

EventEgo3D++: 3D Human Motion Capture from a Head-Mounted Event Camera

Christen Millerdurai^{1,2}, Hiroyasu Akada¹, Jian Wang¹, Diogo Luvizon¹, Alain Pagani², Didier Stricker², Christian Theobalt¹, Vladislav Golyanik¹

¹Visual Computing and Artificial Intelligence, Max Planck Institute for Informatics, SIC, Stuhlsatzenhausweg E1 4, Saarbrücken, 66123, Saarland, Germany.

²Augmented Vision, German Research Center for Artificial Intelligence (DFKI), Trippstadter Str. 122, Kaiserslautern, 67663, Rhineland-Palatinate, Germany.

Contributing authors: christen.millerdurai@dfki.de; hakada@mpi-inf.mpg.de; jianwang@mpi-inf.mpg.de; dluvizon@mpi-inf.mpg.de; alain.pagani@dfki.de; didier.stricker@dfki.de; theobalt@mpi-inf.mpg.de; golyanik@mpi-inf.mpg.de;

Abstract

Monocular egocentric 3D human motion capture remains a significant challenge, particularly under conditions of low lighting and fast movements, which are common in head-mounted device applications. Existing methods that rely on RGB cameras often fail under these conditions. To address these limitations, we introduce EventEgo3D++, the first approach that leverages a monocular event camera with a fisheye lens for 3D human motion capture. Event cameras excel in high-speed scenarios and varying illumination due to their high temporal resolution, providing reliable cues for accurate 3D human motion capture. EventEgo3D++ leverages the LNES representation of event streams to enable precise 3D reconstructions. We have also developed a mobile head-mounted device (HMD) prototype equipped with an event camera, capturing a comprehensive dataset that includes real event observations from both controlled studio environments and in-the-wild settings, in addition to a synthetic dataset. Additionally, to provide a more holistic dataset, we include allocentric RGB streams that offer different perspectives of the HMD wearer, along with their corresponding SMPL body model. Our experiments demonstrate that EventEgo3D++ achieves superior 3D accuracy and robustness compared to existing solutions, even in challenging conditions. Moreover, our method supports real-time 3D pose updates at a rate of 140Hz. This work is an extension of the EventEgo3D approach (CVPR 2024) and further advances the state of the art in egocentric 3D human motion capture. For more details, visit the project page at <https://eventego3d.mpi-inf.mpg.de>.

Keywords: Event-based vision, 3D human pose estimation, Egocentric vision, VR/AR.

1 Introduction

Head-mounted devices (HMDs) hold significant potential to become the next major platform for mobile and pervasive computing, offering

diverse applications in many fields such as education, driving, personal assistance systems, and gaming. HMDs enhance user flexibility, allowing individuals to move freely and explore their surroundings seamlessly. As a result,

egocentric 3D human pose estimation has emerged as an active research area, with numerous studies focusing on recovering 3D human poses using down-facing fisheye RGB cameras mounted on HMDs (Rhodin et al, 2016; Xu et al, 2019; Zhao et al, 2021; Wang et al, 2022a; Akada et al, 2022, 2024; Wang et al, 2023, 2021; Tome et al, 2020; Liu et al, 2023; Li et al, 2023a; Wang et al, 2024a; Kang et al, 2023; Kang and Lee, 2024).

Although these experimental prototypes have demonstrated high 3D human pose estimation accuracy, their setups have several limitations. Firstly, RGB cameras are prone to over- or under-exposure and motion blur, especially in low-light conditions and during rapid movements, which are common in HMD applications. Secondly, these cameras consume relatively high power, making them less efficient for mobile devices. Furthermore, recording image frames synchronously demands high data processing throughput, which can be a significant burden for real-time applications. These limitations are particularly problematic for HMDs, where efficient and reliable performance is crucial.

In light of these challenges, our work is motivated by the observation that many of the challenges associated with RGB-based HMDs can be mitigated through the use of *event cameras*. Event cameras record streams of asynchronous per-pixel brightness changes at high temporal resolution (on the order of microseconds, μs), support an increased dynamic range and consume less power (on the order of tens of mW) than RGB cameras, which consume Watts (Gallego et al, 2020). To leverage these benefits, we build a lightweight HMD that integrates an event camera with a fisheye lens. This setup allows for the precise capture of fast and dynamic movements with much lower power consumption, making itself well-suited for real-time applications. Building on these advantages, we develop a lightweight HMD equipped with an event camera and a fisheye lens, enabling precise capture of fast and dynamic movements at notably lower power consumption. Further details on event camera efficiency can be found in App. A.

However, existing RGB-based pose estimation techniques, particularly learning-based methods, cannot be straightforwardly repurposed for event streams. Also, these methods are typically slow and not ideal for real-time applications. Dedicated

approaches are required to fully leverage the advantages of event cameras, as demonstrated by recent progress in event-based 3D reconstruction across various scenarios (Xu et al, 2020; Rudnev et al, 2021; Zou et al, 2021; Jiang et al, 2024a; Millerdurai et al, 2024b). Furthermore, an egocentric HMD setup utilising an event camera introduces two additional challenges. Firstly, the *moving event camera* generates a significant amount of background events, making it difficult to isolate the user-specific events required for accurate pose estimation. Secondly, event cameras fail to generate events in situations where the HMD user remains stationary and no motion is detected.

Our previous work, EventEgo3D (Millerdurai et al, 2024a) addressed these challenges by introducing a lightweight neural network that processes the egocentric event streams to estimate 3D human pose in real time. By incorporating confidence scores, the network assigns higher weights to human-generated events than background events, enabling robust pose estimation even in the presence of significant background noise. Additionally, a frame buffer mechanism was introduced to maintain stable pose predictions even when only a limited number of events were captured due to the lack of motions.

In this paper, we substantially extend EventEgo3D (Millerdurai et al, 2024a) with EventEgo3D++, which includes several key improvements and additions. Firstly, we improve the 3D pose estimation accuracy of the EventEgo3D framework (Millerdurai et al, 2024a) by incorporating additional supervision through a 2D projection loss and a bone loss. Secondly, in addition to the synthetic dataset (EE3D-S) and the studio-recorded real dataset (EE3D-R) included in EventEgo3D, we introduce a new in-the-wild real dataset (EE3D-W) with 3D ground truth poses, providing additional data for fine-tuning and evaluating our method in outdoor environments. Thirdly, we provide allocentric RGB views and SMPL (Loper et al, 2015) body annotations to the real datasets, thereby providing a more comprehensive dataset for advancing research. The inclusion of in-the-wild data ensures robustness to real-world conditions, while SMPL body annotations provide dense human correspondences, making the datasets

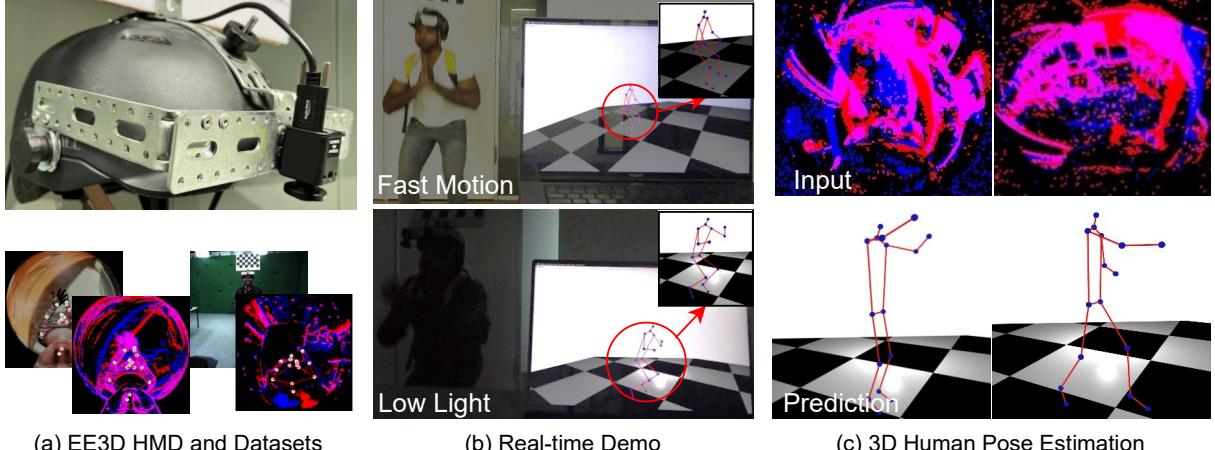


Fig. 1: EventEgo3D++ builds upon the work of EventEgo3D (Millerdurai et al, 2024a) for real-time 3D human motion capture from egocentric event streams: (a) A photograph of our new head-mounted device (HMD) with a custom-designed egocentric fisheye event camera (top) and visualisations of our synthetically rendered dataset and a real dataset recorded with the HMD (bottom); (b) Real-time demo achieving the pose update rate of 140Hz; (c) Visualisation of real event streams (top) and the corresponding 3D human poses from a third-person perspective.

valuable for future research and applicable to a wide range of applications.

The remainder of this paper is organised as follows. Section 2 reviews related work on egocentric 3D human motion capture, event-based 3D reconstruction, and other alternative sensors for 3D human pose estimation. Section 3 provides a detailed description of our EventEgo3D++ method, focusing on the neural network architecture and the newly introduced losses. Section 4 describes the design and implementation of our mobile head-mounted device prototype and the synthetic dataset. Additionally, we outline the recording procedures for the real datasets, including both studio and in-the-wild settings. Section 5 presents a comprehensive evaluation of our method on synthetic and real datasets. Finally, Section 6 discusses the limitations of our approach, and Section 7 offers our concluding remarks.

2 Related Work

We next review related methods for egocentric 3D human pose estimation and event-based 3D reconstruction.

2.1 Egocentric 3D Human Pose Estimation

3D human pose estimation from egocentric monocular or stereo RGB views has been actively studied during the last decade. While the earliest approaches were optimisation-based (Rhodin et al, 2016), the field promptly adopted neural architectures following the state of the art in human pose estimation. Thus, follow-up methods used a two-stream CNN architecture (Xu et al, 2019) and auto-encoders for monocular (Tome et al, 2019, 2020) and stereo inputs (Zhao et al, 2021; Akada et al, 2022, 2024; Kang et al, 2023). Another work focused on the automatic calibration of fisheye cameras widely used in the egocentric setting (Zhang et al, 2021). Recent papers leverage human motion priors and temporal constraints for predictions in the global coordinate frame (Wang et al, 2021); reinforcement learning for improved physical plausibility of the estimated motions (Yuan and Kitani, 2019; Luo et al, 2021); semi-supervised GAN-based human pose enhancement with external views (Wang et al, 2022a) and depth estimation (Wang et al, 2023); and scene-conditioned denoising diffusion probabilistic models (Zhang et al, 2023).

Khirodkar et al (2023) address a slightly different setting and use a multi-stream transformer to capture multiple humans in front-facing egocentric views. Meanwhile, Wang et al (2024a) focus on egocentric whole-body motion capture with a single fisheye camera, utilising FisheyeViT for feature extraction, specialised networks for hand tracking, and a diffusion-based model for refining motion estimates.

All these works demonstrated promising results and pushed the field forward. They, however, were designed for synchronously operating RGB cameras and, hence—as every RGB-based method—suffer from inherent limitations of these sensors (detailed in Sec. 1). Thus, only a few of them support real-time frame rates (Xu et al, 2019; Tome et al, 2019). Moreover, it is unreasonable to expect that RGB-based approaches can be easily adapted for event streams. In contrast, we propose an approach that (for the first time) accounts for the new data type in the context of egocentric 3D vision (events) and estimates 3D human poses at high 3D pose update rates.

Last but not least, none of the existing datasets for the training and evaluation of egocentric 3D human pose estimation techniques and related problems (Rhodin et al, 2016; Xu et al, 2019; Tome et al, 2019; Wang et al, 2021; Zhang et al, 2022; Wang et al, 2023; Pan et al, 2023; Khirodkar et al, 2023; Wang et al, 2022a, 2024b) provide event streams or frames at framerate sufficient to generate events with event steam simulators (Rebecq et al, 2018). To evaluate and train our approach, we synthesise and record the necessary datasets (*i.e.*, synthetic, real, and background augmentation) required to investigate event-based 3D human pose estimation on HMDs.

2.2 Event-based Methods for 3D Reconstruction

Substantial discrepancies between RGB frames and asynchronous event data have spurred the development of specialised 3D pose estimation methods, ranging from purely event-based approaches (Rudnev et al, 2021; Nehvi et al, 2021; Zou et al, 2021; Wang et al, 2022b; Xue et al, 2022; Chen et al, 2022; Rudnev et al, 2023; Millerdurai et al, 2024b) to RGB-event hybrid methods (Xu et al, 2020; Zou et al, 2021; Park

et al, 2024; Jiang et al, 2024b). Although hybrid solutions can offer complementary information, they also significantly increase bandwidth usage, power consumption, and computational overhead—factors that become especially problematic for battery-powered head-mounted displays. For a comparison of bandwidth usage and power consumption between RGB and event cameras, please see App. A. Consequently, our work adopts a purely event-based paradigm.

Within the event-based domain, Nehvi et al (2021) track non-rigid 3D objects (polygonal meshes or parametric 3D models) with a differentiable event stream simulator. Rudnev et al (2021) synthesise a dataset with human hands to train a neural 3D hand pose tracker with a Kalman filter. They introduce a lightweight LNES representation of events for learning as an improvement upon event frames. Next, Xue et al (2022) optimise the parameters of a 3D hand model by associating events with mesh faces using the expectation-maximisation framework assuming that events are predominantly triggered by hand contours. Some works represent events as spatiotemporal points in space and encode them either as point clouds (Chen et al, 2022; Millerdurai et al, 2024b). Consequently, most of these approaches are slow (due to different reasons such as iterative optimisation or computationally expensive operations on 3D point clouds), with the notable exception of EventHands (Rudnev et al, 2021) achieving up to 1kHz hand pose update rates.

In our work, we leverage LNES (Rudnev et al, 2021) because it operates independently of the input event count, facilitates real-time inference, and can be efficiently processed using neural components (*e.g.* CNN layers). Unlike the previously discussed approaches, our method is specifically designed for the egocentric setting and achieves the highest accuracy among all the methods compared. In particular, we incorporate a novel residual mechanism that propagates events (event history) from the previous frame to the current one, prioritising events triggered around the human. This is also helpful when only a few events are triggered due to the lack of motion.

2.3 Alternate Sensors for 3D Human Pose Estimation

Inertial measurement units (IMUs) have been widely used for 3D human pose estimation, often relying on multiple sensors—typically up to six—strategically placed on the head, arms, pelvis, and legs to track body movements (Von Marcard et al, 2017; Huang et al, 2018; Yi et al, 2021; Jiang et al, 2022b; Yi et al, 2022). While these systems can deliver reasonable accuracy, they tend to be cumbersome and inflexible due to the large number of sensors required and the associated calibration demands. Recent advancements have reduced the reliance on multiple sensors, with some systems using as few as three IMUs (Aliakbarian et al, 2022; Winkler et al, 2022; Jiang et al, 2022a; Lee et al, 2023; Jiang et al, 2023; Zheng et al, 2023; Jiang et al, 2025), typically mounted on the head and hands, making them more practical for applications such as virtual reality (VR). However, even with fewer sensors, these systems remain prone to issues like sensor drift and frequent recalibration during rapid motion, limiting their effectiveness in high-dynamic scenarios.

