SIMU-Cap: A Multi-Stage Framework for Sparse IMUs-Based Full-Body Motion Capture

Xiaohai Yu
Beijing University of Posts and Telecommunications
yuxiaohai@bupt.edu.cn

Anlong Ming*
Beijing University of Posts and Telecommunications
mal@bupt.edu.cn

Yinzhe Wang
Beijing University of Posts and Telecommunications
wangyz0602@bupt.edu.cn

Rongkang Zhang
Beijing University of Posts and Telecommunications
zrkkang@bupt.edu.cn

Weihong Yao
Shanghai Vision Era Co., Ltd
yaoweihong@xvgate.com

Huadong Ma
Beijing University of Posts and Telecommunications
mhd@bupt.edu.cn

ABSTRACT

This paper introduces SIMU-Cap, a human motion capture method utilizing sparse inertial measurement unit (IMU) data to address challenges related to motion ambiguity and under-constrained sensor placement. The approach includes a learnable RNN initialization for capturing temporal transitions, a region-specific motion estimation module that combines a Pseudo Velocity Regression Network (VRN), a Velocity Estimation Network (VEN), and a Global Feature Extractor (GFE) to improve pose estimation accuracy. A physics-based optimization module ensures physical plausibility, and a dual PD controller refines joint accelerations. A Trajectory Scale Optimization Network (TSN) corrects trajectory scale discrepancies caused by estimation errors. Evaluations on the Minions dataset demonstrate significant improvements in Mean Per Joint Position Error (MPJPE), Procrustes Aligned MPJPE (PA-MPJPE), and Jitter, enhancing trajectory accuracy and smoothness. SIMU-Cap achieved the best results in this competition, securing the championship on the Minions dataset.

ACM Reference Format:
Xiaohai Yu, Anlong Ming, Yinzhe Wang, Rongkang Zhang, Weihong Yao, and Huadong Ma. 2024. SIMU-Cap: A Multi-Stage Framework for Sparse IMUs-Based Full-Body Motion Capture. In *Proceedings of The 5th International Workshop on Human-centric Multimedia Analysis (MM'24)Proceedings of the 32nd ACM International Conference on Multimedia (MM'24), October 28-November 1, 2024, Melbourne, Australia.* ACM, New York, NY, USA, 5 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Human motion capture is crucial for various applications, including sports analysis, healthcare, rehabilitation, and virtual reality. Traditional optical motion capture systems provide accurate results but suffer from high costs, cumbersome setups, and intrusive requirements involving multiple cameras and markers. In contrast, inertial measurement units (IMUs) offer a more portable and cost-effective alternative. However, when using sparse IMUs, tracking full-body motion becomes challenging due to limited sensor placement, leading to postural ambiguities and inaccuracies.

Existing methods struggle to resolve motion ambiguities, such as distinguishing between similar poses (e.g., standing and sitting) or ensuring physical plausibility in predicted motion. Additionally, errors in estimating joint velocities and trajectories frequently result in issues like foot-ground penetration or mismatches in motion scale.

In this work, we address these challenges with SIMU-Cap, a human motion capture method using sparse IMU data. SIMU-Cap incorporates several key components: a learnable recurrent neural network (RNN) initialization to enhance temporal consistency and reduce motion ambiguity, a Pseudo Velocity Regression Network (VRN) to improve velocity estimation, and a region-specific motion estimation module to capture unique motion dynamics across the upper limbs, torso, and lower limbs.

To further improve physical plausibility, SIMU-Cap introduces a physics-based optimization module to enforce constraints on body dynamics, minimizing artifacts such as body jitter and foot-ground penetration. Despite these enhancements, discrepancies between the scale of the predicted and actual trajectories can still occur. To address this, SIMU-Cap integrates a Trajectory Scale Optimization Network (TSN), which adjusts the predicted trajectory to align with the true scale by leveraging both temporal and motion data.

Our method, tested on the Minions dataset, demonstrates significant performance improvements over baseline methods, achieving better trajectory accuracy, smoothness, and reduced errors in terms of Mean Per Joint Position Error (**MPJPE**), Procrustes Aligned Mean Per Joint Position Error (**PA-MPJPE**), and **Jitter**.

