SPICE: Self-supervised Predictive Coding of Events

by

Tim N.A. den Blanken

to obtain the degree of Master of Science in Robotics
at the Delft University of Technology,
to be defended publicly on Thursday December 18, 2025 at 14:00.

Student number:	5294525	
Project duration:	October 8, 2024 – December 18, 2025	
Thesis committee:	Ph.d. candidate Y. Wu,	TU Delft, daily supervisor
	Prof. dr. G.C.H.E. de Croon,	TU Delft, daily supervisor
	Dr. H. Caesar.	TU Delft, CoR supervisor

An electronic version of this thesis is available at http://repository.tudelft.nl/.

SPICE: Self-supervised Predictive Coding of Events

Tim N.A. den Blanken Yilun Wu Guido C.H.E. de Croon Holger Caesar
Delft University of Technology

Abstract

Event-based cameras provide high temporal resolution, robustness to lighting conditions and low power consumption, but their sparse, temporal data require models that reason over time. In supervised settings, this is increasingly handled with recurrent architectures. In contrast, most self-supervised learning (SSL) methods still adapt non-recurrent RGB techniques, with masking-based objectives that favor spatial reconstruction over temporal understanding. We introduce SPICE: Self-supervised Predictive Coding on Events, an SSL framework tailored to event data that processes longer sequences recurrently and learns by predicting future latent representations rather than reconstructing masked inputs, promoting a more natural objective focyused on anticipating what comes next. SPICE further incorporates an event-specific contrastive loss only operating on active regions. SPICE pre-training improves downstream performance on semantic segmentation, depth estimation and optical flow estimation. Low-dimensional projections confirm that the learned representations are meaningful and avoid collapse, while also revealing limitations in temporal stability and semantic organization, indicating clear directions for future event-specific SSL research. Code is available upon request.

1. Introduction

Event cameras are bio-inspired vision sensors that asynchronously record per-pixel changes in brightness. Unlike conventional frame-based cameras, they produce a continuous stream of events with microsecond latency, high dynamic range, and low power consumption [17]. These properties make them attractive for fast and low-power robotics applications, and they have been successfully applied to various vision tasks [8, 11, 52, 60, 72]. However, their asynchronous and sparse output fundamentally differs from RGB images, which poses unique challenges for learning-based methods.

Despite growing interest, event-based models still trail behind their RGB counterparts on high-level tasks such as semantic segmentation [69] and depth estimation [38]. Two main factors contribute to this gap: (i) the limited availability of large labeled event datasets [8, 11, 20, 72], and (ii) the comparatively limited exploration of architectures and training schemes specifically tailored to the asynchronous and sparse nature of event data. In conventional vision, similar data-scarce regimes have been effectively addressed through self-supervised pre-training, which enables the learning of generalizable visual representations from unlabeled data [10, 12, 22, 27, 49]. These methods have since matured, with architectures, data augmentations, and learning objectives highly optimized for frame-based RGB data.

Encouraged by their success in conventional vision, several works have attempted to extend self-supervised learning (SSL) to event data. Most existing approaches, however, directly adapt RGB image-based methods by first aggregating short time windows of events into image-like representations and then applying standard contrastive [67], reconstruction-based [31, 32, 68], or correlation-based [5] objectives. Aggregating only brief temporal intervals leads to sparse event representations that provide weak or ambiguous learning signals, preventing these methods from exploiting the temporal structure inherent to event streams. Consequently, directly transferring RGB image-oriented SSL pipelines to event data remains suboptimal and highlights the need for methods specifically designed for the characteristics of event cameras. Moreover, recent event-based SSL works increasingly rely on masking strategies, which have shown strong empirical results, but such objectives remain somewhat artificial and unlike how humans naturally learn from continuous sensory input. This motivates us to explore an alternative, more intuitive route grounded in temporal prediction.

We address these limitations with SPICE, a self-supervised pre-training framework explicitly designed for event data. SPICE processes sequences of event voxels over extended temporal horizons and learns to predict future latent representations from past ones, following the intuition that understanding the past enables predicting the future [25, 48]. A contrastive loss is employed for its stability, interpretability, and strong empirical performance in vision pre-training [12, 27]. To handle the sparsity of events, our

model aggregates features recurrently and focuses computation on regions with sufficient activity. Finally, to alleviate data scarcity, we pretrain on events simulated from large RGB datasets, bridging the gap between event-based and frame-based learning.

The main contributions of this work are: (i) a self-supervised learning framework for event data based on *future latent prediction*; (ii) an event-specific contrastive loss; (iii) a loss focusing strategy that restricts optimization to active patches; and (iv) an analysis-driven design process where visualization of representation dynamics using non-linear projection techniques provide insight into semantic structure and temporal stability.

2. Related Work

Self-Supervised Learning for RGB Images and Videos. SSL has become central to visual representation learning, enabling feature extraction from large unlabeled image and video corpora. Existing approaches for RGB data can be grouped into five different paradigms: clustering, reconstruction, contrastive learning, correlation analysis, and self-distillation [14].

Clustering-based methods iteratively group samples and use the cluster assignments as pseudo-labels to train the encoder (e.g., DeepCluster [9], SeLa [1]). Reconstruction approaches train models to predict transformations or masked regions of the input and include both explicit transformation prediction and masked-token style objectives (e.g., Doersch *et al*. [16], BEiT [4], I-JEPA [2]). Contrastive learning enforces instance-level discrimination by pulling together positive pairs and pushing apart negatives; foundational works include Contrastive Predictive Coding (CPC) [48], SimCLR [12], and MoCo [27]. Video extensions exploit temporal consistency between frames or clips (e.g., VideoMoCo [50]) and predictive contrastive schemes such as DPC [25] and its memory-augmented successor MemDPC [26]. To avoid reliance on large negative sets, correlation-based methods such as Barlow Twins [71] and VICReg [6] enforce feature decorrelation and minimum-variance constraints between differently augmented views of the same input. Self-distillation methods (e.g., BYOL [22], DINO [10], and DINOv2 [49]) likewise operate on two augmented views but prevent collapse through asymmetric teacher–student or EMA mechanisms rather than decorrelation or variance constraints; several of these ideas have been adapted to video-level and multi-view settings [7, 54, 66].

Contrastive Self-Supervised Learning. Contrastive SSL aims to learn discriminative representations by pulling together embeddings of related samples while pushing apart those of unrelated ones. The Noise Contrastive Estimation (NCE) framework [23] laid a foundation for this idea, later extended by Contrastive Predictive Coding (CPC) [48], which introduced the InfoNCE loss to maximize mutual information between context and future representations. This objective has since become one of the most widely adopted formulations in contrastive learning, with numerous variants such as weighted and soft versions (Soft-InfoNCE) [33] and formulations supporting multiple positives (MIL-InfoNCE) [46]. This work introduces a variant of the soft version. Around the same time, image-based methods such as SimCLR [12] and MoCo [27] popularized contrastive learning at scale by contrasting global embeddings from augmented views.

For spatiotemporal data, DPC [25] extended CPC by predicting future feature maps rather than global embeddings. This approach directly inspires our method. Its successor, Memory-Augmented DPC (MemDPC) [26], the temporal contrastive pre-training approach of Lorre *et al*. [43], and Contrastive Predictive Coding with Transformer (CPCTR) [40], further refined predictive contrastive learning for video understanding.

Self-Supervised Learning for Event Camera Data. Self-supervised pre-training that operates purely on event data is less explored than its RGB counterpart. Early event-SSL approaches adapted correlation-based losses (e.g., VICReg-inspired joint embedding) to events [5]. Reconstruction and masked-token approaches followed: MEM [32] reconstructs visual tokens for masked regions using targets from a pretrained dVAE [53]; Huang *et al*. [31] voxelize event streams and reconstruct masked local and global voxel groups via semantic uniform masking. Large-scale adaptations of image SSL to events also emerged: ECDDP [68] extends the DINOv2 [49] by enforcing multi-level (patch, context, image) similarities and local-to-global crop comparisons for events. TESPEC [47], currently the strongest event-SSL method, employs a recurrent backbone to integrate longer temporal windows and learns to reconstruct corresponding intensity images, demonstrating the value of temporal aggregation for richer event representations.

Architectures in Event-Based Vision. Most existing self-supervised pre-training methods for event data use feed-forward backbones that process short event windows independently [5, 31, 32, 68]. Accumulating longer time spans into a single input tends to increase motion blur in reconstructed frames, thereby reducing signal quality. A growing alternative is to use recurrent modules that integrate information across short event segments, preserving temporal information. Recurrent architectures, including LSTM [29], GRU [13], and their convolutional extensions ConvLSTM [57] and ConvGRU [3], have been applied across event tasks. Recurrent designs aid object detection and tracking [19, 34, 52], improve dense predictions such as semantic segmentation and depth [15, 28, 39, 58], and support optical flow estimation [21, 35]. In the self-supervised domain, TESPEC [47] demonstrates the value

of recurrence and currently represents the strongest event-based SSL framework. Our work follows a similar architectural philosophy, employing a recurrent backbone to capture longer temporal dependencies in event data.

Handling of Event Camera Data. Data from an event camera is sparse and asynchronous, creating a trade-off between temporal resolution and spatial density. The voxel-grid representation (accumulating events into temporal bins) [73] is widely used but suffers from the density–blur trade-off: long accumulations increase density but blur motion; short accumulations preserve temporal detail but produce very sparse frames. Alternatives include frame-like projections built from event counts or timestamps, learned conversions such as the Event Spike Tensor [18], and selective processing that ignores empty regions (e.g., Event Transformer+ [56], which processes only patches with sufficient activity). Some methods also change the learning target to provide denser supervision: TESPEC [47] reconstructs intensity images from events to create a denser training signal. Our approach tackles sparsity both through the architecture, by using recurrence to aggregate information across short voxel segments, and through the loss, by applying activity-aware contrastive weighting. This prevents the model from enforcing discrimination on empty or uninformative regions.

Event Camera Datasets. Event datasets such as MVSEC [72], DSEC [20], and M3ED [11] offer high-quality benchmarks but are limited in scale for large pre-training. Larger collections (Gen1 [60], 1Mpx [52], DDD17 [8]) help volume but still lag behind RGB corpora in diversity. More recent work generates synthetic event streams from large RGB datasets (e.g., using v2e [30] or ESIM [55]), producing resources like E-TartanAir [64, 68], BlinkFlow [35], BlinkVision [36], TartanAir-V2 [59] and TartanGround [51]. Synthetic data enables large-scale pre-training with varied motion and scene content; in this work, we pretrain on the synthetic TartanAir-V2 dataset to leverage those benefits.

3. Method

The proposed method, SPICE, builds upon the RGB-based DPC framework [25] to pretrain an encoder on event camera data. SPICE learns to predict future latent representations from past observations, encouraging temporally consistent and structure-aware features.

We first outline the key preliminaries, including the InfoNCE objective and characteristics of event data, see Sec. 3.1. Events are converted into voxels that serve as network input, described in Sec. 3.2. The core SPICE framework is detailed in Sec. 3.3, which provides a step-by-step explanation of the main architecture shown in Fig. 1. The pre-training objective aligns predicted and actual future patch embeddings via a weighted contrastive loss, see Sec. 3.4, computed only on active patches to handle sparsity as described in Sec. 3.5.

3.1. Preliminaries

InfoNCE. A central component in contrastive SSL is the InfoNCE loss [48], which encourages the model to output embeddings that have a higher similarity to matching (positive) samples and lower similarity to non-matching (negative) ones.

Given a query embeddings q_i, a positive key k_i, and a set of negative keys k_j ($j \neq i$), the loss is defined as

$$\mathcal{L} = -\mathbb{E}\left[\log \frac{\exp(q_i \cdot k_i/\tau)}{\exp(q_i \cdot k_i/\tau) + \sum_{j\neq i} \exp(q_i \cdot k_j/\tau)}\right] \tag{1}$$

here (q_i, k_i) are positive pairs when $i = j$ and negative pairs otherwise; the dot product is adopted as measurement of similarity. The temperature parameter τ controls the sharpness of the similarity distribution. A lower τ produces a sharper (more selective) distribution, emphasizing hard negatives, while a higher τ leads to a smoother weighting over all samples [62]. Moreover, following the analysis by Wang and Isola [63], this loss can be interpreted as the sum of two complementary objectives: an *alignment term* that pulls positive pairs closer together, and a *uniformity term* that spreads representations uniformly across the embedding space. This balance between alignment and uniformity is central to contrastive learning and motivates later extensions of InfoNCE [33, 46], including ours (see Sec. 3.4).

Event Camera Data. An event camera records changes in logarithmic brightness asynchronously at the pixel level, producing a continuous stream of events $e_i = (x_i, y_i, t_i, p_i)$, where (x_i, y_i) denotes the pixel coordinates, t_i the timestamp, and $p_i \in +1, -1$ the polarity indicating the direction of the brightness change. The entire event stream over a time interval $[t_0, t_N]$ can thus be represented as $\mathcal{E} = \{e_i | i = 1, ..., N\}$. Each event is generated independently by a pixel when the change in logarithmic intensity exceeds a predefined contrast threshold C. Formally, an event is triggered when

$$\Delta L(x_i, y_i, t_i) = L(x_i, y_i, t_i) - L(x_i, y_i, t_i - \Delta t_i) = p_i C \tag{2}$$

where $L(x, y, t) = \log I(x, y, t)$ is the logarithmic brightness, $I(x, y, t)$ the image intensity, and Δt_i the elapsed time since the last event at that pixel.

3.2. Event Representation

Since the DPC framework was originally designed for frame-based inputs, it cannot directly process asynchronous event streams. We therefore convert the raw events into a multi-frame voxel grid [73]. The voxel representation is defined as $v \in \mathbb{R}^{B\times H\times W}$, where B is the number of temporal

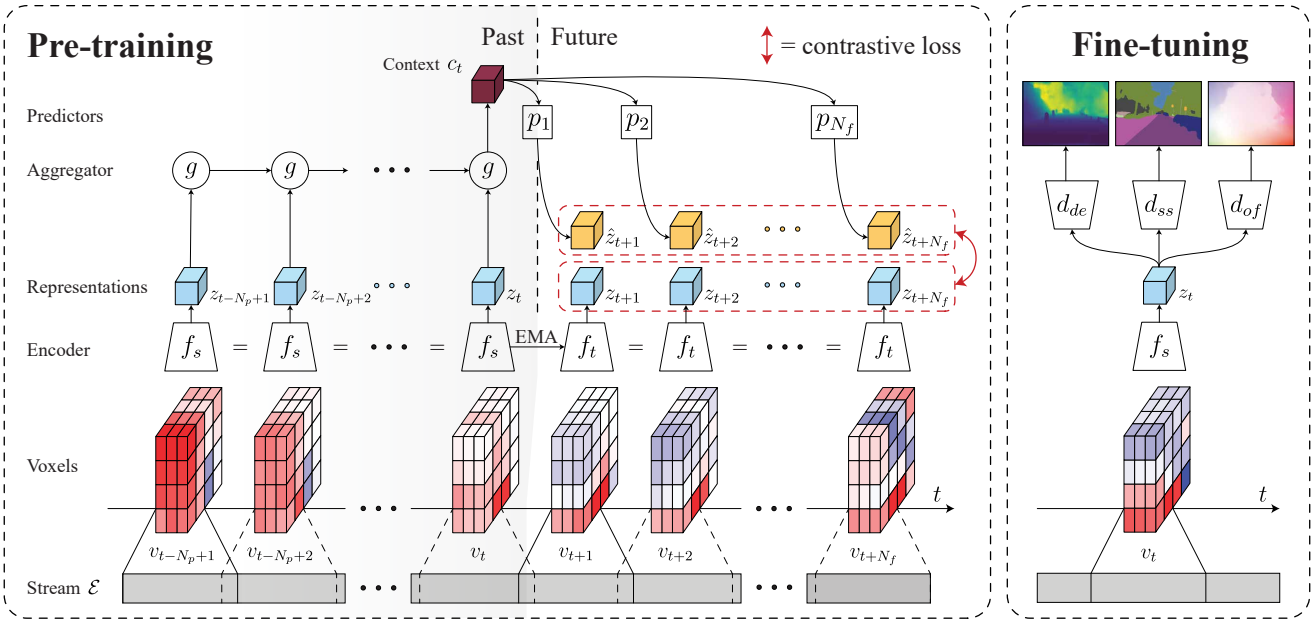


Figure 1. **Overview of SPICE.** A continuous event stream $\mathcal{E}$ is converted into overlapping event voxel grids $v_t{}_{t=1}^{N_p+N_f}$, where N_p and N_f denote the numbers of past and future voxels, respectively. Past voxels are encoded by the student encoder $f_s(\cdot)$ into patch-level representations $z_t{}_{t=1}^{N_p}$, which are aggregated over time by the context module $g(\cdot)$ to form context features c_t. From c_t, a set of predictors $p_i{}_{i=1}^{N_f}$ independently predict future representations $\hat{z}_{t+i}$. Ground-truth future embeddings are obtained by encoding the corresponding future voxels with the EMA-updated teacher encoder $f_t(\cdot)$. A contrastive loss aligns predicted and teacher embeddings at the patch level. After pre-training, only the student encoder is retained and fine-tuned with task-specific heads for depth estimation, semantic segmentation, and optical flow estimation.

