Privacy-Preserving Visual Localization with Event Cameras

Junho Kim, Young Min Kim, Ramzi Zahreddine, Weston A. Welge,
Gurunandan Krishnan, Sizhuo Ma, and Jian Wang

***Abstract*—We consider the problem of client-server localization, where edge device users communicate visual data with the service provider for locating oneself against a pre-built 3D map. This localization paradigm is a crucial component for location-based services in AR/VR or mobile applications, as it is not trivial to store large-scale 3D maps and process fast localization on resource-limited edge devices. Nevertheless, conventional client-server localization systems possess numerous challenges in computational efficiency, robustness, and privacy-preservation during data transmission. Our work aims to jointly solve these challenges with a localization pipeline based on event cameras. By using event cameras, our system consumes low energy and maintains small memory bandwidth. Then during localization, we propose applying event-to-image conversion and leverage mature image-based localization, which achieves robustness even in low-light or fast-moving scenes. To further enhance privacy protection, we introduce privacy protection techniques at two levels. Network level protection aims to hide the entire user's view in private scenes using a novel split inference approach, while sensor level protection aims to hide sensitive user details such as faces with light-weight filtering. Both methods involve small client-side computation and localization performance loss, while significantly mitigating the feeling of insecurity as revealed in our user study. We thus project our method to serve as a building block for practical location-based services using event cameras. Project page including the code is available through this link: https://82magnolia.github.io/event_localization/.**

***Index Terms*—Event cameras, visual localization, camera pose estimation, privacy-preserving computer vision**

I. Introduction

VISUAL localization is a versatile localization method widely used in AR/VR, which aims to find the camera pose with respect to a pre-built 3D map solely using images. Due to the limited amount of compute and storage available in AR/VR devices, conventional systems employ *client-server localization* where edge device (*e.g.* smartphones, AR glasses) users transmit visual information to the service provider for localization [1]–[3]. While this enables effective camera pose estimation by reducing the user-side compute burden, privacy concerns arise due to the nature of image capture [4], [5]. As shown in Figure 1-(a), users of the localization service may be concerned with sharing the current view with the service provider. Further concerns can be raised by uninformed people who are observed by the user and captured in the localization process. Along with privacy concerns, localization systems for mobile devices demand for robust performance in a wide range of conditions including fast motion or dark scenes.

Event cameras are visual sensors that only record brightness *changes* [6], [7] as a stream of events, which have the potential to provide robust, efficient, and privacy-preserving

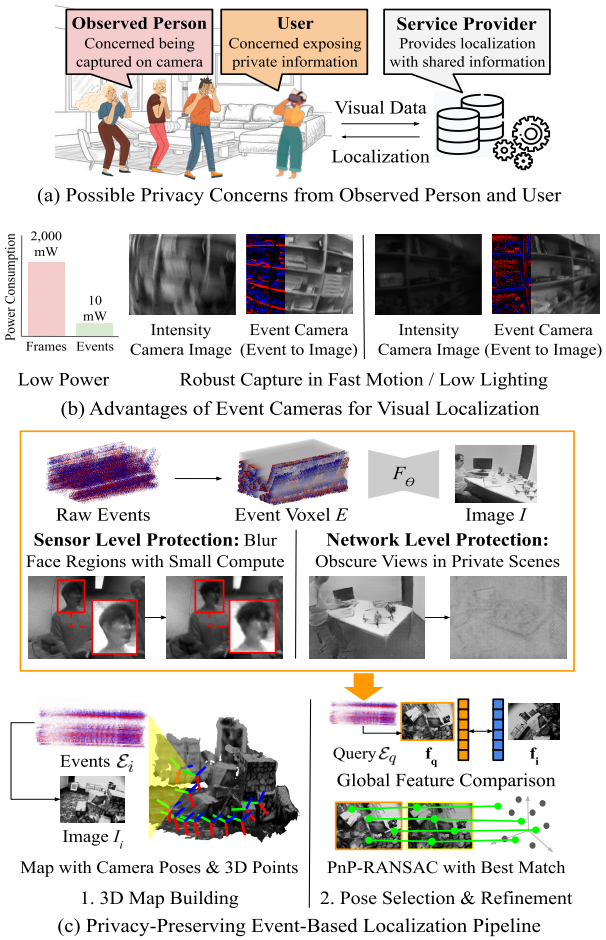


Fig. 1. Overview of our approach. **(a)** Client-server localization introduces privacy concerns. **(b)** Event cameras have numerous hardware benefits for localization. **(c)** We achieve privacy-preserving localization by applying protection techniques tailored to events during events pre-processing (sensor level) and event-to-image conversion (network level) (top), where the results are then used for localization (bottom).

The work was done when Junho Kim was an intern at Snap. (*Corresponding authors: Sizhuo Ma and Jian Wang*)

Junho Kim and Young Min Kim are with Seoul National University, Seoul, 08826, South Korea (e-mail: 82magnolia@snu.ac.kr; youngmin.kim@snu.ac.kr)

Ramzi Zahreddine, Weston A. Welge, Gurunandan Krishnan, Sizhuo Ma, and Jian Wang are with Snap Inc., Santa Monica CA, 90405, USA (e-mail: rzahreddine@snapchat.com, weston.welge@gmail.com, guru@gurukrishnan.com, sma@snapchat.com, jwang4@snapchat.com)

visual localization. As shown in Figure 1-(b), event cameras have a high temporal resolution and dynamic range [8], which are crucial for robust localization in challenging scenarios such as low lighting or fast camera motion. Further, as the power consumption is far lower than normal cameras [6] and the form factor is becoming smaller due to recent advancements in manufacturing [9]–[12], these sensors are highly favorable for machine vision tasks in AR/VR. From a privacy perspective, the sensors only capture a fraction of visual information as shown in Figure 1, and thus an average person cannot confidently identify people solely from events. Nevertheless, this comes at the expense of relatively unstable visual features compared to normal cameras, making direct localization from events difficult.

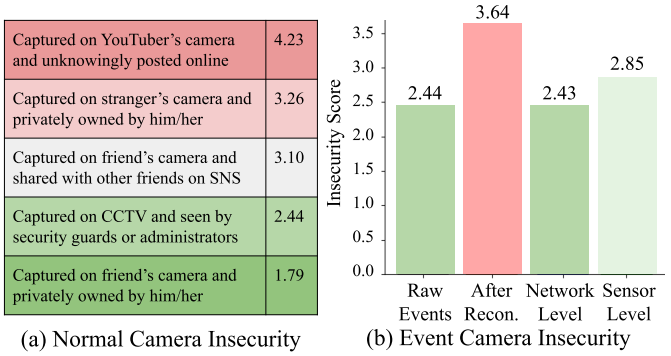
We propose an event-based visual localization method that can perform robust localization while preserving privacy. For localization, we employ event-to-image conversion which allows leveraging mature, powerful image-based localization methods [13], [14] on captured events. Our key observation is that despite the information loss during event capture, the converted images contain robust and salient image features sufficient for localization. The resulting method achieves localization accuracy comparable with an intensity camera in normal scenarios, but significantly better for fast camera motion or low lighting, where localization using normal cameras typically fails. In addition, by combining event-to-image conversion with image-based localization which effectively reduces the domain gap between events and images, our method can outperform existing event-based localization methods. To make our solution amenable to mobile devices, the client is only responsible for a light-weight capture and computation process, while the service provider performs the computationally expensive conversion [15]–[21] and localization steps.

We then design two levels of privacy protection *tailored* for event cameras, as shown in Figure 1-(c). On the **network level**, we observe that naively offloading the computation to the server can lead to privacy breaches. To address such concerns, we propose to split neural network inference with a novel retraining procedure for neural networks that prevents the service provider from reconstructing what the users see. This protection scheme targets users who are willing to use location-based services in private spaces (*e.g.* apartment rooms), where protecting the entire user view should be solicited. On the **sensor level**, we propose novel filtering methods based on the spatiotemporal volume of events that preserve important static landmarks for localization, while blurring facial landmarks without explicit detection. This process targets wider use cases in both private and public spaces: it aims to reduce concerns about being recorded by wearables or mobile devices. The filters are designed to be light-weight which makes it possible to implement such protection on the sensor directly. In practice, since the techniques aim to protect the privacy of different targets (users and observed people), they can be applied jointly.

We design a rigorous evaluation procedure to assess our method on a wide range of localization scenarios involving moving people, low-lighting, or fast camera motion. Specifically, we create two new datasets called EvRooms and EvHumans, and introduce protocols for holistically evaluating privacy protection and localization performance. We also conduct a user study to examine how real users perceive our privacy protection schemes. Experiments show that our approach effectively balances localization performance with privacy protection. We first show that, without privacy protection, the proposed event-to-image approach outperforms existing event-based methods that do not rely on image reconstruction [22]–[24]. We then demonstrate that the proposed privacy preservation techniques efficiently protect user privacy without significantly reducing the localization accuracy. Notably, in the user study, both levels of privacy protection have shown to alleviate privacy concerns of real users, as seen in Figure 2. We will publicly release the dataset along with the accompanying code for benchmarking, which we expect to foster future research in event-based visual localization. To summarize, our key contributions are:

- Proposing the first event-driven effort for robust and private localization targeting mobile environments.
- Novel network level privacy protection for mitigating users' concerns.
- Novel sensor level privacy protection for relieving observed people's concerns.

By synergistically combining event cameras with robust localization and privacy protection, we expect our method to serve as a practical building block for spatial perception systems in mobile environments.



(a) Normal Camera Insecurity (b) Event Camera Insecurity

Fig. 2. User study results on our privacy protection method. The insecurity scores range between 1 and 5, where higher score indicates higher insecurity. (a) We make an initial measurement on how users feel about being captured using normal cameras in various scenarios. (b) Then, we query on event cameras by sequentially showing raw events, event-to-image reconstructions, and privacy protection results.

II. Related Work

1) Event-Based Visual Localization: Due to the high dynamic range and small motion blur, event cameras have been widely studied for visual odometry (VO) or SLAM tasks involving *sequential pose estimation* between temporally adjacent events [25]–[34]. However, re-localizing an event camera with respect to a pre-built 3D map, namely event-based visual localization, is a fairly understudied problem. Unlike the VO or SLAM setup, here poses should be estimated amidst large viewpoint differences between the query events and reference images in the 3D map [13], [35]. In general, despite

recent developments in learning-based feature descriptors for events [36]–[39], approaches that directly use the raw event measurements usually perform inferior to the counterpart using normal images [31], [40], [41]. This is due to the instability of visual features in events compared to normal images and the lack of large scale training data [42]–[44]. Prior works perform direct camera pose estimation using neural networks [24], [40], which requires per-scene training and are inferior to structure-based methods in performance [45]–[47]. Structure-based methods leverage correspondences in 2D and 3D by comparing feature descriptors [13], [14], [35], [48]–[50]. Due to decades of research in image feature descriptors [49], [51], structure-based methods are known to perform stable localization. We leverage event-to-image conversion to combine the mature structure-based paradigm with event cameras, leading to robust localization under challenging conditions.

2) Privacy-Preserving Machine Vision: Machine vision applications in networked environments are subject to privacy breaches as they utilize images captured from the clients [4], [5], [52]. Recent works target privacy protection in split inference scenarios where the costly neural network computation is shared between the client and server [53]–[62]. Such attempts can be viewed as instances of a broader problem in cryptography called Secure Multiparty Computation (SMC) [63], where the goal is to find methods for parties to compute a function (in this case a neural network) over their inputs while hiding the input values from each other. Unlike prior works that mainly target classification tasks where a reduced amount of information is sent to the server, we tackle privacy-preservation for reconstruction which requires sending all the information to reconstruct a full image. This is a more challenging and complex task as it additionally demands hiding the *outputs* of the computation (i.e., reconstructed images) from the server, which differs from the standard SMC setup that mainly focuses on hiding the inputs.