2 METHOD

2.1 Overview

We propose SIMU-Cap, a human motion capture method using sparse inertial sensor data to address motion ambiguity and the under-constrained nature of sparsely distributed IMUs. Our multi-stage approach includes a learnable RNN initialization strategy, a

*Corresponding author

three-region collaborative motion estimation module, and physics-based optimization.

Full-body motion capture with sparse IMUs is under-constrained, particularly for pose ambiguities like distinguishing between standing and sitting still. SIMU-Cap addresses this by using a learnable RNN initialization (Section 2.2) to capture state transitions from historical frames and reduce ambiguity. Additionally, we introduce a Pseudo Velocity Regression Network (VRN) to estimate joint velocities and enhance accuracy.

The human body is divided into three regions (upper limbs, torso, lower limbs), each with a specialized sub-model (Section 2.3) to capture unique motion patterns and minimize weak correlations between joints.

To ensure physical adherence and eliminate artifacts (e.g., foot-ground penetration), SIMU-Cap incorporates physics-based and trajectory scaling optimization modules (Section 2.4, 2.5). An overview of the pipeline is depicted in Figure 1.

2.2 Learnable Initialization for RNN

Tracking full-body motion using sparse inertial sensors is an inherently under-constrained task. Due to the sparse distribution of IMUs, some states (e.g., standing still vs. sitting still) show minimal differences, making them difficult to distinguish. Temporal information must be fully utilized to capture and retain state transitions from historical frames. Existing studies [3, 8] employ bi-directional RNNs, but their fixed-length time windows limit the ability to capture long-term transitions, leading to poor handling of ambiguous poses.

We utilize a unidirectional RNN to preserve complete historical context and more effectively capture key transitions. To address zero-initialized hidden states, we design a learnable RNN initialization. Specifically, an independent fully connected network (FCN) predicts the initial RNN state based on input body posture. The FCN and RNN are jointly optimized during training. During inference, the initial posture is assumed to be calibrated. The FCN is only used for the initial RNN step, maintaining compatibility with optimization libraries.

2.3 Three-Region Collaborative Human Motion Estimation

2.3.1 Pseudo Velocity Regression. Raw acceleration data is highly sensitive to instantaneous dynamics, making it insufficient for continuous joint motion estimation. Studies [2, 6] demonstrate that velocity is more robust for representing joint kinematics and mitigating motion ambiguity. Thus, we employ joint velocity to enhance dynamic information extraction.

Velocity is derived by integrating acceleration data, providing greater robustness compared to raw acceleration. Following [9], we design a Velocity Regression Network (VRN) to predict joint velocities, enhancing the model's ability to capture continuous body dynamics.

We use raw IMU data $X = [X_R, X_{LL}, X_{RL}, X_H, X_{LA}, X_{RA}]$ as input, and obtain velocity $\hat{V} = [\hat{V}_R, \hat{V}_{LL}, \hat{V}_{RL}, \hat{V}_H, \hat{V}_{LA}, \hat{V}_{RA}]$ through the VRN, denoted as $S_{\text{VRN}}(\cdot)$. The input includes raw IMU data X, initial velocity V_0, and predicted pseudo velocity $\hat{V}$:

$$\hat{\boldsymbol{V}} = S_{\text{VRN}}(\boldsymbol{X}, \boldsymbol{V_0}) \tag{1}$$

The VRN module consists of an MLP and a two-layer LSTM. The MLP takes initial velocity as input and transforms it for the LSTM's hidden and memory states. The true velocity V is computed from the true pose θ using forward kinematics (FK).

The pseudo velocity loss L_{vel} is defined as:

$$\begin{aligned} L_{\text{vel}} &= |\boldsymbol{V} - \hat{\boldsymbol{V}}|_2 \\ &= \left| \frac{FK(\boldsymbol{\theta_t}) - FK(\boldsymbol{\theta_{t-1}})}{\Delta t} - S_{\text{VRN}}(\boldsymbol{X}, \boldsymbol{V_0}) \right|_2 \end{aligned} \tag{2}$$

