.

4.1 Latency Results

As mentioned in the previous section, the latency of each evaluated 3D human motion prediction model is measured via the following three values. And, each of these values is reported as an average over 5 inference runs of the model and is in the range of milliseconds.

1. **Average time excluding first run**: Average latency of the model not including the first run. This is the main latency metric the user would focus on when inferring from the model in a real-time scenario.

2. **Average time including first run**: Average latency of the model including the first run. This is reported for the sake of completeness.

3. **Time for first run**: This is the time needed by the model to complete its inference during the first run. This metric is the throwaway time ie, the amount of time the user needs to wait before being able to effectively utilize the inference results of the model.

The reason for making a distinction between the first run and all other runs is that most deep-learning models require a warm-up period. During this first warm-up run, it takes significantly more time for the model to run and complete the inference. Thus, including the time needed for the first run in the average latency of the model skews the metric higher. In practice, this is not something we need to be concerned about since we simply would not use the results from the first run of the model and only utilize the results from the subsequent runs when the model has stabilized in its runtime.

4.1.1 Latency of Models Evaluated as Part of UnPOSed

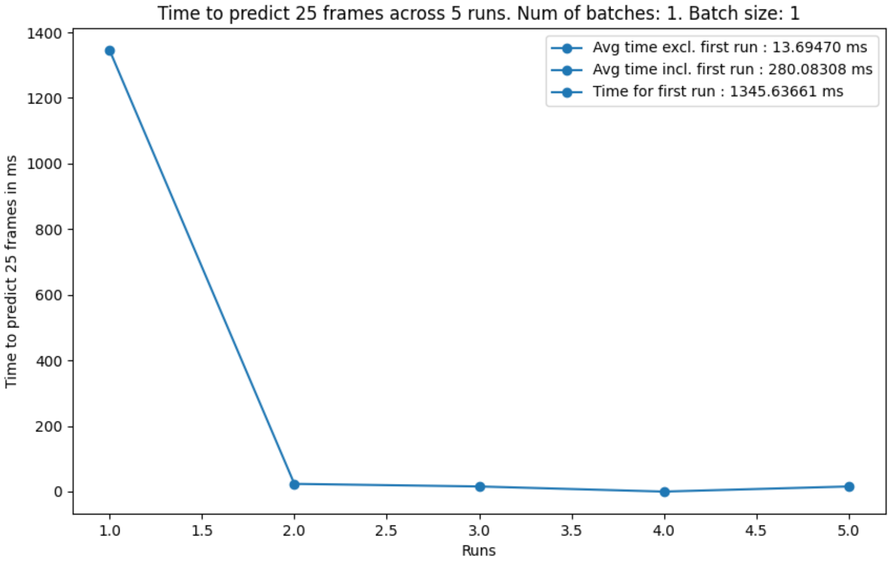

Fig. 4.1: Latency (ms) of the **STS GCN model**. Seed Seq. Len: 10, Target Seq. Len: 25

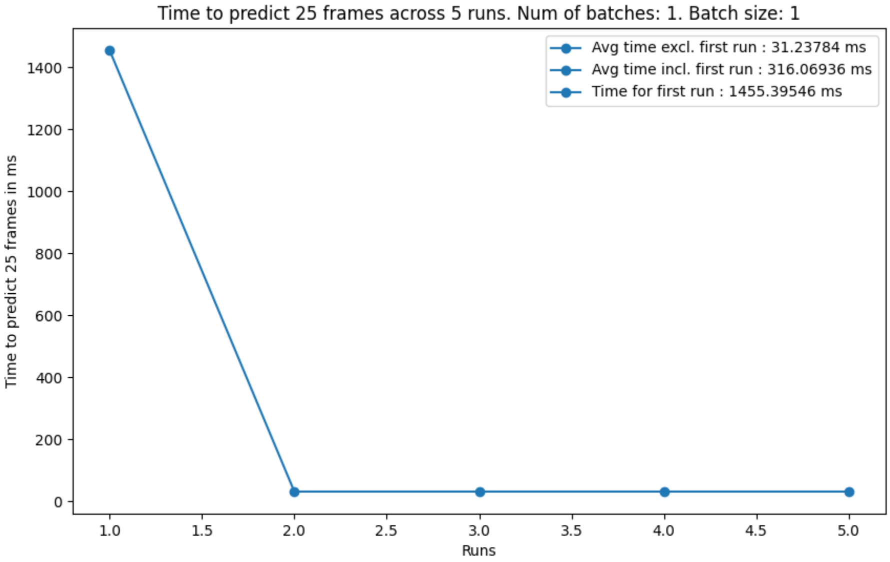

Fig. 4.2: Latency (ms) of the **ST Trans model**. Seed Seq. Len: 10, Target Seq. Len: 25

From Figure 4.1, 4.2, 4.3, we see that amongst the three human motion prediction models evaluated via the UnPOSed framework, our main latency metric ie, "Avg time excl. first run" is least for the STS GCN model at ∼13.7 ms, followed by the PV LSTM model at ∼17.7 ms. The ST Trans model which is a transformer model