Prior works in privacy-preserving visual localization also focus on this aspect, where existing methods suggest lifting the 2D / 3D keypoints to lines [2], [64]–[67], or training a new set of feature descriptors hiding sensitive details [1], [3], [4], [68], [69]. Another direction of recent works try to encrypt the visual data in the sensor level by incorporating specially designed optics [70]–[73]. For example [70] designs a phase mask applicable on the camera lens that hides the sensitive scene content and trains a neural network that decodes the phase mask outputs to get depth map predictions. Inspired by such works, we propose network level privacy protection that prevents possible server-side attacks, and sensor-level protection that hides sensitive visual details in event data.

III. Event-Based Localization Pipeline

Given a short stream of events recorded by an event camera, our method aims to find the 6DoF camera pose within a 3D map as shown in Figure 1-(c). Event cameras are visual sensors that track brightness changes as a stream of events, $\mathcal{E} = \{e_i = (x_i, y_i, t_i, p_i)\}$, where e_i indicates the brightness change of polarity $p_i \in \{+1, -1\}$ at pixel location (x_i, y_i) and timestamp t_i. Compared to conventional image sensors, the outputs are sparse and asynchronous, and thus consumes less memory bandwidth and energy which makes the sensors amenable to mobile vision applications.

Our localization method combines event-to-image conversion with image-based localization. We incorporate such a design choice as our method tackles the previously under-explored problem of finding the event camera pose with respect to a pre-built 3D map, also known as *visual re-localization [74]*. The problem is challenging due to large viewpoint differences compared to SLAM or odometry scenarios [32]–[34]. As a result, existing methods directly using events [24], [36]–[40] often exhibit unstable re-localization performance. Our approach can perform highly accurate localization even in difficult scenarios (fast motion, low lighting) for image-based methods [35], [49] by jointly leveraging the sensor-level benefits of events and the maturity of image-based localization.

Given an input event stream $\mathcal{E}$, let E denote the event voxel grid [15], [18], [75] obtained by taking weighted sums of event polarities within spatio-temporal bins. Event-to-image conversion methods [15], [16], [18], [19], [76] take the event voxels as input and produce images using neural networks, namely $F_\Theta(E) = I$ where Θ denotes the neural network parameters.

A. Structure-Based Localization

Figure 1-(c) shows our localization process. In the mapping stage, a 3D map representaion is built from event streams captured for a scene: $S_e{=}\{\mathcal{E}_1, \ldots, \mathcal{E}_N\}$, with each stream being a spatiotemporal volume of events spanning a short time. In the query stage, a user-captured query stream $\mathcal{E}_q$ is matched against the map to localize the user. To build the map, we first convert the reference events S_e into images $S_i{=}\{I_1, \ldots, I_N\}$. Then we run an off-the-shelf structure-from-motion pipeline COLMAP [77] on S_i. The result is a map containing 3D keypoints and 6DoF pose-annotated reference images.

Next, we extract global features vectors using NetVLAD [48] for each pose-annotated reference image I_i in the 3D map for *candidate pose selection*. In the query stage, we reconstruct an image I_q from the query event stream $\mathcal{E}_q$ and compute the L2 distances between the query and reference image features. We select the top-K nearest poses which serve as candidate poses for further refinement.

Finally, we refine the pose by first performing local feature matching [35], [49] between the query and selected reference images. We count the number of matches found for each query-reference pair and choose the reference view I_r with the largest number of matches. Then we obtain the refined 6DoF pose by retrieving the 3D points visible from I_r and performing PnP-RANSAC [78], [79] between the 2D points in I_q and retrieved 3D points.

By leveraging event-to-image conversion, we can effectively deploy powerful image-based localization methods on events. Nevertheless, for high-quality image recovery the conversion solicits repetitive neural network inferences [15], [19], which can be costly for edge devices. This necessitates the transmission of visual information from edge devices to service

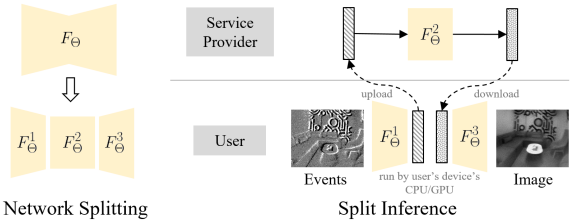


Fig. 3. Split inference setup for network level privacy protection. To save compute while hiding sensitive visual information, the original network weights are split (left) and only the costly intermediate part (F^2_Θ) is shared with the service provider. Then, the compute is distributed between the user and service provider: the user performs the light-weight frontal and lateral network inference and the service provider performs the heavy part (right).

providers, where we propose various techniques for preserving privacy in Section IV.

IV. Privacy-Preserving Localization Overview

We propose two levels of privacy protection during information sharing between the user and service provider: network- and sensor-level protection. **Network-level privacy protection** targets localization in private scenes (*e.g.* apartments, corporate offices), where the user would want to completely hide what they are looking at. **Sensor-level privacy protection** targets a broader range of applications and focuses on hiding non-structural details such as facial regions with small additional computation. Note that similar to prior works in privacy-preserving visual localization [1], [3], [4], we assume an *honest-but-curious* service provider [80] that faithfully provides the required computation (e.g., global/local feature extraction for localization) but is curious and may attempt to extract the client's visual information from the shared data. Therefore, the service provider considered in our work cannot secretly gain access to data that the client has not agreed to share (e.g., raw event data), and as a result, our work focuses on secure information sharing during client-server localization.

V. Network-Level Privacy Protection

Network level privacy protection hides the user's view from the service provider in private spaces. As shown in Figure 3, we suggest *splitting* the event-to-image conversion process between the service provider and user, and strategies for preventing possible server-side attacks. The motivation is twofold: (i) offloading the entire conversion to the server enables revealing the user's view, and (ii) naively splitting inference can still have the server decode the shared intermediate representations. Below we retain our focus on making the event-to-image conversion process privacy-preserving. Once the images are securely reconstructed, it is also possible to apply existing privacy-preserving visual localization methods [1], [3], [64] for further protection.

A. Split Inference Setup

The split inference process operates in three steps, as summarized in Figure 3. Prior to inference, the users divide the event-to-image conversion network to disjoint parts

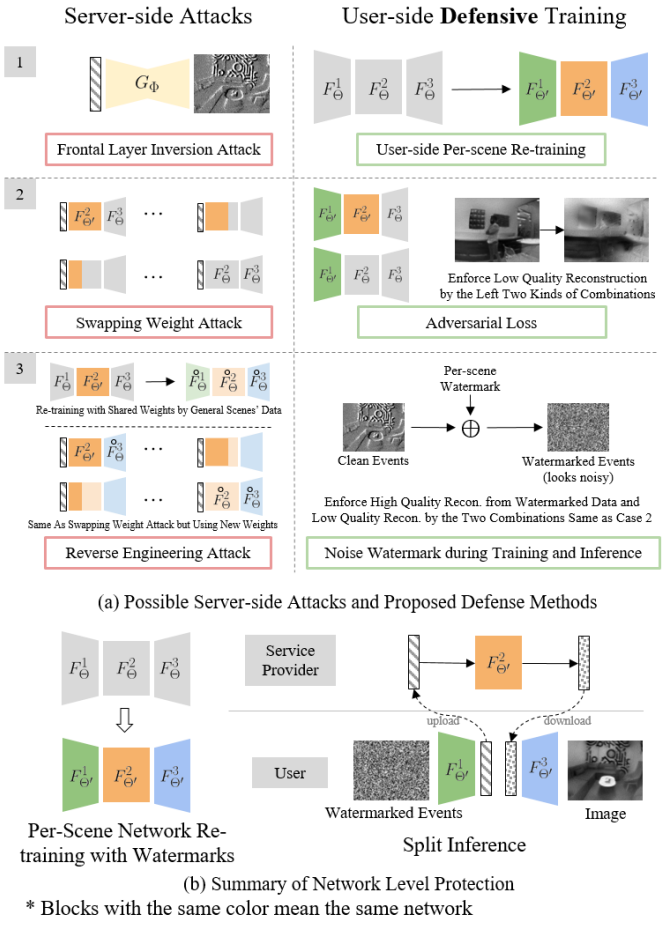


Fig. 4. Network level privacy protection targeting users in private scenes. (a) We identify three possible attacks from the service provider. Case 1: Frontal layer inversion attempts to decode the intermediate activations with a learned network G_Φ. Case 2: Swapping weight attack combines the shared network weights with the publicly available weights (gray) to obtain image reconstructions. Case 3: Reverse engineering operates similarly but with network weights trained on the server side aiming to reverse-engineer the unshared network weights. For defense, we propose per-scene re-training (top) using adversarial losses (middle) and noise-infused event voxels (bottom). (b) In the resulting network level protection, users deploy a privately-trained reconstruction network $F_{\Theta'}$ and share the intermediate part with the server during inference.

$F^1_\Theta, F^2_\Theta, F^3_\Theta$, where Θ denotes publicly available network weights that are pre-trained on datasets containing general scenes [81]. F^2_Θ contains the *majority* of the inference computation and is *the only shared part* with the service provider. During inference, (i) the user performs inference on F^1_Θ, (ii) the result is sent to the service provider to perform F^2_Θ, and (iii) the user retrieves the result to finally perform F^3_Θ. Since the frontal and rear inference is done on-device, it may appear at first glance that this conversion is privacy preserving. However, similar to observations in prior work on split inference [56], [57], we identify three possible server-side attacks to decode user information and propose corresponding defense strategies.

B. Possible Server-Side Attacks and Defenses

1) Frontal Layer Inversion Attacks: First, the service provider could train an inversion network G_Φ that takes the frontal layer activations and regresses the original event voxel.

After that an event-to-image conversion network could be used to obtain an image reconstruction. If the split inference is performed naively with publicly available pre-trained network weights Θ [15], [20], then G_Φ could be easily trained using conventional event camera datasets [42], [43]. To prevent the attack we propose to have the user to quickly re-train a new event-to-image conversion network Θ' using events collected in the private scene. Since the frontal part of the newly trained network $F^1_{\Theta'}$ is unknown to the server, this can prevent training a performant inversion network G_Φ. **Note that re-training is a *one-time* operation for each private scene, and can potentially be shared between users residing in the same scene.** Further, the process does not require large event data as the re-trained network only operates in the private scene. Typically, re-training can be done using approximately one minute of event camera data, which takes 0.5 hours with a commodity GPU and 3 hours with a CPU. We provide a detailed analysis on the computational cost of the re-training process in Appendix C-C.

2) Swapping Weight Attacks: Second, the service provider may use the publicly available network weights Θ and extract reconstructions from the frontal layer activations. For example, since the client needs to offload its computation and share the private weights $F^2_{\Theta'}$ to the server, the service provider can combine it with the public weights for the succeeding layers, *i.e.*, run $F^3_\Theta \circ F^2_{\Theta'}$ on top of the shared frontal layer activations $F^1_{\Theta'}(E)$ to reconstruct an image. Note as in Figure 4-(a.2), Θ' and Θ can be combined at different layers to get an image reconstruction.

To prevent such attacks, we propose to perform re-training with two losses $L{=}L_{\text{recon}}{+}L_{\text{adv}}$, where L_{recon}, L_{adv} are the reconstruction and adversarial losses respectively. Formally, the reconstruction loss is given as follows,

$$L_{\text{recon}} = d(F_\Theta(E), F_{\Theta'}(E)), \tag{1}$$

where $d(\cdot,\cdot)$ is the LPIPS distance [82] and E is the event voxel. While the reconstruction loss ensures similar image conversion quality to be obtained with the new weights Θ', the adversarial loss discourages high-quality reconstruction when layers are swapped. As shown in Figure 4-(a.2) right half, the loss is defined as the sharpness of the image reconstructions made by swapping parts of the new network layers with the original weights,

$$L_{\text{adv}} = s(F^3_\Theta \circ F^2_{\Theta'} \circ F^1_{\Theta'}(E)) + s(F^3_\Theta \circ F^2_\Theta \circ F^1_{\Theta'}(E)), \tag{2}$$

where $s(\cdot)$ is the gradient magnitude computed by applying Sobel filters [83] on the reconstructions. Empirically, we found imposing the two losses in Equation 1 and 2 to be sufficient for preventing swapping weight attacks.