2.3.2 Velocity Estimation Network (VEN). The Velocity Estimation Network (VEN) is used to predict the joint velocities based on the input IMU data, similar to PRN. VEN is composed of a fully connected network (FCN) followed by a recurrent neural network (RNN). The input to VEN is the same as that of PRN, including the IMU data $\boldsymbol{X}$ and initial pose $\boldsymbol{\theta_0}$. The VEN outputs the estimated joint velocities $\hat{v}$ for each joint:

$$\hat{\boldsymbol{v}} = S_{\text{VEN}}(\boldsymbol{X}, \boldsymbol{\theta_0}) \tag{3}$$

The loss function for VEN is the L2 loss, defined as:

$$L_{\text{VEN}} = |\boldsymbol{v} - \hat{\boldsymbol{v}}|_2 \tag{4}$$

Here, $\boldsymbol{v}$ is the true joint velocity, and $\hat{\boldsymbol{v}}$ is the predicted joint velocity from VEN.

2.3.3 Contact Footstep Network (CFN). The Contact Footstep Network (CFN) is designed to predict the foot-ground contact probability c for each footstep. The CFN module consists of an MLP and an RNN. The input to CFN is the IMU data related to the lower limbs, denoted as $\boldsymbol{X_{LL}}$. CFN outputs the contact probability $\hat{c}$:

$$\hat{\boldsymbol{c}} = S_{\text{CFN}}(\boldsymbol{X_{LL}}) \tag{5}$$

The loss function for CFN is the Binary Cross Entropy loss (BCELoss), defined as:

$$L_{\text{CFN}} = -[c \log(\hat{c}) + (1 - c) \log(1 - \hat{c})] \tag{6}$$

where c is the true contact label, obtained by thresholding based on joint velocity and height. Specifically, a joint is labeled as in contact if its velocity is below a certain threshold and its height is within a predefined range. $\hat{c}$ is the predicted contact probability.

2.3.4 Human Motion Estimation. Based on the VRN module's output, we proceed with motion estimation using the Pose Regression Network (PRN), denoted as $S_{\text{PRN}}(\cdot)$, which estimates joint rotations $\hat{\theta}$ from IMU data $\boldsymbol{X}$ and estimated velocity $\hat{V}$:

$$\hat{\boldsymbol{\theta}} = S_{\text{PRN}}(\boldsymbol{X}, \boldsymbol{\theta_0}), \tag{7}$$

where $\boldsymbol{\theta_0}$ is the initial pose, and $S_{\text{PRN}}(\cdot)$ uses a two-layer LSTM. Pose loss L_{pose} is defined as:

$$\begin{aligned} L_{\text{pose}} &= |\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}|_2 \\ &= |\boldsymbol{\theta} - S_{\text{PRN}}(\boldsymbol{X}^{(2)}, \boldsymbol{\theta_0})|_2 \end{aligned} \tag{8}$$

Considering the distinct motion patterns of the upper and lower limbs, and the importance of leveraging spatial information for