bins and (H, W) the spatial resolution. Each voxel accumulates events that fall within its temporal bin according to:

$$t_i^* = (B-1)(t_i - t_{\text{begin}})/(t_{\text{end}} - t_{\text{begin}}) \quad (3)$$

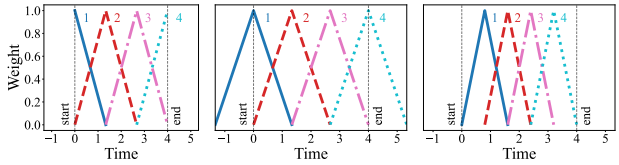
$$v(b,x,y) = \sum_{i|x_i=x,y_i=y} p_i \max(0, 1-|t_i^* - b|) \quad (4)$$

where $b \in [0, B-1]$ indexes the temporal bins. This interpolation scheme distributes each event's contribution to its two nearest temporal bins. However, this formulation leads to an uneven event distribution across bins: the first and last bins receive, on average, only half as many events as the intermediate ones. To address this, we adopt a modified bin weighting scheme (previously Ye *et al*. [70] proposed a similar scheme), which shifts the effective bin centers and ensures uniform accumulation. The temporal normalization and voxelization then become

$$t_i^* = (B+1)(t_i - t_{\text{begin}})/(t_{\text{end}} - t_{\text{begin}}) \quad (5)$$

$$v(b,x,y) = \sum_{i|x_i=x,y_i=y} p_i \max(0, 1-|t_i^* - (b+1)|) \quad (6)$$

To preserve continuity in motion across consecutive voxel grids, we employ an overlap of half a bin duration between successive windows, as illustrated in Fig. 1. The different weighting schemes are shown in Fig. 2.



(a) Original [73] (b) UVG [70] (c) Ours

Figure 2. **Event representation bin weighting schemes.** Events are accumulated into bins, where each event is weighted based on its timestamp. This weight is shown per bin, for a voxel with four bins, spanning 4 time units. (a) Original weights [73], where the first and last bin receive fewer accumulated events. (b) The proposed weights by Ye *et al*. [70], which use events beyond the specified time limits. (c) Our proposed bin weights. Note that ours and (b) essentially only differ in start and end time.

3.3. Self-Supervised Predictive Coding on Events

The goal of our method is to learn representations that capture slowly varying semantics by predicting the future latent representation of an event stream from its past. Intuitively, a model that can accurately infer future dynamics must have developed meaningful internal representations of the input.

Given an event stream $\mathcal{E}$, we first convert it into overlapping voxel grids $v_t{}_{t=1}^{M}$, where each voxel grid $v_t \in$

$\mathbb{R}^{B\times H\times W}$ represents B temporal bins of spatial resolution $H \times W$. Consecutive voxel grids overlap by half the bin duration (see Sec. 3.2). From the voxel grids, we sample training sequences of length $N_v = N_p + N_f$, consisting of N_p *past* and N_f *future* voxels. Each past voxel v_t is then encoded by the student encoder $f_s(\cdot)$, producing a sequence of latent representations $z_{t\,t=1}^{N_p}$, where each

$$z_t = f_s(v_t), \quad z_t \in \mathbb{R}^{H'\times W'\times D} \tag{7}$$

Since the encoder operates on patches, z_t consists of a spatial grid of *patch embeddings* of dimensionality D, maintaining the spatial layout of the input.

The context module $g(\cdot)$ aggregates these representations over time, producing a set of context representations c_t that capture the evolution of local features:

$$c_t = g(z_{t-N_p+1}, z_{t-N_p+2}, ..., z_t) \tag{8}$$

In contrast to the sequential prediction strategy used in DPC, where predictions are generated step-by-step and recursively aggregated, SPICE performs *parallel prediction*. Specifically, from each c_t, a set of prediction heads $\{p_i\}_{i=1}^{N_f}$ independently predict the patch embeddings i steps into the future:

$$\hat{z}_{t+i} = p_i(c_t), \quad i = 1, \ldots, N_f. \tag{9}$$

This design prevents the model from relying on simple propagation of features between consecutive timesteps, forcing c_t to encode information relevant at multiple temporal scales simultaneously. Intuitively, this encourages the formation of richer and more semantically meaningful representations.

Ground-truth future representations are obtained by encoding the corresponding future voxels with the teacher encoder $f_t(\cdot)$, which shares the same architecture as the student encoder $f_s(\cdot)$, but is updated via an exponential moving average (EMA) of the student's parameters [22]. The EMA mechanism ensures that teacher representations evolve smoothly, providing stable learning targets for the student. Alignment between predicted and ground-truth patch embeddings is enforced by our proposed loss function (see Sec. 3.4), which operates at the patch level rather than on a single global embedding.

After pre-training, only the student encoder $f_s(.)$ is retained. The aggregation module, predictor, and teacher encoder are discarded. For downstream evaluation, we attach task-specific heads for semantic segmentation, depth estimation and optical flow estimation. Then we jointly finetune the encoder and corresponding head on the labeled dataset of each task.

3.4. Loss Formulation

SPICE employs a contrastive objective inspired by the InfoNCE loss [48], which encourages positive pairs to align closely in feature space while uniformly pushing apart all negatives. In our setup, pairs correspond to patch embeddings: for each predicted patch embedding $\hat{z}_{i,k}$ (from timestep i, spatial index k), the positive is the corresponding ground-truth embedding $z_{i,k}$ at the same spatiotemporal location. All other patch embeddings serve as negatives.

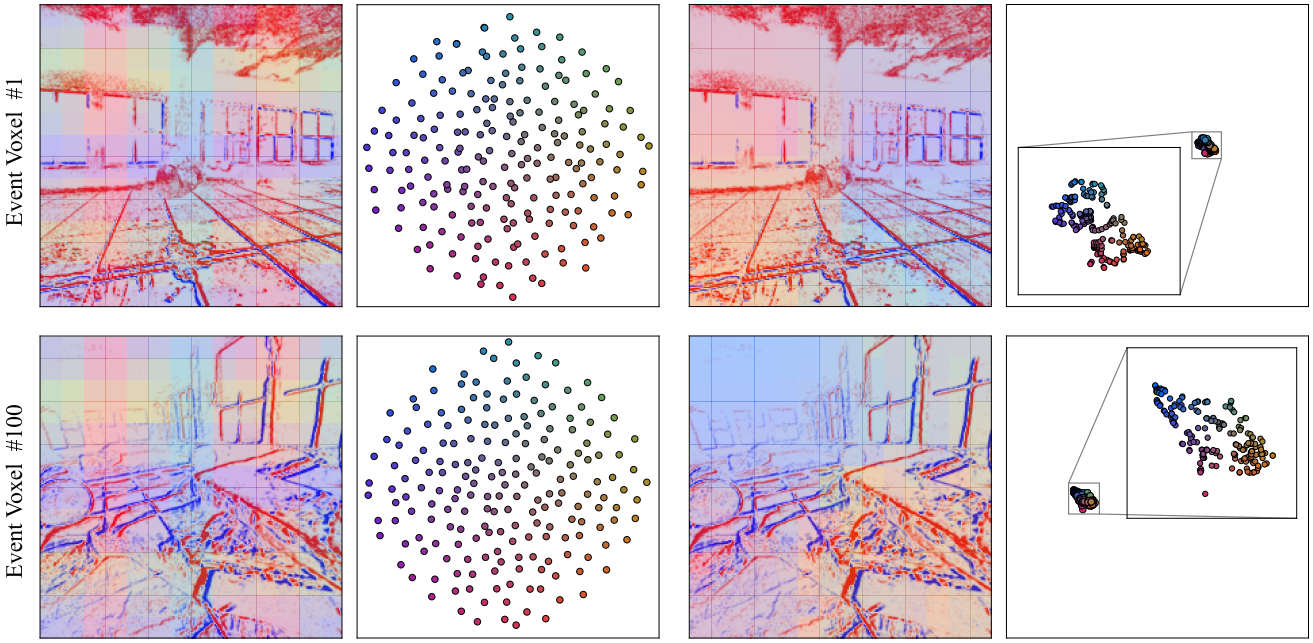
While uniform repulsion of all negatives works well with individual images, this assumption is problematic for event-based video representations. Here, many patches across space or time are naturally correlated, i.e., nearby patches within a voxel or temporally adjacent voxels often carry related semantics. Treating them as equally negative can thus lead to unstable training and representational collapse.

Negative Groups and Imbalance. Each representation $z_t, \hat{z}_t \in \mathbb{R}^{H'\times W'\times D}$ is a spatial grid of $H' \times W'$ patch embeddings, $z_{i,k}, \hat{z}_{i,k} \in \mathbb{R}^D$. Negatives can be grouped into three categories: *temporal negatives* are embeddings at the same spatial location but from other timesteps ($j \neq i$), *spatial negatives* are embeddings at different spatial locations within the same or other timesteps ($k' \neq k$) and *batch negatives* are embeddings from other sequences in the mini-batch. For a sequence with N_f future timesteps, feature map size $H' \times W'$, and batch size B, the ratio of positives to negatives per group is:

$$\begin{aligned} &Pos : N_{temporal} : N_{spatial} : N_{batch} = \\ &1 : (N_f - 1) : (H'W' - 1)N_f : (B - 1)H'W'N_f \end{aligned} \tag{10}$$

This reveals a strong imbalance: batch negatives vastly outnumber temporal or spatial ones. Empirically, this imbalance caused the network to underutilize hard (temporal) negatives, resulting in collapsed patch embeddings that remained nearly constant across time and only varied based on the patch location in the frame instead of its semantic content. A t-SNE [61] analysis of the learned embeddings, shown in Fig. 3, illustrates this effect in more detail.

Negative Group Normalization and Gaussian Weighting. To address this collapse, we first normalize the contribution of each negative group, ensuring equal weight among temporal, spatial, and batch negatives. Within each group, however, still not all negatives are equally "hard." Negatives that are closer in space or time to the anchor should be pushed away less strongly than distant ones. Therefore, we introduce a Gaussian weighting mask that modulates each negative's contribution according to its spatiotemporal distance from the anchor. This results in a loss that combines negative group normalization with Gaussian-weighted negatives, which we refer to as the Gaussian-Weighted InfoNCE (GW-InfoNCE) loss. Fig. 4 illustrates both the three negative groups and the spatiotemporal Gaussian weighting mask. Here, temporal and spatial negatives are intra-negatives, as they come from within the same sequence, while batch negatives are inter-negatives, originating from different sequences. Fig. 5 shows the Gaussian weight mask



(a) All negatives treated equally.

(b) Group normalized and Gaussian-weighted negatives.

Figure 3. **t-SNE** [61] **projection of patch embeddings for two different voxels and two loss configurations.** Both (a) and (b) show the first and last event frame (i.e., bin) of a 100-voxel sequence, alongside the t-SNE projection of all patch embeddings in those frames. Colors are matched between t-SNE points and their corresponding patches in the event frame. (a) Treating all negatives equally leads to uniformly distributed yet nearly static patch embeddings. The near-identical patch colors across event frames indicate that representations are dominated by spatial position rather than semantic content, evidencing a form of representational collapse. (b) Balancing negatives per group and weighting them with a Gaussian mask produces temporally evolving embeddings that capture local variations. While this mitigates collapse, embeddings within each frame remain tightly clustered and differ sharply across time, suggesting that improving both semantic quality and consistency across frames remains an open direction for future work.

on the full input size to highlight its global impact.

Final Formulation. The proposed GW-InfoNCE loss is defined as:

$$\mathcal{L}_{\text{GW-InfoNCE}} = -\sum_{i,k} \log \frac{\mathcal{N}_{pos}(i,k)}{\mathcal{N}_{pos}(i,k) + I_b\mathcal{N}_b(i,k) + I_s\mathcal{N}_s(i,k) + I_t\mathcal{N}_t(i,k)} \tag{11}$$

$$\mathcal{N}_{pos}(i,k) = \exp(\hat{z}_{i,k} \cdot z_{i,k}/\tau) \tag{12}$$

$$\mathcal{N}_b(i,k) = \frac{1}{N_b} \sum_{(j,k')\in\mathcal{B}} \exp(\hat{z}_{i,k} \cdot z_{j,k'}/\tau) \tag{13}$$

$$\mathcal{N}_s(i,k) = \frac{1}{\sum_{(j,k')\in\mathcal{S}} w_{j,k'}} \sum_{(j,k')\in\mathcal{S}} w_{j,k'} \exp(\hat{z}_{i,k}\cdot z_{j,k'}/\tau) \tag{14}$$

$$\mathcal{N}_t(i,k) = \frac{1}{\sum_{(j,k')\in\mathcal{T}} w_{j,k'}} \sum_{(j,k')\in\mathcal{T}} w_{j,k'} \exp(\hat{z}_{i,k}\cdot z_{j,k'}/\tau) \tag{15}$$

Here:

- $\mathcal{B}$,$\mathcal{S}$,$\mathcal{T}$ denote the sets of batch, spatial and temporal negatives, respectively.
- $w_{j,k'}$ is a Gaussian weight based on the spatiotemporal distance between the anchor (i,k) and the negative (j,k'):

$$w_{j,k'} = \exp\left(-\frac{d_{\text{space}}(k,k')^2}{2\sigma_s^2} - \frac{d_{\text{time}}(i,j)^2}{2\sigma_t^2}\right) \tag{16}$$

- I_b, I_s, I_t are importance factors that control the relative weighting of each group (setting all to 1 gives equal group importance).
- N_b is the number of batch negatives.

3.5. Active Patch Calculation

Event data is inherently sparse, meaning that many spatial regions contain little or no activity. To ensure that the contrastive loss focuses on informative regions, we determine *active patches*, i.e., the spatial regions containing meaningful event activity, by analyzing the average intensity of events within each patch. For each voxel to be predicted, the absolute event values are averaged within each patch region to obtain a patch-level activity map. A histogram-based adaptive threshold is then applied per sample: the most frequent activity level (i.e., the bin containing the most

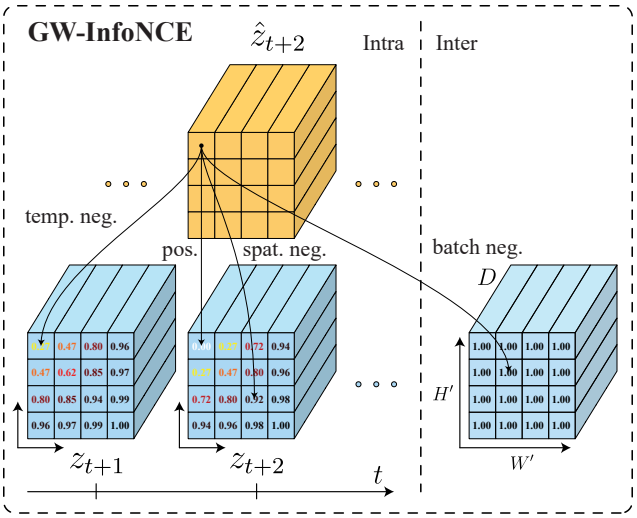
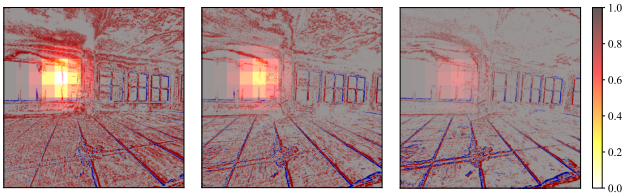


Figure 4. **Patch-level GW-InfoNCE setup.** Temporal and spatial negatives are sampled within the same sequence and Gaussian-weighted based on their spatiotemporal distance to the anchor. The color of the text corresponds to the value of the weight. The Gaussian weight mask is shown in full on an actual training sample in Fig. 5. Batch negatives are drawn from other samples in the mini-batch and are uniformly weighted.