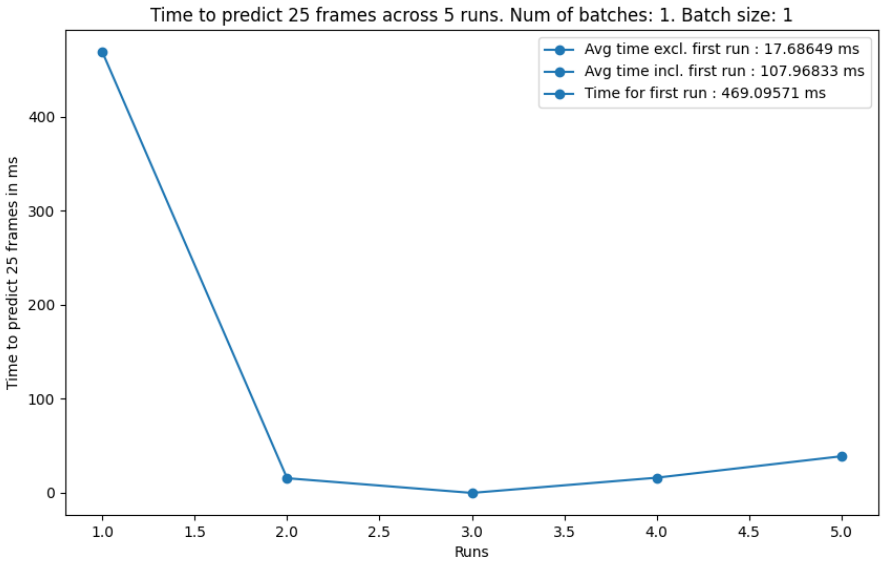

Fig. 4.3: Latency (ms) of the **PV LSTM model**. Seed Seq. Len: 10, Target Seq. Len: 25

is the slowest at ~31.2 ms. This makes the STS GCN model the fastest among the three. An important point to note is that for these non-autoregressive models, this time represents the period after which we have all 25 frames available to us. Hence, depending on the magnitude of the Kinect latency that we're trying to combat, it may even be feasible to utilize the ST Trans model since we could make use of more than 1 frame, say up to 10 frames after one inference run. The Zero Velocity model is just a simplistic baseline model, hence its latency is not measured and reported. We also see that the user would need to wait approximately 1.3s for the first time when utilizing the inference results from the STS GCN model. The PV LSTM model finished its first inference run the fastest, completing it in about 469 ms.

4.1.2 Latency of the Spatio-Temporal Autoregressive Transformer Model

Figure 4.4, 4.5, 4.6, show that our main latency metric ie, "Avg. time excl. first run" is least when the Spatio-Temporal model ran on a Windows-CPU at ~33 ms, followed by ~60.7 ms when it ran on an Ubuntu-CPU via WSL 2. It was slowest when it ran on an Ubuntu-GPU via WSL 2 taking ~183.6 ms. The reason the latency is higher on the GPU as compared to the CPU is because of the memory overhead of moving data back and forth between the CPU and GPU and since this is an autoregressive model, this needs to be done with every frame prediction. Furthermore, unlike the

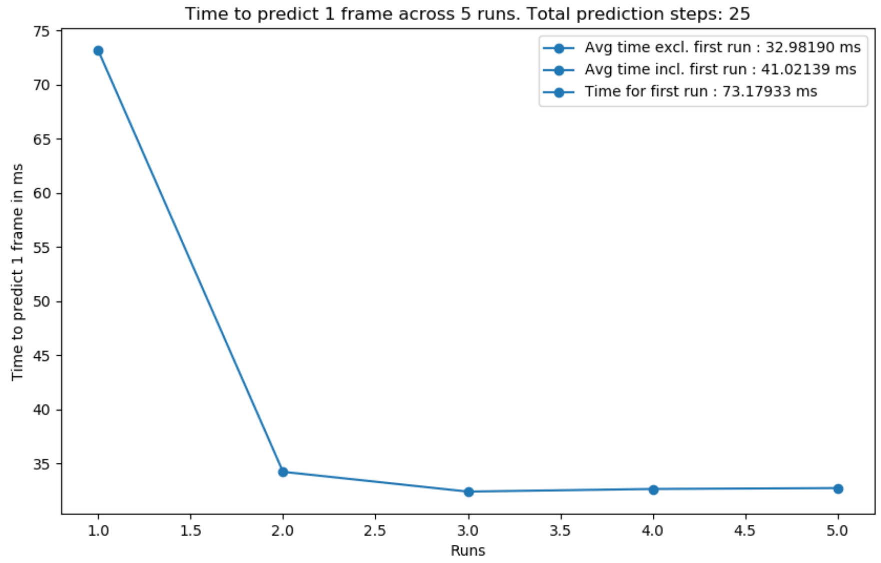

Fig. 4.4: Latency (ms) of the Spatio-Temporal transformer model on **Windows-CPU**. Seed Seq. Len: 120.

