Universiteit Leiden

Master Computer Science

EvHandFPP:
Efficient Event-Based 3D Hand Tracking from First-Person Perspective

Name:	Zhen XU
Student ID:	4037065
Date:	25/08/2025
Specialisation:	Advanced Computing and Systems
1st supervisor:	Dr. Qinyu Chen
2nd supervisor:	Dr.ir. D. Joost Broekens

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Als een stem binnenin je zegt ’je kunt niet schilderen’,

verf en deze stem zal tot zwijgen worden gebracht.

Vincent van Gogh

For my grandmothers

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my supervisor, Dr. Qinyu Chen, who patiently guided me into the world of academic research. It is often said that interest is the best teacher, and the conversations with Qinyu have consistently sparked interest and inspiration, so I have a truly insightful ”Teacher Teacher”.

I would also like to thank my friend and colleague, PhD Student Guorui Lu, for his support in both my research and academic writing. I learned a lot from his solid research experience and strong hands-on ability greatly throughout the research process, allowing me to reduce the inefficiencies of the “idle overhead” along the way.

And many thanks to my family and friends for their help throughout this journey. I am also grateful to my former schools and employers, where the knowledge and skills I acquired have been foundational to my current work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

μs microsecond

AR Augmented Reality

ASIC Application-Specific Integrated Circuit

DNN Deep Neural Network

FPGA Field-Programmable Gate Array

FPS frames per second

HCI Human-Computer Interaction

kHz kilohertz

LNES Locally-Normalised Event Surfaces

M million

ms millisecond

mW milliwatt

RoI Region of Interest

SOTA State-of-the-Art

ViT Vision Transformer

VR Virtual Reality

W watt

XR Extended Reality

ABSTRACT

Hand tracking holds great promise for advanced and intuitive interaction paradigms. However, conventional hand tracking approaches are limited by low temporal resolution and high power consumption, rendering them impractical for real-time applications in resource-constrained environments. In recent years, event-based vision has emerged as a compelling alternative in the hand tracking task, benefiting from its more bio-inspired mechanism of asynchronously capturing pixel-level brightness changes, and the consequent ability to efficiently generate high-speed data streams.

In this thesis, we propose a lightweight, real-time framework for 3D hand tracking from a first-person perspective using a monocular event camera. Our approach leverages an optimized event representation, a region of interest strategy, and a multi-task learning scheme for performance enhancement and cost reduction. To support this deep-learning-based approach, a synthetic event-based vision dataset, simulating hand movements in extended reality scenarios, is generated for training; and real-world event data is collected for evaluation. Experimental results demonstrate that our approach achieves better accuracy compared to prior state-of-the-art approaches, while reducing model parameters by 90% and computational loads by 89%.

CHAPTER 1

INTRODUCTION

1.1 Background

The hand tracking is primarily a vision-based task, which requires performing precise pose estimation of human hands with high degrees of freedom, as well as achieving real-time localization of rapid stochastic movements. Despite these challenges, its broad application potential consistently draws interest from the research community from early on [1, 2, 3, 4]. Notably, the development of deep learning has led to significant advances in hand tracking by improving the accuracy and generalization. As a result, hand tracking has found broader and more impactful applications in several emerging interdisciplinary domains including Human-Computer Interaction (HCI), Virtual Reality (VR) and Augmented Reality (AR), and robotics [5, 6, 7, 8]. **Figure 1.1** illustrates an example of hand tracking in VR/AR scenarios [8], where users' interactions with frequently used objects are collected and analyzed, highlighting its role in enhancing the natural and immersive user experience.

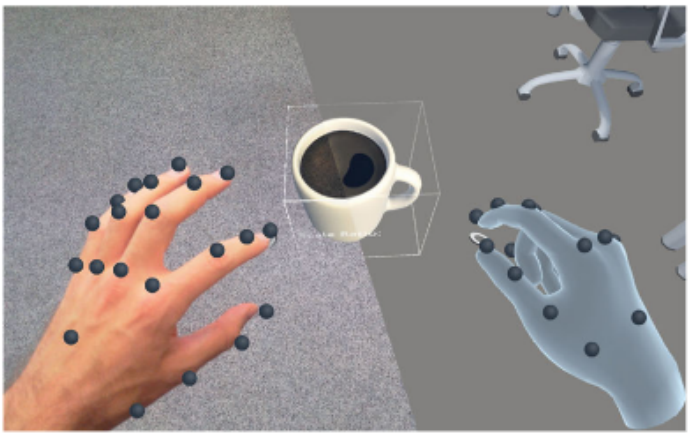

Figure 1.1: Hand Tracking Application: users interact with objects in VR/AR (adapted from [8]).

Prevailing approaches are based on traditional RGB or depth cameras, which acquire images at a fixed frame rate with no relation to the viewed scene transitions. This constant

full-frame acquisition leads to significantly high power consumption, particularly when depth or 3D prediction capabilities are required. Basic 2D cameras operating at only 25-30 frames per second (FPS) but require 200 milliwatt (mW) of power consumption, while 3D sensing systems operating at 30 FPS consume substantially higher power of 3-5 watt (W) [9]. And power reduction methods often come at the cost of lower frame rates, e.g. reducing power consumption by decreasing the depth acquisition frequency by 2 or even 3 times [10]. Under these limitations, such approaches struggle with fast object movements and incur high computational loads, making real-time hand tracking remain a persistent challenge, especially in resource-constrained environments.

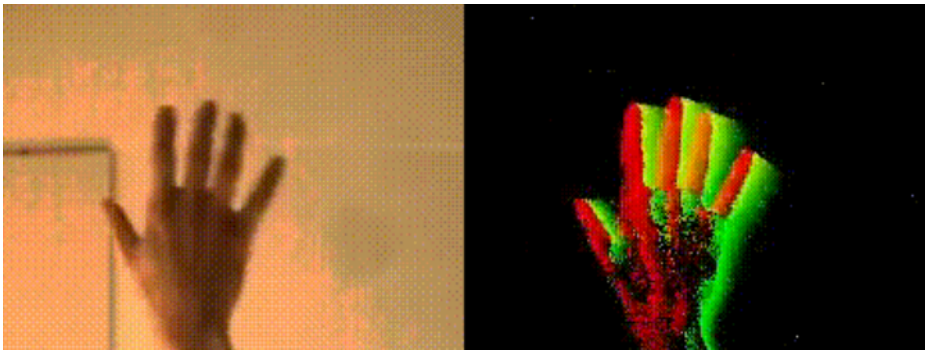

Figure 1.2: RGB Camera vs. Event Camera: 30 FPS vs. μs-level temporal resolution.