(a) z_{t+1} (b) z_{t+2} (c) z_{t+3}

Figure 5. **Gaussian weight mask.** (a-c) display timesteps z_{t+1}, z_{t+2} and z_{t+3} overlaid with a heatmap showing the weight assigned to each patch. A zero weight (only for the patch itself, i.e., the positive) means that it is not considered, while a weight of one means it is maximally considered.

patches) is used to define a threshold, limited by a predefined maximum to avoid outliers. Patches whose mean activity exceeds this threshold are marked as active. Only these active patches are used as anchors and negatives in the loss computation, preventing inactive regions from biasing the representation learning. Fig. 6 shows multiple examples of an event frame and which patches are ignored.

4. Experiments

4.1. Experimental Setup

We evaluate SPICE on semantic segmentation, depth estimation, and optical flow estimation. Following prior works [47, 67, 68], we integrate the pretrained backbone with task-specific heads and perform joint supervised fine-

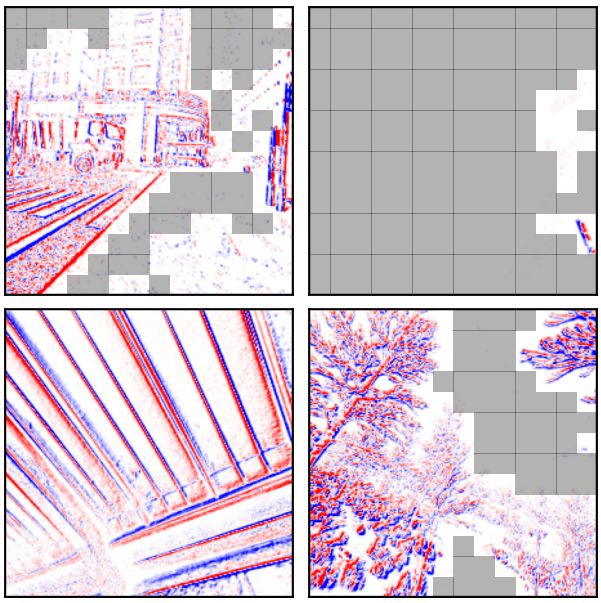


Figure 6. **Active patch masks.** Gray patches indicate regions excluded from the contrastive loss. The mask is computed adaptively from the histogram of patch-wise event intensities, ensuring that dense event frames are fully active, while sparse frames have many inactive patches.

tuning on the respective downstream datasets. This subsection details the pre-training procedure, while the implementation specifics for each downstream task's fine-tuning are provided in their dedicated subsections.

Pre-training Event Dataset. For pre-training, we use the TartanAir-V2 [59] dataset, a large and diverse collection of RGB simulations converted into events using ESIM [55]. TartanAir-V2 contains 63 photorealistic simulation environments chosen to represent a broad spectrum of real-world scenarios. These environments fall into six categories: Indoor, Nature, Rural, Urban, Industrial/Infrastructure, and Historical/Thematic, providing rich diversity in terrain and lighting conditions to support robust generalization.

SPICE Implementation Details. For our backbone, we adopt the Swin Transformer [42] architecture with a patch size of 7 (Swin-T/7), following prior works that demonstrated its superior performance in event-based SSL [47, 68]. The outputs from each Swin-T stage are fused using a feature pyramid network (FPN), which combines multi-scale features into a single representation. To aggregate temporal information, we employ a ConvGRU as the aggregation module, consistent with the design used in DPC. For prediction, we use multiple shallow two-layer perceptrons of identical architecture, operating on the aggregated features. Targets are provided by a teacher encoder whose weights track the student's via EMA with momentum 0.998.

A single input event voxel aggregates 49 ms (see Sec. B.1) of events divided into 6 overlapping bins, each spanning 14 ms. The Swin-T transformer architecture inherently allows pretrained models to be transferred to downstream datasets without constraints on input resolution.

During pre-training, we use a "5pred3" setup: aggregating features from 5 past voxels (covering 217 ms of events) to predict the next 3 voxels (covering 133 ms, with a 7 ms overlap with the input). As such, each training sample consists of 8 consecutive voxels. We train with a batch size of 64 for a total of 15 epochs (19893 steps per epoch) on a single NVIDIA RTX5090.

We use AdamW [44] with a weight decay of 0.01, a peak learning rate of 1×10^{-4}, a warm-up period of 2 epochs, and a cosine annealing learning rate schedule. Loss parameters are set as follows: batch importance $I_b = 0.5$, spatial importance $I_s = 2.0$, temporal importance $I_t = 2.0$, with spatial (σ_s) and temporal (σ_t) Gaussian sigmas both set to 1.25. Finally, the patch embedding dimension D is 768.

Baselines. To evaluate the effectiveness of SPICE, we compare it against three types of baselines:

- **No Pre-training:** For each downstream task, we report results obtained when fine-tuning the model from scratch, without any pre-training.
- **Event-based SSL:** This category includes (former) state-of-the-art self-supervised methods pretrained on event camera data, namely ECDP [67], ECDDP [68], and TESPEC [47].
- **Absolute Best:** For tasks where event-based SSL is not already the top-performing category (i.e., MVSEC depth and DSEC optical flow), we additionally include the best published results overall.

For downstream tasks previously evaluated in these works, we directly report their published results. For tasks without existing evaluations, we fine-tune the respective pretrained backbones using our codebase and with the same training setup to ensure consistent comparison. Additional details on the pre-training and fine-tuning are provided in Sec. B.

4.2. Semantic Segmentation

Setup. Following prior work [47, 68], we attach a UperNet head [65] to the pretrained backbone and fine-tune the network on DSEC. Only the backbone is transferred from pre-training, while all other components are not used. As in ESS [58], we employ a combination of cross-entropy and Dice losses during training. The input event representation remains identical to the one used during pre-training.

We train for 25 epochs (1,008 steps per epoch) with a batch size of 8, using AdamW with a weight decay of 0.05. The learning rate follows a cosine annealing schedule, peaking at 2.5×10^{-4} after 5 warm-up epochs.

Results. Tab. 1 and Fig. 7 show the quantitative and qualitative results respectively. While SPICE does not yet surpass the state of the art, it achieves a substantial improvement over training from scratch. This can especially be seen in Fig. 7, where the no pre-training results are very noisy. This indicates that, although there remains considerable room for improvement, SPICE learns representations that capture information relevant for semantic segmentation.

Table 1. **Quantitative semantic segmentation results on DSEC** [20]. Mean intersection over union (mIoU) and mean class accuracy (mAcc) are used as evaluation metrics. SPICE does not outperform state of the art, but does show significant improvement over the non-pre-training setting. Best results are highlighted in **bold**, second best are underlined.

Method	Backbone	Pre-training	mIoU ↑	mAcc ↑
No pre-training	Swin-T/7	-	52.5	59.8
ECDP [67]	ResNet50	N-ImageNet	59.2	67.5
ECDDP [68]	ResNet50	E-TartanAir	60.6	69.5
↳ ECDDP uses test time augmentations (TTA).				
ECDDP [68]	Swin-T/7	E-TartanAir	<u>61.3</u>	<u>69.6</u>
↳ ECDDP uses test time augmentations (TTA).				
ECDDP [68]	Swin-T/7	E-TartanAir	59.1	67.0
↳ Our reproduced performance of ECDDP, with TTA.				
TESPEC [47]	Swin-T/7	1Mpx	**62.8**	**70.6**
SPICE (**ours**)	Swin-T/7	TartanAir-V2	56.5	64.3

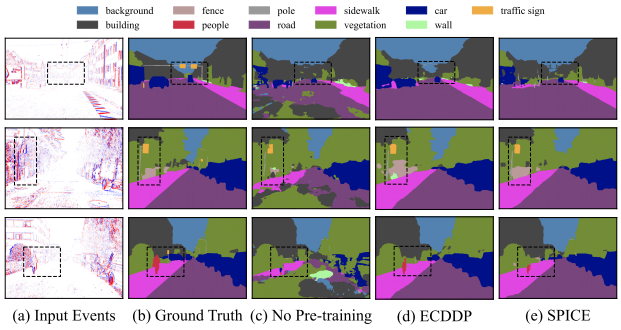


Figure 7. **Qualitative semantic segmentation results on DSEC** [20]. (a) Input events, (b) ground truth, (c) no pre-training, (d) ECDDP, and (e) SPICE. Models pretrained on event data produce markedly cleaner and more structured predictions compared to training from scratch. SPICE yields visually similar results to ECDDP, despite lower quantitative performance. Note that ECDDP employs a different input representation (20-bin event histogram), and thus the shown input events do not directly correspond to its configuration.

4.3. Depth estimation

Setup. For depth estimation, we attach a lightweight decoder to the pretrained backbone. The decoder receives

multi-scale feature maps from each Swin-T stage and progressively upsamples them back to the input resolution. We fine-tune this network on both the MVSEC and DSEC datasets. Since DSEC provides disparity ground truth, we first convert it to depth before training. The input event representation follows the same structure as during pre-training, but we extend the accumulation duration to better match each dataset: 60 ms for DSEC and 400 ms for MVSEC, compared to 49 ms during pre-training (details in Sec. C.1).

Following prior work [47, 68], which builds upon HMNet [24], we adopt the same optimization objectives: a scale-invariant loss and a multi-scale scale-invariant gradient matching loss, weighted 1.0 and 0.25 respectively. The network predicts normalized log-depth values during training, which are converted back to metric depth for evaluation.

For MVSEC, we follow the standard protocol and train exclusively on the `outdoor_day2` sequence, while `outdoor_day1`, `outdoor_night1`, `outdoor_night2`, and `outdoor_night3` are used for evaluation. For DSEC, we employ the commonly used train and validation split (details in Sec. B.2.2).

Training on MVSEC is performed for 15 epochs (1,523 steps per epoch) with a batch size of 8, using AdamW with a weight decay of 0.01. The learning rate peaks at 4×10^{-5} after 5 warm-up epochs and follows a cosine annealing schedule thereafter. For DSEC, we train for 50 epochs (1,433 steps per epoch) with a batch size of 8, AdamW optimizer with weight decay 0.01, and a learning rate peaking at 1×10^{-4} after 2 warm-up epochs, again followed by cosine annealing.

Results. Tabs. 2 and 3 and Figs. 8 and 9 present the quantitative and qualitative results on DSEC [20] and MVSEC [72], respectively. SPICE performs competitively on both datasets, achieving results close to the state of the art. On DSEC, SPICE shows solid results but remains below ECDDP when ECDDP is fine-tuned within our framework using our event representation and decoder. On MVSEC, SPICE surpasses the original ECDDP in the threshold metrics and RMSlog. When re-trained within our framework, ECDDP regains a small advantage, again benefiting from our event representation and decoder. Furthermore, our model without pre-training outperforms the corresponding no pre-training variant of TESPEC. Overall, TESPEC maintains the best performance among all methods.

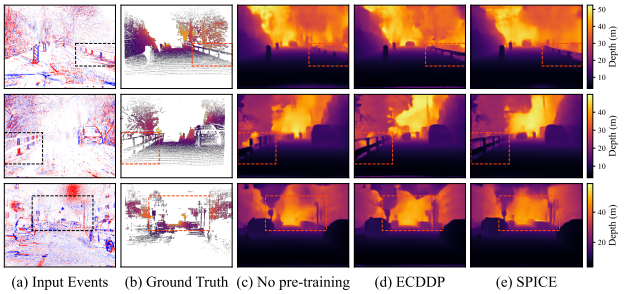


Figure 8. **Qualitative depth estimation results on DSEC** [20]. (a) Input events, (b) ground truth, (c) no pre-training, (d) ECDDP, (e) SPICE. All methods produce reasonable depth maps, with SPICE showing a clear improvement over no pre-training and approaching ECDDP in quality.

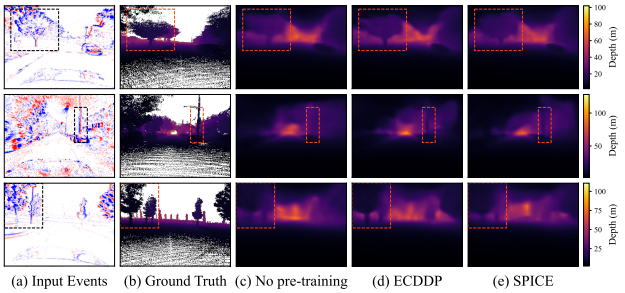


Figure 9. **Qualitative depth estimation results on MVSEC** [72]. (a) Input events, (b) ground truth, (c) no pre-training, (d) ECDDP, (e) SPICE. All models capture coarse scene structure, with SPICE visibly outperforming no pre-training, while ECDDP remains strongest overall.

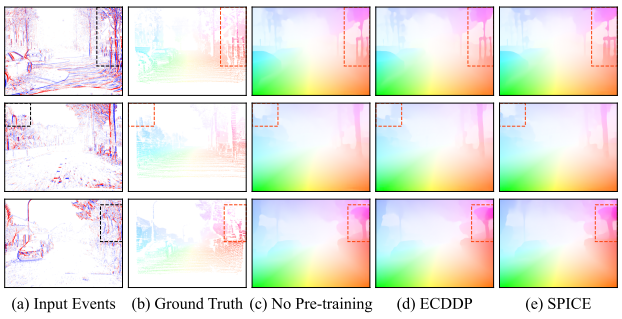


Figure 10. **Qualitative optical flow estimation results on DSEC** [20]. (a) Input events, (b) ground truth, (c) no pre-training, (d) ECDDP, (e) SPICE. Little difference can be observed between the no pre-training setting, ECDDP and our pretrained model. Note that the shown ECDDP results are from our reproduction using our codebase.

4.4. Optical Flow Estimation

Setup. Following ECDDP [68], we adopt the TMA architecture [37] for optical flow estimation. Specifically, we replace TMA's original feature and context encoders with the first two stages of our pretrained Swin-T/7 backbone and fine-tune the resulting network on the DSEC dataset. The training objective is an L1 loss applied across all prediction stages, weighted by a decay factor $\gamma = 0.8$ (as in [37]). We use the same train and validation split as ECDDP (details

Table 2. **Monocular depth estimation results on DSEC [20].** Threshold accuracies (δ_1, δ_2 and δ_3), absolute error (Abs.), root mean squared error (RMS) and root mean squared logarithmic error (RMSlog) are used as evaluation metrics. Best results are highlighted in **bold**, second best are underlined.

Method	Backbone	Pre-training	$\delta_1 \uparrow$	$\delta_2 \uparrow$	$\delta_3 \uparrow$	Abs $\downarrow$	RMS $\downarrow$	RMSlog $\downarrow$
No pre-training	Swin-T/7	-	0.838	0.960	0.990	2.90	4.55	0.18
ECDDP [68]	Swin-T/7	E-TartanAir	**0.880**	**0.975**	**0.994**	**2.47**	**4.03**	**0.15**
↳ No other works present results on DSEC depth, as such we took the only other event SSL work compatible with our codebase and fine-tuned it under the same settings.								
SPICE (**ours**)	Swin-T/7	TartanAir-V2	0.870	0.969	0.991	2.61	4.23	0.16

Table 3. **Monocular depth estimation results on MVSEC [72].** Threshold accuracies (δ_1, δ_2 and δ_3), absolute error (Abs.), root mean squared error (RMS) and root mean squared logarithmic error (RMSlog) are used as evaluation metrics. Best results are highlighted in **bold**, second best are underlined.