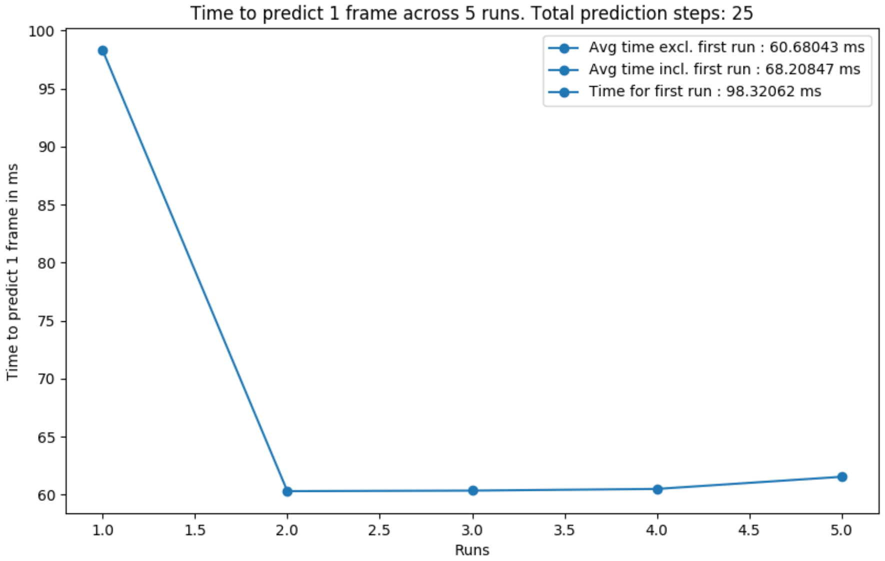

Fig. 4.5: Latency (ms) of the Spatio-Temporal transformer model on **Ubuntu-CPU via WSL2**. Seed Seq. Len: 120.

previously evaluated non-autoregressive models, the "Avg. time excl. first run" here represents the period after which we only have 1 frame available to us.

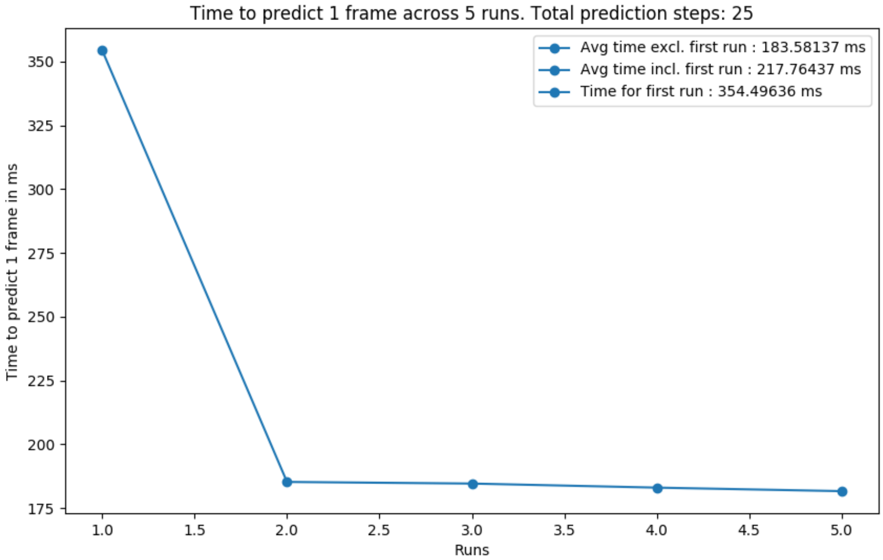

Fig. 4.6: Latency (ms) of the Spatio-Temporal transformer model on **Ubuntu-GPU via WSL2**. Seed Seq. Len: 120.

4.1.3 Latency of the Diffusion-Based Model

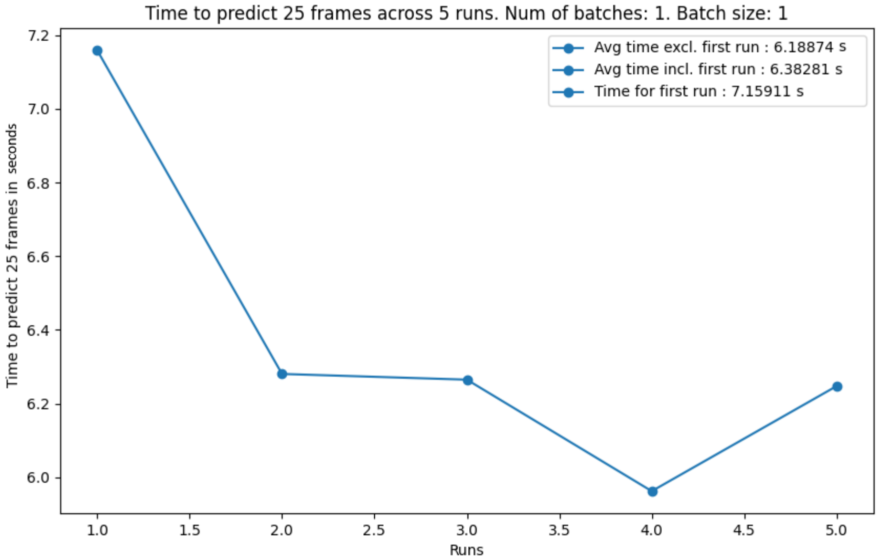

Fig. 4.7: Latency (s) of the **diffusion based model**. Seed Seq. Len: 50, Target Seq. Len: 25.

Figure 4.7, shows that the "Avg. time excl. first run" of the diffusion-based model is ~6.2s while "Time for first run" is ~7.2s. This is reflective of the high computational complexity of the diffusion model and more work is necessary to accelerate the model's runtime performance without impacting its accuracy.

4.1.4 Overview of Latency of all Evaluated Models

In the table below, the abbreviations used are as defined.