In contrast with RGB cameras, the bio-inspired silicon retinas, or event cameras, asynchronously report pixel-level brightness changes, generating a sparse stream of events with microsecond (μs) precision rather than dense images on a fixed clock [11, 12, 13]. This mechanism also enables low power consumption, often as low as 10 mW, by transmitting data only from pixels where motion is detected, rather than processing entire frames [14]. **Figure 1.2** presents the data we capture using a DAVIS346 camera [15] with 346 pixels in width and 260 pixels in height, which we cropped to 240 pixels wide and 180 pixels high for comparison with previous methods. The left panel shows an image of a hand from the RGB sensor with 30 FPS; while the right panel shows the corresponding event-based accumulating image with a time resolution of 1000 FPS, whose specific reconstruction method is detailed in section 4.1.

Prior studies [16, 17, 18, 19] have demonstrated that the high temporal resolution of event cameras enables high-speed and even real-time hand tracking. However, these approaches have not fully exploited the efficiency advantages of event cameras, and thus have failed to show their applicability to resource-constrained devices, like AR/VR headsets. For example, Deep Neural Network (DNN) models with over 10 million (M) parameters are generally adopted, posing significant burdens in such lightweight environments.

Additionally, hand tracking from the first-person perspective is particularly valuable for AR/VR applications, as this perspective is intuitive for direct manipulations with digital content, such as typing on virtual keyboards, tapping, or dragging virtual elements [20, 21]. However, to the best of our knowledge, research on event-based solutions for hand tracking is still at an early stage, and the available datasets are not yet comprehensive. Furthermore, our experiments, demonstrated in the following chapter, reveal that existing approaches do not generalize well to such conditions.

1.2 Contributions

To address the challenges mentioned above, we propose EvHandFPP, an efficient framework focusing on real-time 3D hand tracking from the first-person perspective, and construct the accompanying dataset EvHandFPP-Data.

In summary, our contributions are as follows:

- We construct an event-based hand-tracking dataset EvHandFPP-Data from a first-person perspective, including generated data with 3D labels for training and real data with 2D labels for testing. We make use of a simulator called evsim [17] to generate a synthetic event-based dataset specifically designed to simulate users' views of their own hand movements and interactions within Extended Reality (XR) applications. Labels of our real data are initially annotated using MediaPipe [22] and then manually refined.

- We propose a lightweight framework for real-time 3D hand tracking, with the potential for deployment on resource-limited systems, realized through our proposed event representation, multi-task learning strategy, and Region of Interest (RoI) technique.
  - Using the dataset, we train a model, addressing event data at 1 kilohertz (kHz), that surpasses the performance of the current State-of-the-Art (SOTA) [17], with a reduction in parameters of 90% and a 89% decrease in computation.
  - We propose LNES-Fast, a single-channel event representation that stacks images through time windows under event count limits, thereby reducing redundant information in event images.
  - We propose Wrist-Based Region of Interest, a technique that locates the primary hand position through wrist detection and extracts the RoI, serving as a key module for reducing computational overhead in the framework.
  - We design an auxiliary task based on hand features within the RoI region. This multi-task learning approach helps the network to focus on more useful information and improve the performance of the main task.

CHAPTER 2

RELATED WORK

2.1 3D Hand Reconstruction

Some prior works reconstruct 3D hands by directly learning the mapping from image space to hand pose space [5, 6, 7, 23]. These methods fully leverage the advantage of end-to-end deep learning, enabling efficient inference and demonstrating promising performance and robustness. However, the reconstructed hands from such approaches lack geometric constraints and require large-scale annotated datasets with sufficient diversity. Another class of methods employs predefined hand models as prior knowledge to ensure reconstruction results conform to the physiological hand structure, such as MANO [24] and SMPL [25].

Primary works on 3D hand reconstruction are based on RGB cameras [5, 6, 7, 23, 26, 27, 28, 29, 30, 31] or depth cameras [32, 33, 34, 35, 36, 37]. These cameras offer widespread availability and technological maturity, leading to robust algorithms and comprehensive datasets that enable accurate hand reconstruction under various conditions. However, these methods are limited by temporal resolution constraints and exhibit high power consumption at elevated frame rates, thereby struggling to handle fast-moving objects and perform effectively in resource-constrained environments.

Additionally, to the best of our knowledge, existing works and available datasets focus on general 3D hand reconstruction rather than specifically addressing the hand-back perspective. This mismatch affects the model's generalization in certain applications, e.g. XR. Furthermore, our subsequent experiments demonstrate that existing datasets exhibit poor generalization performance for hand-back reconstruction. Therefore, dedicated methods and datasets for the hand-back perspective are needed.

2.2 Event-Based Vision Techniques

Event-based vision, also known as neuromorphic vision, represents a paradigm shift from traditional frame-based imaging systems [14]. Unlike conventional RGB cameras that capture dense images at fixed intervals, event cameras are bio-inspired silicon retinas that asynchronously report pixel-level brightness changes, generating sparse streams of events with millisecond (ms) precision [11, 12, 13]. This fundamentally different sensing mechanism enables several key advantages: high temporal resolution with event detection at ms-level precision, low power consumption (often as low as 10 mW), and inherent motion blur reduction by transmitting data only from pixels where motion is detected rather than processing entire frames. The asynchronous nature of event cameras makes them particularly well-suited for dynamic scene understanding and high-speed motion capture applications. Each pixel independently responds to brightness changes, resulting in a temporal resolution that can reach up to 1 μs, which is orders of magnitude higher than conventional cameras. This capability effectively eliminates motion blur artifacts that plague traditional vision systems when tracking fast-moving objects.

Recent studies have successfully applied event cameras to 3D hand tracking, demonstrating performance comparable to RGB-based methods while achieving significantly higher frame rates [17, 18, 19]. However, these approaches have failed to fully leverage the low-resource advantages of event cameras, employing DNN models with parameter counts and computational demands that are incompatible with resource-constrained devices.

CHAPTER 3

DATASET CONSTRUCTION

3.1 Synthetic Data

The evsim [17] software can generate 3D right-hand models based on the MANO [24] model. The simulated event data and RGB data from a configured perspective, the 3D coordinates of key joints and their corresponding transformed 12-dimensional labels, detailed in subsection 4.2.1, are all saved with microsecond-precision timestamps. By modifying the camera viewpoint and scripting animations with randomness in gesture transformations, translations, and rotations, we generated in total 720,000 milliseconds of synthetic event camera data for training, and an additional 60,000 milliseconds of synthetic data for evaluation. As shown in **Figure 3.1**, the synthetic data contains timestamped animated (RGB) data, event data, joint coordinates (and the corresponding 12-dimensional labels).