Another line of research fuses IMUs with additional modalities such as RGB data (Gilbert et al, 2019; Von Marcard et al, 2016; Malleson et al, 2017; Guzov et al, 2021; Yi et al, 2023; Dai et al, 2024) or depth maps (Helten et al, 2013), offering improved global positioning or fine-grained pose estimates. Yet, vision-based methods remain sensitive to low-light environments, occlusions, and motion blur, particularly when subjects move rapidly or operate in challenging lighting. Although diffusion-based approaches (Du et al, 2023; Li et al, 2023b; Guzov et al, 2024) have yielded smoother poses, most rely on future frames to achieve robust predictions, making them unsuitable for real-time usage.

In contrast, we propose a purely event-camera-based approach, which operates at high frame rates (*i.e.* 140 fps) and exhibits robustness to challenging conditions like low light and fast motion. By mounting a single event camera on a head-mounted display (HMD), we eliminate the need for additional body-worn sensors, thus simplifying the setup and avoiding drift issues. This setup not only handles

large lighting variations but also naturally accommodates rapid head and body movements, making it especially well-suited for real-time, egocentric 3D human pose estimation.

3 The EventEgo3D++ Approach

Our approach estimates 3D human poses from an egocentric monocular event camera with a fisheye lens. We first explain the event camera model in Sec. 3.1 and then describe the proposed framework in Sec. 3.2.

3.1 Event Camera Preliminaries

Event cameras capture event streams, *i.e.* a 1D temporal sequence that contains discrete packets of asynchronous events that indicate the brightness change of a pixel of the sensor. An event is a tuple of the form $e_i = (x_i, y_i, t_i, p_i)$ with the i -th index representing the event fired at pixel location (x_i, y_i) with its corresponding timestamp t_i and a polarity $p_i \in \{-1, 1\}$. The timestamps t_i of modern event cameras have μs temporal resolution. The event is generated when the change in logarithmic brightness \mathbb{L} at the pixel location (x_i, y_i) exceeds a predefined threshold C , *i.e.*, $|\mathbb{L}(x_i, y_i, t_i) - \mathbb{L}(x_i, y_i, t_i - t_p)| \geq C$, where t_p represents the previous triggering time at the same pixel location. $p = -1$ indicates that the brightness has decreased by C ; otherwise, it has increased if $p = 1$.

Modern neural 3D computer vision architectures (Rudnev et al, 2021; Lan et al, 2023; Jiang et al, 2024a) require event streams to be converted to a regular representation, usually in 2D or 3D. To this end, we adopt the locally normalised event surfaces (LNES) (Rudnev et al, 2021) that aggregate the event tuples into a compact 2D representation as a function of time windows. A time window of size T is constructed by collecting all events between the first event e_0 (relative to the given time window) and e_k , where $t_k - t_0 \leq T$. The events from the time window are stored in the 2D LNES frame denoted by $\mathbf{L} \in \mathbb{R}^{H \times W \times 2}$. For each event within the time window, $e_i \in \{e_1, \dots, e_k\}$, we update the LNES frame by $L(x_i, y_i, p_i) = \frac{t_i - t_0}{T}$, where an event occurring at pixel location (x, y) updates the corresponding pixel in the LNES frame.

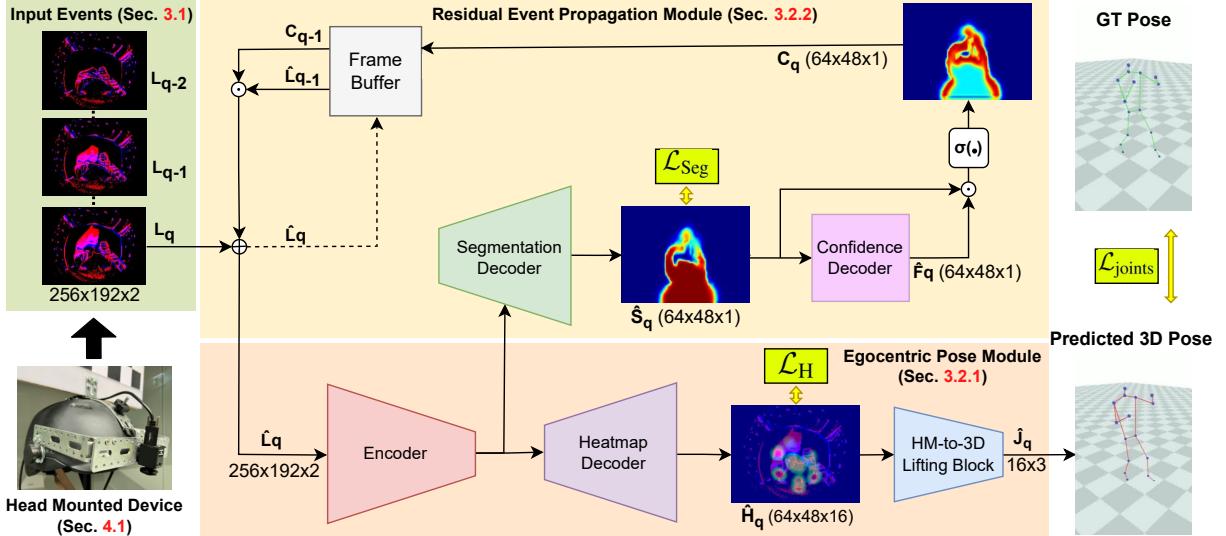


Fig. 2: Overview of our EventEgo3D++ approach. The HMD captures an egocentric event stream, which is then converted to a series of 2D LNES frames (Rudnev et al, 2021) as inputs to our neural architecture to estimate the 3D poses of the HMD user. The residual event propagation module (REPM) emphasises events triggered around the human by considering the temporal context of observations (realised with a frame buffer with event decay based on event confidence). REPM, hence, helps the encoder-decoder (from LNES to heatmaps) and the heatmap lifting module to estimate accurate 3D human poses. The method is supervised with ground-truth human masks, heatmaps and 3D human poses.

Note on Visualisation. For visualisation purposes, we convert each 2-channel LNES frame into a 3-channel (RGB) image by mapping the positive-polarity channel to the red channel, the negative-polarity channel to the blue channel, and setting the green channel to zero.

3.2 Architecture of EventEgo3D++

Our approach takes N consecutive LNES frames $\mathbf{B} = \{\mathbf{L}_1, \dots, \mathbf{L}_N\}$, $\mathbf{L}_q \in \mathbb{R}^{192 \times 256 \times 2}$ as inputs and regresses the camera-centric 3D human body pose per each LNES frame, denoted by $\mathbf{O} = \{\hat{\mathbf{J}}_1, \dots, \hat{\mathbf{J}}_N\}$, $\hat{\mathbf{J}}_q \in \mathbb{R}^{16 \times 3}$; $q \in \{1, \dots, N\}$. $\hat{\mathbf{J}}_q$ include the joints of the head, neck, shoulders, elbows, wrists, hips, knees, ankles, and feet.

The proposed framework includes two modules; see Fig. 2. First, the Egocentric Pose Module (EPM) estimates the 3D coordinates of human body joints. Subsequently, the Residual Event Propagation Module (REPM) propagates events from the previous LNES frame to the current one. The REPM module allows the framework 1) to focus more on the events

triggered around the human (than those of the background) and 2) to retain the 3D human pose when only a few events are generated due to the absence of motions.

3.2.1 Egocentric Pose Module (EPM)

We regress 3D joints from the input \mathbf{L}_q in two steps: 1) 2D joint heatmap estimation and 2) the heatmap-to-3D lifting.

2D joint heatmap estimation. To estimate the 2D joint heatmap, we develop a U-Net-based architecture (Ronneberger et al, 2015). Here, we utilise the Blaze blocks (Bazarevsky et al, 2020) as layers of the encoder and decoder to achieve real-time performance. The encoder and decoder have five layers each (see Fig. 3). The encoder takes \mathbf{L}_q as input and the heatmap decoder generates 2D joint heatmaps with different resolution sizes from each layer. Then, we average them to create the heatmaps of 16 body joints $\hat{\mathbf{H}}_q \in \mathbb{R}^{48 \times 64 \times 16}$ as the final output. For further details on the heatmap averaging scheme, please refer to App. B.

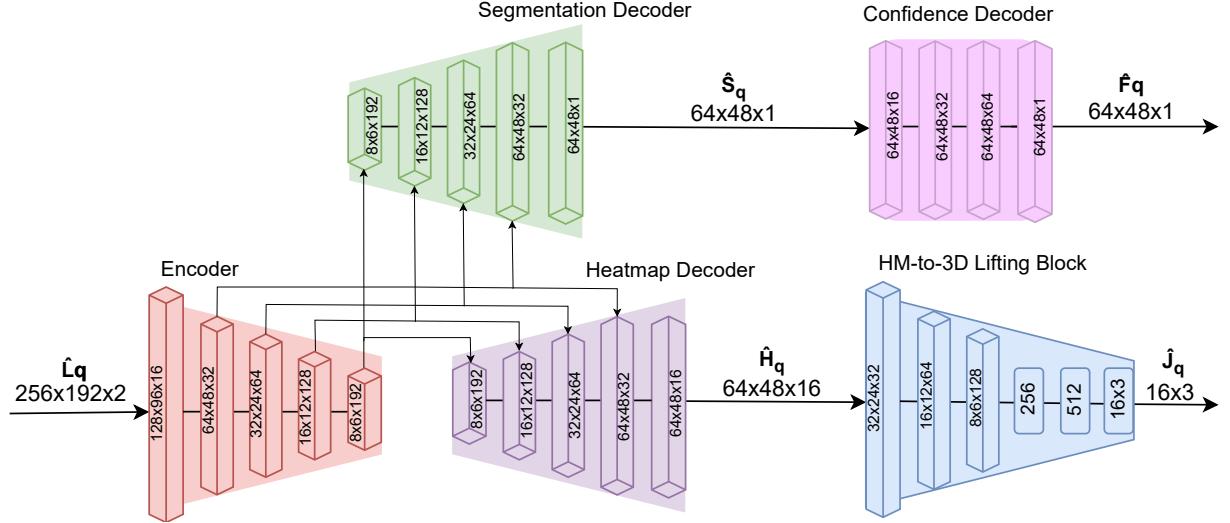


Fig. 3: The network architecture of EventEgo3D++. The Encoder takes the current LNES frame \hat{L}_q as an input. The Heatmap Decoder predicts 2D heatmaps for 16 body joints, which are then fed into the HM-to-3D lifting block to regress 3D joint locations. The Segmentation Decoder generates the human body mask, and the Confidence Decoder subsequently produces a feature map that acts on the human body mask to create a confidence map, highlighting important regions in the egocentric view.

The network is supervised using the mean square error (MSE) between the ground-truth heatmaps and the predicted ones:

$$\mathcal{L}_H = \frac{1}{M_J} \sum_{b=1}^{M_J} \|\hat{\mathbf{H}}_{q,b} \odot V_{q,b} - \mathbf{H}_{q,b} \odot V_{q,b}\|^2, \quad (1)$$

where $\hat{\mathbf{H}}_{q,b}$ and $\mathbf{H}_{q,b}$ are the predicted and ground-truth heatmaps of the b -th joint; $V_{q,b} \in \{0, 1\}$ is the visibility of the b -th joint; M_J is the number of body joints and \odot is the element-wise multiplication. The visibility mask ($V_{q,b}$) ensures that only the joints that are visible and thus relevant for pose estimation contribute to the loss calculation. This is particularly important in scenarios where some joints may be occluded or out of view, such as when the arms are extended or the feet are positioned behind the torso. Applying the visibility mask allows the network training to focus more on the joints that are detectable in the input LNES frames instead of occluded or out-of-view joints.

Heatmap-to-3D Lifting Module. Following previous works (Tome et al., 2019; Pavlakos et al., 2018), the Heatmap-to-3D (HM-to-3D) Lifting module takes the estimated heatmaps as input and outputs the 3D joints $\hat{\mathbf{J}}_q \in \mathbb{R}^{16 \times 3}$. This

module is based on three convolutional layers and three dense layers (see Fig. 3). We supervise the module using three distinct loss terms: the MSE of the 3D joints (3D loss), the MSE of the 2D joints reprojected from the 3D joints (2D reprojection loss), and the error in bone orientations and bone lengths (bone loss).

The 3D loss is computed using the ground-truth joint positions and estimated ones at the frame index q :

$$\mathcal{L}_{J3D} = \frac{1}{M_J} \sum_{r=1}^{M_J} \|\hat{\mathbf{J}}_{q,r} \odot V_{q,r} - \mathbf{J}_{q,r} \odot V_{q,r}\|^2, \quad (2)$$

where M_J is the number of body joints, $V_{q,r} \in \{0, 1\}$ is the visibility of the r -th joint and $\hat{\mathbf{J}}_{q,r}$ and $\mathbf{J}_{q,r}$ are the predicted and ground-truth r -th joint, respectively.

The 2D reprojection loss denoted as \mathcal{L}_{J2D} , compares the 2D projections of the predicted and ground-truth 3D joints, is formulated as:

$$\mathcal{L}_{J2D} = \frac{1}{M_J} \sum_{r=1}^{M_J} \|\Pi(\hat{\mathbf{J}}_{q,r}) \odot V_{q,r} - \Pi(\mathbf{J}_{q,r}) \odot V_{q,r}\|^2, \quad (3)$$

where Π is the camera projection function for the fisheye lens, projecting 3D joints into 2D joints.

The bone loss, denoted as \mathcal{L}_{BA} , captures the difference between predicted and ground-truth bone orientations and lengths, allowing the network to learn the spatial relationships between joints and bones.

For bone orientations, we use a negative cosine similarity loss, defined as:

$$\mathcal{L}_{\theta} = \frac{1}{N_L} \sum_{l=1}^{N_L} \left(1 - \cos(\hat{\mathbf{P}}_{q,l}, \mathbf{P}_{q,l}) \right), \quad (4)$$

where N_L is the number of bones, $\hat{\mathbf{P}}_{q,l} \in \mathbb{R}^3$ is the l -th predicted bone vector, $\mathbf{P}_{q,l} \in \mathbb{R}^3$ is the corresponding ground-truth bone vector, and

$$\cos(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \|\mathbf{b}\|}.$$

This formulation, $1 - \cos(\cdot, \cdot)$, penalizes misalignment between each predicted bone and its ground-truth counterpart.