- AR: Autoregressive, NAR: Non autoregressive
- ST Trans(*): Spatio Temporal Autoregressive Transformer

Name	**Type**	**Average time excl. first run**	**Time for the first run**
STS GCN	NAR	13.69	1345.64
ST Trans	NAR	31.24	1455.40
PV LSTM	NAR	17.69	469.10
ST Trans(*) - Windows CPU	AR	32.98	73.18
ST Trans(*) - WSL2 Ubuntu CPU	AR	60.68	98.32
ST Trans(*) - WSL2 Ubuntu GPU	AR	183.58	354.50
Diffusion Model	NAR	6188.74	7159.11

Tab. 4.1: Overview of Latency of all Evaluated Models. Latency metrics are in the range of milliseconds. The least value is underlined

Overview of the different aspects of these models are listed in Table 3.1

4.2 Accuracy Results

Figure 4.8, compares the accuracy of all 4 models evaluated via UnPOSed. The accuracy is evaluated per frame up to 25 target frames, measured in terms of displacement error in millimeters. The ST Trans model demonstrates the best performance, followed by the STS GCN model. The PV LSTM model is about as accurate as the Zero Velocity baseline model. The figure highlights frame 10 providing a clearer view of the model's rankings up to this point, which aligns with the frames that might be practically usable in real-time scenarios.

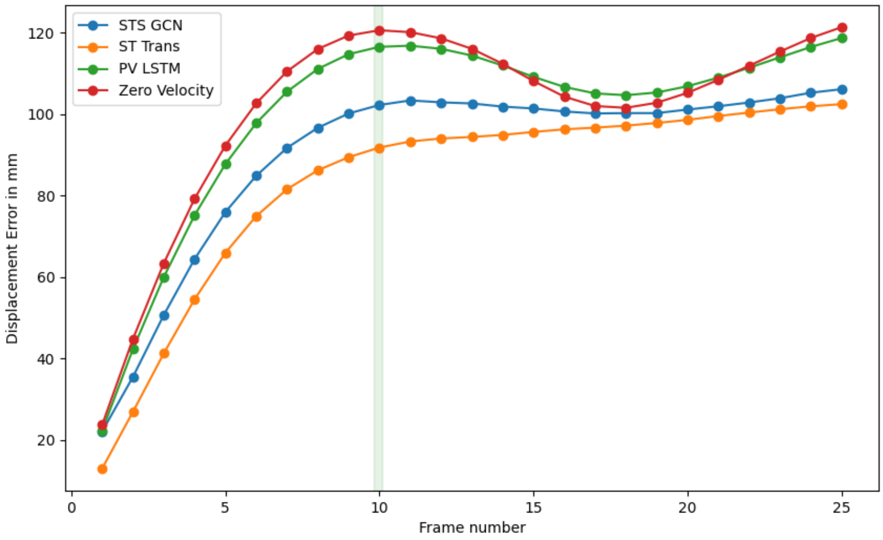

Fig. 4.8: Line chart of per frame error (mm) comparison across all models evaluated via UnPOSed.

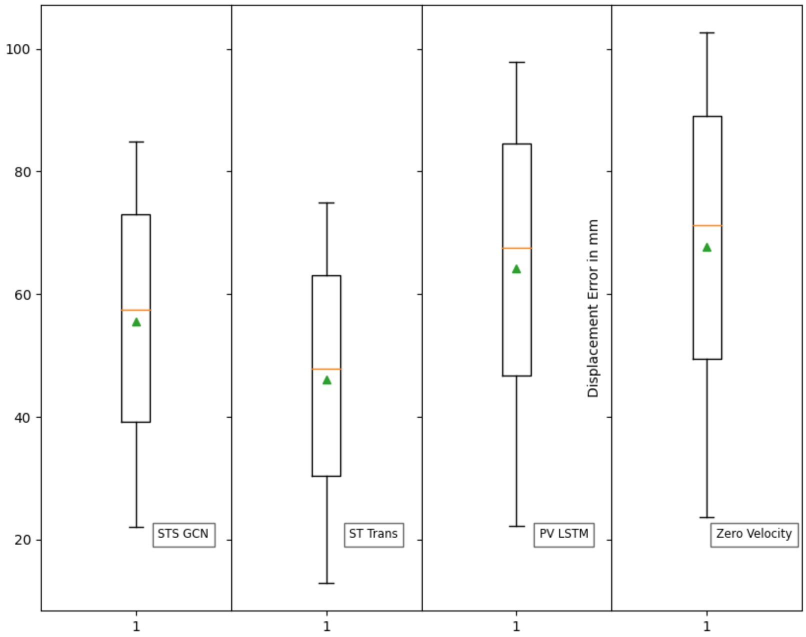

Fig. 4.9: Boxplots of error (mm) comparison across all models evaluated via UnPOSed. First 5 frames

An interesting artifact of the line chart is that we see a dip in the error at around frame 17. This is more clearly visible for the Zero Velocity and the PV LSTM model. It's not entirely clear why this artifact exists but it seems to be due to some specific characteristic of the motion sequences used in the test dataset that these models were evaluated on.

Figure 4.9 and 4.10 display the boxplot of the error metric distribution of the first 5 frames and 10 frames across all 4 models. The ST Trans model outperforms all the other three models over both, 5 frame and 10 frame horizons.