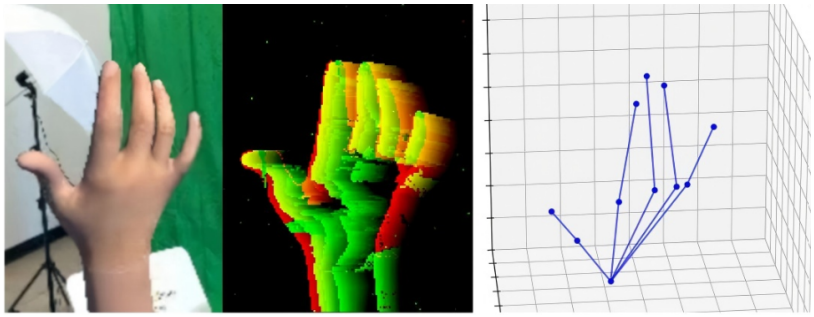

Figure 3.1: Synthetic Data: RGB frames, events and 3D joints

3.2 Real-World Data

We use a DAVIS346 camera to collect real-world data, comprising event data along with co-centered synchronized RGB data at 30 FPS. The duration of our real data reaches up to 60,000 ms, allowing for testing of the real-world performance stability. To ensure the gen-

eralizability of our method, the data are collected from subjects with distinct hand shapes and sizes, and the recorded hand movements encompass common motion patterns in target scenes, such as planar translation, depth movement, wrist rotation, and gesture variation. We use MediaPipe [22] to initially annotate the 2D coordinates of hand joints in RGB frames, and calibrate the labels manually. **Figure 3.2** shows the real-world hand data, including RGB data, event data, and the annotated coordinates of the hand joints.

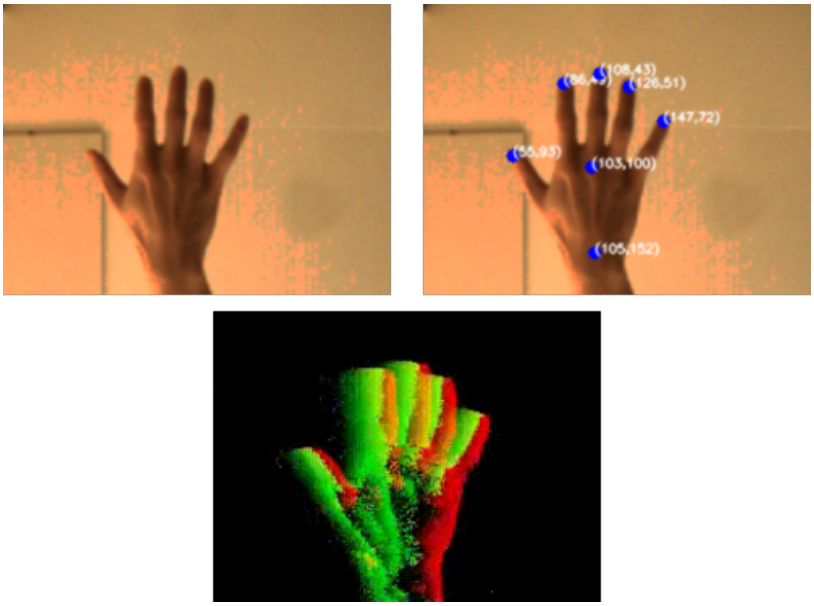

Figure 3.2: Real-World Data: RGB images, 2D joint annotations, events

CHAPTER 4

METHODS

Figure 4.1 presents the flowchart of our method. (a) represents raw event camera data. We will elaborate on our event representation, shown in (b), in section 4.1, which primarily encompasses how we accumulate event data into images and identify the RoI. (c) presents our lightweight multi-task network architecture, which will be discussed in detail in section 4.2. (d) demonstrates the prediction results of our method, including the network's 12-dimensional output, the reconstructed 3D hand joints, and the animated hand based on this model. The formulation of the loss function is presented in section 4.3.

4.1 Event Representation

4.1.1 LNES-Fast

The data generated by event cameras are typically formatted as (t, x, y, p), where t denotes the timestamp, x and y represent spatial coordinates, and p indicates the event polarity. Previous research has proposed various event representations, including Voxel Grid [38, 39], 3D Point Set [40, 41], etc. Stacking events into images or event frames enables the utilization of established computer vision methodologies [14]. Building upon this, Locally-Normalised Event Surfaces (LNES) [17] aggregates events within a fixed time window into a window-normalized 2-channel image, where each channel corresponds to one of the two polarities. Specifically, LNES employs a time window length of 100 ms, with a 99 ms overlap, to ensure a high temporal resolution of 1 kHz. The LNES method accumulates events from earlier time steps starting from the current time point, progressing step by step until reaching the predefined time window size.