For bone lengths, we compute the MSE between the predicted and ground-truth bone vectors, denoted as \mathcal{L}_{BL} :

$$\mathcal{L}_{\text{BL}} = \frac{1}{N_L} \sum_{l=1}^{N_L} \|\hat{\mathbf{P}}_{q,l} - \mathbf{P}_{q,l}\|^2. \quad (5)$$

The overall bone loss \mathcal{L}_{BA} is computed by combining the orientation and length losses:

$$\mathcal{L}_{\text{BA}} = \lambda_{\theta} \mathcal{L}_{\theta} + \lambda_{\text{BL}} \mathcal{L}_{\text{BL}}, \quad (6)$$

where $\lambda_{\theta}=0.001$ and $\lambda_{\text{BL}}=0.001$ are the weights assigned to the orientation and length losses, respectively.

Overall, the combined supervision loss for the joints, denoted as $\mathcal{L}_{\text{joints}}$, is defined as:

$$\mathcal{L}_{\text{joints}} = \lambda_{\text{J3D}} \mathcal{L}_{\text{J3D}} + \lambda_{\text{J2D}} \mathcal{L}_{\text{J2D}} + \lambda_{\text{BA}} \mathcal{L}_{\text{BA}} \quad (7)$$

where we set the weight of each loss as $\lambda_{\text{J3D}}=0.01$, $\lambda_{\text{J2D}}=0.01$, $\lambda_{\text{BA}}=1$.

3.2.2 Residual Event Propagation Module (REPM)

In contrast to stationary camera setups, egocentric cameras mounted on head-mounted

displays (HMDs) experience diverse movement, which affects the number of events they capture. Intense movements by HMD users often result in a large number of events, with a significant portion coming from the background. Conversely, minimal motion results in very few events.

To address these issues, we introduce the Residual Event Propagation Module (REPM). The REPM helps the network focus on events generated by the human body while further incorporating information from previous frames. By focusing on human-generated events, the network ensures that these events are given higher importance than background events. Simultaneously, propagating information from previous frames helps maintain stable pose estimates even when few events are observed.

The REPM comprises the segmentation decoder, the confidence decoder, and the frame buffer. The segmentation decoder estimates human body masks. Next, the confidence decoder takes the body masks as inputs to produce feature maps. These feature maps are then used with the body masks to produce confidence maps that indicate regions of the egocentric view to place more importance on. Lastly, the frame buffer stores the past input frame and its corresponding confidence map, providing weighting to important regions of the current frame (see the top part of Fig. 2).

Segmentation Decoder. The segmentation decoder estimates the human body mask $\hat{\mathbf{S}}_q \in \mathbb{R}^{48 \times 64 \times 1}$ of the HMD user in the egocentric LNES views. The architectures of this module and the heatmap decoder are the same except for the final layer that outputs human body masks.

We use the feature maps from multiple layers of the encoder as inputs to the segmentation decoder (see Fig. 3). The segmentation decoder is supervised by the cross-entropy loss:

$$\mathcal{L}_{\text{seg}} = -\mathbf{S}_q \log(\hat{\mathbf{S}}_q) + (1 - \mathbf{S}_q) \log(1 - \hat{\mathbf{S}}_q), \quad (8)$$

where $\hat{\mathbf{S}}_q$ and \mathbf{S}_q are the predicted and ground-truth segmentation masks, respectively.

Confidence Decoder. The confidence decoder is a four-layer convolution network that takes the human body mask $\hat{\mathbf{S}}_q$ as input and produces a feature map $\hat{\mathbf{F}}_q \in \mathbb{R}^{48 \times 64 \times 1}$. This feature map is then used in combination with $\hat{\mathbf{S}}_q$ to produce the

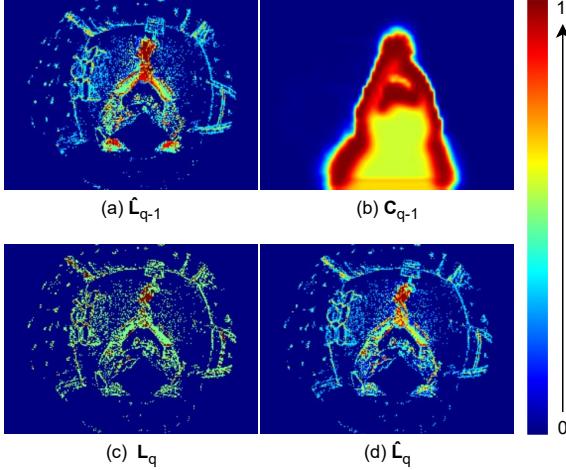


Fig. 4: Visualisation of frame buffering and human-weighted event generation. The frame buffer holds previous input frame $\hat{\mathbf{L}}_{q-1}$ (a) and previous confident map \mathbf{C}_{q-1} (b). $\hat{\mathbf{L}}_{q-1}$ is weighted with \mathbf{C}_{q-1} and added to the current LNES frame \mathbf{L}_q (c) to produce $\hat{\mathbf{L}}_q$ (d). We can observe that the events generated by the subject are highlighted more than the background events.

confidence map $\mathbf{C}_q \in \mathbb{R}^{48 \times 64 \times 1}$:

$$\mathbf{C}_q = \text{sigmoid}(\hat{\mathbf{S}}_q \odot \hat{\mathbf{F}}_q), \quad (9)$$

where “ $\text{sigmoid}(\cdot)$ ” is a sigmoid operation.

Frame Buffer. The frame buffer stores the previous confidence map $\mathbf{C}_{q-1} \in \mathbb{R}^{48 \times 64 \times 1}$ and the previous input frame $\hat{\mathbf{L}}_{q-1} \in \mathbb{R}^{192 \times 256 \times 2}$. Note that we initialise the frame buffer with zeros at the first frame. To compute the current input frame $\hat{\mathbf{L}}_q$, we retrieve \mathbf{C}_{q-1} and $\hat{\mathbf{L}}_{q-1}$ from the frame buffer using the following expression:

$$\hat{\mathbf{L}}_q = (\hat{\mathbf{L}}_{q-1} \odot \mathbf{C}_{q-1}) \oplus \mathbf{L}_q \quad (10)$$

where \mathbf{L}_q denotes the LNES frame at the current time and “ \oplus ” represents an element-wise addition. We normalise the values of $\hat{\mathbf{L}}_q$ to the range of $[-1, 1]$. Note, \mathbf{C}_{q-1} is resized to 192×256 before applying Eqn. (10). See Fig. 4 for an exemplary visualisation of the components used in Eqn. (10).

3.2.3 Loss Terms and Supervision

Overall, our method is supervised by the heatmap loss \mathcal{L}_H (Eqn. 1), the joint loss $\mathcal{L}_{\text{joints}}$ (Eqn. 7) and

the segmentation loss \mathcal{L}_{seg} (Eqn. 8) as follows:

$$\mathcal{L} = \lambda_{\text{joints}} \mathcal{L}_{\text{joints}} + \lambda_H \mathcal{L}_H + \lambda_{\text{seg}} \mathcal{L}_{\text{seg}}, \quad (11)$$

where we set the weight of each loss as $\lambda_{\text{joints}}=1$, $\lambda_H=20$, $\lambda_{\text{seg}}=0.1$.

4 Our Egocentric Setup and Datasets

In this work, we introduce three new datasets: EE3D-R, EE3D-W, and EE3D-S. These datasets are used to train, evaluate, and fine-tune our EventEgo3D++ method. EE3D-R and EE3D-W are real-world datasets captured using our head-mounted device (HMD). The EE3D-R dataset is recorded in a studio environment with controlled lighting and background conditions. In contrast, EE3D-W includes both indoor and outdoor environments in the real world, offering a broader range of scenarios that more accurately represent real-world conditions. EE3D-S is a large-scale synthetic dataset with the same camera parameters applied from our real-world camera. EE3D-S provides a diverse array of human poses within a wide variety of virtual backgrounds. Together, these datasets support a comprehensive approach to developing and refining the EventEgo3D++ method. Moreover, pre-training with the synthetic dataset and further fine-tuning on real-world datasets allows the model to handle both diverse and realistic conditions.

4.1 Real-world Data Capture

In this section, we first describe our experimental head-mounted device (HMD) used to create real-world datasets, *i.e.* EE3D-R and EE3D-W (Sec. 4.1.1). Next, we outline the calibration process for our HMD setup (Sec. 4.1.2). We then detail the procedure for generating ground truth data using the calibrated HMD (Sec. 4.1.3). Finally, we describe the details of the captured datasets, including their diversity and coverage (Sec. 4.1.4).

4.1.1 Head-Mounted Device

Our HMD is a prototypical device consisting of a bicycle helmet with a DVXplorer Mini (2021)

event camera attached to the helmet 3.5cm away from the user’s head; the strap allows a firm attachment on the head. (see Fig. 5) We use a fisheye lens, [Lensagon BF10M14522S118C \(2020\)](#), with a field of view of 190°. The wide field of view effectively covers scenarios where the user’s arms are fully extended. The total weight of the device is $\approx 0.42\text{kg}$. The device is used with a laptop in a backpack for external power supply and real-time on-device computing. The compact design and the flexibility of our HMD allow users to freely move their heads and perform rapid motions.

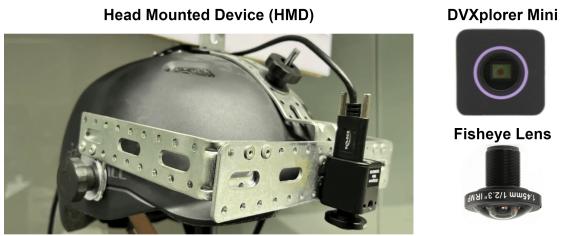


Fig. 5: Our real-world setup. The head-mounted device is equipped with an event camera and a fisheye lens.

4.1.2 Camera Calibration

Intrinsic Calibration. We record an event stream of a moving checkerboard, as described by [Muglikar et al \(2021\)](#), and then convert the stream into a sequence of images using E2VID ([Rebecq et al, 2019a](#)). For the intrinsic calibration, we utilise the Scaramuzza projection model ([Scaramuzza et al, 2006](#)), which can account for the radial distortion and the wide field of view of the fisheye lens on our head-mounted device (HMD). Specifically, we use MATLAB’s Camera Calibrator tool ([MathWorks, 2023](#)) to obtain the projection model parameters.

Extrinsic Calibration. To obtain the egocentric 3D poses and SMPL ([Loper et al, 2015](#)) parameters of the HMD user, we first track the HMD’s position during the motion recording. This can be achieved by calibrating the HMD equipped with a checkerboard as a reference marker in an allocentric RGB multi-camera setup. This step enables us to track the HMD’s position within the coordinate frame of the multi-camera setup, *i.e.* the world coordinate frame. Subsequently,

we perform hand-eye calibration to compute the HMD coordinate frame. Finally, we convert the 3D poses and SMPL parameters from the world coordinate frame into the HMD’s coordinate frame.

To obtain the checkerboard images necessary for the hand-eye calibration, we first generate events from the checkerboard and then convert these events into images using the E2VID ([Rebecq et al, 2019a](#)). To ensure uniform event distribution, we slide the checkerboard diagonally during the event capture process. The final position of the checkerboard after this sliding motion serves as the reference chequerboard position for the calibration procedure. For additional details on the hand-eye calibration, please refer to App. C.1.

4.1.3 Ground Truth Generation

We obtain the 3D human poses and SMPL ([Loper et al, 2015](#)) body parameters using the multi-view motion capture setups, [Captury \(2024\)](#) and [EasyMoCap \(2021\)](#). Specifically, [Captury \(2024\)](#) is a RGB-based multi-view motion capture system that provides accurate human joint positions while [EasyMoCap \(2021\)](#) is used to derive the SMPL parameters from multi-view RGB streams. Subsequently, we transform these 3D human poses and SMPL parameters from the world coordinate frame to the HMD coordinate frame. For more details on the accuracy of the generated ground-truth, please refer to App. C.2.

Additionally, we generate egocentric human body masks, 2D joint coordinates, and joint visibility masks. The joint visibility mask indicates whether a joint is visible or occluded from the egocentric view. For further details on the generation of the human body masks, 2D egocentric joint coordinates, and joint visibility masks, we refer readers to App. C.3

Note on Additional Metadata. Our dataset release includes SMPL body parameters, meshes, and allocentric multi-view RGB streams. These supplementary data are provided solely for future research purposes—such as shape estimation and clothing reconstruction—and are not used in the training or evaluation of our framework.

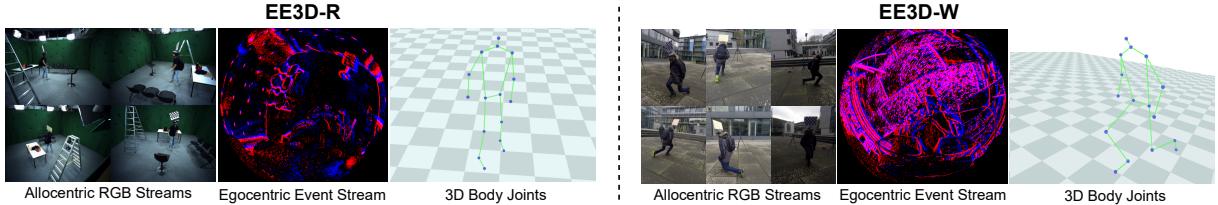


Fig. 6: Visualisation of example data from EE3D-R (left) and EE3D-W (right) datasets.

4.1.4 Real-world Datasets

Following the procedure in the previous sections, we create the real-world datasets, EE3D-R and EE3D-W.

EE3D-R. EE3D-R is a studio dataset that consists of everyday movements, each performed in different manners by various participants. We ask twelve subjects—persons with different body shapes and skin tones—to wear our HMD and perform different motions (e.g. fast) in a multi-view motion capture studio with 30 allocentric RGB cameras recording at 50 fps. (see the left part of Fig. 6)

Each sequence encompasses the following motions: walking, crouching, pushups, boxing, kicking, dancing, interaction with the environment, crawling, sports and jumping. In the sports category, participants perform specific activities—playing basketball, participating in tug of war, and playing golf. Meanwhile, in the interaction with the environment category, the subjects perform actions such as picking up objects from a table, sitting on a chair, and moving the chair.

In total, we collect 12 sequences containing approximately $4.64 \cdot 10^5$ poses spanning around 155 minutes. These sequences include both fast-paced actions (boxing, kicking, dancing, sports, jumping), comprising approximately $2.46 \cdot 10^5$ frames, as well as slower-paced activities in the remaining frames. Figure 7 illustrates the visibility of each joint derived from the SMPL body (see App. C.3 for details on the generation process). We observe that the lower-body joints are predominantly occluded or out-of-view due to camera constraints, with only about 40% visibility for the ankles. For our experiments, we use eight sequences ($2.87 \cdot 10^5$ poses) for training, two sequences ($1.05 \cdot 10^5$ poses) for validation, and two sequences ($7.16 \cdot 10^4$ poses) for testing.