Method	Backbone	Pre-training	$\delta_1 \uparrow$	$\delta_2 \uparrow$	$\delta_3 \uparrow$	Abs $\downarrow$	RMS $\downarrow$	RMSlog $\downarrow$
PCDepth [38]	-	EventScape	0.672	0.845	0.932	-	6.62	0.33
↳ *Best performance in literature.* PCDepth utilizes both events and RGB images as input.								
No pre-training	Swin-T/7	-	0.345	0.731	0.853	5.59	8.67	0.48
↳ Results for no pre-training reported by TESPEC.								
No pre-training	Swin-T/7	-	0.608	0.800	0.906	4.17	7.45	0.37
↳ Results using our codebase.								
ECDP [67]	ResNet50	N-ImageNet	0.611	0.797	0.901	4.06	7.20	0.38
ECDDP [68]	ResNet50	E-TartanAir	0.612	0.809	0.915	3.89	6.81	0.36
ECDDP [68]	Swin-T/7	E-TartanAir	0.618	0.806	0.912	3.86	6.87	0.36
ECDDP [68]	Swin-T/7	E-TartanAir	**0.637**	0.825	0.922	3.81	7.05	0.35
↳ Results using our codebase.								
TESPEC [47]	Swin-T/7	1Mpx	0.634	**0.830**	**0.926**	**3.69**	**6.65**	**0.34**
SPICE (**ours**)	Swin-T/7	TartanAir-V2	0.628	0.815	0.915	3.91	7.15	0.36

in Sec. B.2.3). To fit within the TMA framework, we use a normalized default event voxel grid [73] with 15 temporal bins.

Training is conducted for 150,000 steps using AdamW with a weight decay of 1×10^{-4}. The learning rate peaks at 2×10^{-4} after 1,500 warm-up steps and follows a cosine annealing schedule thereafter.

Results. Tab. 4 and Fig. 10 show the quantitative and qualitative results respectively. Tab. 4 shows the results on the test set using the DSEC optical flow benchmark [21], but Fig. 10 shows samples from the validation set, since the test ground truth is not publicly available. Qualitatively, little difference is visible between the no pre-training setting and the pretrained models, though ECDDP does seem to have a slight edge.

4.5. Ablations

We ablate the main components of SPICE, reporting results on DSEC semantic segmentation (IoU, mAcc) and DSEC and MVSEC depth estimation (δ_1, Abs). All results are summarized in Tab. 5.

Pre-training epochs. Performance generally improves with longer pre-training, up to 15 epochs. Beyond this point, results on DSEC semantic segmentation and MVSEC depth degrade, while DSEC depth continues to improve.

Prediction scheme. We compare the parallel prediction scheme, where separate predictors independently handle different timesteps, to a sequential variant that re-aggregates each prediction as a new context for the next prediction (sharing weights across timesteps). Sequential prediction yields better depth estimation on DSEC only, while parallel prediction performs better on semantic segmentation and MVSEC depth.

Active patches. To mitigate event sparsity, the loss is computed only on active patches (see Sec. 3.5). Instead, computing the loss on all patches improves depth performance but reduces semantic segmentation accuracy.

GW-InfoNCE. Replacing the Gaussian-weighted loss with uniform weighting causes only minor performance changes, suggesting a limited impact of the spatial weighting.

Table 4. **Optical flow estimation results on DSEC** [20]. N-pixel errors (1PE, 2PE and 3PE), endpoint error (EPE) and angular error (AE) are used as evaluation metrics. Best results are highlighted in **bold**, second best are underlined.

Method	Backbone	Pre-training	1PE ↓	2PE ↓	3PE ↓	EPE ↓	AE ↓
EDCPT	-	-	6.87	2.35	1.52	0.625	2.17
↳ *Best performance in literature.* EDCPT remains anonymous at the time of writing.							
No pre-training	Swin-T/7	-	12.75	4.86	2.80	0.802	2.81
ECDDP [68]	ResNet50	E-TartanAir	9.01	3.29	1.98	0.701	2.61
ECDDP [68]	Swin-T/7	E-TartanAir	**8.89**	**3.20**	**1.96**	**0.697**	**2.58**
ECDDP [68]	Swin-T/7	E-TartanAir	12.27	4.60	2.71	0.795	2.80
↳ Since ECDDP does not release fine-tuning code, this is our reproduced result using our codebase.							
SPICE (**ours**)	Swin-T/7	TartanAir-V2	12.41	4.66	2.68	0.799	2.82

Group-normalized negatives. Omitting normalization across negative groups in the loss leads to representation collapse. Fine-tuning such a model performs worse than training from scratch, so results are omitted.

Loss components. The loss consists of three groups of negatives: temporal, spatial and batch. Removing the temporal negatives leads to no meaningful learning, with performance close to or worse than a model without pre-training. On the other hand, removing the batch negatives does not have such detrimental effects. In fact, performance only slightly decreases.

5. Discussion

Main Findings and Interpretations. The proposed SPICE pre-training framework improves performance over the no pre-training setting across all evaluated tasks: semantic segmentation, depth estimation, and optical flow estimation. The most competitive results are achieved in depth estimation, while segmentation and flow still show room for improvement. This could be attributed to limitations in the learned representations: both semantic structure and temporal stability remain underdeveloped, as indicated by the t-SNE analysis. For optical flow, the finetuning stage likely also constrains performance. Although we adopted the procedure described by ECDDP, the absence of released code led to a reproduced pipeline that does not match their reported accuracy, suggesting that details of the original finetuning setup play a significant role.

Furthermore, negative group normalization proved essential for preventing feature collapse and maintaining representational diversity. Restricting the loss to active patches produced more semantically meaningful embeddings, improving segmentation but slightly degrading depth. Since this variant effectively sees fewer samples per epoch, extending training to equalize the number of processed patches may amplify its advantages. The effect of Gaussian weighting remains inconclusive. Among prediction strategies, the parallel formulation was favored as it encourages reasoning over multiple temporal horizons, potentially enriching semantics, whereas sequential propagation might benefit temporally sensitive tasks such as flow, which remains to be investigated.

Relation to Prior Work. TESPEC [47] and ECDDP have shown that event-based pre-training outperforms both supervised and self-supervised RGB-based pre-training. Our results further confirm the superiority of event-specific SSL.

While most prior approaches adopt a masking-based paradigm [32, 47, 68], SPICE demonstrates that future latent prediction is a viable alternative. Masking provides a strong but artificial reconstruction signal focused on semantics. Predictive learning instead encourages understanding of underlying spatiotemporal dynamics. Despite current gaps in segmentation and flow accuracy, the competitive depth results indicate that predictive objectives can capture meaningful structure without explicit reconstruction.

Strengths and Insights. t-SNE analyses and targeted ablations provide valuable insight into how different loss components shape the embedding space. Temporal negatives are crucial for preventing collapse and promoting useful features, yet they also induce abrupt changes across timesteps, highlighting the current tension between semantic richness and temporal stability. Unlike standard contrastive approaches that rely heavily on large batch sizes, SPICE remains robust even with fewer negatives, aided by negative group normalization. Interestingly, the RGB-based DPC baseline does not use such normalization, suggesting that event-based contrastive learning is more sensitive to negative imbalance due to the sparse and asynchronous nature of event data. This sensitivity may also be reinforced by the shift from clip-level (in DPC) to dense prediction tasks (in ours), where each local embedding is important. Overall, this emphasizes that architectural and objective-level design must be tailored to event statistics rather than transferred directly from frame-based methods.

Limitations and Future Work. The main limitation lies

Table 5. **Ablation study on SPICE.** DSEC semantic segmentation is evaluated using mIoU and mAcc, while DSEC and MVSEC depth estimation use δ_1 and Abs. metrics. Bold entries denote the baseline for each ablation, and horizontal lines (midrules) separate different ablation experiments. "E." indicates the number of pre-training epochs. Red arrows (↓ or ↑, depending on whether higher or lower is better) mark a performance decrease, and green arrows indicate an improvement. Percentual changes (in brackets) are computed relative to the no pre-training baseline and the respective ablation baseline.

Ablation	**E.**	**DSEC** - Semantic		**DSEC** - Depth		**MVSEC** - Depth	
		mIoU ↑	mAcc ↑	δ_1 ↑	Abs. ↓	δ_1 ↑	Abs. ↓
More pre-training epochs	20	56.1 ↓ (10%)	63.3 ↓ (22.2%)	0.875 ↑ (17.3%)	2.547 ↓ (21.1%)	0.627 ↓ (5%)	3.916 ↑ (2.7%)
Selected model	15	56.5	64.3	0.870	2.611	0.628	3.909
Less pre-training epochs	10	56.2 ↓ (7.5%)	63.5 ↓ (17.8%)	0.866 ↓ (13.8%)	2.632 ↑ (6.9%)	0.630 ↑ (10%)	3.919 ↑ (3.8%)
Less pre-training epochs	6	55.8 ↓ (17.5%)	62.9 ↓ (31.1%)	0.866 ↓ (13.8%)	2.632 ↑ (6.9%)	0.624 ↓ (20.1%)	4.035 ↑ (47.7%)
Less pre-training epochs	4	55.7 ↓ (20%)	62.9 ↓ (31.1%)	0.864 ↓ (20.7%)	2.622 ↑ (3.6%)	0.625 ↓ (15%)	3.966 ↑ (21.6%)
Less pre-training epochs	2	53.9 ↓ (65%)	61.7 ↓ (57.8%)	0.855 ↓ (51.7%)	2.754 ↑ (47.2%)	0.617 ↓ (55%)	3.998 ↑ (33.7%)
Less pre-training epochs	1	53.5 ↓ (75%)	60.8 ↓ (77.8%)	0.847 ↓ (79.3%)	2.843 ↑ (76.6%)	0.617 ↓ (55%)	4.07 ↑ (61%)
No pre-training epochs	0	52.5 ↓ (100%)	59.8 ↓ (100%)	0.841 ↓ (100%)	2.914 ↑ (100%)	0.608 ↓ (100%)	4.173 ↑ (100%)
Parallel prediction	10	56.2	63.5	0.866	2.632	0.630	3.919
Sequential prediction	10	55.9 ↓ (8.1%)	63.6 ↑ (2.7%)	0.870 ↑ (16%)	2.568 ↓ (22.7%)	0.628 ↓ (9.1%)	3.942 ↑ (9.1%)
Complete loss	6	55.8	62.9	0.866	2.632	0.624	4.035
All patches	6	55.3 ↓ (15.2%)	62.5 ↓ (12.9%)	0.865 ↓ (4%)	2.623 ↓ (3.2%)	0.626 ↑ (12.6%)	3.908 ↓ (92%)
Uniform negatives	6	55.8 ↓ (0%)	63.6 ↑ (22.6%)	0.863 ↓ (12%)	2.666 ↑ (12.1%)	0.625 ↑ (6.3%)	3.859 ↓ (127.5%)
Not normalized negatives	6	-	-	-	-	-	-
↳ This version collapsed and thus no results are reported.							
All negatives	4	55.7	62.9	0.864	2.622	0.625	3.966
No temporal negatives	4	52.7 ↓ (93.7%)	60.3 ↓ (83.9%)	0.837 ↓ (117.4%)	2.928 ↑ (104.8%)	0.601 ↓ (141.2%)	4.251 ↑ (137.7%)
No batch negatives	4	55.2 ↓ (15.6%)	62.5 ↓ (12.9%)	0.861 ↓ (13.1%)	2.661 ↑ (13.4%)	0.622 ↓ (17.7%)	3.981 ↑ (7.2%)

in the overall representation quality, which remains insufficient to reach state-of-the-art performance. Both semantic richness and temporal stability, key ingredients for general, transferable features, are not yet adequately captured. The t-SNE analysis (see also the extended analysis in Sec. E) shows that embeddings vary too abruptly over time and that semantic clusters are not fully coherent. Another limitation is the reliance on a single pre-training dataset (TartanAir-V2). While large and diverse, it differs from the driving-centric downstream datasets. Since the 1Mpx dataset, used by TESPEC, better reflects the characteristics of these tasks, pre-training on it may enhance transfer performance. An open question is whether such pre-training retains robustness on tasks from datasets unrelated to driving. Moreover, the finetuning pipelines for semantic segmentation and optical flow rely on prior work, and are potentially not optimal for our backbone. Developing dedicated, fully controlled fine-tuning frameworks would provide more reliable evaluations. This would also allow for experimentation with multitask fine-tuning. Finally, the current ablation study should be expanded to include optical flow to obtain a more complete understanding of the effects of each component.

Future work should focus on refining the contrastive objective to better balance semantic discrimination and temporal stability. Combining t-SNE-guided loss analysis with controlled ablations could uncover mechanisms that foster both. Overall, SPICE represents an initial step toward a new branch of event-based pre-training centered on predictive understanding rather than masking.

6. Conclusion

In this paper, we propose SPICE, a self-supervised latent prediction framework for event-based representation learning. Our pre-training consistently improves downstream performance over training from scratch, though there remains clear room for improvement. Our experiments underline the importance of event-specific design choices such as negative group normalization and active-patch selection. Our findings through t-SNE analyses highlight key directions for improving future latent prediction schemes, paving the way toward more temporally stable and semantically aware event representations.

Acknowledgments

I would like to express my sincere gratitude to PhD. candidate Yilun Wu for his continuous support, availability, and invaluable guidance throughout this project. I also want to thank prof. Guido C.H.E. de Croon for his insightful discussions and for helping steer the direction of this work. Finally, I would like to thank dr. Holger Caesar for facilitating my thesis.