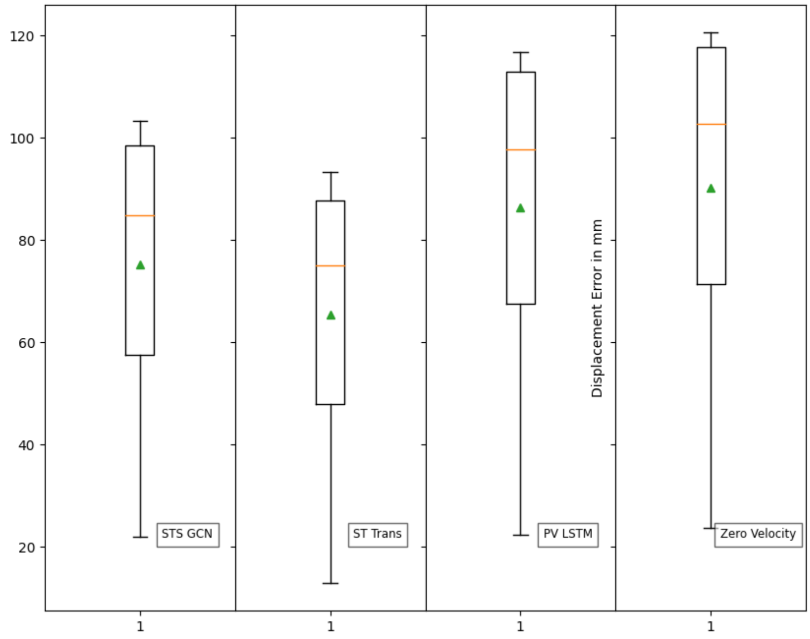

Fig. 4.10: Boxplots of error (mm) comparison across all models evaluated via UnPOSed. First 10 frames

Figure 4.11 compares the accuracy of the Spatio-Temporal autoregressive transformer model with that of the 4 non-autoregressive models evaluated via UnPOSed. The Spatio-Temporal transformer model significantly outperforms all the other models.

However, it's important to note that the Spatio-Temporal transformer model was trained for up to 1000 epochs whereas, as previously noted, the models evaluated via UnPOSed were trained for up to 15 epochs. It's certainly possible that if the UnPOSed models were to be trained for up to 1000 epochs, they may escape their local minima and lead to a better model. Furthermore, the Spatio-Temporal transformer model utilizes a seed sequence of 120 frames. On the other hand, the UnPOSed models use only 10 frames as recommended by the authors of the original research work. Hence, the comparison between these models needs to be considered keeping the above points in mind.

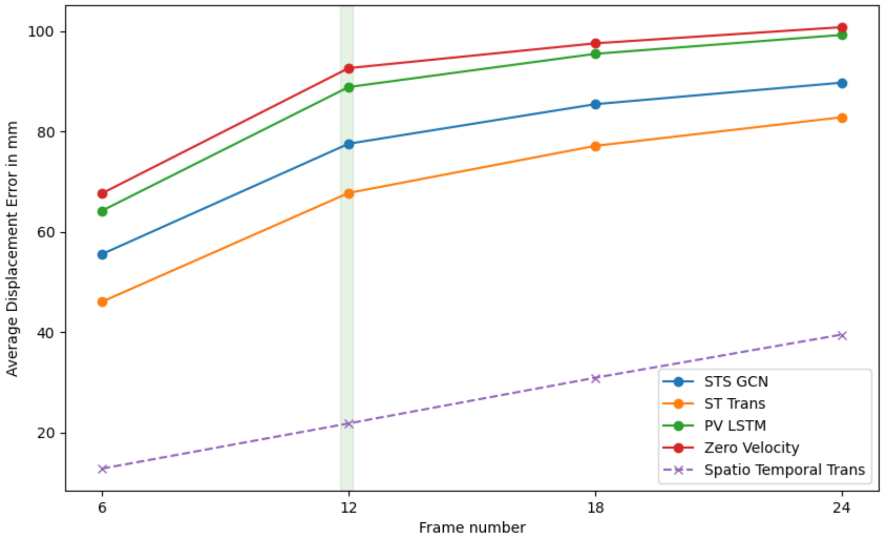

Fig. 4.11: Line chart of average error (mm) comparison across all models.

Conclusion 5

In this research project, the potential to integrate various 3D human motion prediction models with live-tracked human data were investigated, starting with a comprehensive exploration of the relevant literature and related prior work. Both, autoregressive and non-autoregressive models were assessed in terms of their latency and accuracy to determine their use in real-time applications. Moreover, it became clear that non-autoregressive models have been explored more than autoregressive models in this domain.

Specifically, four models including one baseline model were evaluated via UnPOSed, which is an open-source toolbox for 3D human pose prediction. In addition to these, a transformer-based autoregressive model and a diffusion-based model were assessed.

5.1 Selection of a Top Performing Model for Real-Time Applications

Out of the models evaluated, the STS GCN model demonstrated the quickest inference time, taking approximately 13.69 ms. Despite slightly lower accuracy compared to the ST Trans model, the STS GCN model showed promise for real-time applications involving predicting future poses with live-tracked human data.

The Spatio-Temporal autoregressive transformer model had significantly good accuracy but performed poorly in terms of latency, particularly on the GPU. Finally, the diffusion-based model, while impressive due to its ability to fix noise in the data, performed significantly worse in terms of latency, taking over 6 seconds to complete its inference. This is indicative of the high computational cost of diffusion models.

Perhaps, both the Spatio-Temporal autoregressive transformer model and the diffusion-based model could benefit from further optimization efforts.