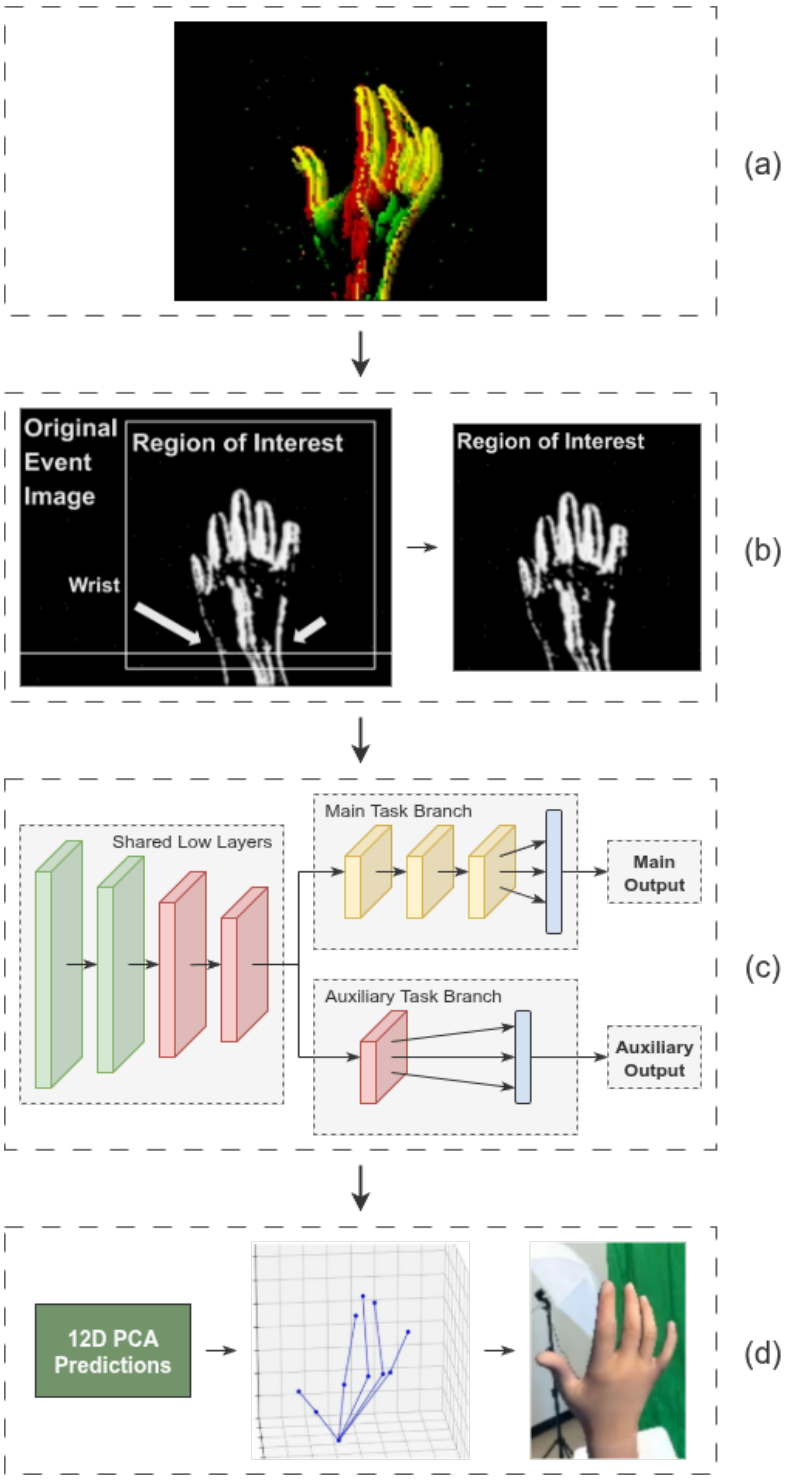

Figure 4.1: Flowchart of EvHandFPP: **(a)** Raw event data, with two polarities illustrated in different colors. **(b)** Our single-channel event representation, cropped to an RoI around the localized wrist. **(c)** Our model design with a multi-task architecture. **(d)** The model outputs, including the 12-dimensional PCA, reconstructed 3D hand joints and simulated hand.

In **Figure 4.2**, (a), (b), and (d) represent events at the current time, events from a time unit prior, and events at the time window size boundary, respectively. Events from earlier time periods are assigned lower weights.

$$Weight = \frac{WindowSize - TimeStep}{WindowSize} \tag{4.1}$$

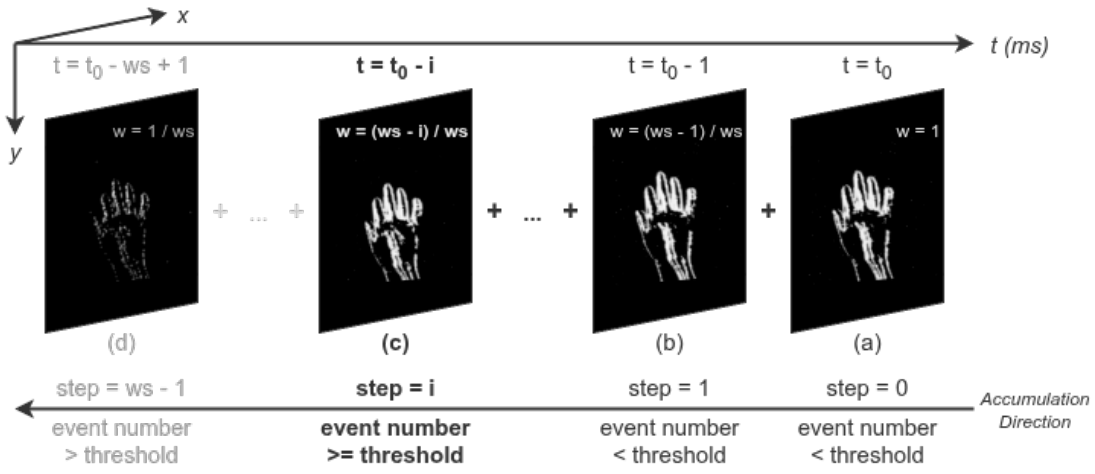

Figure 4.2: Event Accumulation: **(a)**, **(b)**, and **(d)** illustrate events at the current time, at one time step prior, and at the time window boundary, respectively. **(c)** demonstrates events at a certain time step before reaching the time window boundary, where the number of accumulated events exceed the set threshold. The frame brightness indicates temporal weights in the accumulation process, with the corresponding formulas shown in the upper-right corner, where *ws* is the time window size.

However, a fixed temporal window size could lead to either information redundancy or over-sparsity. On one hand, during rapid hand movements, a large number of events are generated within a short time, e.g. 1 ms, sufficient to constitute an image with rich information. In this case, continuing to incorporate events from earlier time is no longer beneficial, but potentially causes the image to be over-blurred. On the other hand, shortening the temporal window may result in overly sparse representations when the hand moves slowly and generates a small number of events.

To address this drawback of LNES, we introduce LNES-Fast, adding an upper bound for the number of incorporated events while preserving the original time window length constraint. We monitor the total number of events being stacked besides counting the time

steps of event aggregation. As shown in **Figure 4.2** (c), the process will be early terminated when the number of events exceeds a predefined threshold, instead of going through the complete time window as (d). While ensuring the event representation is not overly sparse during slow movements by traversing a sufficient length of the window, the early stopping mechanism reduces both interfering information and computational overhead.

4.1.2 Noise Filtering

To mitigate the accumulation of noise that often accompanies event stacking, based on the observation that interfering events typically appear as isolated outliers, we apply Gaussian blurring to suppress the intensity of such noise while enhancing hand-related events. This approach operates only on the current frame, eliminating the need to invoke information from preceding and subsequent frames for filtering, and thereby reducing computational complexity in the temporal dimension.

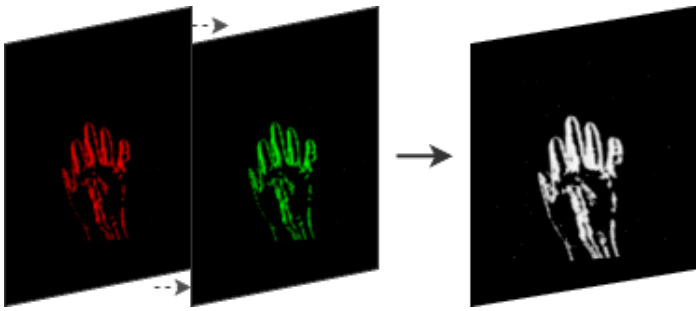

Figure 4.3: Channel Compression: red color for positive events, green for negative.

4.1.3 Channel Compression

When adopting multi-channel representations, polarity augmentation becomes necessary for generalizability, since event polarity is influenced by the environment brightness [14], i.e. background variations can result in different event polarities under the same motion. In this project, we compress the multi-channel representation, accumulating events of different polarities into one channel, as shown in **Figure 4.3**. Our experiment confirms that this simplification does not compromise performance.

4.2 Network Architecture

Based on comparative experiments, shown in chapter 5, we select MobileViT V2 [42] as the lightweight backbone of our method. We employ an RoI technique to accelerate inference to further reduce computational cost. In order to compensate for the performance degradation caused by RoI, we design an auxiliary task, incorporated through a multi-task learning framework, to help the model better focus on more important features.

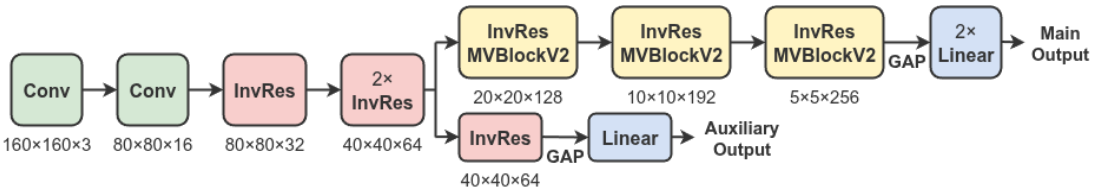

Figure 4.4: Network Architecture: Conv refers to the regular 2D convolutional layer, while GAP denotes global average pooling; InvRes is short for the inverted residual layer, while MVBlockV2 is a kind of lightweight transformer layer, both detailed in subsection 4.2.1. The output shapes are annotated in the format of height × width × channels.