3) Reverse Engineering Attacks: Finally, the service provider may exploit the shared intermediate weights $F^2_{\Theta'}$ and attempt to *reverse-engineer* the unknown parts of the network. Specifically, the service provider could perform a re-training on its own using publicly available event data and the loss functions from Equation 1 and 2, but with the intermediate layer weights initialized to $F^2_{\Theta'}$. After this, the service provider could apply the reverse-engineered weight on the frontal layer activations to obtain an image reconstruction, similar to swapping weight attacks.

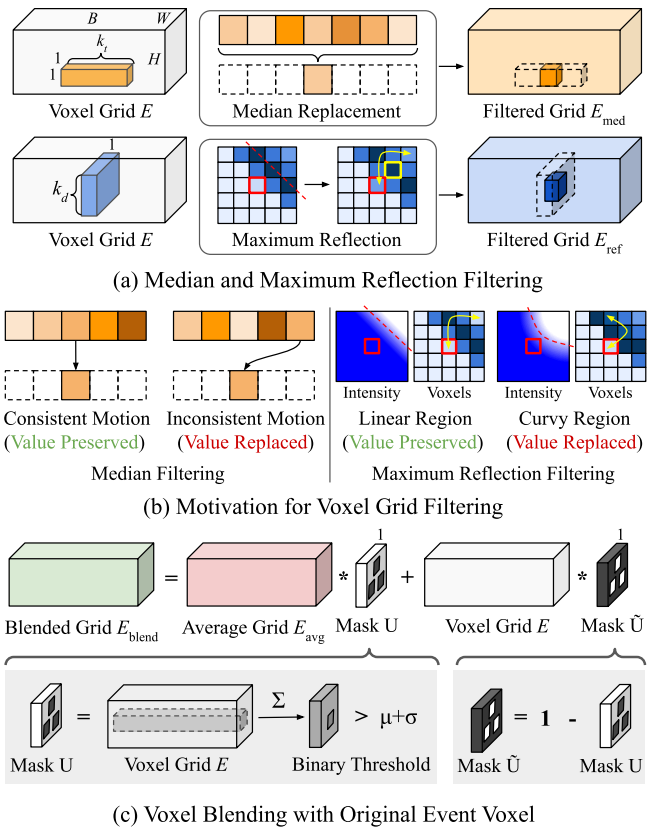


Fig. 5. Sensor-level privacy protection. (a) We attenuate temporally inconsistent regions via median filtering and curvy regions via maximum reflection filtering. (b) While the filtering operations can preserve consistent motion or linear regions, the operations will deliberately scramble the values at other regions, leading to blurry reconstructions. (c) To reduce artifacts, the averaged voxels $E_{\text{avg}}{=}(E_{\text{med}}{+}E_{\text{ref}})/2$ are selectively blended with the original voxels. Here we use a binary mask U that selects averaged voxels E_{avg} only when voxel values are over a threshold.

For defense we propose to add a noise watermark E_{noise} during the client-side re-training process, as shown in Figure 4-(a.3) right half. The noise watermark is fixed for each private scene and unknown to the service provider. The modified training objectives are as follows,

$$L_{\text{recon}} = d(F_\Theta(E), F_{\Theta'}(\tilde{E})), \tag{3}$$

$$L_{\text{adv}} = s(F^3_\Theta \circ F^2_\Theta \circ F^1_{\Theta'}(\tilde{E})) + s(F^3_\Theta \circ F^2_{\Theta'} \circ F^1_{\Theta'}(\tilde{E})), \tag{4}$$

where $\tilde{E} = E + E_{\text{noise}}$ is the noise-infused event voxel. We implement the noise watermark E_{noise} as a voxel grid with each voxels randomly sampled from the normal distribution $\mathcal{N}(0,1)$. During inference, as shown in Figure 4-(b) the user computes $F^1_{\Theta'}$ from the noise-infused voxel grid $\tilde{E}$ and sends it to the server, following the steps from Section V-A. As the noise watermark is unknown to the service provider, it obfuscates the results from user-side re-training and makes reverse engineering attacks to fail. We validate the effectiveness of network level privacy protection in Section VII-C.

VI. Sensor-Level Privacy Protection

Sensor-level privacy protection aims to hide sensitive details of nearby people, such as faces, observed by the user in both private *and* public scenes. While network-level protection can also hide scene details from the service provider, it does not address the concerns on data being abused by the user locally. Sensor-level privacy protection takes the raw event voxels as input and removes temporally inconsistent or curvy regions that are common on faces, while effectively keeping static, straight structure that is important for localization. Thus the protection scheme can balance privacy protection with localization performance. Further, this light-weight processing can potentially be implemented on the sensor directly such that user applications have no access to the raw events.

1) Median Filtering: As shown in Figure 5-(a), median filtering perturbs voxel entries with temporally inconsistent intensity or motion, which are common in events from faces. Given a voxel grid $E \in \mathbb{R}^{B\times H\times W}$ where B denotes the number of temporal bins, we replace each voxel $E(l,m,n)$ with the median value from $E(l-k_t{:}l+k_t,m,n)$ where k_t is half the temporal window size. This is based on the intuition that the event accumulation from regions with constant appearance either *monotonically* increase or decrease, where a detailed exposition is given in Appendix B-B. On the other hand, dynamic entities including human faces deform over time and the resulting voxel regions show irregularities in the temporal domain, which lead to low quality image reconstructions after filtering.

2) Maximum-Reflection Filtering: We further propose maximum-reflection filtering to attenuate curvy regions that often correspond to facial landmarks. For each voxel $E(l,m,n)$ we first find the location (l,m^*,n^*) that attains the maximum event count within the spatial neighborhood $|E(l,m-k_s{:}m+k_s,n-k_s{:}n+k_s)|$, where k_s is half the spatial window size. We then replace $E(l,m,n)$ as the voxel value at the reflected location with respect to (l,m^*,n^*), namely $E(l,2m^*-m,2n^*-n)$. The maximum-reflection filtering preserves event accumulation near lines while replacing other regions with arbitrary values. The intuition for this operation is that event accumulations near step functions are *symmetrical* with respect to the local maximum, while those on the two sides of a curved line are asymmetrical, as shown in Figure 5b. Although lines from real-world scenes are not strictly a step function, we find that in practice the maximum-reflection filtering can well-preserve events near lines while attenuating other regions including faces.

Notice that both median filtering and maximum-reflection filtering are unaware of whether there is a face or not (which can be computationally expensive to detect): they blur pixels that *likely* correspond to a face. While not explicitly detecting faces, such a design choice enables our method to be readily implemented in the sensor level. Further, despite trading off image reconstruction quality, sensor level protection incurs only a small localization performance drop. This is further verified in Section VII-D.

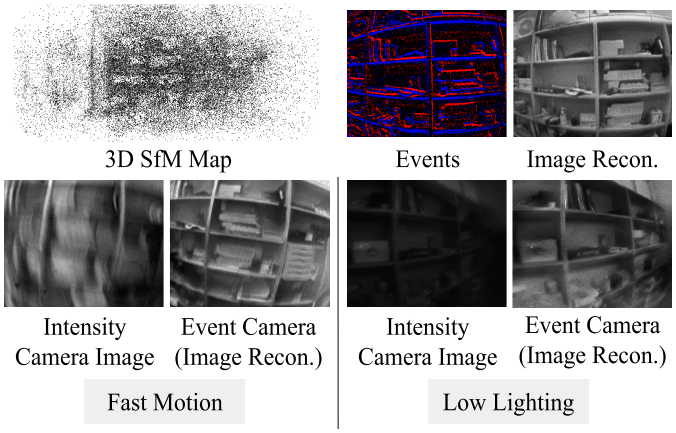
3) Voxel Blending: For voxel grid regions with an insufficient amount of accumulations, the filtering process can incur artifacts as the signal-to-noise ratio is low. Therefore, we blend the filtered voxels with the original event voxel using binary thresholding as depicted in Figure 5-(c). To elaborate, the binary mask $U \in \mathbb{R}^{B\times H\times W}$ is defined as follows,

$$U(l,m,n) = \begin{cases} 1 & \text{if } E_s(m,n) > \mu + \sigma \\ 0 & \text{otherwise,} \end{cases} \tag{5}$$

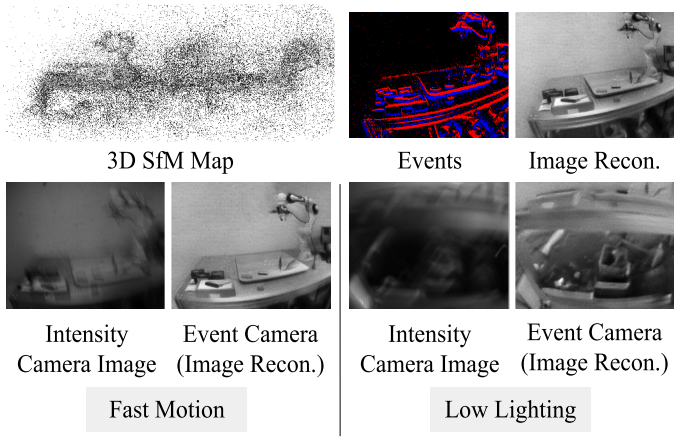
where μ, σ is the mean and standard deviation of the temporally-summed event accumulations, $E_s(m,n) = \sum_l |E(l,m,n)|$. Then, the blended voxel is given as

$$E_{\text{blend}} = U \cdot \left(\frac{E_{\text{med}} + E_{\text{max}}}{2}\right) + (1-U)\cdot E, \tag{6}$$

where $E_{\text{med}}, E_{\text{max}}$ denote the median and maximum-reflection filtered voxels respectively. The resulting blended voxel E_{blend} is then fed to the event-to-image conversion network following the localization pipeline described in Section III.



(a) *Cabinet* Scene from EvRooms



(b) *Robot* Scene from EvRooms

Fig. 6. Visualization of 3D maps, events, images, and event-to-image conversions from EvRooms. While the dataset contains challenging scenarios such as fast camera motion or low lighting, event cameras offer stable visual cues for robust localization.

VII. Experiments

We first share user study results on our event-based localization pipeline (Section VII-A), then validate the performance of our approach using event-to-image conversion (Section VII-B),

and finally show the quantitative and qualitative analysis of privacy protection (Sections VII-C, VII-D). We use three datasets for evaluation: DAVIS240C [84], EvRooms, and EvHumans. EvRooms is a newly collected dataset to evaluate the robustness of event-based localization algorithms amidst challenging external conditions. The dataset is captured in 20 scenes and divided into recordings containing fast camera motion ($\text{EvRooms}^{\text{F}}$) and low lighting ($\text{EvRooms}^{\text{L}}$), where qualitative samples of the 3D structure-from-motion (SfM) maps and images are shown in Figure 6. EvHumans is another newly collected dataset for evaluating privacy-preserving localization amidst moving people. The dataset is captured with 22 volunteers moving in 12 scenes. All three datasets are captured using the DAVIS346 [85] camera.

Furthermore, we use an RTX 2080 GPU and an Intel Core i7-7500U CPU. For event-to-image conversion, we adopt E2VID [15], which is a conversion method widely used in event-based vision applications [86], [87]. Unless specified otherwise, we use $K{=}3$ candidate poses for refinement in our localization pipeline from Section III. For results reporting accuracy, a prediction is considered correct if the translation error is below 0.1 m and the rotation error is below $5.0°$. All translation and rotation error values are median values, following [13].

TABLE I
LOCALIZATION EVALUATION AGAINST EXISTING METHODS.