5.2 Future Work

For the Spatio-Temporal autoregressive transformer model, it could be interesting to train a model with a lower seed sequence length to determine if its latency reduces without having a significant impact on the accuracy. This was currently not taken up as part of this research project due to time and hardware limitations. Additionally, as mentioned previously, the inference of models such as the Spatio-Temporal autoregressive transformer model could be implemented in specialized low latency inference libraries such as WinML or ONNXRuntime to improve their latency, considering the model's already high accuracy.

The evaluated UnPOSed models were trained on a human pose dataset which had a framerate of 25 fps but since the Kinect tracks the human body at 30 fps, it would be interesting to test and assess if this framerate mismatch between the model's training and real-time use scenario affects the model's accuracy in any way. Similar to the previous point, the joints that the models consider to make up a human pose depend on the dataset used to train the model. So if a model is trained on a dataset having, say 3 spine joints but the Kinect tracks only two of those joints, then it would be interesting to look at different ways to fill in the missing data. Perhaps the diffusion-based model could be useful here.

Finally, the diffusion-based model that was evaluated as part of this research project primarily operates in a temporally cascaded fashion ie, it uses two diffusion models, a short-term and a long-term model that operates in sequence to denoise and predict human poses. However, it would be interesting to look at the latency and accuracy of just the short-term diffusion model and determine if the latency improvements outweigh any potential decrease in accuracy.

Bibliography

[Aks+21] Emre Aksan, Manuel Kaufmann, Peng Cao, and Otmar Hilliges. *A Spatio-temporal Transformer for 3D Human Motion Prediction*. 2021 International Conference on 3D Vision (3DV), Dec. 2021, pp. 565–574 (cit. on pp. 6, 8, 12).

[AKH19] Emre Aksan, Manuel Kaufmann, and Otmar Hilliges. *Structured Prediction Helps 3D Human Motion Modelling*. The IEEE International Conference on Computer Vision (ICCV), Oct. 2019 (cit. on p. 5).

[AN04] Kamiar Aminian and Bijan Najafi. *Capturing human motion using body-fixed sensors: outdoor measurement and clinical applications*. Computer Animation and Virtual Worlds, 2004, pp. 79–94 (cit. on p. 3).

[Auv+15] Edouard Auvinet, Franck Multon, Carl-Eric Aubin, Jean Meunier, and Maxime Raison. *Detection of gait cycles in treadmill walking using a Kinect*. Vol. 41. 2. Gait and Posture, 2015, pp. 722–725 (cit. on p. 4).

[AMM] Edouard Auvinet, Franck Multon, and Jean Meunier. *Lower limb movement asymmetry measurement with a depth camera*. Annual International Conference of the IEEE Engineering in Medicine and Biology Society (cit. on p. 4).

[BSA20] Smail Bouhsain, Saeed Saadatnejad, and Alexandre Alahi. *Pedestrian Intention Prediction: A Multi-task Perspective*. European Association for Research in Transportation (hEART), 2020 (cit. on pp. 6, 11).

[BLD21] Wolfgang Büschel, Anke Lehmann, and Raimund Dachselt. *MIRIA: A Mixed Reality Toolkit for the In-Situ Visualization and Analysis of Spatio-Temporal Interaction Data*. Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems, 2021 (cit. on p. 4).

[BKK18] Judith Bütepage, Hedvig Kjellström, and Danica Kragic. *Anticipating many futures: Online human motion prediction and generation for human-robot interaction*. IEEE International Conference on Robotics and Automation (ICRA), 2018, pp. 4563–4570 (cit. on p. 5).

[Cas+14] Fernando Cassola, Leonel Morgado, Fausto de Carvalho, et al. *Online-Gym: A 3D Virtual Gymnasium Using Kinect Interaction*. Vol. 13. Procedia Technology, 2014, pp. 130–138 (cit. on p. 4).

[Che+18] Changan Chen, Yuejiang Liu, Sven Kreiss, and Alexandre Alahi. *Crowd-Robot Interaction: Crowd-Aware Robot Navigation With Attention-Based Deep Reinforcement Learning*. 2019 International Conference on Robotics and Automation (ICRA), 2018, pp. 6015–6022 (cit. on p. 5).

[Dan+21] Lingwei Dang, Yongwei Nie, Chengjiang Long, Qing Zhang, and Guiqing Li. *MSR-GCN: Multi-Scale Residual Graph Convolution Networks for Human Motion Prediction*. Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Oct. 2021, pp. 11467–11476 (cit. on p. 6).

[DA14] Jose Antonio Diego-Mas and Jorge Alcaide-Marzal. *Using Kinect sensor in observational methods for assessing postures at work*. Vol. 45. 4. Applied Ergonomics, 2014, pp. 976–985 (cit. on p. 5).

[Gal+14] Brook Galna, Gillian Barry, Dan Jackson, et al. *Accuracy of the Microsoft Kinect sensor for measuring movement in people with Parkinson's disease*. Vol. 39. 4. Gait and Posture, 2014, pp. 1062–1068 (cit. on p. 4).

[Gon+11] Haifeng Gong, Jack Sim, Maxim Likhachev, and Jianbo Shi. *Multi-hypothesis motion planning for visual object tracking*. 2011 International Conference on Computer Vision, 2011, pp. 619–626 (cit. on p. 5).