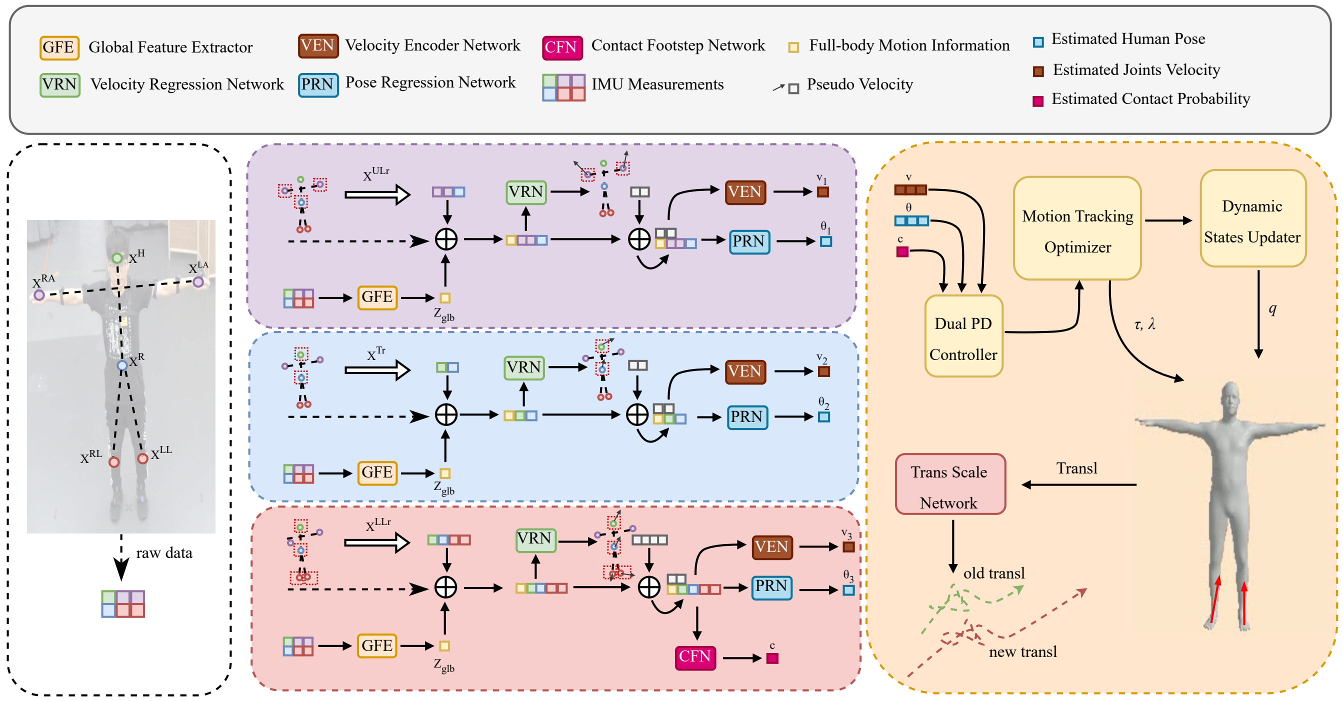
GFE Global Feature Extractor
VRN Velocity Regression Network
VEN Velocity Encoder Network
PRN Pose Regression Network
CFN Contact Footstep Network
IMU Measurements
Full-body Motion Information
Pseudo Velocity
Estimated Human Pose
Estimated Joints Velocity
Estimated Contact Probability
raw data
Motion Tracking Optimizer
Dynamic States Updater
Dual PD Controller
Trans Scale Network
Transl
old transl
new transl

Figure 1: Overview of our human motion capture method. The process begins with sparse IMU data from key body regions. Global features are extracted by the Global Feature Extractor (GFE), followed by pseudo-velocity estimation using the Velocity Regression Network (VRN). The Pose Regression Network (PRN) then predicts joint rotations. The Contact Footstep Network (CFN) identifies foot-ground contacts, and the Velocity Estimation Network (VEN) refines joint velocities. Motion capture is further optimized using a dual proportional-derivative (PD) controller to enforce physical constraints on joint rotations and positions. Finally, the Trajectory Scale Network (TSN) adjusts scale discrepancies, ensuring smooth and realistic motion capture.

robust tracking, we introduce local region modeling. Previous methods [7] used all six IMU measurements as input without accounting for the spatial relationships within the body, which could result in motion ambiguity. Inspired by [4], we divide the body into three regions: upper limbs (ULr), torso (Tr), and lower limbs (LLr), as shown in Figure 1. The IMU measurements and pose estimates are grouped accordingly. For input generation, sensors are partitioned into three sets: $X_{ULr} = [X_R, X_{LA}, X_{RA}]$, $X_{Tr} = [X_R, X_H]$, and $X_{LLr} = [X_R, X_{LL}, X_{RL}, X_H]$.

We design three sub-models to estimate local pseudo-velocities $\hat{V}^l$ and joint rotations $\hat{\theta}^l$ for each region. The full-body joint rotations are then obtained from the combined outputs of these sub-models.

2.4 Physics-Based Motion Optimization

The human motion estimation stage may still produce artifacts such as body jitter and foot-ground penetration due to the lack of real-world kinematic constraints. To address these issues, we introduce a dynamics module that imposes explicit physical constraints, ensuring realistic joint torques and ground reaction forces (GRFs) consistent with the reference motion estimated from the previous stages. [5]

This module takes as input the motion state θ, velocity v, and contact points c, as estimated in the previous sections, and applies a set of physical constraints to optimize the resulting motion. The optimization process includes solving for joint accelerations and forces that are physically plausible, followed by a state update.