References

[1] Yuki Markus Asano, Christian Rupprecht, and Andrea Vedaldi. Self-labelling via simultaneous clustering and representation learning, 2020. 865. 2

[2] Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-Supervised Learning from Images with a Joint-Embedding Predictive Architecture, 2023. 256. 2

[3] Nicolas Ballas, Li Yao, Chris Pal, and Aaron Courville. Delving Deeper into Convolutional Networks for Learning Video Representations, 2016. arXiv:1511.06432 [cs]. 2

[4] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. BEiT: BERT Pre-Training of Image Transformers, 2022. 2814. 2

[5] Sami Barchid, José Mennesson, and Chaabane Djéraba. Exploring Joint Embedding Architectures and Data Augmentations for Self-Supervised Representation Learning in Event-Based Vision. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3903–3912, Vancouver, BC, Canada, 2023. IEEE. 7. 1, 2

[6] Adrien Bardes, Jean Ponce, and Yann LeCun. VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning, 2022. 1233. 2

[7] Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting Feature Prediction for Learning Visual Representations from Video. 2024. 33. 2

[8] Jonathan Binas, Daniel Neil, Shih-Chii Liu, and Tobi Delbruck. DDD17: End-To-End DAVIS Driving Dataset, 2017. arXiv:1711.01458 [cs]. 1, 3

[9] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep Clustering for Unsupervised Learning of Visual Features, 2019. 3349. 2

[10] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging Properties in Self-Supervised Vision Transformers, 2021. 5388. 1, 2

[11] Kenneth Chaney, Fernando Cladera, Ziyun Wang, Anthony Bisulco, M. Ani Hsieh, Christopher Korpela, Vijay Kumar, Camillo J. Taylor, and Kostas Daniilidis. M3ED: Multi-Robot, Multi-Sensor, Multi-Environment Event Dataset. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 4016–4023, Vancouver, BC, Canada, 2023. IEEE. 36. 1, 3

[12] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A Simple Framework for Contrastive Learning of Visual Representations, 2020. 20198. 1, 2

[13] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches, 2014. arXiv:1409.1259 [cs]. 2

[14] Tim N. A. den Blanken. Self-supervised representation learning for event-based vision data: A literature review. Technical report, MAVLab, Delft University of Technology, Delft, The Netherlands, 2025. Unpublished literature review. 2

[15] Anusha Devulapally, Md Fahim Faysal Khan, Siddharth Advani, and Vijaykrishnan Narayanan. Multi-Modal Fusion of Event and RGB for Monocular Depth Estimation Using a Unified Transformer-based Architecture. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2081–2089, Seattle, WA, USA, 2024. IEEE. 2

[16] Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised Visual Representation Learning by Context Prediction, 2016. 3412. 2

[17] Guillermo Gallego, Tobi Delbruck, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew Davison, Joerg Conradt, Kostas Daniilidis, and Davide Scaramuzza. Event-based Vision: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(1):154–180, 2022. 2002. 1

[18] Daniel Gehrig, Antonio Loquercio, Konstantinos G. Derpanis, and Davide Scaramuzza. End-to-End Learning of Representations for Asynchronous Event-Based Data, 2019. 369. 3

[19] Mathias Gehrig and Davide Scaramuzza. Recurrent Vision Transformers for Object Detection with Event Cameras, 2023. arXiv:2212.05598 [cs]. 2

[20] Mathias Gehrig, Willem Aarents, Daniel Gehrig, and Davide Scaramuzza. DSEC: A Stereo Event Camera Dataset for Driving Scenarios, 2021. arXiv:2103.06011 [cs]. 1, 3, 8, 9, 10, 11, 2, 4, 5, 6

[21] Mathias Gehrig, Mario Millhäusler, Daniel Gehrig, and Davide Scaramuzza. E-RAFT: Dense Optical Flow from Event Cameras, 2021. 141. 2, 10

[22] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised Learning, 2020. 6805. 1, 2, 5

[23] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304. JMLR Workshop and Conference Proceedings, 2010. ISSN: 1938-7228. 2

[24] Ryuhei Hamaguchi, Yasutaka Furukawa, Masaki Onishi, and Ken Sakurada. Hierarchical Neural Memory Network for Low Latency Event Processing, 2023. arXiv:2305.17852 [cs]. 9

[25] Tengda Han, Weidi Xie, and Andrew Zisserman. Video Representation Learning by Dense Predictive Coding, 2019. 436. 1, 2, 3, 8, 9, 12

[26] Tengda Han, Weidi Xie, and Andrew Zisserman. Memory-augmented Dense Predictive Coding for Video Representation Learning, 2020. 275. 2

[27] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum Contrast for Unsupervised Visual Representation Learning, 2020. 13505. 1, 2

[28] Javier Hidalgo-Carrió, Daniel Gehrig, and Davide Scaramuzza. Learning Monocular Dense Depth from Events, 2020. arXiv:2010.08350 [cs]. 2

[29] Sepp Hochreiter and Jurgen Schmidhuber. Long short-term memory. *Neural Computation*, 1997. 2

[30] Yuhuang Hu, Shih-Chii Liu, and Tobi Delbruck. v2e: From Video Frames to Realistic DVS Events, 2021. arXiv:2006.07722 [cs]. 3

[31] Zhenpeng Huang, Chao Li, Hao Chen, Yongjian Deng, Yifeng Geng, and Limin Wang. Data-efficient Event Camera Pre-training via Disentangled Masked Modeling, 2024. 0. 1, 2

[32] Simon Klenk, David Bonello, Lukas Koestler, Nikita Araslanov, and Daniel Cremers. Masked Event Modeling: Self-Supervised Pretraining for Event Cameras. In *2024 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2367–2377, Waikoloa, HI, USA, 2024. IEEE. 13. 1, 2, 11

[33] Haochen Li, Xin Zhou, Luu Anh Tuan, and Chunyan Miao. Rethinking Negative Pairs in Code Search, 2023. arXiv:2310.08069 [cs]. 2, 3

[34] Jianing Li, Jia Li, Lin Zhu, Xijie Xiang, Tiejun Huang, and Yonghong Tian. Asynchronous Spatio-Temporal Memory Network for Continuous Event-Based Object Detection. *IEEE Transactions on Image Processing*, 31:2975–2987, 2022. 2

[35] Yijin Li, Zhaoyang Huang, Shuo Chen, Xiaoyu Shi, Hongsheng Li, Hujun Bao, Zhaopeng Cui, and Guofeng Zhang. BlinkFlow: A Dataset to Push the Limits of Event-based Optical Flow Estimation, 2024. 25. 2, 3

[36] Yijin Li, Yichen Shen, Zhaoyang Huang, Shuo Chen, Weikang Bian, Xiaoyu Shi, Fu-Yun Wang, Keqiang Sun, Hujun Bao, Zhaopeng Cui, Guofeng Zhang, and Hongsheng Li. BlinkVision: A Benchmark for Optical Flow, Scene Flow and Point Tracking Estimation using RGB Frames and Events, 2025. arXiv:2410.20451 [cs]. 3

[37] Haotian Liu, Guang Chen, Sanqing Qu, Yanping Zhang, Zhijun Li, Alois Knoll, and Changjun Jiang. TMA: Temporal Motion Aggregation for Event-based Optical Flow. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9651–9660, Paris, France, 2023. IEEE. 25. 9, 4

[38] Haotian Liu, Sanqing Qu, Fan Lu, Zongtao Bu, Florian Roehrbein, Alois Knoll, and Guang Chen. PCDepth: Pattern-based Complementary Learning for Monocular Depth Estimation by Best of Both Worlds, 2024. arXiv:2402.18925 [cs]. 1, 10

[39] Xu Liu, Jianing Li, Xiaopeng Fan, and Yonghong Tian. Event-based Monocular Dense Depth Estimation with Recurrent Transformers, 2022. arXiv:2212.02791 [cs]. 2

[40] Yue Liu, Junqi Ma, Yufei Xie, Xuefeng Yang, Xingzhen Tao, Lin Peng, and Wei Gao. Contrastive predictive coding with transformer for video representation learning. *Neurocomputing*, 482:154–162, 2022. 27. 2

[41] Yuan Liu, Songyang Zhang, Jiacheng Chen, Kai Chen, and Dahua Lin. PixMIM: Rethinking Pixel Reconstruction in Masked Image Modeling, 2023. arXiv:2303.02416 [cs]. 6

[42] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows, 2021. 23443. 7, 1

[43] Guillaume Lorre, Jaonary Rabarisoa, Astrid Orcesi, Samia Ainouz, and Stephane Canu. Temporal Contrastive Pretraining for Video Action Recognition. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 651–659, Snowmass Village, CO, USA, 2020. IEEE. 53. 2

[44] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization, 2019. arXiv:1711.05101 [cs]. 8

[45] Leland McInnes, John Healy, and James Melville. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction, 2020. arXiv:1802.03426 [stat]. 8, 9, 10, 11

[46] Antoine Miech, Jean-Baptiste Alayrac, Lucas Smaira, Ivan Laptev, Josef Sivic, and Andrew Zisserman. End-to-End Learning of Visual Representations from Uncurated Instructional Videos, 2020. arXiv:1912.06430 [cs]. 2, 3

[47] Mohammad Mohammadi, Ziyi Wu, and Igor Gilitschenski. TESPEC: Temporally-Enhanced Self-Supervised Pretraining for Event Cameras, 2025. 0. 2, 3, 7, 8, 9, 10, 11, 6, 12

[48] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation Learning with Contrastive Predictive Coding, 2019. 9908. 1, 2, 3, 5, 6

[49] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning Robust Visual Features without Supervision, 2024. 1619. 1, 2

[50] Tian Pan, Yibing Song, Tianyu Yang, Wenhao Jiang, and Wei Liu. VideoMoCo: Contrastive Video Representation Learning with Temporally Adversarial Examples, 2021. 277. 2

[51] Manthan Patel, Fan Yang, Yuheng Qiu, Cesar Cadena, Sebastian Scherer, Marco Hutter, and Wenshan Wang. TartanGround: A Large-Scale Dataset for Ground Robot Perception and Navigation, 2025. arXiv:2505.10696 [cs] version: 2. 3

[52] Etienne Perot, Pierre de Tournemire, Davide Nitti, Jonathan Masci, and Amos Sironi. Learning to Detect Objects with a 1 Megapixel Event Camera, 2020. arXiv:2009.13436 [cs]. 1, 2, 3

[53] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-Shot Text-to-Image Generation, 2021. 4924. 2

[54] Kanchana Ranasinghe, Muzammal Naseer, Salman Khan, Fahad Shahbaz Khan, and Michael Ryoo. Self-supervised Video Transformer, 2022. 104. 2

[55] Henri Rebecq, Daniel Gehrig, and Davide Scaramuzza. ESIM: an Open Event Camera Simulator. 2018. 3, 7

[56] Alberto Sabater, Luis Montesano, and Ana C. Murillo. Event Transformer+. A multi-purpose solution for efficient event data processing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(12):16013–16020, 2023. 11. 3

[57] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo. Convolutional LSTM

Network: A Machine Learning Approach for Precipitation Nowcasting, 2015. arXiv:1506.04214 [cs]. 2

[58] Zhaoning Sun, Nico Messikommer, Daniel Gehrig, and Davide Scaramuzza. ESS: Learning Event-based Semantic Segmentation from Still Images, 2022. 74. 2, 8

[59] TartanAir. Tartanair-v2, 2025. https://tartanair.org/ [Accessed: 2025-11-03]. 3, 7

[60] Pierre de Tournemire, Davide Nitti, Etienne Perot, Davide Migliore, and Amos Sironi. A Large Scale Event-based Detection Dataset for Automotive, 2020. arXiv:2001.08499 [cs]. 1, 3

[61] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9 (86):2579–2605, 2008. 5, 6, 8, 9, 10, 11

[62] Feng Wang and Huaping Liu. Understanding the Behaviour of Contrastive Loss. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2495–2504, Nashville, TN, USA, 2021. IEEE. 3

[63] Tongzhou Wang and Phillip Isola. Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere, 2022. arXiv:2005.10242 [cs]. 3

[64] Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian Scherer. TartanAir: A Dataset to Push the Limits of Visual SLAM, 2020. arXiv:2003.14338 [cs]. 3

[65] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified Perceptual Parsing for Scene Understanding, 2018. arXiv:1807.10221 [cs]. 8

[66] Jiarui Xu and Xiaolong Wang. Rethinking Self-supervised Correspondence Learning: A Video Frame-level Similarity Perspective, 2021. 104. 2

[67] Yan Yang, Liyuan Pan, and Liu Liu. Event Camera Data Pre-training, 2023. 23. 1, 7, 8, 10

[68] Yan Yang, Liyuan Pan, and Liu Liu. Event Camera Data Dense Pre-training, 2024. 2. 1, 2, 3, 7, 8, 9, 10, 11, 4, 6

[69] Zhen Yao, Xiaowen Ying, and Mooi Choo Chuah. Rethinking RGB-Event Semantic Segmentation with a Novel Bidirectional Motion-enhanced Event Representation, 2025. arXiv:2505.01548 [cs] version: 1. 1

[70] Yaozu Ye, Hao Shi, Kailun Yang, Ze Wang, Xiaoting Yin, Yining Lin, Mao Liu, Yaonan Wang, and Kaiwei Wang. Towards Anytime Optical Flow Estimation with Event Cameras, 2023. 6. 4

[71] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow Twins: Self-Supervised Learning via Redundancy Reduction, 2021. 2438. 2

[72] Alex Zihao Zhu, Dinesh Thakur, Tolga Ozaslan, Bernd Pfrommer, Vijay Kumar, and Kostas Daniilidis. The Multi Vehicle Stereo Event Camera Dataset: An Event Camera Dataset for 3D Perception. *IEEE Robotics and Automation Letters*, 3(3):2032–2039, 2018. arXiv:1801.10202 [cs]. 1, 3, 9, 10, 4, 5, 6, 7

[73] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Unsupervised Event-based Learning of Optical Flow, Depth, and Egomotion, 2018. 569. 3, 4, 10

SPICE: Self-supervised Predictive Coding of Events

Supplementary Material

Appendix A. Additional Experimental Results

For DSEC semantic segmentation, we provide class-wise quantitative results in Tab. 6. Again, it shows ECDDP is clearly ahead, though our method performs slightly better in classifying poles and roads (IoU). Tab. 6 clarifies that ECDDP's overall superior results (see Tab. 1) are mainly the product of its ability to better classify less prevalent classes, such as fences, walls and people. Note that the values presented for ECDDP in the table are our own reproduced results, since the authors did not release their fine-tuning code or checkpoints. Next, we also provide more qualitative results for all downstream tasks.

Table 6. **Class-wise semantic segmentation results on DSEC** [20]. Intersection over union (IoU) and accuracy (Acc) are given per class. Best results are highlighted in **bold**, second best are underlined.

Class	IoU ↑			Acc ↑		
	No Pre-training	ECDDP	SPICE (**ours**)	No Pre-training	ECDDP	SPICE (**ours**)
background	93.2	**94.8**	93.8	96.4	**97.8**	97.0
building	74.1	**79.9**	77.3	88.8	**91.9**	90.9
fence	9.2	**18.4**	13.7	11.3	**22.8**	18.1
people	15.4	**31.0**	30.8	16.4	33.7	**35.9**
pole	31.3	33.3	**33.9**	39.9	**43.7**	43.3
road	92.9	93.5	**93.9**	96.5	96.3	**97.0**
sidewalk	63.0	**68.6**	68.6	81.3	**87.2**	85.4
vegetation	78.8	**82.9**	81.4	87.8	**89.4**	88.8
car	71.0	**78.9**	74.5	81.5	**87.7**	83.6
wall	10.4	**23.4**	10.2	14.1	**33.5**	14.3
traffic sign	38.2	**45.0**	43.6	43.9	**53.2**	52.9

DSEC Semantic. Additional qualitative samples for semantic segmentation on DSEC are shown in Fig. 11. Our predictions improve considerably over the no pre-training case, yet still smaller objects such as fences and people are not detected (2nd and 3rd row) and predictions have jittery boundaries (1st, 3rd and 4th row).

DSEC Depth. Additional qualitative samples for depth estimation on DSEC are shown in Fig. 12. Our pre-training leads to sharper edges and less noisy predictions as compared to the no pre-training setting, especially visible in the 4th row.

MVSEC Depth. Additional qualitative samples for depth estimation on MVSEC are shown in Fig. 13. Our pre-training leads to sharper edges (2nd, 3rd and 4th row), but in general, predictions are quite coarse. Moreover, when performing intermediate fine-tuning on DSEC before fine-tuning on MVSEC we observe a significant performance boost, resulting in $\delta_1 = 0.650$, $\delta_2 = 0.837$, $\delta_3 = 0.929$, Abs $= 3.69$, RMS $= 6.84$, RMSlog $= 0.34$, which is state-of-the-art performance across all metrics except for RMS.

DSEC Flow. Additional qualitative samples for optical flow estimation on DSEC are shown in Fig. 14. Both ECDDP and our pre-training show very little qualitative improvement over the no pre-training setting. This calls for further investigation of the representation quality to improve performance on optical flow.

Appendix B. Methodological Details

B.1. Pre-training Setup

Tab. 7 summarizes the key hyperparameters of the pre-training framework. The full pre-trained network contains 35.7M trainable parameters: 27.5M for the Swin-T/7 student encoder, 1.1M for the feature pyramid network (FPN) combining outputs from all Swin-T stages, 3.5M for the ConvGRU aggregator, and 1.2M for each of the three predictor heads. The teacher encoder has the same number of parameters as the student, but its weights are updated as an exponential moving average (EMA) of the student's parameters and are therefore not trainable.

Backbone and Feature Pyramid Network. The Swin-T/7 backbone follows the standard Swin Transformer architecture [42]. An upsample FPN is applied on top of the four Swin-T feature stages to align all feature maps to a common spatial resolution and channel dimension. A dedicated branch processes each stage output: the first two are downsampled by factors of 4 and 2 (via average pooling), the third is used directly with a 1×1 convolution, and the fourth is upsampled by a factor of 2. All branches are projected to the same channel width ($8\times$ the base embedding dimension) using 1×1 convolutions, and the resulting feature maps are averaged to form a unified output. A learnable weighting mechanism, initialized uniformly as [0.25, 0.25, 0.25, 0.25], allows the network to adaptively emphasize different Swin-T stages if beneficial. This weighting behavior remains untested but represents an interesting direction for future exploration.