The architecture of the entire network is illustrated in **Figure 4.4**. Initially, the input image is 180 pixels in height, 240 pixels in width, and has 1 channel. The selected RoI has a shape of 160×160×1, while the offset x and y coordinates in the original image are fed into the linear fully connected layer of the main task for restoring prediction results in the original spatial system. The RoI feature map first passes through a basic feature extraction layer comprising conventional 2D convolutional layers and inverted residual layers from MobileViT V2 [42]. Subsequently, it proceeds through two parallel branches: an auxiliary task branch consisting of an inverted residual layer, global average pooling, and a linear layer; and a main task branch composed of inverted residual layers, MVBlockV2, which is also from MobileViT V2, global average pooling, and double linear layers.

4.2.1 Lightweight Backbone Selection

Previous studies have adopted models with large numbers of parameters, such as ResNet-18 which has over 11 M parameters [43], as their backbones [17, 19]. However, our goal is to design a lightweight system, and these models clearly do not meet our requirements. Therefore, we retain ResNet-18 only as a baseline, and focus our experiments on lightweight models specifically designed for resource-constrained devices, including ShuffleNet V2 [44], MobileNet V3 [45], and MobileViT V2 [42]. Ultimately, we select MobileViT V2 as the backbone of our model, as it achieves the fastest convergence and best performance under comparable parameter sizes while avoiding the excessive computational overhead typically associated with other Vision Transformer (ViT) models.

To further reduce computational complexity and make the model more amenable to hardware deployment, we simplify two modules within the model. First, we replace the SiLU activation function with the simplified ReLU, which is more hardware-friendly for computation. Second, we substitute the Softmax function in the linear self-attention module with its Taylor series expansion approximation. Experimental validation chapter 5 demonstrates that these two optimization measures do not result in accuracy degradation.

The backbone's output is 12-dimensional, consisting of three parts: the first 6 dimensions represent the principal components of the MANO model, dimensions 7 to 9 correspond to the 3D translation of the hand, and the final 3 dimensions encode hand rotation.

4.2.2 Wrist-Based Region of Interest

Compared to RGB cameras, stacking event camera data into images requires additional processing. However, the inherent spatial coordinate information in event data can be effectively utilized to quickly estimate the approximate location of a target, which enables the potential implementation of the RoI method. The RoI technique reduces computational load by focusing only on cropped regions of the image that contain relevant target information, thereby decreasing the size of the area that needs to be processed.

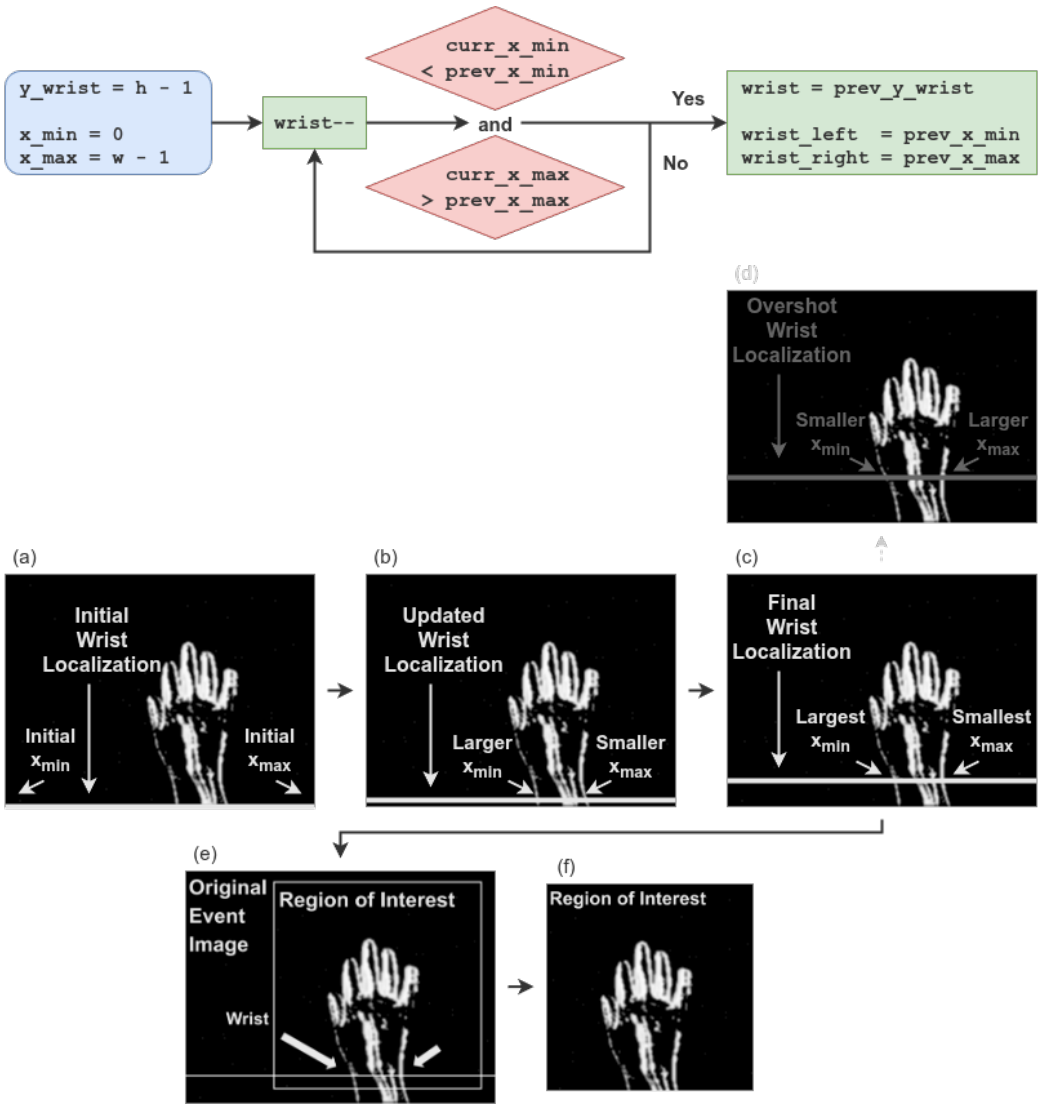

Figure 4.5: Wrist-Based Region of Interest: We identify potential hand contour boundaries (x_{min} and x_{max}) with spacing above a threshold to locate the wrist and define the RoI. (a) Search begins from the bottom row, initializing boundaries at image edges. (b) Identify larger x_{min}s and smaller x_{max}s. (c) The wrist corresponds to maximum x_{min} and minimum x_{max}. (d) When x_{min} decreases and x_{max} increases, the previous minimal spacing position indicates the wrist. (e)(f) The RoI is selected based on wrist y-coordinate and boundary values.