[GWK05] Daniel Grest, Jan Woetzel, and Reinhard Koch. *Nonlinear Body Pose Estimation from Depth Images*. Springer Berlin Heidelberg - Pattern Recognition, 2005, pp. 285–292 (cit. on p. 4).

[Hig+21] Yosuke Higuchi, Nanxin Chen, Yuya Fujita, et al. *A Comparative Study on Non-Autoregressive Modelings for Speech-to-Text Generation*. 2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), 2021, pp. 47–54 (cit. on p. 8).

[Ion+14] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. "Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.7 (2014), pp. 1325–1339 (cit. on p. 13).

[KSS16] Koppula, Hema S., and Ashutosh Saxena. *Anticipating Human Activities Using Object Affordances for Reactive Robotic Response*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2016, pp. 14–29 (cit. on p. 5).

[KGP08] Lucas Kovar, Michael Gleicher, and Frédéric Pighin. *Motion graphs*. ACM SIGGRAPH 2008 Classes, 2008 (cit. on p. 5).

[Lam21] Alex Lamb. *A Brief Introduction to Generative Models*. 2021 (cit. on p. 5).

[LS17] Przemyslaw A Lasota and Julie A Shah. *A multiple-predictor approach to human motion prediction*. IEEE International Conference on Robotics and Automation (ICRA), 2017, pp. 2300–2307 (cit. on p. 5).

[Lev+12] Sergey Levine, Jack M. Wang, Alexis Haraux, Zoran Popovic, and Vladlen Koltun. *Continuous character control with low-dimensional embeddings*. ACM Transactions on Graphics (TOG), 2012, pp. 1–10 (cit. on p. 5).

[M+20] Menolotto M, Komaris DS, Tedesco S, O'Flynn B, and Walsh M. *Motion Capture Technology in Industrial Applications: A Systematic Review*. Sensors (Basel - Switzerland), 2020 (cit. on p. 4).

[Ma+22] T. Ma, Y. Nie, C. Long, Q. Zhang, and G. Li. *Progressively Generating Better Initial Guesses Towards Next Stages for High-Quality Human Motion Prediction*. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2022, pp. 6427–6436 (cit. on p. 6).

[Mah+19] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. “AMASS: Archive of Motion Capture as Surface Shapes”. In: *International Conference on Computer Vision*. Oct. 2019, pp. 5442–5451 (cit. on p. 12).

[Man+20] Karttikeya Mangalam, Ehsan Adeli, Kuan-Hui Lee, Adrien Gaidon, and Juan Carlos Niebles. *Disentangling human dynamics for pedestrian locomotion forecasting with noisy supervision*. IEEE/CVF Winter Conference on Applications of Computer Vision, 2020, pp. 2784–2793 (cit. on p. 5).

[Mao+19] Wei Mao, Miaomiao Liu, Mathieu Salzmann, and Hongdong Li. *Learning Trajectory Dependencies for Human Motion Prediction*. Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Oct. 2019 (cit. on p. 6).

[MBR17] Julieta Martinez, Michael J. Black, and Javier Romero. *On human motion prediction using recurrent neural networks*. Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2017, July 2017, pp. 4674–4683 (cit. on p. 5).

[MVO21] Angel Martínez-González, Michael Villamizar, and Jean-Marc Odobez. *Pose Transformers (POTR): Human Motion Prediction With Non-Autoregressive Transformers*. Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops, Oct. 2021, pp. 2276–2284 (cit. on pp. 6, 8).

[MGB07] Brice Michoud, Erwan Guillou, and Saïda Bouakaz. *Real-Time and Markerless 3D Human Motion Capture Using Multiple Views*. Workshop on Human Motion, 2007 (cit. on p. 4).

[PGA18] Dario Pavllo, David Grangier, and Michael Auli. *QuaterNet: A Quaternion-based Recurrent Model for Human Motion*. May 2018 (cit. on p. 5).

[Ped+14] Fabrizio Pedersoli, Sergio Benini, Nicola Adami, and Riccardo Leonardi. *XKin: an open source framework for hand pose and gesture recognition using kinect*. Vol. 30. The Visual Computer, Oct. 2014 (cit. on p. 4).

[SFS21] Armin Saadat, Nima Fathi, and Saeed Saadatnejad. *Towards Human Pose Prediction using the Encoder-Decoder LSTM*. ICCVW 2021, 2021 (cit. on p. 6).

[SJA22] Saeed Saadatnejad, Yi Zhou Ju, and Alexandre Alahi. *Pedestrian 3D Bounding Box Prediction*. European Association for Research in Transportation (hEART), 2022 (cit. on pp. 6, 11).

[Saa+23a] Saeed Saadatnejad, Mehrshad Mirmohammadi, Matin Daghyani, et al. *Toward Reliable Human Pose Forecasting with Uncertainty*. 2023 (cit. on pp. 6, 10, 11).

[Saa+23b] Saeed Saadatnejad, Ali Rasekh, Mohammadreza Mofayezi, et al. *A generic diffusion-based approach for 3D human pose prediction in the wild*. International Conference on Robotics and Automation (ICRA), 2023 (cit. on pp. 6, 13).

[Sof+21] Theodoros Sofianos, Alessio Sampieri, Luca Franco, and Fabio Galasso. *Space-Time-Separable Graph Convolutional Network for Pose Forecasting*. Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021 (cit. on pp. 6, 11).