EE3D-W. EE3D-W is an in-the-wild dataset recorded under varying lighting conditions in three different scenes: indoor environments, outdoor areas with concrete flooring, and outdoor areas with grass. We capture various motions of six subjects in a multi-view motion capture setup with 6 allocentric RGB cameras recording at 60 fps. (See the right part of Fig. 6.) The motion types in EE3D-W are similar to those specified in EE3D-R. This resulted in nine sequences totaling approximately $4.18 \cdot 10^5$ poses over 116 minutes, with roughly $1.845 \cdot 10^5$ frames containing fast-paced motion. As shown in Fig. 7, the in-the-wild dataset exhibits lower overall joint visibility compared to EE3D-R. This is primarily because frequent head movements during outdoor activities cause parts of the body to intermittently move in and out of the camera’s field of view, thereby increasing occlusions. For our experiments, we use five sequences ($2.28 \cdot 10^5$ poses) for training, two sequences ($9.32 \cdot 10^4$ poses) for validation, and two sequences ($9.24 \cdot 10^4$ poses) for testing.

4.2 Synthetic Data Setup

In addition to the real-world datasets, we propose EE3D-S, a large-scale synthetic dataset. In the following, We first describe the virtual human character wearing the HMD and virtual scenes (Sec. 4.2.1). Next, we explain the rendering and generation of the egocentric event stream (Sec. 4.2.2). We then outline the ground truth generation for the proposed dataset (Sec. 4.2.3). Finally, we introduce an event augmentation strategy aimed at reducing the domain gap between real-world datasets (Sec. 4.2.4).

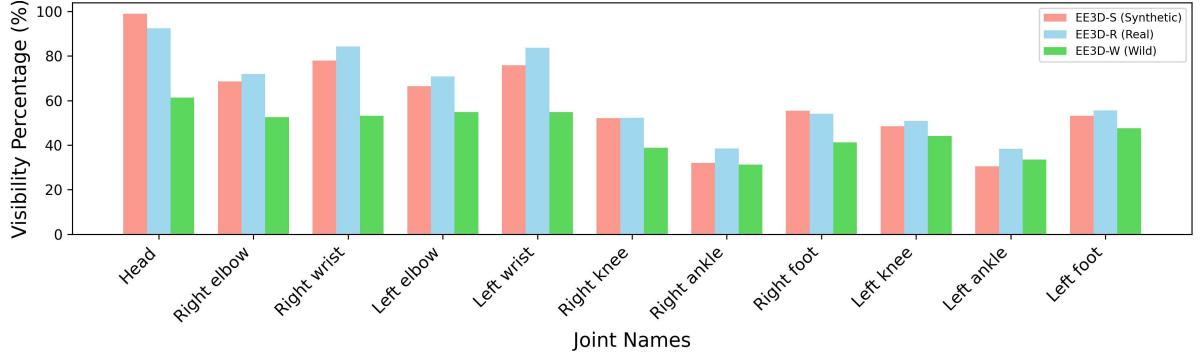


Fig. 7: Joint visibility for SMPL body in our proposed datasets. The visibility percentage is computed as the proportion of samples where each joint is visible from the egocentric perspective.

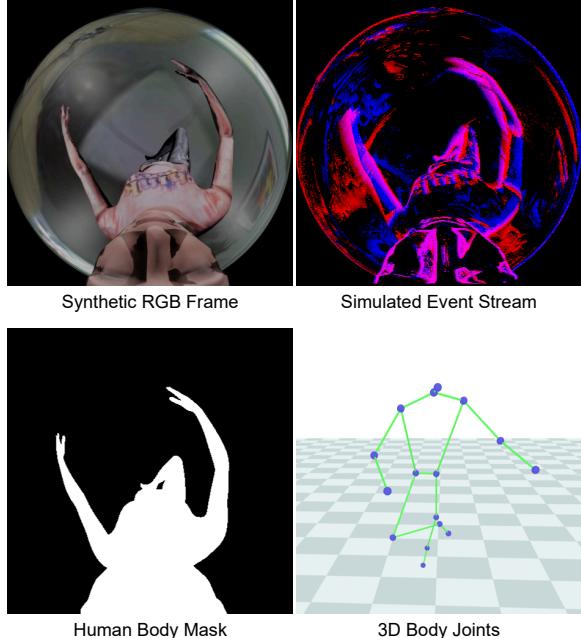


Fig. 8: Visualisations of sample data from EE3D-S.

4.2.1 Virtual Human Character and Background Scene

We utilise SMPL body models as virtual human users for our HMD, following Xu et al (2019). Body textures are randomly sampled from the SURREAL dataset (Varol et al, 2017), and animations are driven by motions from the CMU MoCap dataset (CMU, 2006). When generating event data, we sample motions at high frame rates (Gehrig et al, 2020) using linear interpolation of SMPL parameters.

As a background scene, we use a $26m^2$ sized 2-dimensional plane with textures sampled from the LSUN dataset (Yu et al, 2015). The scenes are illuminated by four randomly placed point lights within a 5-meter radius of the HMD.

4.2.2 Rendering and Event Stream Generation

We render egocentric views using a fisheye camera positioned near the virtual human’s face, emulating the real-world HMD setup. We apply random perturbations to the fisheye camera position to account for head size variations and HMD movement. This allows for simulating real-world scenarios where the camera position relative to the user’s head may slightly shift. We use real-world intrinsic camera parameters (Sec. 4.1.2) to render RGB frames and human body masks. The rendered RGB frames are then processed by VID2E (Gehrig et al, 2020) to generate the event streams. Sample data of EE3D-S is shown in Fig. 8.

In total, we synthesise 946 motion sequences containing approximately $6.34 \cdot 10^6$ 3D human poses and $1.419 \cdot 10^{11}$ events. As shown in Fig. 7, joint visibility is predominantly reduced in the lower body, while the head remains largely unobstructed. For our experiments, we use 860 sequences ($5.75 \cdot 10^6$ poses) for training, 43 sequences ($3.79 \cdot 10^5$ poses) for validation, and 43 sequences ($2.15 \cdot 10^5$ poses) for testing. For further details on the configurations used to create the synthetic dataset, we refer readers to App. E.

4.2.3 Ground Truth Generation

We extract 3D body joints from the SMPL model, including the head, neck, shoulders, elbows, wrists, hips, knees, ankles, and feet. Additionally, we derive 2D joints, human body masks, and visibility masks as outlined in App. C.3.

4.2.4 Event Augmentation

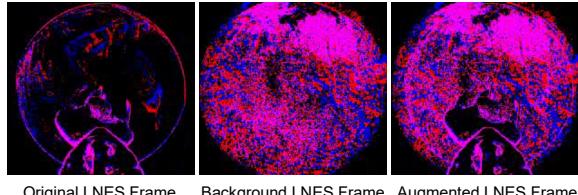


Fig. 9: An example scenario of our event augmentation technique. The original LNES frame (left) is augmented with an LNES frame of background events (middle) to create an augmented LNES frame (right).

Models trained on synthetic data often fail to generalise effectively to real-world scenarios with diverse backgrounds. To address this issue, we propose an event-wise augmentation technique for background events of the synthetic dataset, EE3D-S (see Fig. 9). First, we capture sequences of both outdoor and indoor scenes without humans with a handheld event camera, creating background event streams. These streams are then converted to 2D background LNES frames \mathbf{L}_B (center image in Fig. 9). Subsequently, We apply the human body mask \mathbf{S}_B from EE3D-S to \mathbf{L}_B , obtaining a background LNES frame without a region corresponding to a human body in the original LNES frame, denoted as \mathbf{L}_A . Finally, we add \mathbf{L}_A to the original LNES frames \mathbf{L}_q from EE3D-S to generate the augmented frame \mathbf{L}_{aug} (right image in Fig. 9). \mathbf{L}_{aug} serves as the input to our network.

5 Experimental Evaluation

This section describes the implementation details of our experiments (Sec. 5.1), our results including numerical comparisons to the most related methods (Sec. 5.2), an ablation study validating the contributions of the core method modules

(Sec. 5.3) as well as comparisons in terms of the runtime and architecture parameters (Sec. 5.4). Finally, we show a real-time demo (Sec. 5.5).

5.1 Implementation Details

We implement our method in PyTorch (Paszke et al, 2019) and use Adam optimiser (Kingma and Ba, 2015) with a batch size of 27. For the EE3D-S dataset, we adopt a learning rate of 10^{-3} for $8 \cdot 10^5$ iterations. For the EE3D-R dataset, we train our network with a learning rate of 10^{-4} for $1.5 \cdot 10^4$ iterations. For the EE3D-W dataset, we use a learning rate of 10^{-4} for $1.2 \cdot 10^4$ iterations. All modules of our EventEgo3D++ architecture are jointly trained. The network is supervised using the most recent ground-truth human pose within the time window T when constructing the LNES frame, *i.e.* the ground-truth pose is aligned with the latest event in the LNES. We set $T = 33$ ms and $N = 20$ for our experiments. For additional details on how the LNES frames are constructed, please refer to App. F. The performance metrics are reported on a single GeForce RTX 3090. The real-time demo is performed on a laptop equipped with a single 4GB Quadro T1000 GPU housed in a backpack as illustrated in Fig. 1-(b).

We compare our method EventEgo3D++ with EventEgo3D (Millerdurai et al, 2024a), the CVPR version of our work. In addition, we adapt three existing 3D pose estimation methods for our problem setting:

- Xu et al (2019) and Tome et al (2019) are egocentric RGB-based methods: We modify their first convolution layer to accept the LNES representation. Specifically, we replace the original 3-channel input convolution, which is designed for RGB images, with a 2-channel input convolution layer that is compatible with the LNES representation.
- Rudnev et al (2021) is an event-based method that takes LNES as input and estimates hand poses: We modify its output layer to regress 3D human poses. Specifically, we modify the output linear layer to predict the 3D body joints $\hat{\mathbf{J}} \in \mathbb{R}^{16 \times 3}$.

For a fair comparison, we adopt the same training strategy, *i.e.* learning rates and iterations, for all of the competing methods as ours. We follow previous works (Xu et al, 2019; Zhao

et al., 2021; Akada et al., 2022, 2024; Wang et al., 2021, 2022a, 2023, 2024b) to report the Mean Per Joint Position Error (MPJPE) and MPJPE with Procrustes alignment (Kendall, 1989) (PA-MPJPE).

5.2 Comparisons to the Related State of the Art

Experiment on EE3D-S. Firstly, we evaluate our approach on the test set of our synthetic EE3D-S dataset. To ensure a fair comparison, We train our method and all the competing methods (Tome et al., 2019; Xu et al., 2019; Rudnev et al., 2021; Millerdurai et al., 2024a) with the training set of our EE3D-S dataset.

From Table 1, we observe that our method achieves the lowest MPJPE of 98.67 mm on average, outperforming our previous work (Millerdurai et al., 2024a) as well as all other competing methods. Our method demonstrates superior performance in estimating lower body joints, offering a 9% improvement over Rudnev et al. (2021), with gains exceeding 11% on the ankle and foot joints. This robustness is particularly notable given the significant radial distortion caused by the fisheye lens in our setup, which makes the feet appear much smaller in the input compared to the upper body. Despite this distortion, our method effectively estimates the position of the feet and other small joint areas, highlighting its accuracy and reliability in challenging conditions.

Experiment on EE3D-R. In this experiment, we first pretrain all methods on the EE3D-S dataset. We then fine-tune these methods using the EE3D-R dataset and evaluate their performance on the EE3D-R test set. While the EE3D-S dataset includes a wide range of human motions, there is a domain gap between the synthetic and real-world cases. This gap arises from factors such as uncontrolled and diverse movement patterns, as well as wearer-specific variability, including differences in posture and movement style. Fine-tuning the pose estimation methods on real-world data can further reveal their potential in real-world scenarios.

From Table 2, we observe that our method significantly outperforms all of our comparison methods by a large margin. Specifically, our method achieves improvements of 5% in MPJPE

on average compared to the best-competing method, *i.e.* EventEgo3D (Millerdurai et al., 2024a). It is also worth noting that our method demonstrates a superiority over the competing methods especially in complex motions involving interaction with the environment, crawling, kicking, sports and dancing. These motions often come with fast-paced and jittery movements of the HMD, generating substantial background event noise. Notably, our method excels in handling such challenging scenarios.

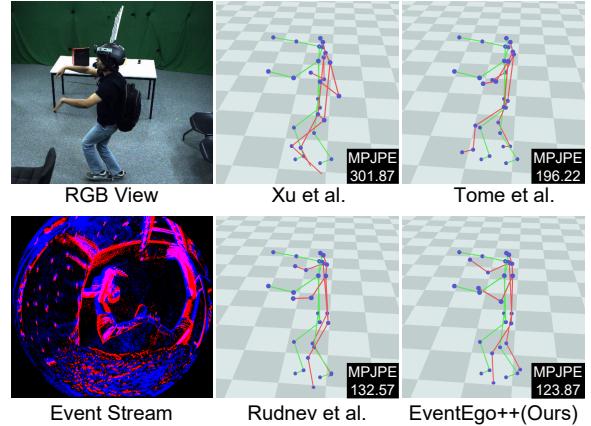


Fig. 10: Qualitative results on EE3D-R. The MPJPE values are shown in the figures. 3D pose predictions and ground-truth poses are visualised in red and green, respectively.

Fig. 10 shows visual outputs from our approach compared to other methods. The input LNES frame is noisy and the events generated by the hand sometimes exhibit very close proximity to those generated by the background. In such scenarios, the competing methods often struggle, predicting incorrect hand positions. However, our method estimates reasonably accurate 3D poses even in the presence of noisy background events.

Experiment on image-based reconstructions of EE3D-R. In this experiment, we first convert the event streams into image sequences using Rebecq et al (2019b). We then train and evaluate the RGB-based methods (Xu et al., 2019; Tome et al., 2019) on these reconstructed image sequences. From Table 4, we observe that our method, which directly processes event streams, significantly outperforms the RGB-based methods by a large

Method	Metric	Head	Neck	Shoulder	Elbow	Wrist	Hip	Knee	Ankle	Foot	Avg. (σ)
Tome et al (2019)	MPJPE	21.33	30.80	63.07	148.09	233.09	106.88	199.07	287.17	313.75	172.15 (97.4)
	PA-MPJPE	69.93	64.59	65.75	115.83	202.93	79.62	120.17	164.88	180.53	124.62 (49.85)
Xu et al (2019)	MPJPE	71.03	80.13	95.91	182.47	225.35	107.76	196.74	333.84	351.37	196.15 (97.98)
	PA-MPJPE	110.67	108.05	112.80	165.64	205.74	97.77	135.88	189.40	196.22	151.60 (40.48)
Rudnev et al (2021)	MPJPE	6.08	14.11	31.18	76.19	99.30	71.54	118.20	203.14	210.92	102.57 (68.59)
	PA-MPJPE	39.00	35.67	41.06	70.58	97.07	68.33	84.07	117.60	123.17	79.90 (30.07)
Millerdurai et al (2024a)	MPJPE	19.41	16.38	37.23	71.43	106.61	82.97	122.88	188.19	203.20	103.80 (62.03)
	PA-MPJPE	45.60	36.05	43.09	68.22	103.91	58.89	82.55	113.44	121.52	79.06 (29.47)
EventEgo3D++ (Ours)	MPJPE	18.79	20.63	35.45	68.24	97.37	73.92	118.68	181.77	194.26	98.67 (59.57)
	PA-MPJPE	35.09	32.13	36.19	60.55	87.17	51.72	76.55	98.35	107.00	68.89 (26.01)

Table 1: Numerical comparisons on the EE3D-S dataset (in mm). “ σ ” denotes the standard deviation of MPJPE or PA-MPJPE across body joints. EventEgo3D++ outperforms all other competing methods, particularly in lower body joints, achieving the best MPJPE. Additionally, our method improves lower body performance by 6% compared to EventEgo3D (Millerdurai et al, 2024a).