Method	Description	t-error (m)	R-error (°)	Acc.
Direct	PoseNet [46]	0.15	15.94	0.05
	SP-LSTM [40]	0.19	20.30	0.03
	AECRN [23]	0.15	15.16	0.05
Structure-Based	Binary Event Image [22]	0.07	3.77	0.54
	Timestamp Image [88]	0.06	3.18	0.58
	Complementary Recon. Filter [89]	0.11	6.42	0.33
	Linear Inverse Recon. [90]	0.14	10.39	0.22
Ours	Event-to-Image Conversion	**0.04**	**2.29**	**0.69**

(a) Event-Based Localization Methods Comparison

Dataset Split	Method	t-error (m)	R-error (°)	Acc.
`Normal`	Intensity Camera	**0.04**	**1.77**	0.72
	Event Camera	0.05	2.00	**0.73**
`Low Lighting`	Intensity Camera	0.26	10.90	0.26
	Event Camera	**0.05**	**2.53**	**0.68**
`Fast Motion`	Intensity Camera	0.18	6.25	0.26
	Event Camera	**0.05**	**1.82**	**0.72**

(b) Event Cam. vs. Intensity Cam., Using Image-Based Localization

Method	t-error (m)	R-error (°)	Acc.
No Protection	**0.04**	**2.29**	**0.69**
Network Level Protection	0.05	2.58	0.64
Sensor Level Protection	0.05	2.50	0.66
Joint Protection	0.06	2.88	0.62

(c) Localization Evaluation Including Joint Protection

A. User Study

Before we conduct a detailed empirical analysis on each component of our localization pipeline, we share user study results to illustrate how people would feel about our approach. We request 39 volunteers to answer a survey that assesses how insecure people feel about various capturing scenarios, where the insecurity is scored from 1 to 5 with larger scores indicating higher insecurity. As shown in Figure 2, the survey first makes an assessment of being captured with normal cameras for various situations such as tourist spots and CCTV. Then the participants are asked about their feeling of insecurity with event cameras after seeing (1) raw event measurements, (2) image reconstructions from events, and (3) image reconstructions after network/sensor level protection. We share the full survey along with the detailed answers of the subjects in the Appendix C-E.

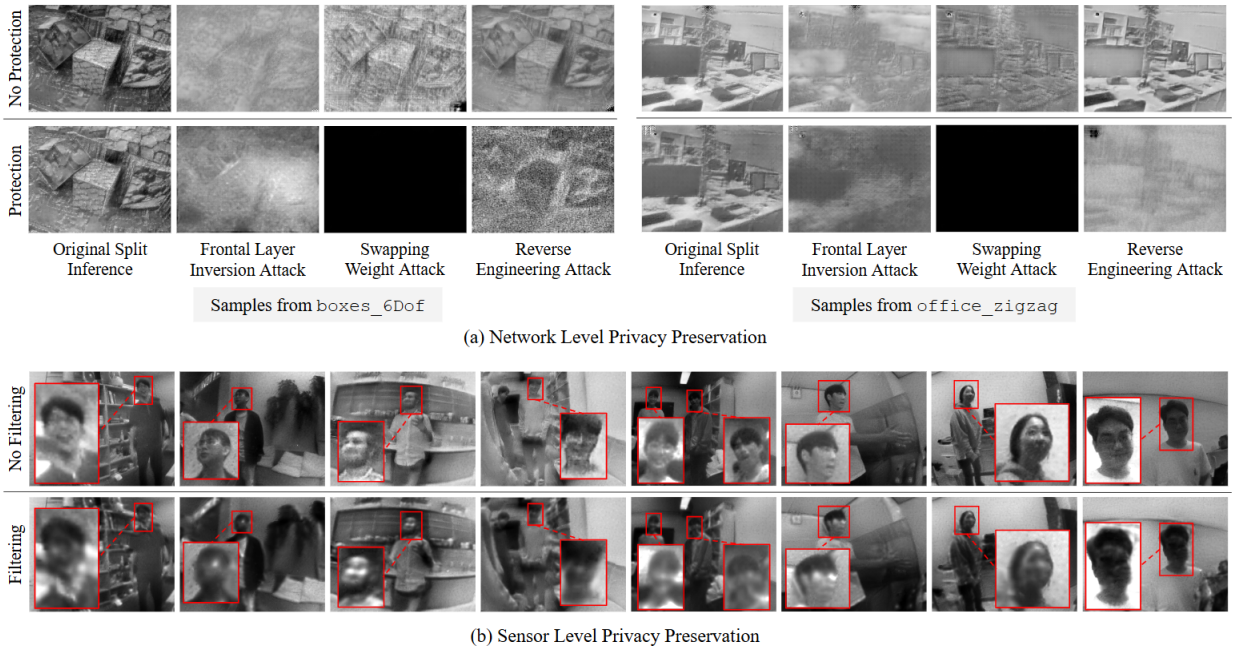
In the initial assessment from Figure 2-(a) people have varying levels of insecurity depending on the capturing scenario. The averaged insecurity scores provide a rough translation cue for interpreting the scores obtained for event cameras and privacy protections. In Figure 2-(b), the subjects first give a low insecurity score when they see the raw events but increase their score once they observe that image reconstruction is possible. The scores drop after people observe the sensor level protection results, to a level roughly equivalent to 'being captured on CCTV / friend's camera'. Furthermore, the scores are even lower for network level protection, as the scene content is completely hidden. The results show that our method can indeed alleviate the concerns presented by users of AR/VR services and observed people.

B. Localization Performance Analysis

1) Event-Based Localization Comparison: We use the DAVIS240C dataset [84] for evaluation, and consider seven baselines: direct methods (PoseNet [46], SP-LSTM [40], AECRN [23]), and structure-based methods taking as input various event representations (binary event image [22], timestamp image [88]) or light-weight image reconstructions from events (complementary reconstruction filter [89], linear inverse reconstruction [90]). Note the light-weight reconstruction methods rely on simple optimization techniques for converting events to images, which entail small computation costs but often exhibit inferior reconstruction quality compared to learning based methods as used in our approach [15], [20].

Table I-(a) shows the localization results of our method and the baselines. All structure-based methods outperform direct methods, as the pose refinement step using PnP-RANSAC [79], [91] enables accurate localization. Among the structure-based methods, our method outperforms the event representation baselines [22], [88] as the event-to-image conversion mitigates the domain gap and enables leveraging stable image feature descriptors [48], [49]. A similar trend is present when comparing against light-weight image reconstruction baselines [89], [90], where the low fidelity of these methods hinders visual feature matching, ultimately deteriorating localization performance. Therefore, our design choice of incorporating event-to-image conversion is crucial for robust performance to handle a variety of scenarios including fast camera motion and low lighting.

2) Image-Based Localization Comparison: To motivate our focus on event cameras, we implement an exemplary image-based localization method by replacing the input modality in



(a) Network Level Privacy Preservation

(b) Sensor Level Privacy Preservation

Fig. 7. Qualitative results of privacy protection. With protection, scene details in (a) and facial information in (b) are significantly removed.

our pipeline from events to intensity images that are captured as a parallel stream by DAVIS cameras. We create three splits from DAVIS240C and EvRooms: `normal`, `low lighting`, and `fast motion`, where details on data preparation are deferred to Appendix A.

Table I-(b) compares localization results under various settings. The performance of the two methods are on par in the `normal` split, as intensity camera-based localization can confidently extract good global/local features in normal conditions. However, the performance gap largely increases in the `low lighting`, and `fast motion` splits, as the motion blur and low exposure make feature extraction difficult. Due to the high dynamic range and temporal resolution of event cameras, our method can perform robust localization even in these challenging conditions.

3) Localization with Privacy Protection: We further evaluate the localization performance while using our privacy protection methods in the DAVIS240C dataset [84]. Note as shown in Figure 4-(b) network-level protection is activated with the re-trained network for image reconstruction in our localization pipeline. As shown in Table I-(c), the localization accuracy only drops mildly both for sensor and network level, which proves that our protection schemes can effectively balance secure image reconstruction with stable localization.

While network level protection hides the visual content from the service provider, observed people in private spaces may still feel uncomfortable about being captured and stored on mobile devices. Table I-(c) shows that only a small performance decrease occurs even when both levels of protection are applied. Thus our method can address privacy concerns from both users and observed people without significantly sacrificing the utility of localization.

C. Network Level Privacy Preservation Evaluation

We use six scenes from the DAVIS240C dataset to evaluate network level privacy protection. For each scene, we re-train an event-to-image conversion network following Section V. The re-training is quickly done using Adam [92] with learning rate 10^{-4} and batch size 2 for 10 epochs. Then for each trained model, we perform frontal layer inversion and reverse engineering attacks, where the models are trained using events generated from MS-COCO [81] following [15]. From the E2VID [15] architecture we use the first two layers as the frontal part ($F^1_{\Theta'}$), the last two layers as the rear part ($F^3_{\Theta'}$), and the rest as the middle part ($F^2_{\Theta'}$).

1) Attack Protection Assessment: We first assess how our method can prevent possible attacks from the service provider. For each attack, we simulate the procedure from Section V by performing image reconstruction where the frontal part of the inference uses the client's network $F_{\Theta'}$ and the latter part uses the service provider's network. Note that weight swapping is done at various network locations, and we report the averaged image difference between the attacks and the reconstructions from the original network F_Θ.

As shown in Table II, the reconstruction quality (*i.e.*, deviation from the original network inference) after network level protection is low for all attack scenarios, proving that privacy is protected. Simply per-scene re-training with random initialization offers a defense against frontal layer inversion attacks, which can be inferred from large image deviations. Adversarial loss (Equation 4) plays a key role in blocking swapping weight attacks ('Ours' vs 'Ours w/o Adversarial Loss'). For reverse-engineering attacks, a large similarity gap occurs from applying noise watermarking ('Ours' vs 'Ours w/o Noise Watermarking') which indicates the crucial role of this procedure for preventing the attack. We show visualizations of the attacks in Figure 7-(a), where all attacks fail after protection.

TABLE II
RECONSTRUCTION QUALITY (MAE) OF POSSIBLE ATTACKS USING NETWORK LEVEL PRIVACY PRESERVATION, WHERE PER-SCENE TRAINING IS PERFORMED. NOTE 'VANILLA RE-TRAINING' REFERS TO USING RECONSTRUCTION LOSS L_{RECON} ONLY.

Method \ Attack Type	Frontal Layer Inversion	Swapping Weight	Reverse Engineering
Vanilla Re-training	0.3524	0.4613	0.2165
Ours w/o Adversarial Loss	0.4212	0.4746	0.3517
Ours w/o Noise Watermark	0.4472	**0.9760**	0.3237
Ours (Re-training + Adv. Loss + Noise Wtm.)	**0.4947**	0.9587	**0.4415**

2) Computational Efficiency and Transmission Bandwidth Analysis: In addition to privacy preservation, network level protection also reduces the computational burden of running the entire event-to-image conversion on-device. This is similar in spirit to conventional client-server localization systems [1], [3], [64] that perform light-weight compute on the client side and offload majority of the computation to the server. To assess the computational efficiency, we first compare the event-to-image conversion runtime on CPU, GPU, and our method that performs splitting between the two. Here the CPU and GPU are used to model the edge device and service provider respectively. The runtime of our method is 0.08s, which is significantly lower than only using CPU (0.82s), and comparable to the case only using GPU (0.01s). To further explain such a speedup, we measure the number of floating point operations (FLOPS) for computing each part of the event-to-image conversion network. Computing intermediate parts of the conversion network (F^2_Θ: 13.82 GFLOPS) is much larger than the frontal and latter parts (F^1_Θ: 0.02 GFLOPS, F^3_Θ: 1.14 GFLOPS), which suggests that computation can be largely reduced by offloading computations of the intermediate network to the server. While these results may differ from the actual characteristics of edge devices and service providers, our method can efficiently distribute the computation and reduce the burden on the edge device side.

We finally analyze the bandwidth usage of transferring intermediate network outputs. The average size of the intermediate network outputs to be transferred back and forth between the client and server is 0.75 MB. This is comparable to the size of the data being transferred in recent split inference works [56] (1.27 MB per network inference), and would consume only a small bandwidth from a commodity 100 MBps WiFi or 20 MBps 4G / 100 MBps 5G cellular network. Therefore, transferring intermediate network outputs will not be a large bottleneck compared to the actual neural network computation. Note that in practice, the split network inference only needs to be performed *sparsely* for re-localization and camera tracking after re-localization can be performed using more light-weight event-based methods [28].

D. Sensor Level Privacy Protection Evaluation

We use the EvHumans dataset to assess sensor level protection in face blurring and localization. In all experiments, we set the half spatial/temporal window size as k_s=23, k_t=13.