Aggregation and Prediction Modules. The ConvGRU aggregation module is adopted directly from DPC [25]. Each predictor consists of two sequential 1×1 convolutional layers separated by a GELU activation. Both convolutions preserve the feature dimensionality, operating on and outputting tensors with the same hidden size as the aggregator.

Training Data and Temporal Setup. Due to storage con-

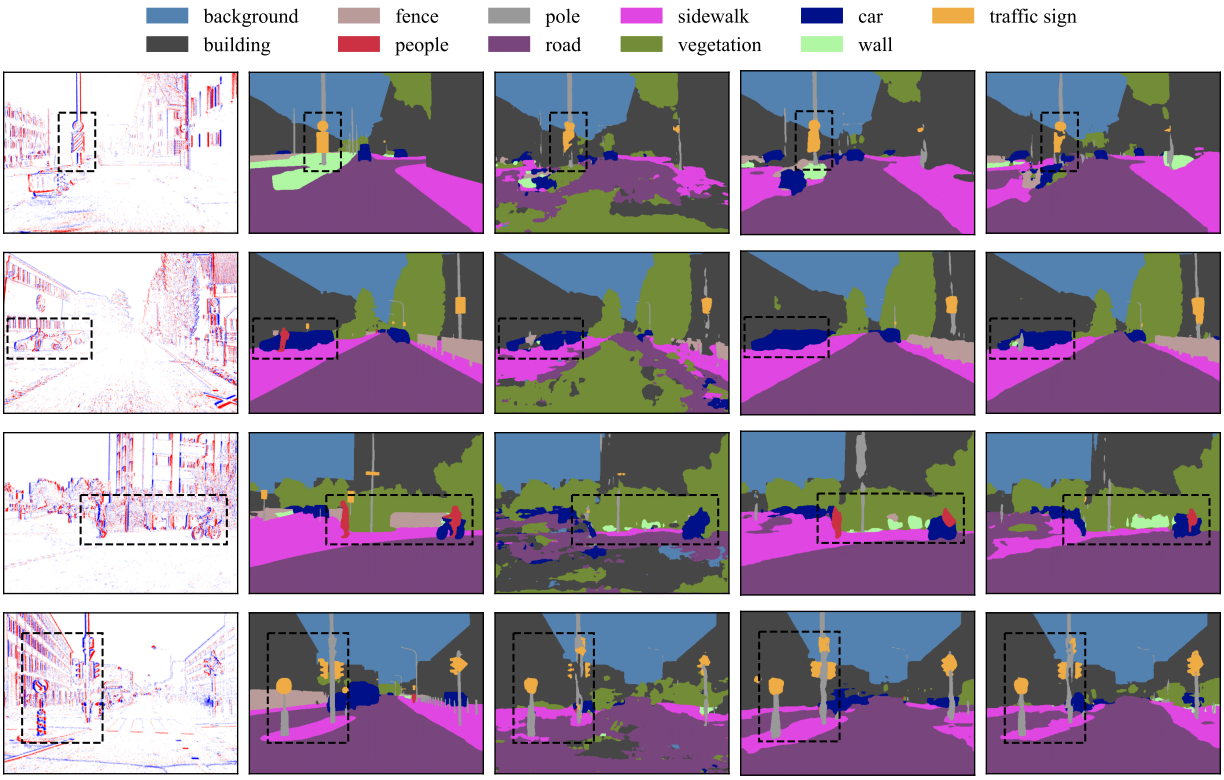


Figure 11. **Additional qualitative semantic segmentation results on DSEC** [20].

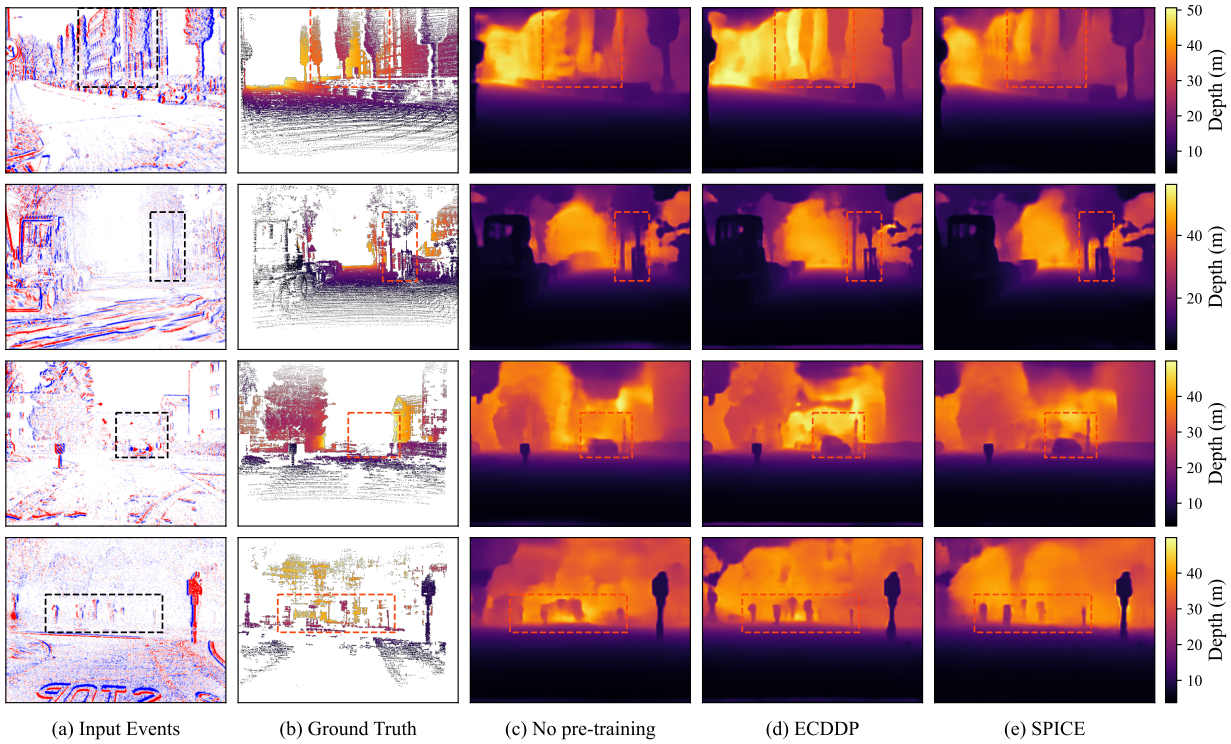
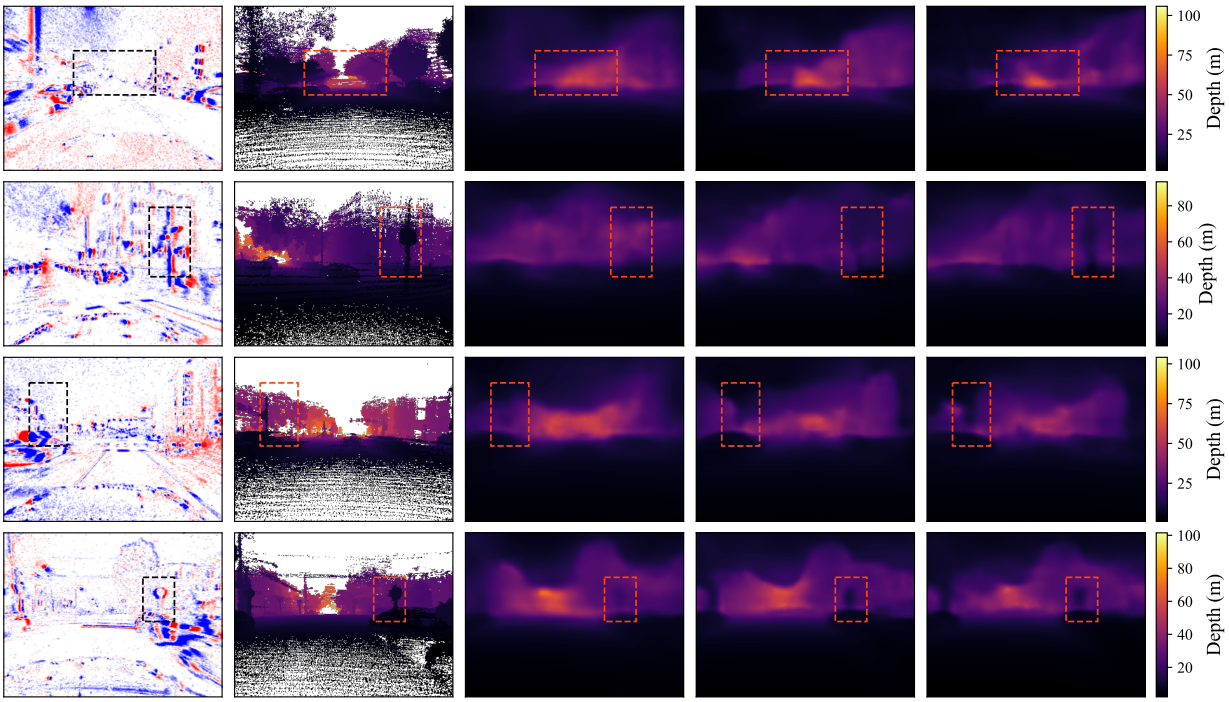
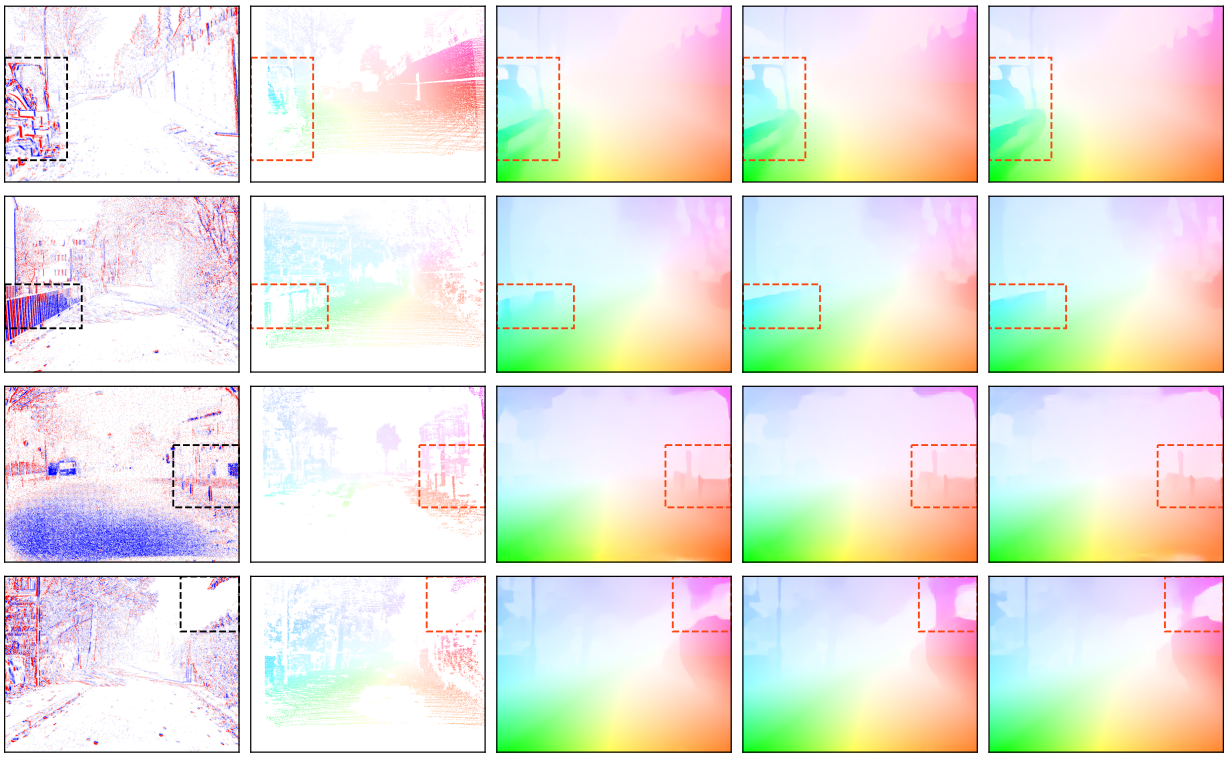


Figure 12. **Additional qualitative depth estimation results on DSEC** [20].



(a) Input Events (b) Ground Truth (c) No pre-training (d) ECDDP (e) SPICE

Figure 13. **Additional qualitative depth estimation results on MVSEC** [72].



(a) Input Events (b) Ground Truth (c) No Pre-training (d) ECDDP (e) SPICE

Figure 14. **Additional qualitative optical flow estimation results on DSEC** [20].

straints, approximately 82% of the full TartanAir-V2 dataset is used, corresponding to about 14.85 hours of event data per training epoch. We aggregate 49ms of events per sample. This somewhat unusual window length results from the framework's millisecond-level indexing and the need for half-bin overlapping voxel grids. With six bins and a target of 50ms total duration, 49ms is the closest feasible choice, corresponding to a half-bin temporal span of 7ms.

Table 7. **Pre-training parameters.** Overview of all relevant pre-training parameters.

Parameter	Value
Dataset	TartanAir-V2
Batch size	64
Training steps	298,394
Optimizer	AdamW
Peak learning rate	1×10^{-4}
Warmup steps	39,786
Scheduler	Cosine annealing
Gradient clipping	1.0
Precision	BF16-Mixed
Augmentations	Tab. 8a
Representation	Our event voxel grid with 6 temporal bins
Loss	GW-InfoNCE
Teacher EMA	0.998
Contrastive temperature	0.15
Single sample dimension	$8 \times 6 \times 224 \times 224$
Prediction scheme	Parallel
Sequence length (voxels)	8
Prediction steps (voxels)	3
GPU	1× GTX5090
Training time	59.5h

B.2. Fine-tuning Details

B.2.1. Semantic Segmentation

We fine-tune our encoder on DSEC [20], which is a dataset that consists of 65 driving sequences, of which 11 have semantic maps available. Our fine-tuning codebase is taken from ECDDP [68] and adapted with our event representation. Relevant hyperparameters are detailed in Tab. 9.

Data split. Tab. 10 lists the used train and validation split.

Model size. The pretrained Swin-T7 backbone has 27.5M parameters. The attached heads, UperNet and FCNHead, are taken from ECDDP and have 31.4M and 0.9M parameters, respectively.

B.2.2. Depth Estimation

We fine-tune our encoder on DSEC [20] and MVSEC [72]. The former has 41 driving sequences with ground truth depth, while the latter has 5. Relevant hyperparameters are detailed in Tab. 9.

(a) SPICE pre-training.

Augmentation	Parameters	Output size	Probability
Random Resized Crop	scale=[0.8, 1.0], ratio=[0.75, 1.33]	224×224	1.0
Random Horizontal Flip	-	-	0.5

(b) DSEC semantic segmentation.

Augmentation	Parameters	Output size	Probability
Random Resized Crop	scale=[0.8, 1.0], ratio=[0.75, 1.33]	480×640	1.0
Random Horizontal Flip	-	-	0.5

(c) DSEC depth estimation.

Augmentation	Parameters	Output size	Probability
Random Resized Crop	scale=[0.8, 1.0], ratio=[0.75, 1.33]	480×640	1.0
Random Horizontal Flip	-	-	0.5

(d) MVSEC depth estimation.

Augmentation	Parameters	Output size	Probability
Random Resized Crop	scale=[0.8, 1.0], ratio=[0.75, 1.33]	260×346	1.0
Random Horizontal Flip	-	-	0.5

(e) DSEC optical flow estimation.