Method	Metric	Walk	Crouch	Pushup	Boxing	Kick	Dance	Inter. with env.	Crawl	Sports	Jump	Avg. (σ)
Tome et al (2019)	MPJPE	140.34	173.93	157.29	177.07	181.12	212.61	169.80	144.80	207.56	165.57	173.01 (23.62)
	PA-MPJPE	104.34	119.89	102.39	124.28	121.64	132.86	111.89	88.94	120.15	110.32	113.67 (12.76)
Xu et al (2019)	MPJPE	86.09	153.53	199.34	133.15	114.00	104.44	114.52	187.95	128.21	114.10	133.53 (36.42)
	PA-MPJPE	59.11	113.31	147.13	102.50	91.75	79.65	85.83	138.12	98.10	89.19	100.47 (26.52)
Rudnev et al (2021)	MPJPE	74.82	178.23	105.68	128.93	112.45	98.14	110.05	120.51	110.16	106.19	114.52 (26.54)
	PA-MPJPE	56.77	108.34	84.15	100.39	91.84	78.16	74.62	83.47	84.83	86.09	84.87 (14.08)
Millerdurai et al (2024a)	MPJPE	70.88	163.84	97.88	136.57	103.72	88.87	103.19	109.71	101.02	97.32	107.30 (25.78)
	PA-MPJPE	52.11	99.48	75.53	104.66	86.05	71.96	70.85	77.94	77.82	80.17	79.66 (14.83)
EventEgo3D++ (Ours)	MPJPE	68.67	157.41	88.63	123.57	102.31	84.95	95.73	109.38	94.9	95.94	102.15 (23.01)
	PA-MPJPE	50.06	100.76	66.29	94.52	84.26	66.91	68.2	75.73	72.23	75.83	75.48 (13.95)

Table 2: Numerical comparisons on the EE3D-R dataset (in mm). “ σ ” denotes the standard deviation of MPJPE or PA-MPJPE across actions. Our EventEgo3D++ outperforms existing approaches on most activities by a substantial margin and achieves 11% improvement over Rudnev et al (2021).

Method	Metric	Walk	Crouch	Pushup	Boxing	Kick	Dance	Inter. with env.	Crawl	Sports	Jump	Avg. (σ)
Tome et al (2019)	MPJPE	469.01	555.70	425.97	547.19	732.93	620.50	508.09	577.70	528.96	604.00	557.01 (81.12)
	PA-MPJPE	104.11	125.07	126.80	101.61	122.85	130.24	111.36	113.05	129.12	123.63	118.78 (9.90)
Xu et al (2019)	MPJPE	218.96	234.88	221.28	209.71	232.84	212.79	218.56	228.13	253.04	238.96	245.32 (12.60)
	PA-MPJPE	245.41	247.50	255.32	230.69	297.02	249.14	247.17	259.65	275.16	269.37	257.64 (17.79)
Rudnev et al (2021)	MPJPE	163.47	174.45	171.59	151.29	199.97	182.98	189.28	172.09	211.41	205.06	182.16 (18.23)
	PA-MPJPE	92.29	109.63	110.43	77.29	98.32	105.00	95.32	92.42	113.87	101.77	99.63 (10.38)
Millerdurai et al (2024a)	MPJPE	177.70	185.86	181.70	149.22	187.12	176.62	178.65	170.90	211.38	188.90	180.81 (14.81)
	PA-MPJPE	96.77	110.64	110.62	71.12	90.05	101.32	94.23	91.26	110.53	104.76	98.13 (11.74)
EventEgo3D++ (Ours)	MPJPE	164.63	160.88	171.49	145.81	172.32	163.61	164.30	151.32	193.63	173.87	166.19 (12.47)
	PA-MPJPE	93.44	96.69	105.23	69.62	89.75	97.72	90.33	85.12	104.57	98.19	93.07 (9.86)

Table 3: Numerical comparisons on the EE3D-W dataset (in mm). “ σ ” denotes the standard deviation of MPJPE or PA-MPJPE across actions. Our method, EventEgo3D++, outperforms existing approaches with the lowest MPJPE on most activities. We see an improvement of 10% over Rudnev et al (2021) in interaction with the environment (Inter. with env.), showing the robustness of our method against events generated by the environment.

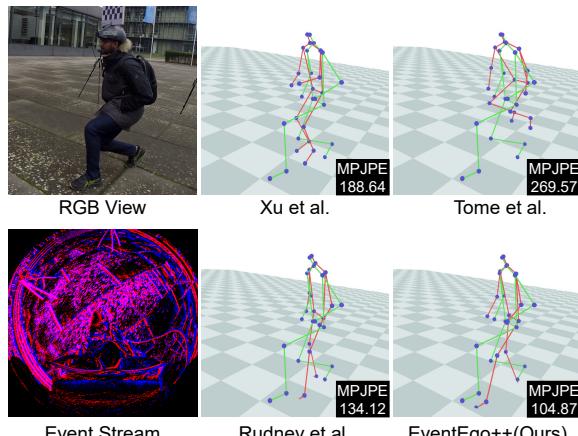


Fig. 11: Qualitative results on EE3D-W. The MPJPE values are shown in the figures. 3D pose predictions and ground-truth poses are visualised in red and green, respectively.

margin. Specifically, we achieve an average improvement of 57% in MPJPE when compared to the best-performing RGB-based method, Tome

Method	MPJPE	PA-MPJPE
*Tome et al (2019)	237.28	117.3
*Xu et al (2019)	295.17	121.91
†EventEgo3D++ (Ours)	102.15	75.48

Table 4: Numerical comparisons on the EE3D-R dataset (in mm). Methods marked with * process reconstructed images obtained from event streams using Rebecq et al (2019b), while the method marked with † processes event streams directly.

et al (2019). This performance gap can likely be attributed to artefacts introduced during the image reconstruction process. When there is significant motion of the person or background, the event camera produces a large number of events, leading to relatively clear reconstructions (see Fig. D4) However, in scenarios with sparse events—such as those with slower or minimal

motion—the reconstructed images degrade dramatically, making it difficult for RGB-based methods to accurately estimate human poses. Figure 12 illustrates this issue: although the event data captures the lower body (*e.g.* the right leg), these details are lost in the reconstructed images, leading to poorer performance by RGB-based methods. In contrast, our method, which leverages the raw event streams, continues to produce reasonably accurate 3D poses even under these challenging conditions. For additional details on the conversion process, we refer readers to App. D.

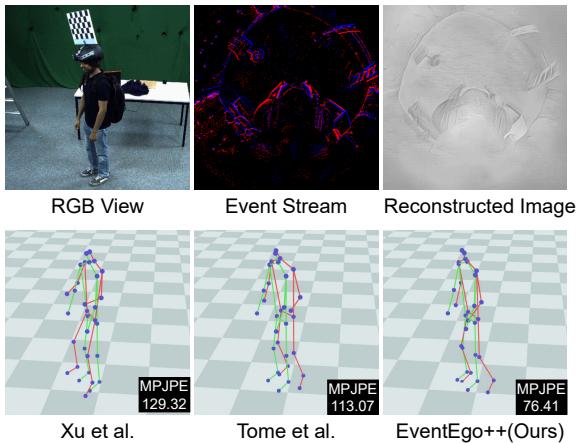


Fig. 12: Qualitative results on EE3D-R. The MPJPE values are shown in the figures. 3D pose predictions and ground-truth poses are visualised in red and green, respectively. Both Xu et al (2019) and Tome et al (2019) process reconstructed images obtained from event streams, whereas EventEgo++ (Ours) directly processes the event streams.

Experiment on EE3D-W. We are also interested in pose estimation performance in in-the-wild real-world scenarios, *i.e.* EE3D-W. Therefore, in this experiment, we initially pretrain all methods on the EE3D-S dataset and then fine-tune them using the training set of EE3D-W for the evaluation on the test set of EE3D-W. From Tab. 3, we observe that our approach achieves the best MPJPE and PA-MPJPE scores among all methods. Compared to other competing methods, there is a significant performance improvement, ranging from a 8%

improvement over Rudnev et al (2021) to a 70% improvement over Tome et al (2019) in the MPJPE. Furthermore, we achieve high accuracy in specific motions, such as crawling, crouching, pushups, and boxing. This reflects our strength in handling diverse and complex human activities. Additionally, we achieve the lowest standard deviation σ of the 3D errors on average. This result indicates that our method is robust across different types of motion, consistently providing accurate 3D pose estimations for a wide range of activities. Fig. 11 shows visual outputs from our approach compared to other methods. The comparison methods fail to handle the substantial amount of events generated by the background scene. In this challenging scenario, however, our method estimates reasonably accurate 3D poses.

5.3 Ablation Study

We next perform an ablation study to systematically evaluate the contributions of the core modules of our method as shown in Tab. 5.

	Seg. D	FB	Conf. D	\mathcal{L}_{J2D}	\mathcal{L}_{BA}	VM	MPJPE	PA-MPJPE
(I)							111.01	85.58
(II)	✓						108.85	84.98
(III)	✓		✓				107.58	83.95
(IV)	✓	✓	✓	✓			107.30	79.66
(V)	✓	✓	✓	✓	✓		106.50	77.93
(VI)	✓	✓	✓	✓	✓	✓	104.73	75.79
(VII)	✓	✓	✓	✓	✓	✓	102.15	75.48

Table 5: Ablation study of our approach.

Seg. D (segmentation decoder), FB (frame buffer), Conf. D (confidence decoder), \mathcal{L}_{J2D} (2D reprojection loss), \mathcal{L}_{BA} (bone loss) and VM (visibility mask). We report the MPJPE and PA-MPJPE evaluated on the EE3D-R dataset. The first row (I) represents the baseline that includes only the egocentric pose module (EPM).

In Tab. 5, we first define our baseline method by the Egocentric Pose Module (EPM) without the REPM (I). We next systematically examine the impact of the REPM. Adding the segmentation decoder to the baseline (II) improves the performance by 2% in the MPJPE. Incorporating the frame buffer along with the segmentation decoder (III) enables past events

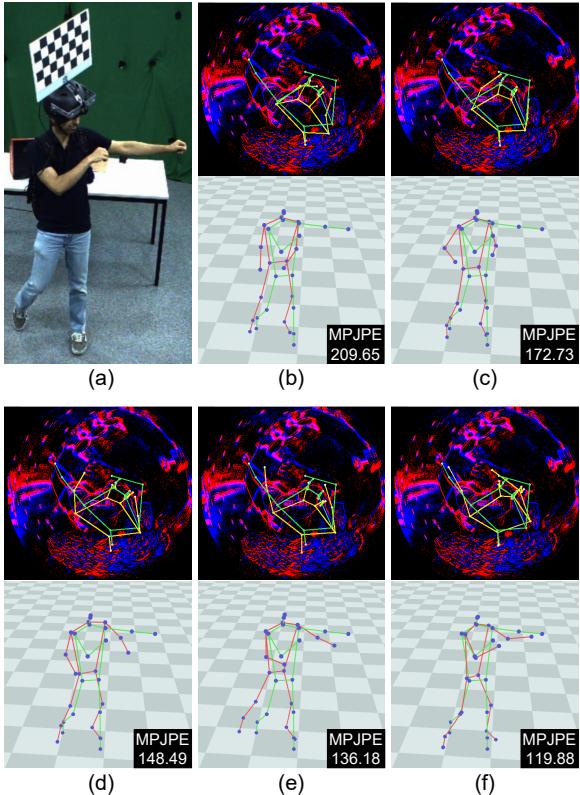


Fig. 13: Qualitative ablation study of our approach on EE3D-R. (a) Reference RGB view, (b) baseline (EPM only), (c) inclusion of segmentation decoder (Seg. D), (d) inclusion of frame buffer (FB) with Seg. D, (e) inclusion of confidence decoder (Conf. D) with FB and Seg. D, (f) inclusion of 2D reprojection, bone losses, and the visibility mask in (e). The MPJPE values are shown in the figures. 3D pose predictions and ground-truth poses are visualised in red and green, respectively. The 2D reprojection of the predicted 3D joints is shown in yellow.

to propagate to the current frame, resulting in a further 1% improvement in MPJPE. Additionally, introducing the confidence decoder (IV) significantly enhances performance, *e.g.* by 5% in the PA-MPJPE. These results validate the effectiveness of each component in REPM.

We also introduce a 2D reprojection loss (V) to refine the alignment of predicted 3D poses with the observed 2D event streams, yielding an additional 1% improvement in MPJPE and a 2% improvement in PA-MPJPE.

The integration of bone loss (VI) and visibility mask (VII) further improves our method’s accuracy. Specifically, incorporating the bone loss (VI) ensures anatomically plausible bone orientations and lengths, resulting in an additional 1% improvement in MPJPE. Furthermore, applying the visibility mask (full model) excludes occluded or out-of-view joints from 3D and 2D joint supervision. This prevents the model from directly learning the positions of these invisible joints. Instead, the model estimates their positions based on bone orientations and lengths. This approach enables more accurate pose predictions by leveraging the spatial relationships between joints and bones even in cases of occlusion or partial views. By integrating these losses, our method achieves the best MPJPE and PA-MPJPE scores, with improvements of over 8% and 11%, respectively, compared to the baseline.

Dataset	Config.	\mathcal{L}_{J2D}	\mathcal{L}_{BA}	MPJPE	PA-MPJPE
EE3D-S	(A)			100.80	74.63
	(B)	✓		101.27	72.76
	(C)	✓	✓	98.67	68.89
EE3D-W	(A)			177.28	100.84
	(B)	✓		174.30	95.76
	(C)	✓	✓	166.19	93.07

Table 6: Ablation study of additional losses. \mathcal{L}_{J2D} (2D reprojection loss) and \mathcal{L}_{BA} (bone loss). We report the MPJPE and PA-MPJPE evaluated on the EE3D-S and EE3D-W datasets with the visibility masks enabled.

To validate these findings across datasets, we evaluate the 2D reprojection and bone loss terms on both EE3D-S and EE3D-W in Tab. 6. Let (A) represent the model without additional losses. Adding the 2D reprojection loss (B) consistently reduces errors by a few percentage points on both datasets, indicating that enforcing tight alignment between estimated 3D poses and the 2D projections helps refine pose predictions. Furthermore, adding bone loss supervision (C) yields additional improvements in MPJPE, with a larger reduction of 5% observed in EE3D-W. This greater improvement is likely due to the more frequent and severe occlusions in the in-the-wild

dataset (see Fig. 7). By combining bone loss with the other supervisory signals, the model more effectively recovers joint positions by utilising information from nearby visible joints. This enables the inference of anatomically consistent poses, even in scenarios where parts of the human body are occluded.