1) Face Blurring Assessment: We examine face blurring in terms of low-level image characteristics and high-level semantics. For evaluation, we generate 9,755 image reconstruction pairs from the event streams with/without sensor level protection. Also, we use the publicly available FaceNet [93] and DeepFace [94] libraries for face detection and description.

TABLE III
EVALUATION ON SENSOR LEVEL PROTECTION.

Method	t-error	R-error	Acc.	# of Faces	Sharpness	Re-ID Acc.
No Protection	**0.04**	**0.99**	**0.84**	1034	0.0956	0.9387
w/o Blending	0.06	1.61	0.64	**106**	**0.0286**	**0.4670**
w/o Max Reflection	**0.05**	**1.17**	**0.77**	354	0.0483	0.5708
w/o Median Filtering	**0.05**	1.23	0.75	231	0.0461	0.5613
Ours	**0.05**	1.28	0.73	192	0.0475	0.5377

(a) Localization and Face Protection Evaluation

Region	PSNR (↓)	SSIM (↓)	MAE (↑)
Face	18.1620	0.3572	0.1974
Background	21.1243	0.6677	0.1391

(b) Reconstruction Quality of Faces and Background

Table III summarizes the evaluation results. For low-level image analysis, in Table III-(a) we report the average sharpness of the faces detected from the reconstructed images. To ensure fair comparison, we first run face detection on the non-filtered image reconstruction and use the detection results to crop both filtered/non-filtered versions. The sharpness largely drops after filtering, which indicates that our sensor level protection can effectively blur facial landmarks. Table III-(b) further supports this claim, where we measure the image similarity between the two image reconstructions separately for face and background regions. The similarity metrics are much higher for background regions, meaning that our method can keep important localization cues ample in the background while blurring out faces. Some exemplary results are shown in Figure 7-(b), where the faces are blurred out from filtering while the background features remain much less intact.

For high-level analysis, Table III-(a) reports the face detection and grouped face re-identification results. We apply a face detection algorithm [95] on the image reconstructions, where the number of detected faces largely decreases after filtering. We further analyze how the filtering obfuscates facial features with grouped face re-identification. In this task, we first divide the faces of volunteers in EvHumans to disjoint groups and apply face re-identification [96] on the detected faces to check whether it belongs to a certain group or not. Additional details regarding the task are explained in the Appendix C-D1. Similar to face detection, re-identification accuracy largely drops after filtering, indicating the efficacy of our method to obfuscate facial semantics.

2) Localization Evaluation and Ablation Study: We evaluate localization while using sensor level protection, where we pass the filtered voxels to our main localization pipeline. As shown in Table III-(a), only a small drop in accuracy occurs. While attenuating facial features, sensor level protection can preserve important features for localization.

We finally perform an ablation study on the key components of sensor level protection. As shown in Table III-(a), using the two filters makes an optimal trade-off between privacy protection and localization performance. If we ablate the median or max reflection filters, the number of detected faces increases which indicates that the faces are less protected. However, if

we ablate voxel blending, the localization accuracy drastically decreases. Each component in the sensor level protection is necessary for effective privacy-preserving localization.

VIII. Conclusion

We propose a robust event-based localization algorithm that can simultaneously protect user privacy. Our method exploits event-to-image conversion to adapt structure-based localization on event cameras for robust localization. To protect privacy during the conversion, we propose network and sensor level protection. Network level protection aims to hide the entire view for users in private scenes, whereas sensor level protection targets hiding facial landmarks. Both levels of protection are light-weight, and our experiments show that the protections incur only small performance drops in localization. We thus expect our method to be used as a practical pipeline for event-based machine vision systems.

A. Limitations and Future Work

While we present a first attempt in privacy-preserving event-based localization targeting mobile applications, we acknowledge a number of limitations that solicit future work.

a) Privacy Protection under a More Capable Adversary: Our network-level protection assumes an *honest-but-curious* service provider, that only has access to information that the client has agreed to share. However, our protection scheme may fail for a more capable adversary. To elaborate, if the service provider gains access to the frontal and lateral neural network weights which are originally kept secretly to the client, then the service provider can run the full inference pipeline to decode the shared neural network activations. In addition, if the service provider can access the raw event data captured by the client, it can run event-to-image conversion on the event data to decode the client's original view. We expect leveraging recent developments in event data encryption [97]–[99] to offer protection against such attacks.

b) Handling Other Biometric Features for Sensor-Level Protection: While our sensor-level protection can efficiently blur pixels that likely correspond to a face, we acknowledge that other biometric features such as gait or body shapes may still disclose sensitive information [100]–[104]. Note this has also been reported by several participants in the user study from Section VII-A. While increasing the level of blurring can hide other biometric features, this will come at the cost of lower localization performance. Efficient biometric obfuscation methods that preserve the localization performance can be a promising future direction.

c) Performance Analysis on Real Mobile Hardware: In the experiments section, we measured various runtime and bandwidth characteristics of our method through software-level simulations. However, experiments using real mobile hardware will better illustrate the computational requirements of the proposed method. While such experiments are more difficult to conduct as they require designing and manufacturing real mobile hardware, we acknowledge their importance for accurately measuring real-world feasibility.

d) Computational Burden for Re-training: Despite the re-training procedure only requiring small amounts of event camera data, we expect the following directions to further reducing the re-training burden. First, a single user can retrain the neural network and share it with other users in the private scene. This way, only one user needs to conduct neural network retraining. Alternatively, multiple users in the private scene may train a single neural network in a distributed manner [105], [106], which may reduce the training burden for each user.

e) Hardware Cost of Event Cameras: Despite recent efforts from manufacturers [11], [12], we acknowledge that many event cameras available for purchase today are of relatively high cost compared to conventional RGB cameras. However, the sensors are increasingly being deployed with a small form factor for mobile setups such as eye tracking in AR/VR devices [9], [10], [107] or smartphone cameras [108]. Note the pixel size of modern event cameras ($4 \sim 5\mu$m) [109] is comparable to that of commodity CMOS image sensors ($1.7 \sim 3.45\mu$m) [110]. Finally, we expect that due to economies of scale, larger utilization of these sensors will reduce the overall cost in the future.

IX. Acknowledgment

The authors would like to thank Yicheng Wu for helpful discussions, volunteers to help construct our EvHumans dataset which is used in our experiment, and volunteers to help our user study on privacy preserving evaluation.

Appendix A
Dataset Details

A. DAVIS240C

We use six scenes from DAVIS240C [84] for localization evaluation in Section VII-B. Specifically, we use `dynamic_6DoF`, `poster_6DoF`, `boxes_6DoF`, `hdr_boxes`, `hdr_poster`, and `office_zigzag` for evaluation. Among these scenes `hdr_boxes` and `hdr_poster` are recorded under low light conditions.

B. EvRooms

We collect EvRooms using the DAVIS346 camera [85] from 19 scenes, with 7 scenes additionally recorded under low lighting. As explained in Section III, we obtain the 3D maps by first converting short event streams to images [15] and using an off-the-shelf 3D reconstruction software COLMAP [77]. The dataset contains pose annotations for 18323 images converted from events in fast camera motion (EvRooms$^{\text{F}}$), along with 5022 images for events captured in low lighting (EvRooms$^{\text{L}}$). For localization evaluation in Section VII-B, we set the `Normal` split as the four scenes in DAVIS240C (`dynamic_6DoF`, `poster_6DoF`, `boxes_6DoF`, `office_zigzag`). To evaluate localization in more challenging scenarios, we set the `Low Lighting` split as the two scenes in DAVIS240C (`hdr_boxes`, `hdr_poster`) along with EvRooms$^{\text{L}}$, and the `Fast Motion` split as the scenes in EvRooms$^{\text{F}}$. We are planning to release the EvRooms dataset in public.

C. EvHumans

For evaluating localization amidst moving people, we use the newly captured dataset called EvHumans. Similar to EvRooms, this dataset was captured using the DAVIS346 camera [85] and we obtain the 3D maps using the event-to-image conversions with COLMAP [77]. The dataset consists of 20 scenes with 16 volunteers, where each scene on average contains $2\sim3$ moving people. Prior to dataset capture, we obtained the consent from all 16 volunteers under the conditions that the dataset is not made public. We additionally obtained approval from the volunteers whose faces appear in the paper figures.

APPENDIX B
SENSOR LEVEL PROTECTION DETAILS

A. Accelerating Filtering

We further reduce the runtime of sensor level protection by exploiting the fact that the blending process only keeps the filtered values for pixels with sufficient number of event accumulations. Namely, we sparsely apply the filtering operation only on the pixel regions where the binary mask in Equation 5 is non-zero. This simple optimization reduces the runtime from 4.32 s to 0.15 s for processing 3×10^5 events spanning approximately 0.45 s. Note that the runtime is measured from filters implemented using pure Python code, and the process could be further accelerated by re-implementing the filters with faster languages such as C++.

B. Justification of Median Filtering

Among the two voxel filtering steps, median filtering attenuates temporally inconsistent voxel regions. Here we give a mathematical justification of median filtering based on the event generation equation [6]. First, let $L(\mathbf{u}, t)$ denote the log intensity at pixel $\mathbf{u}$ in time t. Then, assuming constant illumination and motion, we can write the event accumulation from a short time interval $[t - \Delta t, t]$ as follows [6],

$$\Delta L(\mathbf{u}, t) = \nabla L(\mathbf{u}, t) \cdot \mathbf{v} \Delta t, \tag{7}$$

where $\mathbf{v}$ is the apparent velocity at each pixel (optical flow). Similarly, the event accumulations over a short time Δt for the neighboring timestamps $t \pm \Delta t$ could be expressed as $\Delta L(\mathbf{u}, t \pm \Delta t) = \nabla L(\mathbf{u}, t \pm \Delta t) \cdot \mathbf{v} \Delta t$. Using Taylor expansion and the constant velocity/illumination assumption, we have

$$L(\mathbf{u}, t \pm \Delta t) = L(\mathbf{u} \mp \mathbf{v}\Delta t, t), \tag{8}$$

$$L(\mathbf{u} \mp \mathbf{v}\Delta t, t) = L(\mathbf{u}, t) \mp \nabla L(\mathbf{u}, t) \cdot \mathbf{v} \Delta t. \tag{9}$$

By applying Equation 9 on Equation 7, we have

$$\nabla L(\mathbf{u}, t \pm \Delta t) = \nabla L(\mathbf{u}, t) \mp \nabla^2 L(\mathbf{u}, t) \cdot \mathbf{v} \Delta t, \tag{10}$$

$$\Delta L(\mathbf{u}, t \pm \Delta t) = \Delta L(\mathbf{u}, t) \mp \mathbf{v} \cdot \nabla^2 L(\mathbf{u}, t) \cdot \mathbf{v} (\Delta t)^2, \tag{11}$$

which indicates that in regions with constant velocity and illumination, the event accumulations are either monotone increasing or decreasing for a short period of time. Therefore, applying median filtering on voxels can preserve the accumulation values for temporally consistent regions while perturbing the values for other regions.

C. Justification of Maximum-Reflection Filtering

In maximum-reflection filtering we use the fact that pixels near straight edges have image gradient magnitudes $|\nabla L(\mathbf{u}, t)|$ that are line symmetric. Since the pixel velocity $\mathbf{v}$ for static objects are near constant under a moving camera, the event equation in Equation 7 implies that the event accumulations are nearly symmetrical (Figure 5-(b)). Thus, events near straight edges are preserved while those at curvy regions are obfuscated. Although curvy regions do not necessarily correspond to faces, faces are mostly composed of curvy regions. Therefore this filtering provides a conservative way to obfuscate faces, at the cost of slight decrease (1 cm, 0.3° from Table III) in localization performance.

APPENDIX C
EXPERIMENT DETAILS

A. Event Voxels for Image Conversion

In all our experiments we first package the input events to event voxel grids and apply event-to-image conversion methods [15], [18]. Given an event stream $\mathcal{E}$ spanning ΔT seconds, the event voxel is defined by taking weighted sums of event polarities within spatio-temporal bins. Formally, each entry of the event voxel $E \in \mathbb{R}^{B \times H \times W}$ is given as follows,

$$E(l, m, n) = \sum_{\substack{x_i = l \\ y_i = m}} p_i \max(0, 1 - |n - t_i^*|), \tag{12}$$

where $t_i^* = \frac{B-1}{\Delta T}(t_i - t_0)$ is the normalized event timestamp. For all our experiments, we set the number of temporal bins for the event voxel as $B{=}50$.