[Tor+19] Chris Torkar, Saeed Yahyanejad, Horst Pichler, Michael W. Hofbaur, and Bernhard Rinner. *RNN-based Human Pose Prediction for Human-Robot Interaction*. Proceedings of the Joint ARW & OAGM Workshop, May 2019, pp. 76–80 (cit. on p. 5).

[UFF06] Raquel Urtasun, David J. Fleet, and Pascal V. Fua. *3D People Tracking with Gaussian Process Dynamical Models*. 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), 2006, pp. 238–245 (cit. on p. 5).

[Wag+18] Fabien B Wagner, Jean-Baptiste Mignardot, Camille G Le Goff-Mignardot, et al. *Targeted neurotechnology restores walking in humans with spinal cord injury*. Nature - 563(7729), 2018, pp. 65–71 (cit. on p. 5).

[Wan+13] Yanbin Wang, Rohit Dubey, Nadia Magnenat-Thalmann, and Daniel Thalmann. *An immersive multi-agent system for interactive applications*. Vol. 29. The Visual Computer, 2013 (cit. on p. 4).

[Wen+16] Nikolaus Wenger, Eduardo Martin Moraud, Jerome Gandar, et al. *Spatiotemporal neuromodulation therapies engaging muscle synergies improve motor control after spinal cord injury*. Nature medicine - 22(2), 2016, pp. 138–145 (cit. on p. 5).

[YH09] Katsu Yamane and Jessica K. Hodgins. *Simultaneous tracking and balancing of humanoid robots for imitating human motion capture data*. 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009, pp. 2510–2517 (cit. on p. 3).

[YDY03] Hiromasa Yoshimoto, Naoto Date, and Satoshi Yonemoto. *Vision-based real-time motion capture system using multiple cameras*. Proceedings of IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, MFI2003, 2003, pp. 247–251 (cit. on p. 4).

[ZL16] Xudong Zhu and Kin Fun Li. *Real-Time Motion Capture: An Overview*. 2016 10th International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS), 2016, pp. 522–525 (cit. on pp. 3, 4).

Webpages

[@] URL: https://www.tensorflow.org/install/source_windows#gpu (cit. on p. 12).

[@] URL: https://github.com/NVIDIA/tensorflow/issues/58 (cit. on p. 13).

[@] *Accelerating TensorFlow on NVIDIA A100 GPUs*. URL: https://developer.nvidia.com/blog/accelerating-tensorflow-on-a100-gpus/ (cit. on p. 12).

[@AA] Abdenour Amamra and Nabil Aouf. *Real-Time Robust Tracking of Moving Robots with Multiple RGBD Consumer Cameras*. URL: https://arxiv.org/ftp/arxiv/papers/2110/2110.15815.pdf (cit. on p. 7).

[@] *CUDA Compatibility*. URL: https://docs.nvidia.com/deploy/cuda-compatibility/ (cit. on p. 12).

[@] *How To Install TensorFlow 1.15 for NVIDIA RTX30 GPUs (without docker or CUDA install)*. URL: https://www.pugetsystems.com/labs/hpc/how-to-install-tensorflow-1-15-for-nvidia-rtx30-gpus-without-docker-or-cuda-install-2005/ (cit. on p. 12).

[@Mah+] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. *AMASS: Archive of Motion Capture As Surface Shapes*. URL: https://amass.is.tue.mpg.de/ (cit. on p. 12).

[@Mic22a] Microsoft. *Azure Kinect body tracking joints*. 2022. URL: https://learn.microsoft.com/en-us/azure/kinect-dk/body-joints (visited on Sept. 2, 2024) (cit. on p. 6).

[@Mic22b] Microsoft. *Azure Kinect DK hardware specifications*. 2022. URL: https://learn.microsoft.com/en-us/azure/kinect-dk/hardware-specification (visited on Sept. 2, 2024) (cit. on p. 4).

[@] *Papers With Code - Human Pose Forecasting*. URL: https://paperswithcode.com/task/human-pose-forecasting (cit. on p. 9).

[@Rap14] Ilana Rapp. *Motion Capture Actors: Body Movement Tells the Story*. 2014. URL: https://web.archive.org/web/20140703113656/http://www.nycastings.com/dmxreadyv2/blogmanager/v3_blogmanager.asp?post=motioncaptureactors (visited on July 3, 2014) (cit. on p. 3).

[@] *Your GPU Compute Capability*. URL: https://developer.nvidia.com/cuda-gpus (cit. on p. 12).

List of Figures

List of Tables

Colophon

This thesis was typeset with $\LaTeX\,2_\varepsilon$. It uses the *Clean Thesis* style developed by Ricardo Langner. The design of the *Clean Thesis* style is inspired by user guide documents from Apple Inc.

Download the *Clean Thesis* style at `http://cleanthesis.der-ric.de/`.

Declaration

I hereby certify that I have authored this document entitled "Short-term Prediction of User Motion from Live Tracking Data" independently and without undue assistance from third parties. No other than the resources and references indicated in this document have been used. There were no additional persons involved in the intellectual preparation of the present document other than mentioned in the acknowledgment section. I am aware that violations of this declaration may lead to subsequent withdrawal of the research project.

Dresden, April 14, 2024

Divyendu Dutta

Digitally signed by Divyendu Dutta
Date: 2024.04.14 13:48:25 +02'00'

Divyendu Dutta