Augmentation	Parameters	Output size	Probability
Resize	fx=[$2^{-0.2}$,$2^{0.4}$], fy=[$2^{-0.2}$,$2^{0.4}$]	480·fx×640·fy	0.8
Random Crop	-	288×384	1.0
Random Horizontal Flip	-	-	0.5
Random Vertical Flip	-	-	0.1

Table 8. **Augmentations.** Overview of augmentations used during pre-training and fine-tuning.

Data split. Tabs. 11 and 12 list the train and validation sequences used for DSEC and MVSEC, respectively.

Model size. For both datasets, the pretrained Swin-T7 backbone has 27.5M parameters and the attached depth decoder has 3.9M parameters.

B.2.3. Optical Flow Estimation

We fine-tune our encoder on DSEC [20], which has 18 driving sequences with ground truth optical flow. We follow ECDDP [68] in using TMA [37] architectures for fine-tuning. Since ECDDP did not release fine-tuning code for optical flow, we implemented it ourselves based on the details in their supplementary material. Relevant hyperparameters are detailed in Tab. 9.

Data split. Tab. 13 lists the used train and validation split, which is the same one as TMA used.

Model size. The first two stages of the pretrained Swin-T/7 backbone have 1.2M parameters; as such, both the feature encoder and context encoder have 1.2M parameters each. The motion feature encoder has 0.8M parameters, the motion pattern aggregation module has 0.1M parameters and the update network has 2.5M parameters. TMA [37] provides details on each of these modules.

Table 9. **Fine-tuning parameters.** Overview of all relevant fine-tuning parameters for all downstream tasks.

Parameter	DSEC Semantic	DSEC Depth	MVSEC Depth	DSEC Flow
Batch size	8	16	8	6
Training steps	25,200	71,699	22,845	150,000
Optimizer	AdamW	AdamW	AdamW	AdamW
Peak learning rate	2.5×10^{-5}	1×10^{-4}	4×10^{-5}	2×10^{-4}
Min. learning rate	1×10^{-6}	1×10^{-5}	6×10^{-5}	1×10^{-6}
Warmup steps	5,040	2,868	7,615	1,500
Scheduler	Cosine annealing	Cosine annealing	Cosine annealing	Cosine annealing
Gradient clipping	None	None	None	1.0
Precision	FP16	FP32	FP32	FP32
Augmentations	Tab. 8b	Tab. 8c	Tab. 8d	Tab. 8e
Representation	Our event voxel grid with 6 temporal bins	Our event voxel grid with 6 temporal bins	Our event voxel grid with 6 temporal bins	Default normalized event voxel grid with 15 temporal bins
Loss	$1\times$ DiceLoss + $1\times$ CrossEntropyLoss	$1\times$ ScaleInvariantLoss + $0.25\times$ MultiScaleGradientLoss	$1\times$ ScaleInvariantLoss + $0.25\times$ MultiScaleGradientLoss	L1 SequenceLoss with $\gamma = 0.8$
Keep patch embed. weights	True	True	True	False
↳ When using the same input representation for pre-training and fine-tuning, the weights of the patch embedding layer of the Swin-T/7 can also be used.				
Single sample dimension	6×480×640	6×480×640	6×260×346	15×288×384
GPU	$1\times$ A100 (MIG 20GB)	$1\times$ GTX5090	$1\times$ GTX5090	$1\times$ GTX5090
Training time	5.5h	3.0h	0.7h	11.3h

Table 10. Train and validation split for DSEC [20] Semantic.

Train	Validation
zurich_city_00_a	zurich_city_13_a
zurich_city_01_a	zurich_city_14_c
zurich_city_02_a	zurich_city_15_a
zurich_city_04_a	
zurich_city_05_a	
zurich_city_06_a	
zurich_city_07_a	
zurich_city_08_a	

Table 11. Train and validation split for DSEC [20] Depth.

Train	Validation
interlaken_00_{e,d,f}	interlaken_00_{c,g}
zurich_city_00_{a,b,d,e,f}	thun_00_a
zurich_city_01_{b,c,d,e,f}	zurich_city_01_a
zurich_city_02_{a,b,c,d,e}	zurich_city_04_a
zurich_city_03_{a}	zurich_city_06_a
zurich_city_04_{b,c,d,e,f}	zurich_city_09_a
zurich_city_05_{a,b}	
zurich_city_07_{a}	
zurich_city_08_{a}	
zurich_city_09_{b,c,d,e}	
zurich_city_10_{a,b}	
zurich_city_11_{b,c}	

Table 12. Train and validation split for MVSEC [72] Depth.

Train	Validation
outdoor_day2	outdoor_day1
	outdoor_night1
	outdoor_night2
	outdoor_night3

Table 13. Train and validation split for DSEC [20] Flow.

Train	Validation
thun_00_a	zurich_city_05_b
zurich_city_01_a	zurich_city_06_a
zurich_city_02_{a,c,d,e}	zurich_city_10_b
zurich_city_03_a	zurich_city_11_c
zurich_city_05_a	
zurich_city_07_a	
zurich_city_08_a	
zurich_city_09_a	
zurich_city_10_a	
zurich_city_11_{a,b}	

B.3. Computational Resources

As shown in Tab. 7, we pretrain for 59.5 hours on a single RTX5090. TESPEC reports training times of 45h, 120h, and 135h for ECDP, ECDDP, and their own method, respectively, using four RTX6000 GPUs. Considering that a single RTX5090 and four RTX6000 offer comparable computational performance, these training times show that our method is rather efficient.

Appendix C. Dataset Analysis

C.1. Dataset Statistics

To fully leverage the pre-trained model, we aim to keep the input representation consistent between pre-training and fine-tuning. Limiting shifts in the input distribution helps the network concentrate on the target task rather than adapting to altered input statistics. We therefore analyzed each dataset to obtain its event statistics and, based on the results in Tab. 14, aligned the input representations using the average number of events per pixel per second. This yields accumulation times of 60 ms for DSEC and 400 ms for MVSEC, providing comparable event densities across datasets and tasks.

Table 14. **Dataset statistics.** Overview of relevant statistics of the used datasets.

Parameter	TartanAir-V2	DSEC [20]	MVSEC [72]
Sensor size	640×640	480×640	260×346
Number of sequences	500	65	5
Total events	1876G	51.1G	913M
Total duration in sec.	75893	3754	1829
Average events per sec.	24.718M	13.620M	0.500M
Median events per sec.	15.286M	11.271M	0.404M
Std. events per sec.	17.779M	5.618M	0.137M
Average events per pixel per sec	49.56	41.32	4.95
Average positive events ratio	0.515	0.534	0.391
Average negative events ratio	0.485	0.466	0.609

C.2. MVSEC Dataset Quality Discussion

The MVSEC dataset [72] has been instrumental in advancing event-based vision and remains a common choice for depth estimation, especially for event-based SSL methods [47, 68]. However, several limitations now make it less suitable as a benchmark compared to newer datasets such as DSEC [20].

MVSEC provides only a few sequences (typically five used for depth tasks) with limited variation in motion and environment. DSEC, in contrast, offers 41 sequences with ground truth and greater diversity. The event rate in MVSEC is also low, over eight times fewer events per pixel per second than DSEC. Moreover, MVSEC's resolution is relatively low, while DSEC offers higher-resolution recordings that can be downsampled if desired. Finally, MVSEC exhibits an unbalanced polarity distribution (39.1% positive, 60.9% negative).

The quality of the MVSEC ground-truth depth also introduces several challenges. First, events and depth maps exhibit small but noticeable misalignments (Fig. 15a). Second, some objects are incomplete or partially missing in the depth annotations (Figs. 15b and 15c). Third, the depth maps can be sparse or discontinuous, resulting in ambiguous supervision signals (Figs. 15d and 15e). Finally, various LiDAR artifacts remain embedded in the ground truth (Figs. 15f to 15j).

Given these factors, we would recommend adopting DSEC as the default benchmark for event-based depth estimation. Although originally designed for disparity estimation, DSEC provides calibrated camera parameters that allow direct conversion to depth. The conversion script used in this work is available upon request.

Appendix D. Additional Discussion and Motivations

D.1. Latent Space Prediction

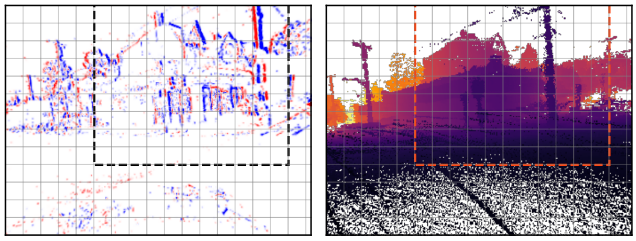
Recent state-of-the-art methods for event-based SSL, such as ECDDP [68] and TESPEC [47], rely on masking strategies. ECDDP employs a student-teacher framework where the student learns to reconstruct the masked representations provided by the teacher, while TESPEC aims to reconstruct missing regions in an intensity image derived from events.

Although these approaches are effective, our design departs from this paradigm in favor of latent space prediction without masking. Masking is primarily a spatial completion task, effective for reconstructing missing semantic content in static images, but it may not align naturally with the structure of event data, and in a broader sense, the way humans learn from continuous sensory input. Events encode fine-grained spatiotemporal dynamics rather than static semantics, and we argue that predictive learning over time better exploits this property. In our view, learning to predict future representations from past observations encourages a model to capture both spatial and temporal regularities.

We also choose to predict in the latent space rather than the raw input space. Direct pixel-level prediction would require the model to reconstruct all fine details of the input signal, demanding complex generative modeling and emphasizing low-level reconstruction rather than abstract understanding. Prior work in masked modeling, such as PixMIM [41], shows that reconstruction targets strongly bias the learned features (e.g., toward high- or low-frequency content), underlining that pixel reconstruction can overemphasize details that are not necessarily the most informative. Following the reasoning of CPC [48], predicting latent features instead allows the model to focus on information shared between past and future representations, maximizing mutual information between meaningful abstractions rather than raw pixels. This encourages the network to encode features that are predictive and invariant.

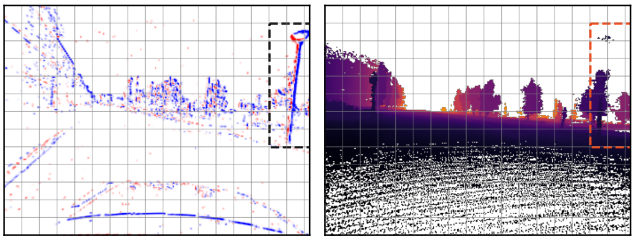
D.2. Temporal Prediction Range

As noted by the authors of CPC [48], next-step prediction mainly exploits the local smoothness of a signal, leading to representations that capture short-term continuity. When the prediction horizon extends further into the future, the mutual information between past and future decreases, forc-



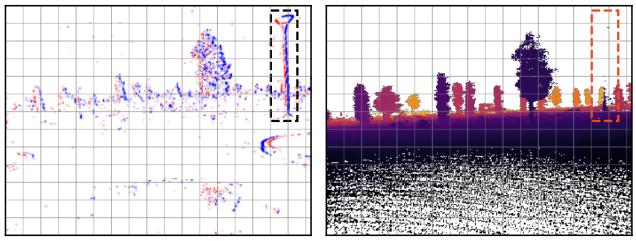
(a) Input Events (b) Ground Truth Depth

(a) Sequence `outdoor_day2`, index 11573: Misaligned ground truth.



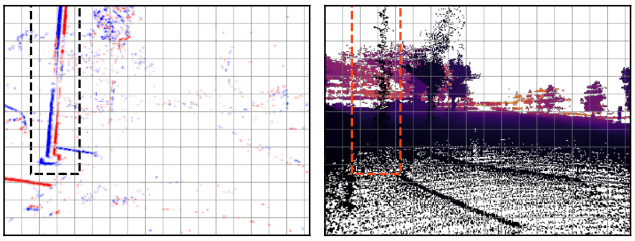
(a) Input Events (b) Ground Truth Depth

(b) Sequence `outdoor_day1`, index 4516: Incomplete pole.



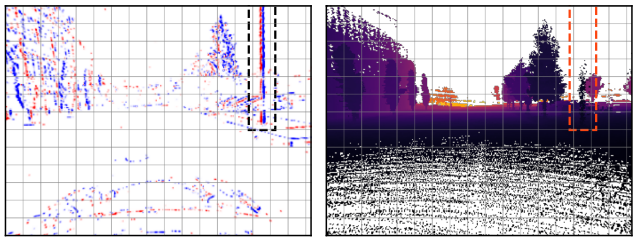
(a) Input Events (b) Ground Truth Depth

(c) Sequence `outdoor_day1`, index 1536: Incomplete pole.



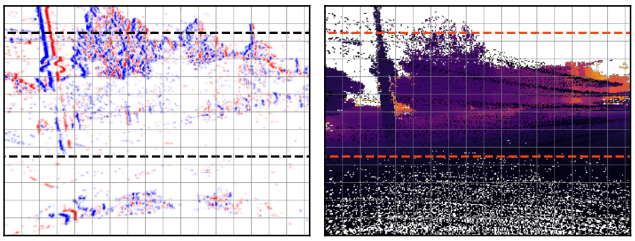
(a) Input Events (b) Ground Truth Depth

(d) Sequence `outdoor_day1`, index 4976: Non dense pole.



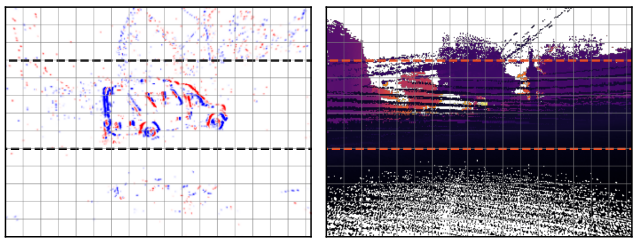
(a) Input Events (b) Ground Truth Depth

(e) Sequence `outdoor_day1`, index 1746: Non dense pole.



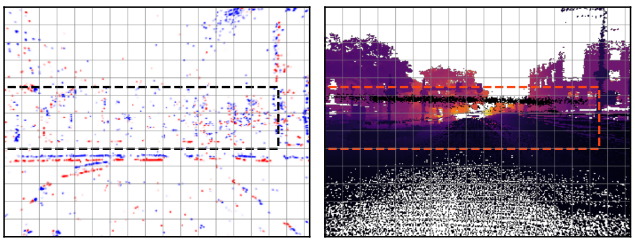
(a) Input Events (b) Ground Truth Depth

(f) Sequence `outdoor_day2`, index 4693: Artifacts.



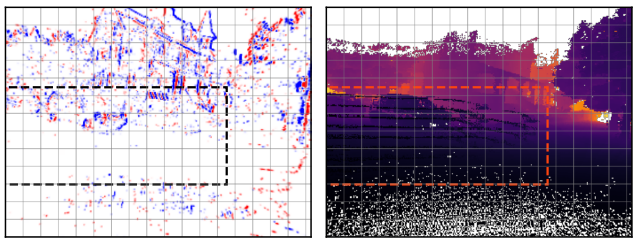
(a) Input Events (b) Ground Truth Depth

(g) Sequence `outdoor_day2`, index 4553: Artifacts.



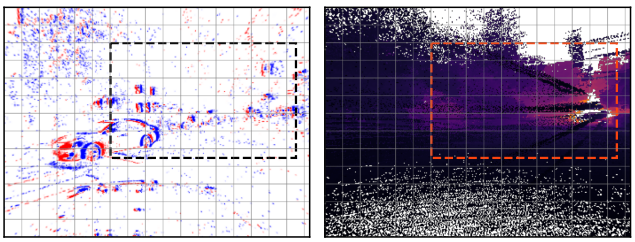
(a) Input Events (b) Ground Truth Depth

(h) Sequence `outdoor_night1`, index 2623: Artifacts.



(a) Input Events (b) Ground Truth Depth

(i) Sequence `outdoor_day2`, index 1183: Artifacts.



(a) Input Events (b) Ground Truth Depth

(j) Sequence `outdoor_night2`, index 438: Artifacts.