Configuration	MPJPE	PA-MPJPE
Without Augmentation	105.62	78.74
With Augmentation	102.15	75.48

Table 7: Comparison of our approach with and without event augmentation. Lower values indicate better performance.

We also examine the impact of event augmentations during pretraining on EE3D-S, as shown in Tab. 7. Disabling these augmentations degrades generalisation performance on EE3D-R, resulting in a 3% increase in MPJPE. This result highlights the importance of event augmentation in capturing the variability of real-world event noise and preventing the model from overfitting to the training data’s limited noise patterns.

Finally, we present a hyperparameter tuning study in Tab. 8, where we vary each loss term’s weight by up to a factor of 10. Our method exhibits minimal sensitivity to these changes: on average, the MPJPE varies by approximately 1 mm, suggesting that the contribution of each term remains stable over a broad range of loss weightings.

We also provide qualitative ablation studies on the core modules of our approach in Fig. 13, Fig. 14, and Fig. 15. From Fig. 13, we observe that the baseline (b) is highly susceptible to noisy events. This significantly affects the network outputs, especially in the hand pose with a very high MPJPE value. Although this issue can be mitigated by adding the segmentation decoder (c) to some extent, it still struggles to estimate the correct hand position. The introduction of Frame Buffer (d) results in a significant performance improvement because it can utilise residual events from the previous frame weighted by the human body mask. Moreover, the additional inclusion of the confidence decoder

(e) further improves the visual quality of pose estimation. Finally, supervising our framework with the 2D reprojection loss, bone loss and visibility masks (f) plays a key role in producing the best visual outputs.

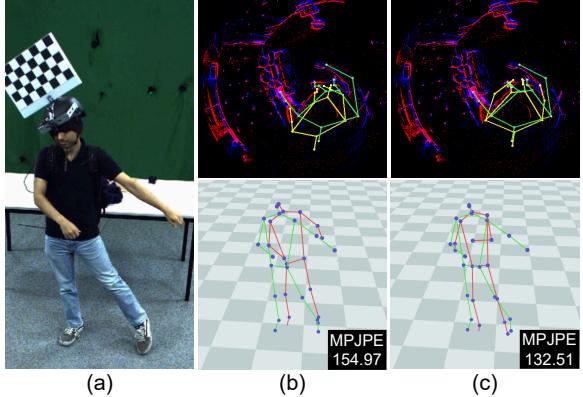


Fig. 14: Qualitative ablation study of 2D reprojection loss on EE3D-R. (a) Reference RGB view, (b) our model without the loss, (c) inclusion of 2D reprojection loss. The MPJPE values are shown in the figures. 3D pose predictions and ground-truth poses are visualised in red and green, respectively. The 2D reprojection of the predicted 3D joints is shown in yellow.

In Fig. 14, we visually examine the impact of the 2D reprojection loss (c) in a more challenging motion, such as dancing. Similarly, in Fig. 15, we analyse the influence of bone loss (c) and visibility masks (d) in another demanding motion, namely crawling. Despite significant occlusions from the egocentric views, the proposed components enable accurate estimation of human body poses and demonstrate their effectiveness in handling complex scenarios.

5.4 Runtime Performance

EventEgo3D++ and EventEgo3D (Millerdurai et al, 2024a) support real-time 3D human pose update rates of 140Hz. From Tab. 9, we see that both methods has the lowest number of parameters and floating point operations (FLOPs) compared to the competing methods. Rudnev et al (2021) is the fastest approach and the third-best in terms of 3D accuracy. We achieve the second-highest number of pose updates per

	λ_{J3D}	λ_{H}	λ_{seg}	\mathcal{L}_{J2D}	λ_{θ}	\mathcal{L}_{BL}	Weights	MPJPE	PA-MPJPE
(II)	✓						0.01 (current)	112.29	86.39
							0.1 (10x)	112.50	86.26
(III)	✓	✓					20 (current)	109.67	78.15
							200 (10x)	110.04	77.96
(IV)	✓	✓	✓				0.1 (current)	108.48	78.98
							1 (10x)	108.16	77.99
(V)	✓	✓	✓	✓			0.01 (current)	106.31	80.15
							0.1 (10x)	107.20	79.99
(VI)	✓	✓	✓	✓	✓		0.001 (current)	102.83	76.04
							0.01 (10x)	103.34	76.51
(VII)	✓	✓	✓	✓	✓	✓	0.001 (current)	102.15	75.48
							0.01 (10x)	102.50	76.15

Table 8: Ablation study of loss hyperparameters. λ_{J3D} (3D joint loss), λ_{H} (heatmap loss), λ_{seg} (segmentation loss), λ_{J2D} (2D reprojection loss), λ_{θ} (bone orientation loss) and λ_{BL} (bone length Loss). ✓ highlights the loss being ablated, while ✓ indicates the other losses enabled with their respective “current” weights. We report the MPJPE and PA-MPJPE evaluated on the EE3D-R dataset.

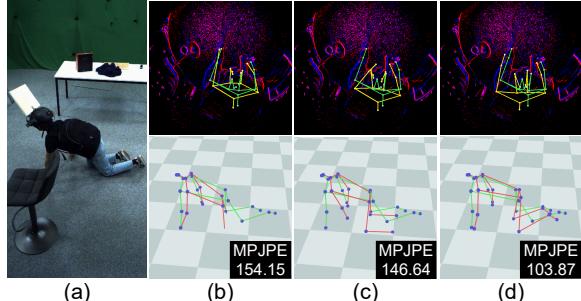


Fig. 15: Qualitative ablation study of bone loss and visibility mask on EE3D-R. (a) Reference RGB view, (b) our model without bone loss and visibility mask, (c) with bone loss, (d) with both bone loss and visibility mask. MPJPE values are displayed. Predicted 3D poses are in red, ground-truth poses are in green, and 2D reprojections are in yellow.

second. This result highlights that our approach is well-suited for mobile devices due to its low memory and computational requirements as well as its low power consumption, due to the event

camera. Since Rudnev et al (2021) use direct regression of 3D joints, their method is faster, while all other methods use heatmaps as an intermediate representation to estimate the 3D joints. Furthermore, the operations by Rudnev et al (2021) are well parallelisable, which explains its high pose update rate. Meanwhile, Xu et al (2019) and Tome et al (2019) are not designed for event streams and achieve lower 3D accuracy.

5.5 Real-time Demo

Event cameras provide high temporal event resolution and can operate under low-lighting conditions due to their excellent high dynamic range properties. EventEgo3D++ runs at real-time 3D pose update rates, and we design a real-time demo setup; see Fig. 1-(b) with a third-person view. Our portable HMD enables a wide range of movements, and the on-device computing laptop housed in the backpack allows us to capture in-the-wild sequences.

We showcase two challenging scenarios, *i.e.* with fast motions and in a poorly lit

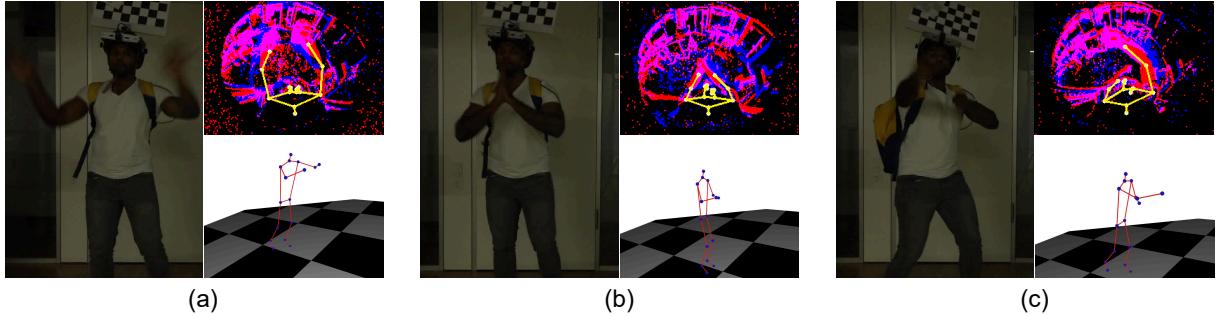


Fig. 16: Qualitative results of our method on in-the-wild motion sequences. (a) Waving, (b) Clapping and (c) Boxing. Our method accurately regresses 3D poses even in low-light conditions. Although the RGB stream experiences significant motion blur due to the fast movement of hands as seen in (a) and (c), our approach effectively utilises the event stream to capture the 3D poses.

Method	Params	FLOPs	Pose Update Rate
Tome et al (2019)	77.01M	11.46G	77.07
Xu et al (2019)	82.18M	44.06G	68.65
Rudnev et al (2021)	11.2M	3.58G	489.56
Millerdurai et al (2024a)	1.25M	416.84M	139.88
EventEgo3D++ (Ours)	1.25M	416.84M	139.88

Table 9: Comparisons of model efficiency: number of parameters, FLOPs, and runtime (pose update rate). EventEgo3D (Millerdurai et al., 2024a) and EventEgo3D++ (Ours) maintain the same number of parameters and FLOPs, achieving the lowest values in both metrics while still maintaining a good pose update rate. The enhancements in EventEgo3D++ improve accuracy without increasing complexity, refining the EventEgo3D framework.

environment that would lead to increased exposure time and motion blur in images captured by mainstream RGB cameras. Fig. 16 illustrates some of the challenging motions performed during the demo, highlighting that our method accurately estimates 3D poses for each motion. Notably, in Fig. 16-(a), a fast-paced waving motion is depicted, and our method successfully recovers the 3D poses in this dynamic scenario.

6 Limitations

EventEgo3D++ achieves substantial progress in event-based egocentric pose estimation, particularly in challenging scenarios involving fast

motion or low-light conditions, where it surpasses traditional RGB-based methods by producing more robust pose estimates. Nevertheless, several factors constrain the theoretical "upper bound" of an event-only approach. First, event cameras detect changes in brightness rather than absolute intensities. This can cause jitter in the estimated poses when subtle shifts in clothing generate unexpected events, but this is a less pronounced issue in RGB-based methods. Second, despite the inherent advantages of event cameras, sensor noise, spurious events, or environmental artefacts (*e.g.* flickering lights) can degrade performance. Finally, while our REPM module mitigates the effects of minimal motion by aggregating events, *extended* periods of little or no user movement yield fewer events, allowing sensor noise to dominate and destabilise pose estimates.

Furthermore, our framework employs Locally-Normalised Event Surfaces (LNES; Sec. 3.1) to convert the event stream into a 2D representation. This step may introduce uncertainties when multiple events triggered at the same pixel location within a time window overwrite each other, potentially discarding valuable spatiotemporal details. Alternative methods, such as those proposed by Chen et al (2022) and Millerdurai et al (2024b), aim to preserve the event stream's spatiotemporal representation and could enhance the performance of event-based systems. Nonetheless, it is important to note that these methods have been developed for static event cameras. When transitioning to *moving event cameras*, new challenges arise, particularly the significant increase in the number of events

generated from the background. While event sampling strategies offer a potential solution to this issue, the effectiveness of importance sampling specifically targeting events generated by the human body remains an unexplored area. Addressing this challenge could present a promising direction for future research in event-based pose estimation using egocentric cameras.

7 Conclusion

In this work, we present *EventEgo3D++*, an enhanced framework for egocentric 3D human motion capture from event cameras. Building upon the existing EventEgo3D framework, EventEgo3D++ introduces additional loss functions and a new in-the-wild dataset (EE3D-W). We have further expanded our datasets (EE3D-S, EE3D-R, and EE3D-W) by incorporating parametric human models, as well as allocentric multi-view RGB recordings for the EE3D-R and EE3D-W datasets. This expanded and diverse dataset provides a comprehensive resource to support and advance future research in the field. Experimental results demonstrate that EventEgo3D++ achieves state-of-the-art accuracy at real-time pose update rates, excelling in scenarios involving rapid motions and low-light conditions—areas where egocentric event sensing proves particularly advantageous. Our method effectively handles sparse and noisy event inputs, maintaining robust performance across a wide range of challenging conditions. These findings highlight the potential of event-based cameras for egocentric 3D vision tasks and pave the way for future research in areas such as motion analysis, action recognition, and human-computer interaction.

Acknowledgement. This research has been partially funded by the ERC Consolidator Grant 4DReply (GA Nr. 770784) and the EU project FLUENTLY (GA Nr. 101058680). Hiroyasu Akada is also supported by the Nakajima Foundation.

Data Availability. The datasets used in this paper—EE3D-R, EE3D-W, and EE3D-S—are publicly available and can be accessed from the project page at <https://eventego3d.mpi-inf.mpg.de>.

Appendix A Efficiency of Event Cameras

We evaluate the efficiency of event cameras along two dimensions: (1) the power consumption of our HMD equipped with an event camera, and (2) the bandwidth required to transmit event data over a fixed time window T .

Energy Efficiency of Event Cameras. We measure the power draw of the HMD using a precision USB power analyser to record watts (W) and milliamperes (mA). On average, the device consumes ~ 0.25 W (~ 50 mA), notably lower than typical RGB cameras that often exceed 1 W. Furthermore, no significant variation in power usage is observed between stationary and fast-motion scenarios, whether indoors or outdoors. This stability, despite rapid head movements or dynamic backgrounds, highlights the suitability of event cameras for continuous, real-time egocentric applications.

Event Camera Bandwidth Requirements. We measure the bandwidth consumption on a representative EE3D-W sequence ($S2$), featuring outdoor, in-the-wild conditions that generate a large number of events from both the wearer’s body and the background. Fig. A1 plots the per-frame bandwidth usage for this sequence, showing an average of approximately $6.6 \cdot 10^5$ bytes per frame. Each event is a 13-byte tuple (x, y, t_s, p) , where x and y each require 4 bytes, t_s requires 8 bytes, and p requires 1 byte. These events are accumulated over a time window $T = 16.66$ ms, matching the 60 fps rate of the allocentric RGB cameras. By comparison, an RGB frame at 1920×1080 encodes each pixel in 3 bytes (RGB), resulting in $1920 \times 1080 \times 3 \approx 6.22 \cdot 10^6$ bytes per frame—about 9.4 times higher than our event-stream data. Even at a lower resolution of 640×480 , which matches our event camera, RGB data requires about 1.39 times more bandwidth than the event stream.