B. Localization Evaluation

1) Query/Reference Split: We explain the query and reference sets for evaluating localization. Recall from Section III that we build the 3D map from events-to-image conversions in the reference set and measure localization accuracy using the query set. For evaluation in EvRooms and DAVIS240C, we use the first 70% of the event streams for reference and the rest for querying. On the other hand for EvHumans, we randomly slice 70% of the event streams for reference and the rest for querying. The distinction is made for EvHumans because the dataset does not contain significant visual overlaps between the frontal and latter events as the capturing was made while constantly tracking the humans close by.

2) Baselines: In Section VII-B we consider nine baselines for event-based visual localization: direct methods (PoseNet [46], SP-LSTM [40], AECRN [23]), and structure-based methods taking as input various event representations (binary event image [22], timestamp image [88]) or light-weight image reconstructions from events (Scheerlink et al. [89], Zhang et al. [90]). All the direct methods are trained to directly regress the 6DoF pose within a scene, and needs to be re-trained for each scene. We train the networks separately for each scene in DAVIS240C [84], using the events in the reference split. Here the pose annotations from the 3D map are used to ensure that the network pose predictions are consistent with our method's prediction. The networks are all trained for

1400 epochs using a batch size of 20 and a learning rate of $1e{-}4$ with Adam [92].

The structure-based methods are contrived baselines that either directly package the input events for localization or leverage simple optimization techniques for converting events to images. Two conventional packaging methods are tested, namely the binary event image caching the pixel-wise event occurrences and timestamp image [88] caching the pixel-wise most recent event timestamps. In addition, we consider two optimization-based image conversion methods, where Scheerlinck et al. [89] performs numerical integration on event polarities and Zhang et al. [90] performs linear programming for image reconstruction. We use the poses from the 3D maps built from event-to-image conversions, but replace the images to corresponding input representations during localization.

C. Network Level Privacy Protection Evaluation

1) Evaluation Protocol: We elaborate on evaluating network level privacy protection, where the evaluation process is summarized in Figure A.1. We use the 6 scenes from DAVIS240C [84] as in Section VII-B and first conduct user-side private re-training for each scene (Figure A.1: **Evaluation Setup**). Here we train each conversion network using the events from the reference set described in Appendix C-B. After this, using the shared weights $F^2_{\Theta'}$, we conduct server-side reverse engineering with general scenes, namely 3,000 event sequences each spanning 2 seconds generated from the MS-COCO dataset [81] following [15], [18]. We find that fine-tuning the shared weights improved attack performance compared to freezing them, so we report the results from fine-tuning. Then, we train the frontal layer inversion network G_Φ that takes the frontal activations of the publicly available network as input and learns to regress the original event voxel (Figure A.1: **Evaluation Type 1**). We use the UNet architecture [111] to implement G_Φ and use events from MS-COCO dataset [81]. Finally, to make attacks for each scene we swap the weights during split inference using the original network weights Θ or the reverse engineering weights Θ' (Figure A.1: **Evaluation Type 2**). Note for the frontal layer inversion attack we take the shared intermediate activation and apply the inversion network G_Φ to recover the original event voxel. The result is then fed to the reconstruction network F_Θ to obtain the recovered image.

2) Network Re-training Procedure: We describe the detailed steps on re-training neural networks in the user side for network-level protection. First, the user residing in a private scene collects a small set of event camera data (approximately spanning 1 minute) and converts the raw events to voxels for re-training following Equation 12. Then, the user trains the randomly initialized reconstruction network using the training objectives from Equation 3 and 4 which includes the noise watermark added to each training sample. Finally, the user shares the intermediate part of the trained network with the service provider. During inference, given a stream of events, the user first performs inference with the frontal layer and sends the results to the service provider (Figure A.1: **Evaluation Type 2**). The service provider then performs the intermediate layer inference and returns the result to the user. The user finally performs the lateral layer inference to obtain the image reconstruction.

TABLE A.1
COMPUTATION COST COMPARISON OF RE-TRAINING E2VID [15] FOR PRIVACY PROTECTION AGAINST THE ORIGINAL SETUP AND POSENET [46], A LEARNING-BASED EVENT LOCALIZATION METHOD. THE TRAINING TIME FOR E2VID (ORIGINAL) IS OMITTED AS THE TRAINING CODE IS NOT PUBLICLY AVAILABLE.

Method	E2VID [15] (Original)	E2VID [15] (Private)	PoseNet [46]
Event Duration (s)	2000.00	70.70	70.70
Number of Epochs	160	10	1400
Training Time (hr)	N/A	0.5	48

3) Re-training Overhead Quantification: We provide a quantification of the re-training process for network-level protection from Section V. Table A.1 shows the training data characteristics (average duration of event data, number of epochs, and training time) of our retraining scheme compared against the original training setup of E2VID [15] and PoseNet [46], a learning-based localization algorithm. Our retraining is done for 10 epochs in all experiments, which takes on average 0.5 hours using an RTX 2080 GPU and 3 hours using an Intel Core i7-7500U CPU. Further, the training process consumes on average 178.63 W on the GPU and 52.75 W on the CPU. In addition, the amount of event data used for training our event-to-image conversion network is much smaller than the original setup for training E2VID: approximately 1 minute of event camera data is required. Therefore, our retraining scheme can operate quickly compared to other methods. Further, note that the retraining process is a one-time operation for each private scene: once the network is retrained, it can be continuously deployed within the same scene for visual localization.

D. Sensor Level Privacy Protection Evaluation

1) Group Re-Identification Evaluation Protocol: We elaborate on the details of the grouped face re-identification task from Table III and Section VII-D. We first divide the faces of volunteers in EvHumans to disjoint groups and apply face re-identification [96] on the detected faces to check whether it belongs to a certain group or not. To elaborate, we first run face detection [93] on the non-filtered reconstructed images and obtain the bounding boxes for the detected faces. We then use the bounding boxes to crop facial regions in the filtered image reconstructions similar to the image sharpness evaluation.

For each scene, we make two groups $\mathcal{G}_{\text{in}}$ containing the faces of all the people appearing in the scene and $\mathcal{G}_{\text{out}}$ containing the faces of the remaining people. Then we further split $\mathcal{G}_{\text{in}}$ into two groups $\mathcal{G}^{\text{ref}}_{\text{in}}, \mathcal{G}^{\text{test}}_{\text{in}}$ with $\mathcal{G}^{\text{ref}}_{\text{in}}$ containing the first 80% of the faces and $\mathcal{G}^{\text{test}}_{\text{in}}$ containing the rest. Finally, for each face in $\mathcal{G}^{\text{test}}_{\text{in}}$ we (i) extract the ArcFace [96] descriptor and compare against the descriptors extracted for faces in $\mathcal{G}^{\text{ref}}_{\text{in}}$ and $\mathcal{G}_{\text{out}}$, and (ii) choose the group with the smallest L2 descriptor distance. We finally measure the ratio of predictions with the correct group re-identifications.

2) Comparison against Gaussian and Mean Filtering: We make comparisons against more simpler design choices for

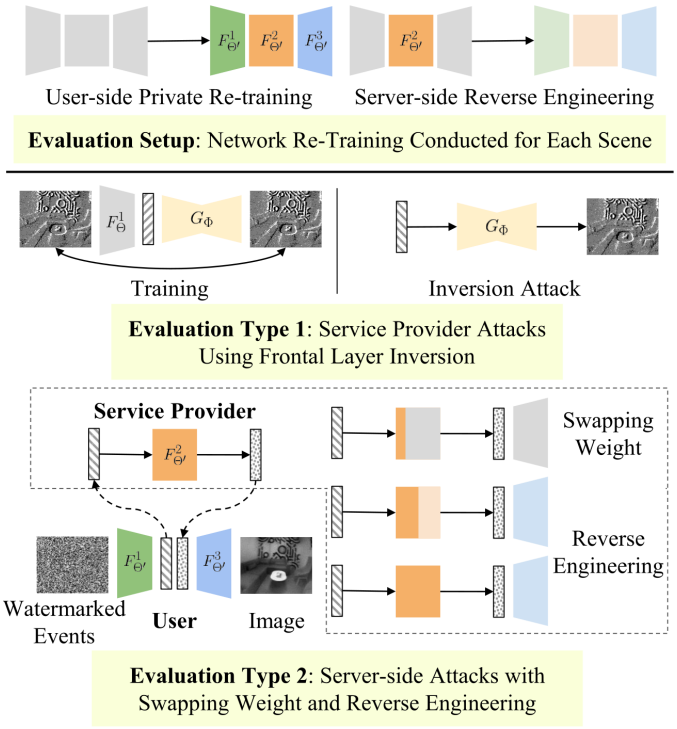


Fig. A.1. Evaluation procedure of network level privacy protection. **Evaluation Setup**: For each tested scene, we first train two sets of neural networks: private re-training in the user side (left) and reverse engineering in the service provider side (right). **Evaluation Type 1**: For frontal layer inversion attack, the service provider trains a network that learns to recover the original event voxel from intermediate layer activations. The recovered voxels are then fed to an event-to-image conversion network for evaluation. **Evaluation Type 2**: Then, to evaluate swapping weight and reverse engineering attacks we perform inferences using network weights available to the service provider. Specifically, the frontal layer activations (dashed line box) are decoded using the two types of weights available to the service provider at various swapping locations: publicly available weights (gray) for swapping weight attacks and reverse engineered weights for reverse engineering attacks (light orange).

TABLE A.2
PRIVACY PROTECTION COMPARISON AGAINST SIMPLE FILTERING METHODS, NAMELY GAUSSIAN AND MEAN FILTERING.

Method	t-error	R-error	Acc.	# of Faces	Sharpness	Re-ID Acc.
No Filtering	**0.04**	**0.99**	**0.84**	1034	0.0956	0.9387
Gaussian Filtering	0.11	2.85	0.42	**55**	**0.0341**	**0.4151**
Mean Filtering	**0.04**	1.12	0.77	372	0.0449	0.5755
Ours	0.05	1.28	0.73	192	0.0475	0.5377

voxel filtering, namely Gaussian and mean filtering. The filtering operations are performed with a spatio-temporal kernel size of $5\times5\times5$, and the blending operations is kept identically as our sensor level protection method. Note the hyperparameters for filtering were chosen to attain the optimal balance between privacy and localization performance. The results are summarized in Table A.2. Gaussian filtering results in harsher removal of feature points and incurs large drop in localization accuracy. Mean filtering on the other hand shows smaller localization performance drop but fails in protecting facial information which could be indicated from the large number of face detections. Our method better balances between privacy protection and localization performance preservation.

3) Additional Ablation Study: We conduct an additional ablation study on the effect of various hyperparameters on privacy protection and localization performance of sensor-level protection. Table A.3 summarizes the number of faces detected, which measures privacy protection, and localization accuracy under variations in filter size and voxel blending. We follow the experimental setup from Section VII-D for evaluation. While the performance of our method does not largely fluctuate by the choice of hyperparameters, our design choice of setting the median filter size $k_t = 13$, maximum-reflection filter size $k_s = 23$, and using voxel blending shows the optimal balance between privacy protection and localization accuracy.

TABLE A.3
EVALUATION ON THE EFFECT OF VARIOUS HYPERPARAMETERS ON MEDIAN (MED.) AND MAXIMUM-REFLECTION (MAX-REF.) FILTERS FOR SENSOR-LEVEL PROTECTION PERFORMANCE.

Method	Med. Filter Size	Max-Ref. Filter Size	Voxel Blending	Loc. Acc.	# of Faces
w/o Voxel Blending	13	23	✗	0.64	**106**
w/ Larger Med. Filter	21	23	◯	0.71	185
w/ Smaller Med. Filter	7	23	◯	0.72	210
w/ Larger Max-Ref. Filter	13	31	◯	0.72	184
w/ Smaller Max-Ref. Filter	13	11	◯	**0.75**	278
Ours	13	23	◯	0.73	192

"I don't think there's a need to hide information other than face information during the pre-processing."