2.4.1 Basic Physical Model. We represent the human body as a torque-controlled floating base system, where the mass distribution is consistent with [5]. The body's configuration is defined by its joint positions $\boldsymbol{\theta}$, velocities $\dot{\boldsymbol{\theta}}$, and accelerations $\ddot{\boldsymbol{\theta}}$. The model follows the equation of motion [1]:

$$\boldsymbol{\tau} + J_c(\boldsymbol{q})^T \boldsymbol{\lambda} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \tag{9}$$

where $\boldsymbol{M}$ is the inertia matrix, J_c is the contact Jacobian matrix, $\boldsymbol{\lambda}$ represents external contact forces (e.g., ground reaction forces), and $\boldsymbol{h}$ represents the non-linear terms such as gravity, Coriolis, and centrifugal forces.

Similar to the approach in [7], to compute realistic joint accelerations, we apply a dual proportional-derivative (PD) controller to the joint rotations and positions. The joint angular acceleration $\ddot{\boldsymbol{\theta}}_{\text{des}}$ is computed as:

$$\ddot{\boldsymbol{\theta}}_{\text{des}} = k_{p_\theta}(\boldsymbol{E}(\boldsymbol{\varphi}) - \boldsymbol{\theta}) - k_{d_\theta}\dot{\boldsymbol{\theta}} \tag{10}$$

Similarly, the joint linear acceleration $\ddot{\boldsymbol{r}}_{\text{des}}$ is computed using the estimated reference positions and velocities:

$$\ddot{\boldsymbol{r}}_{\text{des}} = k_{p_r}(\boldsymbol{r}_{\text{ref}} - \boldsymbol{r}) - k_{d_r}\dot{\boldsymbol{r}} \tag{11}$$

2.4.2 Motion Capture Optimizer. The motion capture optimizer solves a quadratic programming problem to estimate joint accelerations $\ddot{\boldsymbol{q}}$, joint torques $\boldsymbol{\tau}$, and ground reaction forces $\boldsymbol{\lambda}$, ensuring that the character's motion adheres to physical laws. The optimization problem is formulated as follows:

$$\min_{\ddot{\boldsymbol{q}},\boldsymbol{\lambda},\boldsymbol{\tau}} \mathcal{E}_{\text{PD}} + \mathcal{E}_{\text{reg}} \tag{12}$$

subject to the following constraints:

$$\begin{aligned} \boldsymbol{\tau} + \boldsymbol{J}_{\text{c}}^{T}\boldsymbol{\lambda} = \boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{h} \quad & \text{(Equation of motion)} \\ \boldsymbol{\lambda} \in \mathcal{F} \quad & \text{(Friction cone constraint)} \\ \dot{\boldsymbol{r}}_{\text{j}}(\ddot{\boldsymbol{q}}) \in C \quad & \text{(No sliding constraint)} \end{aligned} \tag{13}$$

2.4.3 Motion Update. After solving for the accelerations and forces, we update the state of the character using a finite difference method. The positions and velocities are updated as follows:

$$\boldsymbol{q}^{(t+1)} = \boldsymbol{q}^{(t)} + \dot{\boldsymbol{q}}^{(t)}\Delta t, \tag{14}$$

$$\dot{\boldsymbol{q}}^{(t+1)} = \dot{\boldsymbol{q}}^{(t)} + \ddot{\boldsymbol{q}}^{(t)}\Delta t \tag{15}$$

where $\boldsymbol{q}^{(t)}$ and $\dot{\boldsymbol{q}}^{(t)}$ are the current positions and velocities, $\ddot{\boldsymbol{q}}^{(t)}$ is the computed acceleration, and Δt is the simulation time step.

2.5 Trajectory Scale Optimization

After optimizing the physical plausibility of the model's output for human motion, we obtain relatively accurate pose information and physically consistent trajectory data. However, due to inevitable discrepancies between the estimated human pose, joint velocities, and contact probabilities with their true values, these errors may cause the physically optimized trajectories to either overestimate or underestimate certain depth translations. This manifests as scale discrepancies between the predicted trajectory and the ground truth. To address this issue, we propose a Trajectory Scale Network, denoted as TSN.

TSN is designed to leverage both inter-frame temporal information and scale-misaligned trajectory data to generate accurate, scale-consistent trajectories. By incorporating physically consistent pose and trajectory information along with temporal modeling, TSN produces trajectories that exhibit physical plausibility, smoothness, and uniform scale alignment.