There are various approaches to select RoI. For example, predefining multiple candidate regions and, during data input, counting the number of events within each region to select the one with the highest event density as RoI [46]. Our method is inspired by the anatomical observation that the wrist is the narrowest part between the human arm and hand. **Figure 4.5** demonstrates our RoI method. We perform a simple line-by-line upward

scan of the event data from the bottom, identifying the row where the difference between the maximum and minimum x-coordinates of the events is minimized. This row is considered to correspond to the wrist position, which is then used to define RoI. We set the RoI size to 160×160 based on our observation, which is sufficient to cover the hands under most conditions. By applying RoI, we reduce the computational load at the cost of an acceptable performance decrease, as shown in chapter 5.

4.2.3 Multi-Task Learning

In order to improve the accuracy, we attempt to design a multi-task learning architecture. We find that the characteristic of event data to disregard hand details while preserving contour information makes it particularly well-suited for computing hand centroid, standard deviation, feature ellipses, and other statistical descriptors. While applying the RoI method, we identify the hand region, where the calculation of parameters such as the mean, variance, and characteristic ellipse of the hand-related event distribution becomes computationally convenient. These parameters are closely correlated with the 3D spatial position of the hand. Based on this observation, we introduce an additional output head at an intermediate layer of the backbone model, using these parameters as labels. This auxiliary task encourages the lower and intermediate layers of the model to learn more meaningful features. As a result, it improves the performance of our approach. Furthermore, since the auxiliary branch can be removed during inference, it introduces no additional computational overhead at deployment.

The labels and outputs of the auxiliary task have seven dimensions, including the normalized values of the following information: the mean and standard deviation of event coordinates along the x and y axes, the two eigenvalues, and the orientation angle of the characteristic ellipse of the event distribution.

4.3 Loss

The total loss consists of the main and the auxiliary task loss. To prevent the auxiliary task from excessively interfering with the optimization direction, we reduce its weight to 0.5.

$$Loss_{Total} = Loss_{Main} + 0.5 \times Loss_{Aux} \tag{4.2}$$

4.3.1 Main Task Loss

We compute separate loss terms for each of the three components in the main task, MANO, translation, and rotation mentioned in subsection 4.2.1, and assign different weights to them when aggregating the overall main task loss. In determining the weights w, we consider both maintaining comparable magnitudes across the three loss components and accounting for their varying contributions to overall performance. Based on experimental results, we adopt the following weighting scheme:

$$Loss_{Main} = \frac{(l_{MANO} \times 6 \times w_{MANO} + l_{Trans} \times 3 \times w_{Trans} + l_{Rot} \times 3 \times w_{Rot})}{12} \tag{4.3}$$

where $w_{MANO} = 10, w_{Trans} = 10000, w_{Rot} = 20$.

4.3.2 Auxiliary Task Loss

For the auxiliary task, we computed the total mean squared error of all seven components, the mean and standard deviation of event coordinates along the x and y axes, the two eigenvalues, and the orientation angle of the characteristic ellipse of the event distribution, as mentioned in subsection 4.2.3, without assigning different weights to each component.

CHAPTER 5

EXPERIMENTAL RESULTS

We conduct experiments on real-world event-based hand data to evaluate the effectiveness of our proposed method, including both necessity verification against prior state-of-the-art works and ablation analyses to examine the importance of our approach components.

5.1 Training Settings

We train our models with PyTorch Lightning [47] on a single NVIDIA RTX 3090 GPU. We set the random seed to 42, the batch size to 32, and the training epochs to 20.

5.2 Evaluation Metrics

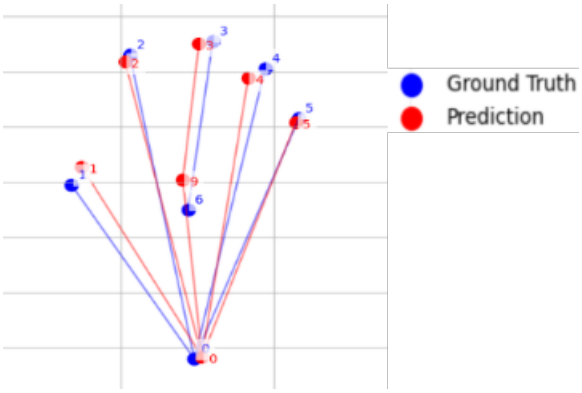

Figure 5.1: 2D Evaluation

5.2.1 2D Metric

The PCK metric, referring to the root-aligned percentage of correct keypoints, is usually adopted in hand tracking works [28]. To make the 2D-PCK, which includes no explicit depth information, more comparable, V. Rudnev et al. [17] proposed 2D-PCKp, the 2D-PCK normalized by the palm length, and 2D-AUCp, the corresponding area under the

curve. According to this metric, the predicted results from the neural network models are fed into the software evsim [17] to calculate the 3D hand joints based on the MANO model [24], which are then projected onto the 2D image plane to compute the 2D-PCKp. **Figure 5.1** compares the predictions against the ground truth labels using this metric.

5.2.2 3D Metric

3D coordinates transformed from predicted outputs via the evsim software can be directly compared with 3D ground-truth labels to compute 3D-PCK and 3D-AUC. However, due to the inherent difficulties in obtaining 3D annotations for real-world data, our evaluation of this metric is limited to synthetic datasets.

5.3 Comparison with Prior Works

Table 5.1 presents a comparison between our method and other works. The EventHands model [17], primarily trained on third-person right-hand data, achieves a 2D-AUCp of 0.77. This is considered the state-of-the-art performance for event-based hand tracking under this metric. However, when applied to our task, which basically involves first-person perspective right-hand data, its performance drops significantly, yielding a 2D-AUCp of only 0.12. This highlights the necessity of our proposed approach. For a fair comparison, we also train and evaluate a model using ResNet-18 as the backbone, as adopted by EventHands, and achieve a 2D-AUCp of 0.77. This suggests that the difficulty level of our task is comparable to that of EventHands. Our proposed method achieves a 2D-AUCp of 0.84 on this task, surpassing the performance of previous approaches. Furthermore, our model exhibits a 90% reduction in parameter count and an 89% reduction in computational cost compared to prior work, demonstrating a previously unseen potential for lightweight scenarios.