"With this preprocessing, it's hard to distinguish the silhouettes of people, therefore less concerning in the perspective of privacy."

"The reconstructed images absolutely hide the sensitive details."

"I think people cannot recognize my facial information after pre-processing takes place."

"I would feel less insecure with pre-processing since it blurs out the facial details to the point where individuals are hardly verifiable."

"I feel the average person will not be able to recognize the original face behind the filtered results."

"Network level privacy protection seems to do a really good job in hiding the scene details."

"I think faces are the most 'unique' features to recognize a person. The proposed pre-processing can hide facial landmarks very well. It would be better if we can also hide gestures and other biometric features, but they are less 'unique' compared to faces."

"It might be possible to identify someone by his/her unique body features or personal habits."

"It can be dangerous if one can reverse the pre-processing, or try to track someone using gait and gesture information."

"I'm not sure if the pre-processing will provide enough protection if there is something like a 'color' event camera. It may be more harder to discern people because all the information is in grayscale."

Fig. A.2. Positive and negative comments from the user study about our privacy protection method.

E. User Study

1) User Comments: Along with the insecurity scoring, we requested the users to optionally leave comments about their evaluations. We share the user comments in Figure A.2. Overall, the positive comments emphasize the fact that for sensor level protection the facial details are sufficiently removed and for network level protection the reconstructions successfully hide the entire user's view. Nevertheless, there were some negative comments about the sensor level protection that other signals such as unique gestures or body features are not obscured. Devising a privacy protection method that can cover a wider range of biometric signals is left as future work.

2) Full Survey: We share the full survey content used for the user study as a separate file `survey.pdf`. Note that we showed multiple videos for each scenario in the actual user study, and for sensor level evaluation we additionally showed samples of cropped faces reconstructed after processing.

REFERENCES

[1] M. Dusmanu, J. Schönberger, S. Sinha, and M. Pollefeys, "Privacy-preserving image features via adversarial affine subspace embeddings," in *CVPR*, 2021.

[2] M. Geppert, V. Larsson, P. Speciale, J. L. Schönberger, and M. Pollefeys, "Privacy preserving structure-from-motion," in *CVPR*, 2020.

[3] T. Ng, H. J. Kim, V. T. Lee, D. DeTone, T.-Y. Yang, T. Shen, E. Ilg, V. Balntas, K. Mikolajczyk, and C. Sweeney, "Ninjadesc: Content-concealing visual descriptors via adversarial learning," in *CVPR*, 2022.

[4] D. Dangwal, V. T. Lee, H. J. Kim, T. Shen, M. Cowan, R. Shah, C. Trippel, B. Reagen, T. Sherwood, V. Balntas, A. Alaghi, and E. Ilg, "Analysis and mitigations of reverse engineering attacks on local feature descriptors," *arXiv*, 2021.

[5] C. Cachin, I. Keidar, and A. Shraer, "Trusting the cloud," *SIGACT News*, 2009.

[6] G. Gallego, T. Delbrück, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. Davison, J. Conradt, K. Daniilidis, and D. Scaramuzza, "Event-based vision: A survey," *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[7] 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*, 2008.

[8] Y. Fu, M. Li, W. Liu, Y. Wang, J. Zhang, B. Yin, X. Wei, and X. Yang, "Distractor-aware event-based tracking," *IEEE Transactions on Image Processing*, vol. 32, pp. 6129–6141, 2023.

[9] S. Mentasti, F. Lattari, R. Santambrogio, G. Careddu, and M. Matteucci, "Event-based eye tracking for smart eyewear," in *ETRA*, 2024.

[10] "Event-based vision for eye tracking," https://www.prophesee.ai/event-based-vision-eye-tracking/, accessed: 2025-04-08.

[11] "Event-based vision for mobile imaging," https://www.prophesee.ai/event-based-vision-mobile/, accessed: 2025-04-08.

[12] "Event-based vision sensor (evs) technology," https://www.sony-semicon.com/en/technology/industry/evs.html, accessed: 2025-04-08.

[13] P.-E. Sarlin, C. Cadena, R. Siegwart, and M. Dymczyk, "From coarse to fine: Robust hierarchical localization at large scale," in *CVPR*, 2019.

[14] M. Humenberger, Y. Cabon, N. Guérin, J. Morat, J. Revaud, P. Rerole, N. Pion, C. R. de Souza, V. Leroy, and G. Csurka, "Robust image retrieval-based visual localization using kapture," *CoRR*, 2020.

[15] H. Rebecq, R. Ranftl, V. Koltun, and D. Scaramuzza, "Events-to-video: Bringing modern computer vision to event cameras," in *CVPR*, 2019.

[16] L. Wang, I. S. M. Mostafavi, Y. Ho, and K. Yoon, "Event-based high dynamic range image and very high frame rate video generation using conditional generative adversarial networks," in *CVPR*, 2019.

[17] C. Scheerlinck, N. Barnes, and R. Mahony, "Continuous-time intensity estimation using event cameras," in *ACCV*, 2019.

[18] C. Scheerlinck, H. Rebecq, D. Gehrig, N. Barnes, R. E. Mahony, and D. Scaramuzza, "Fast image reconstruction with an event camera," in *WACV*, 2020.

[19] T. Stoffregen, C. Scheerlinck, D. Scaramuzza, T. Drummond, N. Barnes, L. Kleeman, and R. Mahoney, "Reducing the sim-to-real gap for event cameras," in *ECCV*, 2020.

[20] P. R. G. Cadena, Y. Qian, C. Wang, and M. Yang, "Spade-e2vid: Spatially-adaptive denormalization for event-based video reconstruction," *IEEE Transactions on Image Processing*, vol. 30, pp. 2488–2500, 2021.

[21] B. Ercan, O. Eker, C. Saglam, A. Erdem, and E. Erdem, "Hypere2vid: Improving event-based video reconstruction via hypernetworks," *IEEE Transactions on Image Processing*, vol. 33, pp. 1826–1837, 2024.

[22] G. Cohen, S. Afshar, G. Orchard, J. Tapson, R. Benosman, and A. van Schaik, "Spatial and temporal downsampling in event-based visual classification," *IEEE Transactions on Neural Networks and Learning Systems*, 2018.

[23] H. Lin, M. Li, Q. Xia, Y. Fei, B. Yin, and X. Yang, "6-dof pose relocalization for event cameras with entropy frame and attention networks," in *Proceedings of the 18th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry*, ser. VRCAI '22. New York, NY, USA: Association for Computing Machinery, 2023. [Online]. Available: https://doi.org/10.1145/3574131.3574457

[24] Y. Jin, L. Yu, G. Li, and S. Fei, "A 6-dofs event-based camera relocalization system by cnn-lstm and image denoising," *Expert Systems with Applications*, 2021.

[25] H. Kim, A. Handa, R. Benosman, S.-H. Ieng, and A. Davison, "Simultaneous mosaicing and tracking with an event camera," in *BMVC*, 2014.

[26] H. Kim, S. Leutenegger, and A. J. Davison, "Real-time 3d reconstruction and 6-dof tracking with an event camera," in *ECCV*, 2016.

[27] J. Jiao, H. Huang, L. Li, Z. He, Y. Zhu, and M. Liu, "Comparing representations in tracking for event camera-based SLAM," *CoRR*, 2021.

[28] J. Hidalgo-Carrió, G. Gallego, and D. Scaramuzza, "Event-aided direct sparse odometry," in *CVPR*, 2022.

[29] G. Gallego, J. E. Lund, E. Mueggler, H. Rebecq, T. Delbruck, and D. Scaramuzza, "Event-based, 6-dof camera tracking from photometric depth maps," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[30] S. Bryner, G. Gallego, H. Rebecq, and D. Scaramuzza, "Event-based, direct camera tracking from a photometric 3d map using nonlinear optimization," in *ICRA*, 2019.

[31] "Slam handbook," https://github.com/SLAM-Handbook-contributors/slam-handbook-public-release, accessed: 2025-04-08.

[32] S. Klenk, M. Motzet, L. Koestler, and D. Cremers, "Deep event visual odometry," in *3DV*, 2024.

[33] N. Messikommer, C. Fang, M. Gehrig, G. Cioffi, and D. Scaramuzza, "Data-driven feature tracking for event cameras with and without frames," *TPAMI*, 2025.

[34] N. Messikommer*, C. Fang*, M. Gehrig, and D. Scaramuzza, "Data-driven feature tracking for event cameras," *CVPR*, 2023.

[35] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperGlue: Learning feature matching with graph neural networks," in *CVPR*, 2020.

[36] A. J. Lee and A. Kim, "Eventvlad: Visual place recognition with reconstructed edges from event cameras," in *IROS*, 2021.

[37] J. Ren, X. Jiang, Z. Li, D. Liang, X. Zhou, and X. Bai, "Minima: Modality invariant image matching," in *CVPR*, 2025.

[38] S. L. Yannick Burkhardt, Simon Schaefer, "Superevent: Cross-modal learning of event-based keypoint detection for slam," *arXiv preprint arXiv:2504.00139*, 2025.

[39] Z. Huang, L. Sun, C. Zhao, S. Li, and S. Su, "Eventpoint: Self-supervised interest point detection and description for event-based camera," in *WACV*, 2023.

[40] A. Nguyen, T.-T. Do, D. G. Caldwell, and N. G. Tsagarakis, "Real-time 6dof pose relocalization for event cameras with stacked spatial lstm networks," in *CVPR Workshops*, 2019.

[41] T. Pan, J. He, C. Chen, Y. Li, and C. Feng, "Nyc-event-vpr: A large-scale high-resolution event-based visual place recognition dataset in dense urban environments," in *ICRA*, 2025.

[42] J. Kim, J. Bae, G. Park, D. Zhang, and Y. M. Kim, "N-imagenet: Towards robust, fine-grained object recognition with event cameras," in *ICCV*, 2021.

[43] G. Orchard, A. Jayawant, G. K. Cohen, and N. Thakor, "Converting static image datasets to spiking neuromorphic datasets using saccades," *Frontiers in Neuroscience*, vol. 9, p. 437, 2015.

[44] B. Kueng, E. Mueggler, G. Gallego, and D. Scaramuzza, "Low-latency visual odometry using event-based feature tracks," in *IROS*, 2016.

[45] T. Sattler, Q. Zhou, M. Pollefeys, and L. Leal-Taixe, "Understanding the limitations of cnn-based absolute camera pose regression," in *CVPR*, 2019.

[46] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization," *ICCV*, 2015.

[47] F. Walch, C. Hazirbas, L. Leal-Taixe, T. Sattler, S. Hilsenbeck, and D. Cremers, "Image-based localization using lstms for structured feature correlation," in *ICCV*, 2017.

[48] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," in *CVPR*, 2016.

[49] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superpoint: Self-supervised interest point detection and description," in *CVPR Deep Learning for Visual SLAM Workshop*, 2018.

[50] B. Fan, J. Zhou, W. Feng, H. Pu, Y. Yang, Q. Kong, F. Wu, and H. Liu, "Learning semantic-aware local features for long term visual localization," *IEEE Transactions on Image Processing*, vol. 31, pp. 4842–4855, 2022.

[51] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–110, 2004.

[52] S. Hu, Q. Wang, J. Wang, Z. Qin, and K. Ren, "Securing sift: Privacy-preserving outsourcing computation of feature extractions over encrypted image data," *IEEE Transactions on Image Processing*, vol. 25, no. 7, pp. 3411–3425, 2016.

[53] S. Kumawat and H. Nagahara, "Privacy-preserving action recognition via motion difference quantization," in *ECCV*, 2022.

[54] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft, "Privacy-preserving face recognition," in *Privacy Enhancing Technologies*, 2009.

[55] Y. Wang, J. Liu, M. Luo, L. Yang, and L. Wang, "Privacy-preserving face recognition in the frequency domain," in *AAAI*, 2022.