The TSN module is built upon a Transformer architecture, consisting of transformation layers, positional encoding layers, multiple Transformer encoder layers, and output prediction layers. The acceleration data is mapped from its original input dimension to the model's internal feature space through a linear layer. This transformation enables the subsequent Transformer layers to process data within a unified feature space.

The new trajectory prediction can be expressed as:

$$\boldsymbol{T}_{\text{pred}} = \text{TSN}(\boldsymbol{T}_{\text{old}}) \tag{16}$$

where $\boldsymbol{T}_{\text{old}}$ represents the old, scale-misaligned trajectory, and $\boldsymbol{T}_{\text{pred}}$ is the new, scale-consistent trajectory generated by TSN.

The TSN loss function is based on the L2 Loss, ensuring the alignment between the predicted trajectory and the ground truth:

$$L_{\text{TSN}} = \left|\boldsymbol{T}_{\text{pred}} - \boldsymbol{T}_{\text{gt}}\right|_2 \tag{17}$$

where $\boldsymbol{T}_{\text{pred}}$ denotes the predicted trajectory after scale optimization, and $\boldsymbol{T}_{\text{gt}}$ represents the ground truth trajectory. Through this optimization process, TSN refines the trajectory to achieve both physical realism and scale consistency, making it robust for various motion prediction tasks.

3 EXPERIMENT

In this section, we provide an evaluation of our proposed human motion capture method within the context of the competition. We utilize the Minions dataset, provided by the competition organizers, for both training and testing our models. The experimental setup, implementation details, and an analysis of the results compared to the baseline method are presented.

3.1 Dataset

The Minions dataset, provided by the competition, features a wide variety of human motions captured using sparse Inertial Measurement Units (IMUs) placed on key body regions, including the upper limbs, torso, and lower limbs. It includes activities such as walking, running, sitting, standing, and dynamic movements, making it a comprehensive benchmark for evaluating motion capture performance. Each motion sequence is accompanied by synchronized IMU data and ground truth joint positions, allowing for precise assessment of our method's effectiveness.

3.2 Implementation Details

Our model was implemented using the PyTorch framework. The learnable RNN initialization was realized using a fully connected network (FCN) with two hidden layers, each containing 128 neurons and ReLU activation functions. The RNN employed was a two-layer Long Short-Term Memory (LSTM) network with 256 hidden units per layer.

The Velocity Regression Network (VRN) comprised a Multi-Layer Perceptron (MLP) with three hidden layers (256, 128, 64) followed by a two-layer LSTM, designed to capture temporal dependencies in the IMU data. For the Three-Region Collaborative Motion Estimation, the body was divided into upper limbs, torso, and lower limbs, each processed by specialized sub-models with identical architectures to the VRN.

The Pose Regression Network (PRN) utilized a two-layer LSTM with 256 hidden units to predict joint rotations from IMU data and estimated velocities. The Physics-Based Motion Optimization module was implemented following the approach described in our method section, utilizing a dual proportional-derivative (PD) controller for joint rotations and positions. The Trajectory Scale Network (TSN) was built upon a Transformer architecture with 4 encoder layers, 8 attention heads, and a hidden dimension of 512, trained to align predicted trajectories with ground truth data.

We trained all networks using the Adam optimizer with an initial learning rate of 1×10^{-4}, a batch size of 64, and trained for 100 epochs. Early stopping based on validation loss was employed to prevent overfitting, with a patience of 10 epochs. Data augmentation techniques, including random noise addition and temporal scaling, were applied to enhance model robustness.

3.3 Evaluation Metrics

We evaluated our method using the following metrics provided by the competition guidelines:

- **Mean Per Joint Position Error (MPJPE)**: Measures the average Euclidean distance between the predicted and ground truth joint positions.
- **Procrustes Aligned Mean Per Joint Position Error (PA-MPJPE)**: Similar to MPJPE but after a rigid alignment (Procrustes analysis) to account for scale and rotation differences.
- **Jitter**: Quantifies the smoothness of the predicted motion by measuring temporal fluctuations in joint positions.

3.4 Quantitative Results

Table 1 presents the performance of different methods on the competition dataset. Our proposed method demonstrates significant improvements over the baseline and the old method across all metrics.