Appendix B 2D Joint Heatmap Estimation

We estimate 2D joint heatmaps using the Heatmap Decoder. We produce heatmaps at

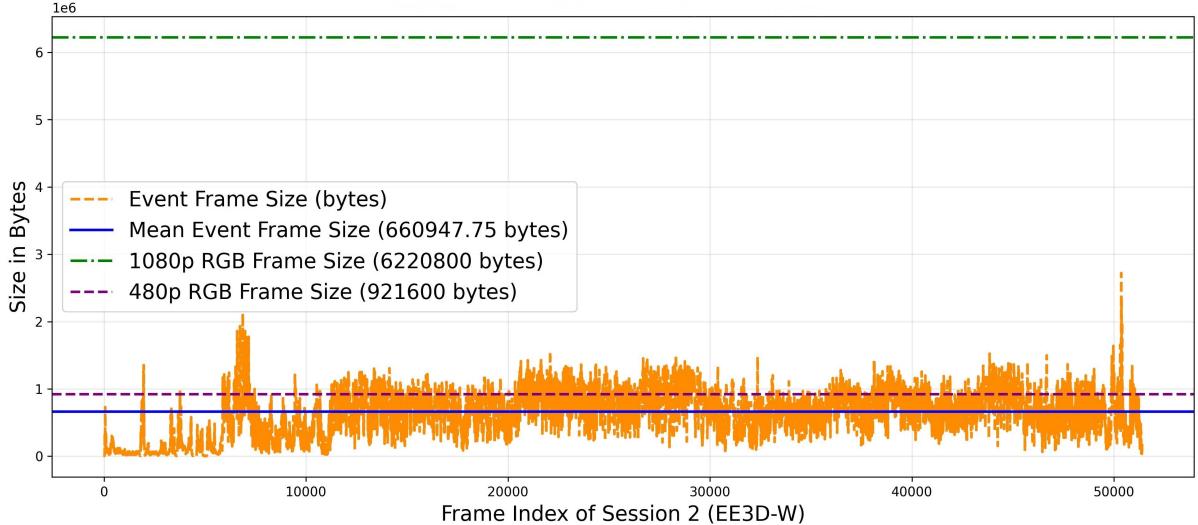


Fig. A1: Bandwidth comparison between event streams and RGB frames.

different resolutions from the layers of the decoder. Specifically, we utilise layers 2, 3, 4, and 5, extracting the first 16 feature maps from each layer. Each feature map corresponds to a heatmap for each body joint.

These heatmaps are then upsampled to a common resolution of 48×64 . After upsampling, we average the heatmaps from all the selected layers to produce the final heatmaps $\hat{\mathbf{H}}_q \in \mathbb{R}^{48 \times 64 \times 16}$, which represent the 2D joint heatmaps for the body joints.

Appendix C Real World Data Capture

C.1 Head Mounted Device Calibration

To obtain the ground-truth pose of the HMD user, we first calibrate the HMD using an allocentric RGB multi-camera setup. This calibration allows us to determine the HMD’s position in the multi-camera setup’s coordinate frame *i.e.* the world coordinate frame. Finally, we compute the world-to-device transformation matrix, denoted by \mathbf{M}_{WE} , which maps the world coordinate frame to the HMD coordinate frame. This lets us obtain the user’s 3D pose within the HMD’s coordinate system.

The position of the HMD in the world coordinate frame is obtained through hand-eye

calibration, following the approach of Rhodin et al (2016). In this process, a checkerboard, referred to as the “head-checkerboard,” is mounted on top of the HMD. This checkerboard is a surrogate for the event camera’s position, enabling precise tracking of the HMD within the world coordinate system. We compute the \mathbf{M}_{WE} matrix in two steps. First, we obtain the transformation from the world to the head-checkerboard coordinate frame, denoted as \mathbf{M}_{WC} . Next, we calculate the transformation from the head-checkerboard to the event camera, denoted by \mathbf{M}_{CE} . Specifically, \mathbf{M}_{WE} is defined as:

$$\mathbf{M}_{WE} = (\mathbf{M}_{CE} \cdot \mathbf{M}_{WC}) \quad (C1)$$

The \mathbf{M}_{WC} matrix is obtained by solving the pose of the head-checkerboard in the world coordinate frame. We apply the PnP algorithm (Itseez, 2015) on the images obtained from the multi-view RGB setup for the pose computation. Meanwhile, the \mathbf{M}_{CE} matrix is obtained through the following steps:

- Generate a checkerboard image using the event camera: we first capture an event stream of a checkerboard placed at the bottom HMD, referred to as the “floor-checkerboard,” while keeping the HMD stationary. To create a uniform distribution of events in both vertical and horizontal directions, the checkerboard is slid diagonally. The captured event stream is then converted into image sequences using

E2VID (Rebecq et al, 2019a). From these sequences, we select the image that captures the last position of the floor-checkerboard after the slide. Finally, we compute its pose, \mathbf{M}_E , in the HMD coordinate system using the PnP algorithm. A visualisation is shown in Fig. C2-(c).

- While maintaining the positions of both the floor-checkerboard and the HMD from the previous step, we use an external RGB camera to capture an image sequence that includes both the head-checkerboard and floor-checkerboard. We then select the images where the calibration patterns for both checkerboards are detected. With these selected images, we compute the poses of the head-checkerboard (\mathbf{M}_H) and floor-checkerboard (\mathbf{M}_F) relative to the external RGB camera using the PnP algorithm.
- Finally, the \mathbf{M}_{CE} matrix is obtained through the following transformation:

$$\mathbf{M}_{CE} = \mathbf{M}_E \cdot \mathbf{M}_F^{-1} \cdot \mathbf{M}_H. \quad (\text{C2})$$

A visualisation of the calibrated setup is shown in Fig. C2-(d).

C.2 Accuracy of Ground Truth

We acquire 3D human poses and SMPL (Loper et al, 2015) parameters using two multi-view motion capture pipelines: [Captury](#) (2024) for accurate 3D joints and [EasyMoCap](#) (2021) for SMPL parameter recovery.

EE3D-R Dataset. Captured with a state-of-the-art commercial system ([Captury](#), 2024) at 50 fps under high illumination, EE3D-R uses 30 cameras to minimise motion blur and maximise tracking accuracy. This setup aligns with prior literature on multi-view pose capture (Xu et al, 2020; Wang et al, 2021, 2022a, 2023, 2024b; Akada et al, 2024; Wang et al, 2024a; Millerdurai et al, 2024b) and ensures robust 3D reference poses.

EE3D-W Dataset. In contrast, EE3D-W is filmed at 60 fps using 6 cameras in outdoor settings, leveraging the same [Captury](#) (2024) technology. Although fewer cameras are employed, the system remains sufficient for accurate ground-truth capture, following best practices used in prior works for outdoor

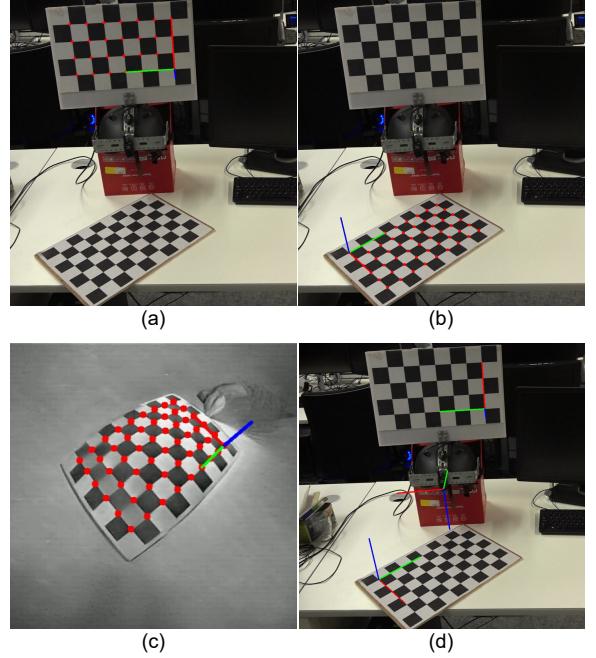


Fig. C2: Hand-eye calibration for determining event camera position relative to checkerboard on head-mounted device (a) The coordinate frame of the head-checkerboard is obtained using the external RGB camera. (b) The coordinate frame of the floor-checkerboard is obtained using the external RGB camera. (c) The coordinate frame of the floor-checkerboard is obtained using the event camera. (d) After hand-eye calibration is performed, the event camera is localised with respect to the head-checkerboard.

environments (Elhayek et al, 2016; Mehta et al, 2018; Xu et al, 2019).

In both datasets, each event in the egocentric event stream is synchronised with the allocentric RGB frames up to the frame’s timestamp. Together, EE3D-R and EE3D-W provide diverse, well-calibrated benchmarks, facilitating robust evaluations of egocentric 3D human pose estimation.

C.3 Ground Truth Generation

We obtain the 3D human poses and SMPL (Loper et al, 2015) parameters within the world coordinate frame using the multi-view RGB camera setup (see Fig. C3). Subsequently, we apply the world-to-device transformation matrix

$\mathbf{M}_{\text{WE}} \in \mathbb{R}^{4 \times 4}$ to convert these 3D human poses and SMPL parameters from the world coordinate frame to the HMD coordinate frame. Specifically, we use the following transformations:

$$\mathbf{J} = \mathbf{M}_{\text{WE}} \cdot \mathbf{G} \quad (\text{C3})$$

$$\mathbf{S}_{\mathbf{E}} = \mathbf{M}_{\text{WE}} \cdot \mathbf{S}_{\mathbf{W}} \quad (\text{C4})$$

Here, $\mathbf{G} \in \mathbb{R}^{16 \times 3}$ represents the world 3D human pose, $\mathbf{J} \in \mathbb{R}^{16 \times 3}$ represents the egocentric 3D human pose, $\mathbf{S}_{\mathbf{W}} \in \mathbb{R}^{6890 \times 3}$ is the world SMPL mesh and $\mathbf{S}_{\mathbf{E}} \in \mathbb{R}^{6890 \times 3}$ is the egocentric SMPL mesh. Additionally, we derive the 2D egocentric joint coordinates, represented as $\mathbf{J}_{\text{2D}} \in \mathbb{R}^{16 \times 2}$, by projecting the egocentric 3D poses using the intrinsics of the event camera.



Fig. C3: Visualisation of the calibrated HMD and 3D human body pose. We employ a multi-view camera setup to simultaneously track the 3D human body pose and the position of a checkerboard in the world coordinate frame. The 3D poses obtained are subsequently projected onto the coordinate frame of the HMD. To establish the coordinate frame of the HMD, we determine a suitable transformation matrix that maps points from the checkerboard’s coordinate frame to the HMD’s coordinate frame. Given the known position of the checkerboard, this transformation matrix allows us to derive the egocentric 3D pose.

Also, we generate human body masks and visibility masks for each joint, in addition to obtaining the 3D human poses and SMPL parameters. The joint visibility mask $V \in \{0, 1\}$ indicates whether a joint is visible or occluded from the egocentric view. We use [Blender \(2020\)](#) to create the human body masks and the joint

visibility masks. We first set up a SMPL body model of the user and an egocentric virtual camera with the same intrinsic parameters and position as our real-world event camera. To render the human body masks, we use Mist render layers in Blender’s Cycles renderer. Next, we obtain the joint visibility masks by shooting rays from the virtual camera to each 3D body joint. When a ray intersects with the SMPL body for the first time, we query the nearest vertices of the intersection. If the nearest vertices belong to the corresponding body part of the targeted 3D body joint, we mark that body joint as visible. Conversely, if the nearest vertices do not belong to the relevant body part, the 3D body joint is considered occluded. Additionally, if a 3D joint is occluded, we also mark the corresponding 2D joint as occluded. The body parts are identified using the predefined human part segmentation mesh provided by [Loper et al \(2015\)](#).

Appendix D Reconstructing Images from the Event Stream

We utilise E2VID [Rebecq et al \(2019b\)](#) to generate image reconstructions from the event stream. The frame duration (event window) is set to 20 ms to align with the ground-truth frame timing of EE3D-R. As shown in Fig. D4, the reconstructed images often exhibit artefacts, particularly in scenarios with minimal human motion. For instance, during low-motion actions such as walking (left part of Fig. D4), the reconstructed images fail to accurately capture the human figure. In contrast, during high-motion actions, such as punching (right part of Fig. D4), the reconstructed images can recover the human figure properly. To ensure precise synchronization, each event window is aligned with the corresponding ground-truth frame number, maintaining consistency between the ground-truth 3D poses and the reconstructed images.

Appendix E Synthetic Data Generation

To simulate human motions as captured by event cameras, we linearly interpolated SMPL body parameters from the SURREAL dataset at a

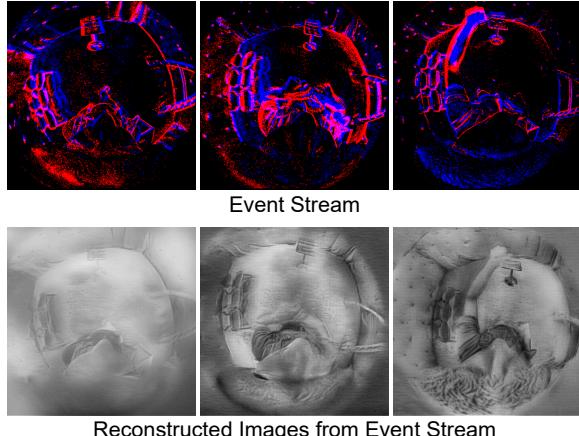


Fig. D4: Exemplary event streams and their corresponding image reconstructions. The reconstructed images lose significant details of the human body, especially when the motion of the human is minimal.

frequency of 480Hz. The dataset is created by generating RGB frames and human body masks through the Image and Mist render layers in Blender’s Cycles renderer (Blender, 2020). The 3D body joints used for training our EventEgo3D++ method, denoted as $\mathbf{J} = \{\mathbf{J}_1, \dots, \mathbf{J}_N\}$, where $\mathbf{J} \in \mathbb{R}^{16 \times 3}$, are derived from the SMPL body joints represented by $\mathbf{S} = \{\mathbf{S}_1, \dots, \mathbf{S}_N\}$, where $\mathbf{S} \in \mathbb{R}^{45 \times 3}$. Specifically, we map the joints as follows:

$$\mathbf{G} = \{\mathbf{S}_{16}, \mathbf{S}_{13}, \mathbf{S}_{18}, \mathbf{S}_{20}, \mathbf{S}_{22}, \mathbf{S}_{17}, \mathbf{S}_{19}, \mathbf{S}_{21}, \mathbf{S}_3, \mathbf{S}_6, \mathbf{S}_9, \mathbf{S}_{12}, \mathbf{S}_2, \mathbf{S}_5, \mathbf{S}_8, \mathbf{S}_{11}\},$$

where \mathbf{S}_i denotes the i -th SMPL joint index from the set $\{1, 2, \dots, 45\}$. Each joint in \mathbf{J} corresponds to a specific body part: the head, neck, right shoulder, right elbow, right wrist, left shoulder, left elbow, left wrist, right hip, right knee, right ankle, right foot, left hip, left knee, left ankle, and left foot, respectively.

Appendix F Input representation

We use the LNES representation (Rudnev et al, 2021) to aggregate events over a time window without applying any temporal overlap. Nevertheless, we have conducted experiments using explicitly overlapping LNES frames with a

7 ms temporal resolution matching our network’s runtime performance of 140 fps—which yields an MPJPE of 102.26 and a PA-MPJPE of 75.62 on the EE3D-R dataset. These results are nearly identical to our default setting using non-overlapping LNES frames (MPJPE: 102.15, PA-MPJPE: 75.48).