[56] M. Yang, Z. Li, J. Wang, H. Hu, A. Ren, X. Xu, and W. Yi, "Measuring data reconstruction defenses in collaborative inference systems," in *NeurIPS*, 2022.

[57] Z. He, T. Zhang, and R. B. Lee, "Model inversion attacks against collaborative inference," in *ACSAC*, 2019.

[58] A. E. Eshratifar, M. S. Abrishami, and M. Pedram, "Jointdnn: An efficient training and inference engine for intelligent mobile cloud computing services," *IEEE Transactions on Mobile Computing*, 2021.

[59] A. Banitalebi-Dehkordi, N. Vedula, J. Pei, F. Xia, L. Wang, and Y. Zhang, "Auto-split: A general framework of collaborative edge-cloud ai," in *KDD*, 2021.

[60] Z. Li, M. Yang, Y. Liu, J. Wang, H. Hu, W. Yi, and X. Xu, "GAN you see me? enhanced data reconstruction attacks against split inference," in *NeurIPS*, 2023.

[61] H. Liu, M. E. Fouda, A. M. Eltawil, and S. A. Fahmy, "Split dnn inference for exploiting near-edge accelerators," in *EDGE*, 2024.

[62] A. Singh, V. Sharma, R. Sukumaran, J. Mose, J. Chiu, J. Yu, and R. Raskar, "Simba: Split inference—mechanisms, benchmarks and attacks," in *ECCV*, 2024.

[63] Y. Lindell, "Secure multiparty computation (MPC)," Cryptology ePrint Archive, Paper 2020/300, 2020. [Online]. Available: https://eprint.iacr.org/2020/300

[64] P. Speciale, J. Schönberger, S. Sinha, and M. Pollefeys, "Privacy preserving image queries for camera localization," in *ICCV*, 2019b.

[65] P. Speciale, J. Schönberger, S. B. Kang, S. N. Sinha, and M. Pollefeys, "Privacy preserving image-based localization," in *CVPR*, 2019a.

[66] L. Pan, J. L. Schönberger, V. Larsson, and M. Pollefeys, "Privacy preserving localization via coordinate permutations," in *ICCV*, 2023.

[67] K. Chelani, A. Benbihi, F. Kahl, T. Sattler, and Z. Kukelova, "Obfuscation based privacy preserving representations are recoverable using neighborhood information," in *3DV*, 2025.

[68] Q. Zhou, S. Agostinho, A. Osep, and L. Leal-Taixé, "Is geometry enough for matching in visual localization?" in *ECCV*, 2022.

[69] M. Pietrantoni, M. Humenberger, T. Sattler, and G. Csurka, "Segloc: Learning segmentation-based representations for privacy-preserving visual localization," in *CVPR*, 2023.

[70] Z. Tasneem, G. Milione, Y.-H. Tsai, X. Yu, A. Veeraraghavan, C. Manmohan, and F. Pittaluga, "Learning phase mask for privacy-preserving passive depth estimation," in *ECCV*, 2022.

[71] J. Tan, S. S. Khan, V. Boominathan, J. Byrne, R. Baraniuk, K. Mitra, and A. Veeraraghavan, "Canopic: Pre-digital privacy-enhancing encodings for computer vision," in *ICME*, 2020.

[72] T. Winkler, A. Erdelyi, and B. Rinner, "Trusteye.m4: Protecting the sensor — not the camera," in *AVSS*, 2014.

[73] F. Pittaluga and S. J. Koppal, "Privacy preserving optics for miniature vision sensors," in *CVPR*, 2015.

[74] E. Brachmann and C. Rother, "Learning less is more - 6D camera localization via 3D surface regression," in *CVPR*, 2018.

[75] A. Z. Zhu, L. Yuan, K. Chaney, and K. Daniilidis, "Unsupervised event-based optical flow using motion compensation," in *ECCV Workshops*, 2019.

[76] N. Messikommer, D. Gehrig, A. Loquercio, and D. Scaramuzza, "Learning to see in the dark with events," in *ECCV*, 2020.

[77] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *CVPR*, 2016.

[78] J. A. Hesch and S. I. Roumeliotis, "A direct least-squares (dls) method for pnp," in *ICCV*, 2011.

[79] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography." *Commun. ACM*, 1981.

[80] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, "Gazelle: A low latency framework for secure neural network inference," in *USENIX*, 2018.

[81] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *ECCV*, 2014.

[82] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *CVPR*, 2018.

[83] N. Kanopoulos, N. Vasanthavada, and R. L. Baker, "Design of an image edge detection filter using the sobel operator," *IEEE Journal of solid-state circuits*, 1988.

[84] E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck, and D. Scaramuzza, "The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and slam," *The International Journal of Robotics Research*, 2017.

[85] Inivation, "Davis 346 camera," https://shop.inivation.com/products/davis346, 2022, accessed: 2024-03-01.

[86] T. Fischer and M. Milford, "Event-based visual place recognition with ensembles of temporal windows," *IEEE Robotics and Automation Letters*, 2020.

[87] M. Muglikar, M. Gehrig, D. Gehrig, and D. Scaramuzza, "How to calibrate your event camera," in *CVPR Workshops*, 2021.

[88] P. K. J. Park, B. H. Cho, J. M. Park, K. Lee, H. Y. Kim, H. A. Kang, H. G. Lee, J. Woo, Y. Roh, W. J. Lee, C. Shin, Q. Wang, and H. Ryu, "Performance improvement of deep learning based gesture recognition using spatiotemporal demosaicing technique," in *ICIP*, 2016.

[89] C. Scheerlinck, N. Barnes, and R. Mahony, "Continuous-time intensity estimation using event cameras," in *Asian Conf. Comput. Vis. (ACCV)*, December 2018, pp. 308–324.

[90] Z. Zhang, A. J. Yezzi, and G. Gallego, "Formulating event-based image reconstruction as a linear inverse problem with deep regularization using optical flow," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 7, pp. 8372–8389, 2023.

[91] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o(n) solution to the pnp problem," *International Journal Of Computer Vision*, 2009.

[92] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.

[93] FaceNet, "Facenet-pytorch," https://github.com/timesler/facenet-pytorch, 2022, accessed: 2024-03-01.

[94] S. I. Serengil and A. Ozpinar, "Lightface: A hybrid deep face recognition framework," in *2020 Innovations in Intelligent Systems and Applications Conference (ASYU)*. IEEE, 2020, pp. 23–27. [Online]. Available: https://doi.org/10.1109/ASYU50717.2020.9259802

[95] J. Xiang and G. Zhu, "Joint face detection and facial expression recognition with mtcnn," in *ICISCE*, 2017.

[96] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *CVPR*, 2019.

[97] P. Zhang, S. Zhu, and E. Y. Lam, "Event encryption: Rethinking privacy exposure for neuromorphic imaging," *Neuromorphic Computing and Engineering*, 2024.

[98] W. Shariff, M. S. Dilmaghani, P. Kielty, M. Moustafa, J. Lemley, and P. Corcoran, "Event cameras in automotive sensing: A review," *IEEE Access*, 2024.

[99] S. Zhu, C. Wang, J. Huang, P. Zhang, J. Han, and E. Y. Lam, "Neuromorphic encryption: combining speckle correlography and event data for enhanced security," *Advanced Photonics Nexus*, 2024.

[100] Y. Wang, B. Du, Y. Shen, K. Wu, G. Zhao, J. Sun, and H. Wen, "Ev-gait: Event-based robust gait recognition using dynamic vision sensors," in *CVPR*, 2019.

[101] K. Ma, Y. Fu, C. Cao, S. Hou, Y. Huang, and D. Zheng, "Learning visual prompt for gait recognition," in *CVPR*, 2024.

[102] K. Ma, Y. Fu, D. Zheng, C. Cao, X. Hu, and Y. Huang, "Dynamic aggregated network for gait recognition," in *CVPR*, 2023.

[103] D. Ye, C. Fan, J. Ma, X. Liu, and S. Yu, "Biggait: Learning gait representation you want by large vision models," in *CVPR*, 2024.
[104] R. Liu and C. Vondrick, "Humans as light bulbs: 3d human reconstruction from thermal reflection," in *CVPR*, Jun. 2023.
[105] N. Ivkin, D. Rothchild, E. Ullah, I. Stoica, R. Arora *et al.*, "Communication-efficient distributed sgd with sketching," in *NeurIPS*, 2019.
[106] S. Teerapittayanon, B. McDanel, and H.-T. Kung, "Distributed deep neural networks over the cloud, the edge and end devices," in *ICDCS*, 2017.
[107] A. N. Angelopoulos, J. N. Martel, A. P. Kohli, J. Conradt, and G. Wetzstein, "Event-based near-eye gaze tracking beyond 10,000 hz," *IEEE TVCG*, 2021.
[108] "Hybrid vision sensors and integrated machine vision solutions," https://www.alpsentek.ch/scene/detail#mobile-phone/, accessed: 2025-07-22.
[109] "Imx636 event sensor," https://www.prophesee.ai/event-camera-evk4/, accessed: 2025-07-22.
[110] "Sensor and pixel sizes of industrial cameras," https://www.vision-doctor.com/en/camera/sensor-pixel-sizes.html/, accessed: 2025-07-22.
[111] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *MICCAI*, 2015.

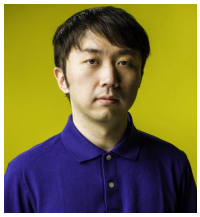
Junho Kim received the bachelor's degree from Seoul National University and the Ph.D. degree from the same university in 2025. He is currently a postdoctoral scholar at Seoul National University. He has published papers at prestigious venues including CVPR, ICCV, and ECCV. His research interests include 3D reconstruction, scene understanding, and event-based vision.

Young Min Kim is an Associate Professor in the Department of Electrical and Computer Engineering at Seoul National University, Seoul, Korea, where she is leading a 3D vision lab. She received a B.S. from Seoul National University in 2006 and an M.S. and Ph.D. in electrical engineering from Stanford University in 2008 and 2013, respectively. Before joining SNU, she was a Senior Research Scientist at the Korea Institute of Science and Technology (KIST). Her research interest lies in 3D vision, where she combines computer vision, graphics, and robotics algorithms to solve practical problems. She serves as an area chair in CVPR, ICCV, ACCV, program committee for Pacific Graphics, AAAI, and technical papers committee for SIGGRAPH Asia. She is also a program chair for 3DV 2026.

Ramzi Zahreddine received his B.S. in Optical Engineering from Rose-Hulman Institute of Technology, and his M.S. and Ph.D. in Electrical Engineering from University of Colorado Boulder. He is currently an Optical Sensing Incubation Engineer at Snap Inc. His research interests center around the intersection of computational imaging and novel optical components applied to augmented reality systems.

Weston A. Welge received the Ph.D. degree in optical sciences from The University of Arizona in 2016. He currently leads the camera and sensing hardware team at Snap Inc. responsible for sensing R&D for augmented reality glasses.

Gurunandan Krishnan is the Sr. SVP of Machine Learning at OtoNexus Medical Technologies. He holds a Masters degree in Computer Science from Columbia University. His research interests include computer vision, imaging, acoustics, augmented reality and machine learning at the edge.



Sizhuo Ma is a Research Scientist at Snap Inc. He obtained a Ph.D. degree from University of Wisconsin-Madison in 2022. He received his bachelor's degree from Shanghai Jiao Tong University. His research interests include computational imaging, computational photography and low-level vision. He has published papers at prestigious journals and conferences including CVPR, ECCV, Siggraph, MobiCom, ISMAR, etc.

Jian Wang is a Staff Research Scientist at Snap Inc., focusing on computational photography and imaging. He has published in top-tier venues such as CVPR, MobiCom, and SIGGRAPH, and has contributed numerous features to production. He has received the best paper award from SIGGRAPH Asia, 2024, and the 4th IEEE International Workshop on Computational Cameras and Displays, and the best poster award from IEEE Conference on Computational Photography 2022. He has served as an Area Chair for CVPR, NeurIPS, ICLR, ICML, etc. Jian holds a Ph.D. from Carnegie Mellon University.