Method	MPJPE ↓	PA-MPJPE ↓	Jitter ↓
Competition Baseline	692.72	84.06	**0.00**
Old Method	573.04	31.93	0.48
Old Method + TSN	536.30	31.93	0.43
Proposed Method	**298.21**	**27.54**	0.23

Table 1: Performance comparison between different methods based on Mean Per Joint Position Error (MPJPE), Procrustes Aligned Mean Per Joint Position Error (PA-MPJPE), and Jitter. Lower values indicate better performance.

As shown in Table 1, our proposed method achieved substantial improvements:

- **MPJPE**: Reduced from 573.04 (Old Method) to 298.21, indicating a significant enhancement in joint position accuracy.
- **PA-MPJPE**: Decreased from 31.93 (Old Method) to 27.54, reflecting better alignment with the ground truth after Procrustes analysis.
- **Jitter**: Improved from 0.48 (Old Method) to 0.23, demonstrating smoother and more stable motion capture.

The integration of the Trajectory Scale Network (TSN) with the Old Method further improved MPJPE and Jitter, reducing them to 536.30 and 0.43, respectively. This highlights the effectiveness of TSN in refining motion trajectories for enhanced accuracy and smoothness.

3.5 Qualitative Results

Figure 2 illustrates the results of our proposed method. The estimated body poses and trajectories exhibit superior accuracy and smoothness, effectively mitigating common artifacts such as foot-ground penetration and body jitter.

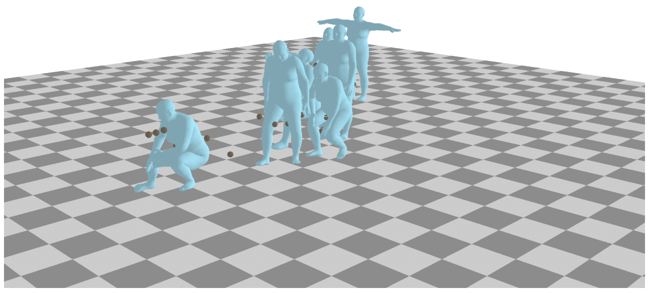
Figure 2: Estimated body poses and trajectories using our proposed method. The results demonstrate accurate and smooth motion capture, effectively reducing artifacts.

3.6 Discussion

The experimental results demonstrate that our proposed method significantly outperforms the baseline, achieving superior performance in the competition. Our approach secured the championship, underscoring its effectiveness in human motion capture using sparse IMU data.

Despite these achievements, our method may face limitations in extremely dynamic motions or when IMU data quality is compromised. Future work could explore integrating additional sensor modalities, such as camera-based inputs, or leveraging more sophisticated optimization techniques to further enhance performance. Additionally, expanding the dataset to include a wider variety of motions and subjects could improve the generalizability of our approach.

REFERENCES

[1] Roy Featherstone. 2014. *Rigid body dynamics algorithms.* Springer.

[2] Sachini Herath, Hang Yan, and Yasutaka Furukawa. 2020. Ronin: Robust neural inertial navigation in the wild: Benchmark, evaluations, & new methods. In *2020 IEEE international conference on robotics and automation (ICRA).* IEEE, 3146–3152.

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

[4] Gun-Hee Lee and Seong-Whan Lee. 2021. Uncertainty-aware human mesh recovery from video by learning part-based 3d dynamics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision.* 12375–12384.

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

[6] Dongseok Yang, Doyeon Kim, and Sung-Hee Lee. 2021. Lobstr: Real-time lower-body pose prediction from sparse upper-body tracking signals. In *Computer Graphics Forum*, Vol. 40. Wiley Online Library, 265–275.

[7] Xinyu Yi, Yuxiao Zhou, Marc Habermann, Soshi Shimada, Vladislav Golyanik, Christian Theobalt, and Feng Xu. 2022. Physical inertial poser (pip): Physics-aware real-time human motion tracking from sparse inertial sensors. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.* 13167–13178.

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

[9] Yu Zhang, Songpengcheng Xia, Lei Chu, Jiarui Yang, Qi Wu, and Ling Pei. 2024. Dynamic Inertial Poser (DynaIP): Part-Based Motion Dynamics Learning for Enhanced Human Pose Estimation with Sparse Inertial Sensors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 1889–1899.