Furthermore, when using an overlapping configuration with a 1 ms temporal resolution, we obtain an MPJPE of 100.54 and a PA-MPJPE of 73.97—corresponding to a 1.58% reduction in MPJPE and a 2.00% reduction in PA-MPJPE compared to our default configuration. However, since our network can only process frames at an effective rate of approximately 7 ms per frame ($1000/140 \approx 7$ ms), this 1 ms configuration is not feasible for real-time operation.

References

- Akada H, Wang J, Shimada S, et al (2022) Unrealego: A new dataset for robust egocentric 3d human motion capture. In: European Conference on Computer Vision (ECCV)
- Akada H, Wang J, Golyanik V, et al (2024) 3d human pose perception from egocentric stereo videos. In: Computer Vision and Pattern Recognition (CVPR)
- Aliakbarian S, Cameron P, Bogo F, et al (2022) Flag: Flow-based 3d avatar generation from sparse observations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 13253–13262
- Bazarevsky V, Grishchenko I, Raveendran K, et al (2020) Blazepose: On-device real-time body pose tracking. arXiv preprint arXiv:200610204
- Blender (2020) Blender - a 3D modelling and rendering package. Blender Foundation, Blender Institute, Amsterdam, URL <http://www.blender.org>
- Captury (2024) Capturstudio - markerless mocap of humans from pre-recorded, multi-view video footage. <http://www.thecaptury.com/>
- Chen J, Shi H, Ye Y, et al (2022) Efficient human pose estimation via 3d event point cloud. In: International Conference on 3D Vision (3DV)

- CMU (2006) Cmu graphics lab motion capture database. URL <http://mocap.cs.cmu.edu/>
- Dai P, Zhang Y, Liu T, et al (2024) Hmd-poser: On-device real-time human motion tracking from scalable sparse observations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 874–884
- Du Y, Kips R, Pumarola A, et al (2023) Avatars grow legs: Generating smooth human motion from sparse tracking inputs with diffusion model. In: CVPR
- DVXplorer Mini (2021) DVXplorer mini specification. <https://netsket.kr/img/custom/board/DVXplorer-Mini.pdf>
- EasyMoCap (2021) Easymocap - make human motion capture easier. <https://github.com/zju3dv/EasyMocap>
- Elhayek A, de Aguiar E, Jain A, et al (2016) Marconi—convnet-based marker-less motion capture in outdoor and indoor scenes. IEEE transactions on pattern analysis and machine intelligence 39(3):501–514
- Gallego G, Delbrück T, Orchard G, et al (2020) Event-based vision: A survey. IEEE transactions on pattern analysis and machine intelligence 44(1):154–180
- Gehrig D, Gehrig M, Hidalgo-Carrió J, et al (2020) Video to events: Recycling video datasets for event cameras. In: Computer Vision and Pattern Recognition (CVPR)
- Gilbert A, Trumble M, Malleson C, et al (2019) Fusing visual and inertial sensors with semantics for 3d human pose estimation. International Journal of Computer Vision 127:381–397
- Guzov V, Mir A, Sattler T, et al (2021) Human poseitioning system (hps): 3d human pose estimation and self-localization in large scenes from body-mounted sensors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 4318–4329
- Guzov V, Jiang Y, Hong F, et al (2024) Hmd²: Environment-aware motion generation from single egocentric head-mounted device. arXiv preprint arXiv:240913426
- Helten T, Muller M, Seidel HP, et al (2013) Real-time body tracking with one depth camera and inertial sensors. In: Proceedings of the IEEE international conference on computer vision, pp 1105–1112
- Huang Y, Kaufmann M, Aksan E, et al (2018) Deep inertial poser: Learning to reconstruct human pose from sparse inertial measurements in real time. ACM Transactions on Graphics (TOG) 37(6):1–15
- Itseez (2015) Open source computer vision library. <https://github.com/itseez/opencv>
- Jiang J, Streli P, Qiu H, et al (2022a) Avataposer: Articulated full-body pose tracking from sparse motion sensing. In: European conference on computer vision, Springer, pp 443–460
- Jiang J, Streli P, Meier M, et al (2023) Egoposer: Robust real-time ego-body pose estimation in large scenes. arXiv e-prints pp arXiv–2308
- Jiang J, Li J, Zhang B, et al (2024a) Evhandpose: Event-based 3d hand pose estimation with sparse supervision. IEEE Transactions on Pattern Analysis and Machine Intelligence
- Jiang J, Zhou X, Wang B, et al (2024b) Complementing event streams and rgb frames for hand mesh reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 24944–24954
- Jiang J, Streli P, Luo X, et al (2025) Manikin: biomechanically accurate neural inverse kinematics for human motion estimation. In: European Conference on Computer Vision, Springer, pp 128–146
- Jiang Y, Ye Y, Gopinath D, et al (2022b) Transformer inertial poser: Real-time human motion reconstruction from sparse imus with simultaneous terrain generation. In: SIGGRAPH Asia 2022 Conference Papers, pp

- Kang T, Lee Y (2024) Attention-propagation network for egocentric heatmap to 3d pose lifting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)
- Kang T, Lee K, Zhang J, et al (2023) Ego3dpose: Capturing 3d cues from binocular egocentric views. In: SIGGRAPH Asia 2023 Conference Papers. Association for Computing Machinery, New York, NY, USA, SA '23, <https://doi.org/10.1145/3610548.3618147>, URL <https://doi.org/10.1145/3610548.3618147>
- Kendall DG (1989) A survey of the statistical theory of shape. *Statistical Science* 4(2):87–99
- Khirodkar R, Bansal A, Ma L, et al (2023) Ego-humans: An ego-centric 3d multi-human benchmark. In: International Conference on Computer Vision (ICCV)
- Kingma D, Ba J (2015) Adam: A method for stochastic optimization. In: International Conference on Learning Representations (ICLR)
- Lan C, Yin Z, Basu A, et al (2023) Tracking fast by learning slow: An event-based speed adaptive hand tracker leveraging knowledge in rgb domain. arXiv preprint arXiv:230214430
- Lee S, Starke S, Ye Y, et al (2023) Questenvsim: Environment-aware simulated motion tracking from sparse sensors. In: ACM SIGGRAPH 2023 Conference Proceedings, pp 1–9
- Lensagon BF10M14522S118C (2020) Lensagon bf10m14522s118 specification. <https://www.lensation.de/pdf/BF10M14522S118.pdf>
- Li J, Liu K, Wu J (2023a) Ego-body pose estimation via ego-head pose estimation. In: Computer Vision and Pattern Recognition (CVPR)
- Li J, Liu K, Wu J (2023b) Ego-body pose estimation via ego-head pose estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 1–9
- Liu Y, Yang J, Gu X, et al (2023) Egofish3d: Egocentric 3d pose estimation from a fisheye camera via self-supervised learning. *IEEE Transactions on Multimedia*
- Loper M, Mahmood N, Romero J, et al (2015) SMPL: A skinned multi-person linear model. *ACM Trans Graphics (Proc SIGGRAPH Asia)* 34(6):248:1–248:16
- Luo Z, Hachiuma R, Yuan Y, et al (2021) Dynamics-regulated kinematic policy for egocentric pose estimation. *Advances in Neural Information Processing Systems (NeurIPS)*
- Malleson C, Gilbert A, Trumble M, et al (2017) Real-time full-body motion capture from video and imus. In: 2017 international conference on 3D vision (3DV), IEEE, pp 449–457
- MathWorks (2023) Matlab version: 9.14.0 (r2023a). URL <https://www.mathworks.com>
- Mehta D, Sotnychenko O, Mueller F, et al (2018) Single-shot multi-person 3d pose estimation from monocular rgb. In: 3D Vision (3DV), 2018 Sixth International Conference on, IEEE, URL <http://gvv.mpi-inf.mpg.de/projects/SingleShotMultiPerson>
- Millerdurai C, Akada H, Wang J, et al (2024a) Eventego3d: 3d human motion capture from egocentric event streams. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 1186–1195
- Millerdurai C, Luvizon D, Rudnev V, et al (2024b) 3d pose estimation of two interacting hands from a monocular event camera. In: International Conference on 3D Vision (3DV)
- Muglikar M, Gehrig M, Gehrig D, et al (2021) How to calibrate your event camera. In: Conference on Computer Vision and Pattern Recognition (CVPR) Workshops
- Nehvi J, Golyanik V, Mueller F, et al (2021) Differentiable event stream simulator for non-rigid 3d tracking. In: CVPR Workshop on Event-based Vision

- Pan X, Charron N, Yang Y, et al (2023) Aria digital twin: A new benchmark dataset for egocentric 3d machine perception. In: International Conference on Computer Vision (ICCV)
- Park J, Moon G, Xu W, et al (2024) 3d hand sequence recovery from real blurry images and event stream. In: European Conference on Computer Vision, Springer, pp 343–359
- Paszke A, Gross S, Massa F, et al (2019) Pytorch: An imperative style, high-performance deep learning library. Advances in Neural Information Processing Systems (NeurIPS)
- Pavlakos G, Zhu L, Zhou X, et al (2018) Learning to estimate 3d human pose and shape from a single color image. In: Computer Vision and Pattern Recognition (CVPR)
- Rebecq H, Gehrig D, Scaramuzza D (2018) Esim: an open event camera simulator. In: Conference on Robot Learning (CORL)
- Rebecq H, Ranftl R, Koltun V, et al (2019a) Events-to-video: Bringing modern computer vision to event cameras. Computer Vision and Pattern Recognition (CVPR)
- Rebecq H, Ranftl R, Koltun V, et al (2019b) High speed and high dynamic range video with an event camera. IEEE transactions on pattern analysis and machine intelligence 43(6):1964–1980
- Rhodin H, Richardt C, Casas D, et al (2016) Egocap: egocentric marker-less motion capture with two fisheye cameras. ACM Transactions on Graphics (TOG) 35(6):1–11
- Ronneberger O, Fischer P, Brox T (2015) U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention
- Rudnev V, Golyanik V, Wang J, et al (2021) Eventhands: Real-time neural 3d hand pose estimation from an event stream. In: International Conference on Computer Vision (ICCV)
- Rudnev V, Elgharib M, Theobalt C, et al (2023) Eventnerf: Neural radiance fields from a single colour event camera. In: Computer Vision and Pattern Recognition (CVPR)
- Scaramuzza D, Martinelli A, Siegwart R (2006) A toolbox for easily calibrating omnidirectional cameras. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
- Tome D, Peluse P, Agapito L, et al (2019) xr-egopose: Egocentric 3d human pose from an hmd camera. In: International Conference on Computer Vision (ICCV)
- Tome D, Alldieck T, Peluse P, et al (2020) Selfpose: 3d egocentric pose estimation from a headset mounted camera. IEEE Transactions on Pattern Analysis and Machine Intelligence 45(6):6794–6806
- Varol G, Romero J, Martin X, et al (2017) Learning from synthetic humans. In: Computer Vision and Pattern Recognition (CVPR)
- Von Marcard T, Pons-Moll G, Rosenhahn B (2016) Human pose estimation from video and imus. IEEE transactions on pattern analysis and machine intelligence 38(8):1533–1547
- Von Marcard T, Rosenhahn B, Black MJ, et al (2017) Sparse inertial poser: Automatic 3d human pose estimation from sparse imus. In: Computer graphics forum, Wiley Online Library, pp 349–360
- Wang J, Liu L, Xu W, et al (2021) Estimating egocentric 3d human pose in global space. In: International Conference on Computer Vision (ICCV)
- Wang J, Liu L, Xu W, et al (2022a) Estimating egocentric 3d human pose in the wild with external weak supervision. In: Computer Vision and Pattern Recognition (CVPR)
- Wang J, Luvizon D, Xu W, et al (2023) Scene-aware egocentric 3d human pose estimation. Computer Vision and Pattern Recognition (CVPR)

- Wang J, Cao Z, Luvizon D, et al (2024a) Egocentric whole-body motion capture with fisheyevit and diffusion-based motion refinement. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 777–787
- Wang J, Cao Z, Luvizon D, et al (2024b) Egocentric whole-body motion capture with fisheyevit and diffusion-based motion refinement. In: Computer Vision and Pattern Recognition (CVPR)
- Wang Z, Chaney K, Daniilidis K (2022b) Evac3d: From event-based apparent contours to 3d models via continuous visual hulls. In: European Conference on Computer Vision (ECCV)
- Winkler A, Won J, Ye Y (2022) Questsim: Human motion tracking from sparse sensors with simulated avatars. In: SIGGRAPH Asia 2022 Conference Papers, pp 1–8
- Xu L, Xu W, Golyanik V, et al (2020) Eventcap: Monocular 3d capture of high-speed human motions using an event camera. In: Computer Vision and Pattern Recognition (CVPR)
- Xu W, Chatterjee A, Zollhoefer M, et al (2019) Mo²Cap²: Real-time mobile 3d motion capture with a cap-mounted fisheye camera. *IEEE Transactions on Visualization and Computer Graphics* 25(5):2093–2101
- Xue Y, Li H, Leutenegger S, et al (2022) Event-based non-rigid reconstruction from contours. In: British Machine Vision Conference (BMVC)
- Yi X, Zhou Y, Xu F (2021) Transpose: Real-time 3d human translation and pose estimation with six inertial sensors. *ACM Transactions On Graphics (TOG)* 40(4):1–13
- Yi X, Zhou Y, Habermann M, et al (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, pp 13167–13178
- Yi X, Zhou Y, Habermann M, et al (2023) Egolocate: Real-time motion capture, localization, and mapping with sparse body-mounted sensors. *ACM Transactions on Graphics (TOG)* 42(4):1–17
- Yu F, Zhang Y, Song S, et al (2015) Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. arXiv preprint arXiv:150603365
- Yuan Y, Kitani K (2019) Ego-pose estimation and forecasting as real-time pd control. In: International Conference on Computer Vision (ICCV)
- Zhang S, Ma Q, Zhang Y, et al (2022) Egobody: Human body shape and motion of interacting people from head-mounted devices. In: European conference on computer vision (ECCV)
- Zhang S, Ma Q, Zhang Y, et al (2023) Probabilistic human mesh recovery in 3d scenes from egocentric views. In: International Conference on Computer Vision (ICCV)
- Zhang Y, You S, Gevers T (2021) Automatic calibration of the fisheye camera for egocentric 3d human pose estimation from a single image. In: Winter Conference on Applications of Computer Vision
- Zhao D, Wei Z, Mahmud J, et al (2021) Egoglass: Egocentric-view human pose estimation from an eyeglass frame. In: International Conference on 3D Vision (3DV)
- Zheng X, Su Z, Wen C, et al (2023) Realistic full-body tracking from sparse observations via joint-level modeling. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp 14678–14688
- Zou S, Guo C, Zuo X, et al (2021) Eventhpe: Event-based 3d human pose and shape estimation. In: International Conference on Computer Vision (ICCV)