Table 5.1: Comparison with SOTA

Method	Task	2D-AUCp ↑	Params ↓	FLOPs ↓
EventHands	Hand Tracking	0.77	11.2 M	1.648 G
EventHands	FPP Hand Tracking	0.11	11.2 M	1.648 G
Retrained ResNet-18	FPP Hand Tracking	0.77	11.2 M	1.648 G
Ours	FPP Hand Tracking	**0.85**	**1.1 M**	**0.185 G**

5.4 Ablation Studies

5.4.1 Backbones

Previous research has compared the performance of various general-purpose models for event-based hand tracking. However, their substantial parameter counts render them unsuitable for resource-constrained lightweight systems. A lot of lightweight models have been proposed and validated for edge device deployment, and we specifically select ShuffleNet V2 [44], MobileNet V3 [45], and MobileViT V2 [42] for comparative experiments, where we ensure their parameter counts are at a consistent level for fair comparison. Their performance, parameter counts, and computational loads are detailed in **Table 5.2**. These results are obtained based on original image inputs without auxiliary task branches.

Ultimately, we chose MobileViT V2 as the backbone for our proposed method. MobileViT V2 benefits from the superior performance of the ViT architecture, while its computational load is optimized through the integration of the linear attention mechanism and is considerably lower than that of general-purpose models like ResNet. Although the computational demand of MobileViT V2 is not the lowest among all the lightweight models we evaluated, our RoI method effectively mitigates this drawback.

It is worth mentioning that due to non-negligible variations in the parameter counts of the officially released version of the selected lightweight models, we prune them to the same to ensure a fair comparison. Subsequent experiments are conducted using the official 0.5x parameter version of MobileViT V2 to ensure optimal performance.

Table 5.2: Comparison among Backbones

Backbone	**2D-AUCp ↑**	**Params ↓**	**FLOPs ↓**
ResNet-18	0.77	11.2 M	1.648 G
ShuffleNet V2	0.78	**0.1 M**	0.025 G
MobileNet V3	0.76	**0.1 M**	**0.011 G**
MovileViT V2	**0.80**	**0.1 M**	*0.023 G*

5.4.2 Architectures

While the SiLU activation function in MobileViT V2 offers advantages such as smoothness and non-monotonicity, its implementation on hardware (e.g. Field-Programmable Gate Array (FPGA) or Application-Specific Integrated Circuit (ASIC)) presents significant challenges [48, 49]. Given this consideration, and acknowledging that event images are inherently simpler than RGB images, we replace all activation functions within MobileViT V2 with the comparatively simpler ReLU. Comparative experiments confirmed that this substitution results in negligible performance loss.

Another computationally intensive component is the softmax function. To accommodate lightweight systems, we replace the original softmax function with a Taylor series expansion approximation. Comparative experiments demonstrated that this modification introduced no discernible performance loss on our specific task.

5.4.3 Event Representations

Our event representation builds upon LNES by incorporating efficiency optimizations. We represent events of different polarities within a single channel, and terminate the stacking process prematurely once the event count exceeds a defined threshold. Comparative experiments demonstrate that these modifications not only achieve the anticipated reduction in computational load but also result in no performance degradation.

5.4.4 RoI and Multi-Task Learning

Comparative experiment, based on the 0.5× version of MobileViT V2, reveals that the RoI method indeed yields computational optimizations of over 40%, with an ignorable reduction in performance, as shown in Table 5.3.

Furthermore, our experiments confirmed that placing the auxiliary task's fully connected layer after a middle layer of the model results in the highest performance improvement. The experimental results are presented in **Table 5.3**. We disregard the parameter and computational loads associated with the auxiliary tasks in our multi-task architecture design, as their corresponding branches will be pruned in the inference stage.

Table 5.3: Comparison for RoI and Multi-Task Learning

RoI	**Aux Task**	**2D-AUCp ↑**	**FLOPs ↓**
No	No	*0.85*	0.322 G
Yes	No	0.84	**0.185 G**
No	Yes	**0.87**	0.322 G
Yes	**Yes**	*0.85*	**0.185 G**

CHAPTER 6
CONCLUSION

This thesis presents an efficient lightweight framework for real-time 3D hand tracking from an first-person perspective using event cameras. Our proposed method addresses the limitations of low temporal resolution and high cost of existing RGB-based and event-based approaches by leveraging the unique properties of event-based vision.

The key technical contributions include an optimized event representation with adaptive accumulation, a single-channel representation strategy, an RoI technique, and a multi-task learning framework that leverages event coordinate statistics. We introduce the first event-based hand tracking dataset specifically designed for first-person perspective, comprising 720,000 milliseconds of synthetic training data and 60,000 milliseconds of real-world evaluation data. Our ablation studies validate the effectiveness of each component, demonstrating that our approach balances accuracy and efficiency.

Our solution achieves better performance with a 2D-AUCp of 0.84, outperforming SOTA approaches while reducing model parameters by over 10-fold (from 11.2 M to 1.1 M) and decreasing computational costs by nearly 90% (from 1.648 GFLOPs to 0.185 GFLOPs). This framework enables practical deployment of hand tracking in AR/VR applications where first-person hand interaction is crucial. By processing event data at 1 kHz with minimal computational requirements, this work opens new possibilities for ubiquitous real-time hand tracking technology on edge devices.

Future work may explore model quantization and FPGA deployment with acceleration designs to further enhance efficiency, adoption of bio-inspired spiking neural network models that naturally align with event-based data processing, and improvement of robustness and adaptability in challenging scenarios including occlusion and dynamic environments. Our publicly available method and dataset will facilitate continued research.

REFERENCES

[1] J. M. Rehg and T. Kanade, "Visual tracking of high dof articulated structures: An application to human hand tracking," in *Computer Vision — ECCV '94*, J.-O. Eklundh, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 35–46, ISBN: 978-3-540-48400-4.

[2] Y. Wu and T. Huang, "Hand modeling, analysis and recognition," *IEEE Signal Processing Magazine*, vol. 18, no. 3, pp. 51–60, 2001.

[3] Y. Wu, J. Lin, and T. Huang, "Analyzing and capturing articulated hand motion in image sequences," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 12, pp. 1910–1922, 2005.

[4] M. de La Gorce, D. J. Fleet, and N. Paragios, "Model-based 3d hand pose estimation from monocular video," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 9, pp. 1793–1805, 2011.

[5] C. Zimmermann and T. Brox, "Learning to estimate 3d hand pose from single rgb images," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct. 2017.

[6] U. Iqbal, P. Molchanov, T. B. J. Gall, and J. Kautz, "Hand pose estimation via latent 2.5d heatmap regression," in *Proceedings of the European Conference on Computer Vision (ECCV)*, Sep. 2018.

[7] L. Ge *et al.*, "3d hand shape and pose estimation from a single rgb image," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2019.

[8] J. Liebers, S. Brockel, U. Gruenefeld, and S. S. and, "Identifying users by their hand tracking data in augmented and virtual reality," *International Journal of Human–Computer Interaction*, vol. 40, no. 2, pp. 409–424, 2024.

[9] A. Colaco, A. Kirmani, N.-W. Gong, T. Mcgarry, L. Watkins, and V. K. Goyal, "3dim: Compact and low power time-of-flight sensor for 3d capture using parametric signal processing," in *Proc. Int. Image Sensor Workshop*, Citeseer, 2013, pp. 349–352.

[10] D. Rotman, O. Cohen, and G. Gilboa, "Frame rate reduction of depth cameras by rgb-based depth prediction," in *2016 IEEE International Conference on the Science of Electrical Engineering (ICSEE)*, 2016, pp. 1–5.

[11] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128× 128 120 db 15 μs latency asynchronous temporal contrast vision sensor," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, 2008.

[12] C. Posch, D. Matolin, and R. Wohlgenannt, "A qvga 143 db dynamic range frame-free pwm image sensor with lossless pixel-level video compression and time-domain cds," *IEEE Journal of Solid-State Circuits*, vol. 46, no. 1, pp. 259–275, 2011.

[13] B. Son *et al.*, "4.1 a 640×480 dynamic vision sensor with a 9µm pixel and 300meps address-event representation," in *2017 IEEE International Solid-State Circuits Conference (ISSCC)*, 2017, pp. 66–67.

[14] G. Gallego *et al.*, "Event-based vision: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 1, pp. 154–180, 2022.

[15] iniVation AG, *Davis346 doc*, Jul. 2025.

[16] Y. Xue, H. Li, S. Leutenegger, and J. Stückler, *Event-based non-rigid reconstruction from contours*, 2022. arXiv: 2210.06270 [cs.CV].

[17] V. Rudnev *et al.*, "Eventhands: Real-time neural 3d hand pose estimation from an event stream," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2021, pp. 12 385–12 395.

[18] C. Millerdurai *et al.*, "3d pose estimation of two interacting hands from a monocular event camera," in *2024 International Conference on 3D Vision (3DV)*, 2024, pp. 291–301.

[19] J. Jiang, J. Li, B. Zhang, X. Deng, and B. Shi, "Evhandpose: Event-based 3d hand pose estimation with sparse supervision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 9, pp. 6416–6430, 2024.

[20] J. Grubert, L. Witzani, E. Ofek, M. Pahud, M. Kranz, and P. O. Kristensson, "Effects of hand representations for typing in virtual reality," in *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, 2018, pp. 151–158.

[21] W. Lin, L. Du, C. Harris-Adamson, A. Barr, and D. Rempel, "Design of hand gestures for manipulating objects in virtual reality," in *Human-Computer Interaction. User Interface Design, Development and Multimodality*, M. Kurosu, Ed., Cham: Springer International Publishing, 2017, pp. 584–592.

[22] F. Zhang *et al.*, *Mediapipe hands: On-device real-time hand tracking*, 2020. arXiv: 2006.10214 [cs.CV].

[23] X. Zhang, Q. Li, H. Mo, W. Zhang, and W. Zheng, "End-to-end hand mesh recovery from a monocular rgb image," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2019.

[24] J. Romero, D. Tzionas, and M. J. Black, "Embodied hands: Modeling and capturing hands and bodies together," *ACM Transactions on Graphics*, vol. 36, no. 6, pp. 1–17, Nov. 2017.

[25] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, "Smpl: A skinned multi-person linear model," *ACM Trans. Graph.*, vol. 34, no. 6, Oct. 2015.

[26] Y. Cai, L. Ge, J. Cai, and J. Yuan, "Weakly-supervised 3d hand pose estimation from monocular rgb images," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 666–682.

[27] A. Spurr, J. Song, S. Park, and O. Hilliges, "Cross-modal deep variational hand pose estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 89–98.

[28] F. Mueller *et al.*, "Ganerated hands for real-time 3d hand tracking from monocular rgb," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2018.

[29] L. Yang, S. Li, D. Lee, and A. Yao, "Aligning latent spaces for 3d hand pose estimation," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 2335–2343.

[30] A. Spurr, U. Iqbal, P. Molchanov, O. Hilliges, and J. Kautz, "Weakly supervised 3d hand pose estimation via biomechanical constraints," in *Proceedings of the European conference on computer vision (ECCV)*, Springer, 2020, pp. 211–228.

[31] X. Chen *et al.*, "Mobrecon: Mobile-friendly hand mesh reconstruction from monocular image," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 20 544–20 554.

[32] C. Wan, T. Probst, L. Van Gool, and A. Yao, "Crossing nets: Combining gans and vaes with a shared latent space for hand pose estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 680–689.

[33] J. Malik *et al.*, "Handvoxnet: Deep voxel-based network for 3d hand shape and pose estimation from a single depth map," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 7113–7122.

[34] S. Yuan *et al.*, "Depth-based 3d hand pose estimation: From current achievements to future goals," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2636–2645.

[35] L. Ge, Y. Cai, J. Weng, and J. Yuan, "Hand pointnet: 3d hand pose estimation using point sets," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8417–8426.

[36] G. Moon, J. Y. Chang, and K. M. Lee, "V2v-posenet: Voxel-to-voxel prediction network for accurate 3d hand and human pose estimation from a single depth map," in *Proceedings of the IEEE conference on computer vision and pattern Recognition*, 2018, pp. 5079–5088.

[37] L. Fang, X. Liu, L. Liu, H. Xu, and W. Kang, "Jgr-p2o: Joint graph reasoning based pixel-to-offset prediction network for 3d hand pose estimation from a single depth image," in *Proceedings of the European conference on computer vision (ECCV)*, Springer, 2020, pp. 120–137.

[38] P. Bardow, A. J. Davison, and S. Leutenegger, "Simultaneous optical flow and intensity estimation from an event camera," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016.

[39] L. Wang, Y.-S. Ho, and K.-J. Yoon, "Event-based high dynamic range image and very high frame rate video generation using conditional generative adversarial networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2019.

[40] R. Benosman, C. Clercq, X. Lagorce, S.-H. Ieng, and C. Bartolozzi, "Event-based visual flow," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 2, pp. 407–417, 2014.

[41] Y. Sekikawa, K. Hara, and H. Saito, "Eventnet: Asynchronous recursive event processing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2019.

[42] S. Mehta and M. Rastegari, *Separable self-attention for mobile vision transformers*, 2022. arXiv: 2206.02680 [cs.CV].

[43] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, 2015. arXiv: 1512.03385 [cs.CV].

[44] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, *Shufflenet v2: Practical guidelines for efficient cnn architecture design*, 2018. arXiv: 1807.11164 [cs.CV].

[45] A. Howard *et al.*, *Searching for mobilenetv3*, 2019. arXiv: 1905.02244 [cs.CV].

[46] S. Tan *et al.*, "Toward efficient eye tracking in ar/vr devices: A near-eye dvs-based processor for real-time gaze estimation," *IEEE Transactions on Circuits and Systems I: Regular Papers*, pp. 1–13, 2025.

[47] W. Falcon and The PyTorch Lightning team, *PyTorch Lightning*, version 1.4, Mar. 2019.

[48] S. Elfwing, E. Uchibe, and K. Doya, *Sigmoid-weighted linear units for neural network function approximation in reinforcement learning*, 2017. arXiv: 1702.03118 [cs.LG].

[49] P. Ramachandran, B. Zoph, and Q. V. Le, *Searching for activation functions*, 2017. arXiv: 1710.05941 [cs.NE].