Figure 15. **Issues with ground truth depth in MVSEC** [72]. Examples of misaligned ground truth (a), incomplete objects (b and c), ambiguous regions (d and e) and artifacts (f to j).

ing the model to rely on more global and persistent structures, such as objects and motion patterns. This motivates our choice to predict multiple timesteps ahead. So far, we have adopted a relatively modest prediction range (3 steps; 133ms), but extending it to much longer horizons (e.g., beyond 1000ms) presents an interesting direction for future work.

D.3. Negative Group Normalization

We observed that normalizing the contribution of each negative group (temporal, spatial, and batch) is essential to prevent representation collapse, whereas the RGB-based DPC baseline did not require such normalization. We attribute this sensitivity partly to the sparse and asynchronous nature of event data, making patches less discriminative.

However, as mentioned, the difference may not solely stem from the data modality. DPC primarily targets clip-level representation learning for action recognition, where the context embedding is used to predict a single label for an entire sequence. In contrast, our framework supports dense prediction tasks, such as segmentation, depth and optical flow, where local patch embeddings carry more importance. This difference in supervision granularity likely accentuates the impact of negative imbalance.

To further investigate this phenomenon, a promising direction would be to follow DPC [25] and perform a t-SNE analysis of the global context embeddings across different sequences (as shown in their supplementary Figure 6), rather than at the patch level as done in our current analysis. Such an experiment could help reveal whether normalization primarily affects global contextual organization or local embedding diversity.

Appendix E. Representation Space Analysis

t-SNE [61] has been a central diagnostic tool in this work, offering intuition about the structure of the learned embedding space. However, relying on a single nonlinear projection risks drawing conclusions that are specific to the visualization method. We therefore provide additional analysis along three complementary dimensions. First, we reproduce Fig. 3 using UMAP [45] to verify whether the same collapse behaviour appears when group-normalized negatives are removed, see Sec. E.1. Second, we assess the representational capacity of the pretrained model by comparing its patch embeddings to those of a randomly initialized network, using both t-SNE and UMAP, see Sec. E.2. Third, to contextualize performance against the (prior) state of the art, ECDDP, we compare embeddings at a backbone stage rather than using the entire backbone with FPN, see Sec. E.3. Each of these analyses is presented in its own subsection. Before doing so, we describe the visualization procedure used throughout.

Setup. To visualize the embedding structure of an event sequence, each event voxel grid is processed by the backbone (or backbone with FPN, depending on the experiment) to obtain patch-level feature vectors. All patches from the sequence are jointly projected into a two-dimensional space using a nonlinear manifold embedding method (t-SNE or UMAP) to ensure consistent relationships across time. Per event voxel, each projected point is assigned a unique color, which is overlaid with partial transparency on the corresponding spatial patch in the event frame. Spatial regions with similar colors therefore reflect neighborhoods in feature space, providing an interpretable view of the model's learned representation. Note that the colormap is relative to each frame, and as such, the same color does not necessarily carry the same meaning across frames.

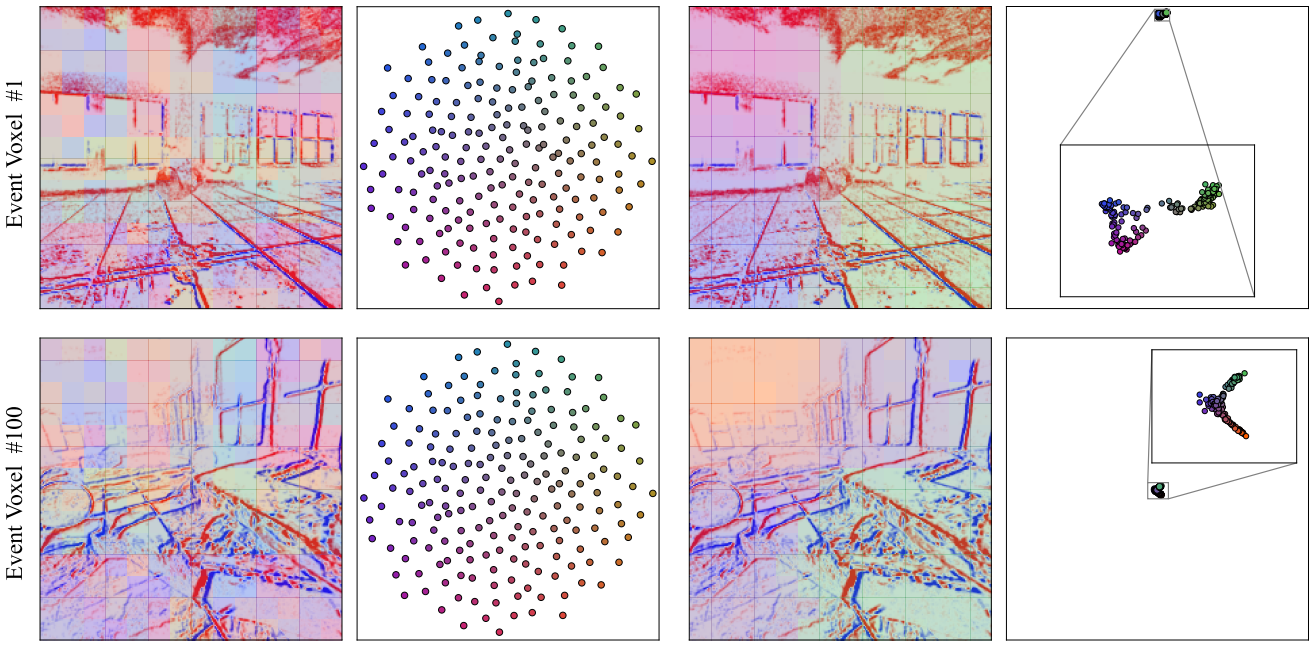
E.1. Collapse analysis with UMAP

While developing SPICE, we initially saw that the fine-tuning performance of pretrained models matched or underperformed random initialization. This indicated collapse, which also showed in the t-SNE [61] projection of patch embeddings, as shown in Fig. 3. After normalizing negative groups, fine-tuning improved, confirming the collapse. The remaining question was whether the apparent temporal invariance in Fig. 3a was genuine or (partially) a t-SNE artifact. Repeating the analysis with UMAP [45] (Fig. 16) yields the same static structure, showing that the embeddings were indeed nearly unchanged over time.

E.2. Learned Representation Space

To assess what the model actually learns during pretraining, we compare the patch embeddings produced by our pretrained SPICE model (Swin-T/7 with FPN, as in Fig. 3) to those from a randomly initialized network. We project embeddings from several event frames of the same sequence using both t-SNE and UMAP. The resulting visualizations (Fig. 17) reveal clear differences in structure and temporal stability between the two models.

The randomly initialized model (Figs. 17a and 17c) produces embeddings that primarily reflect the presence or absence of events, but exhibit no coherent clustering; patches from any given frame are scattered throughout the projection space. In contrast, the SPICE model (Figs. 17b and 17d) yields more organized representations: embeddings are clustered more compactly, and different frames occupy distinct regions of the space. Some emerging semantic structure is visible, for example, sky patches tend to map to similar areas, but the clustering remains coarse. Moreover, the substantial variation in embeddings across frames with similar or overlapping content indicates limited temporal consistency. In other words, the model distinguishes different scene elements but does not reliably map the same element across frames to similar regions of the



(a) All negatives treated equally.

(b) Group normalized and Gaussian-weighted negatives.

Figure 16. **UMAP [45] projection of patch embeddings for two different voxels and two loss configurations.** This figure is exactly the same as Fig. 3, but using UMAP projection instead of t-SNE [61] projection. The conclusion is the same: treating all negatives equally (a) leads to representation collapse, while balancing the negatives per group and using a Gaussian mask (b) leads to more semantically structured embeddings.

representation space. Combined with the coarse semantic grouping observed, this suggests significant room for improving the stability and granularity of the learned representations.

E.3. Comparison against ECDDP

Because our pretrained backbones do not yet match state-of-the-art downstream performance, it is instructive to compare our learned representations to those of a now prior state-of-the-art method, ECDDP. A direct comparison at the backbone with FPN level is not possible: ECDDP's FPN introduces strong checkerboard artifacts due to a `ConvTranspose2d` layer, which we avoided in our own design. Instead, we extract embeddings from the third stage of the Swin-T/7 backbone, whose spatial resolution matches the FPN output without additional pooling or upsampling. Fig. 18 and Fig. 19 show t-SNE and UMAP projections of patch embeddings from Swin-T/7 stage-3 for a randomly initialized model, for ECDDP, and for SPICE.

The randomly initialized backbone (Figs. 18a and 19a) shows little meaningful structure: similarly to Fig. 17 embeddings react mainly to the presence or absence of events and remain weakly organized, with only mild grouping by visual appearance in UMAP. ECDDP (Figs. 18b and 19b) exhibits great improvements: clear semantic clusters emerge, and these clusters remain stable across time, e.g., floor, shelves, and ceiling consistently occupy similar regions across frames. SPICE (Figs. 18c and 19c) also forms recognizable semantic groupings, but the cluster locations drift more across time, indicating weaker temporal consistency compared to ECDDP.

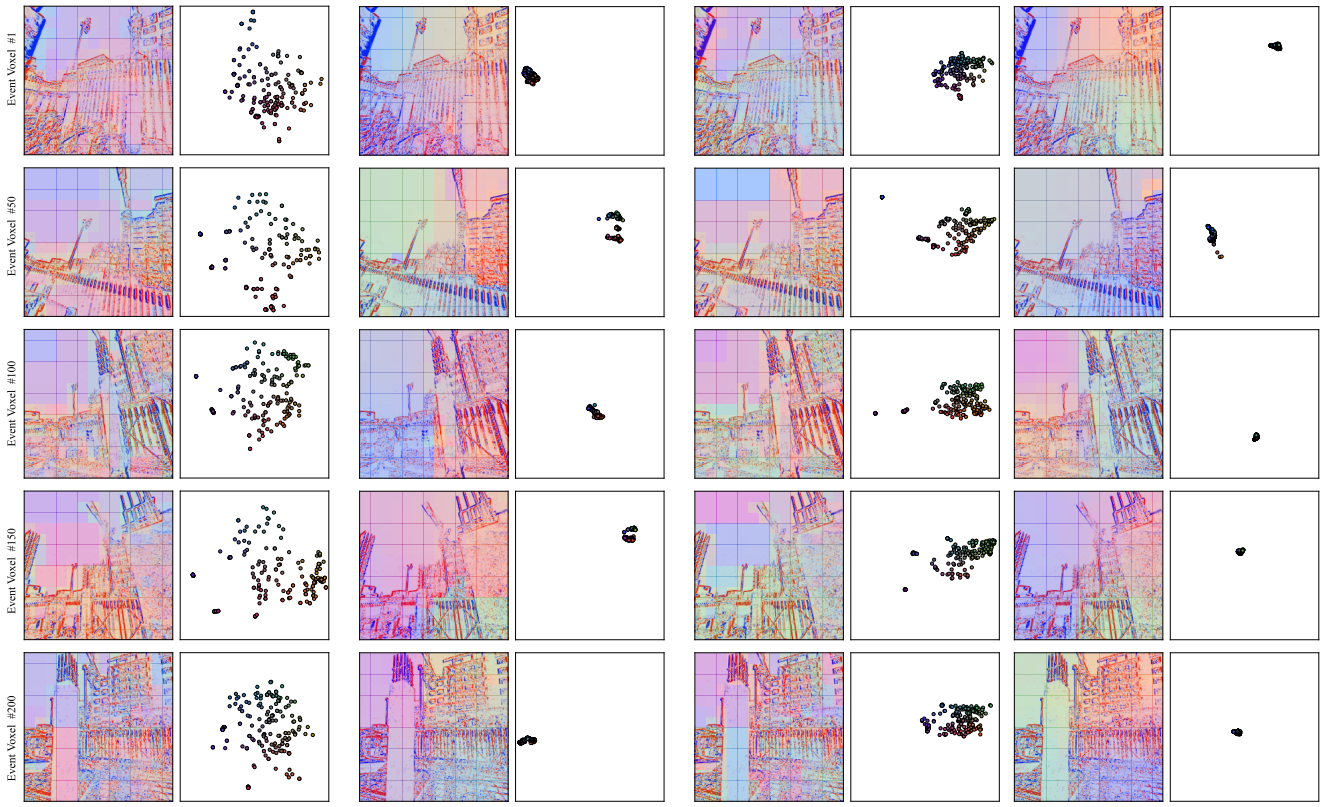
Appendix F. Limitations and Future Work

The primary limitation remains the overall representation quality, which lacks sufficient semantic structure and temporal stability. This shortcoming underpins and motivates all proposed future investigations. A further limitation concerns benchmarking: most of our comparisons are made against ECDDP [25], which was the state of the art at the time of experimentation. However, more recent methods such as TESPEC [47] have since become available. Their code was released too late to include a full comparison, but future work should incorporate such baselines for a more complete evaluation.

F.1. Proposed Future Experiments

Based on our insights, several experimental directions could further enhance SPICE and deepen understanding of its behavior:

- **Contrastive temperature schedule.** The current implementation uses a fixed temperature for the contrastive

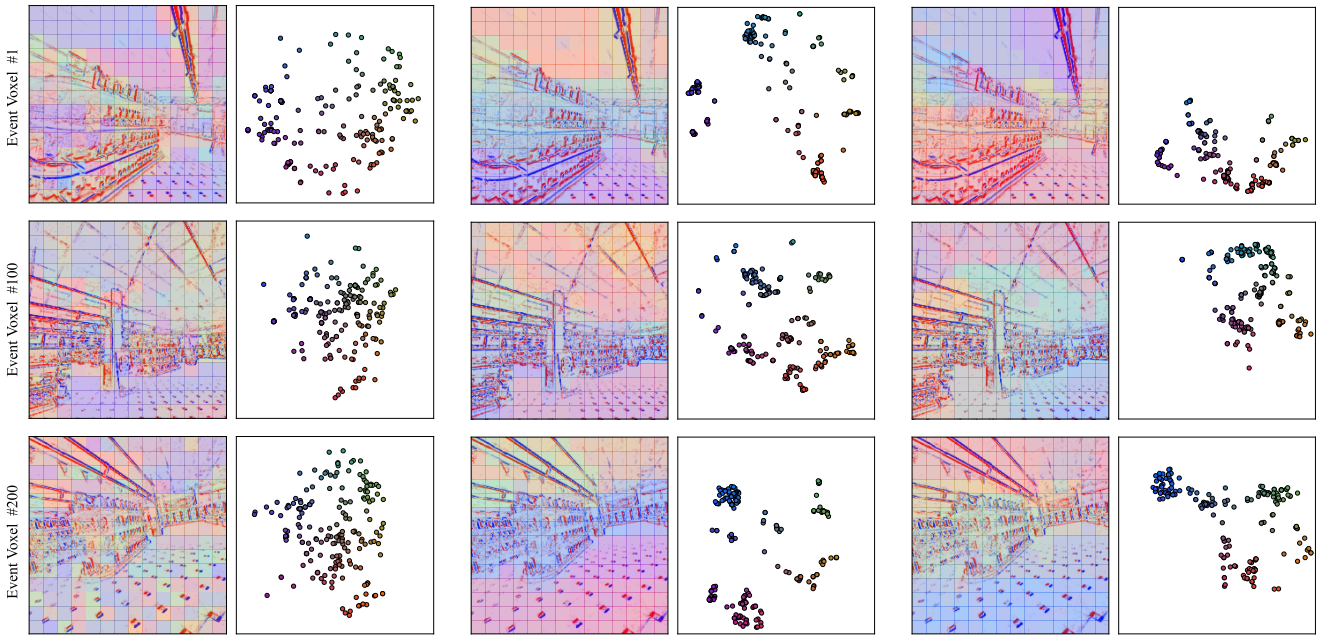


(a) Random initialization, t-SNE. (b) SPICE pretrained, t-SNE. (c) Random initialization, UMAP. (d) SPICE pretrained, UMAP.

Figure 17. **t-SNE [61] and UMAP [45] projections of patch embeddings for a randomly initialized and a SPICE pretrained model.** Colors are matched between projected points and their corresponding patches in the event frame. The pretrained model exhibits more compact and temporally varying embeddings.

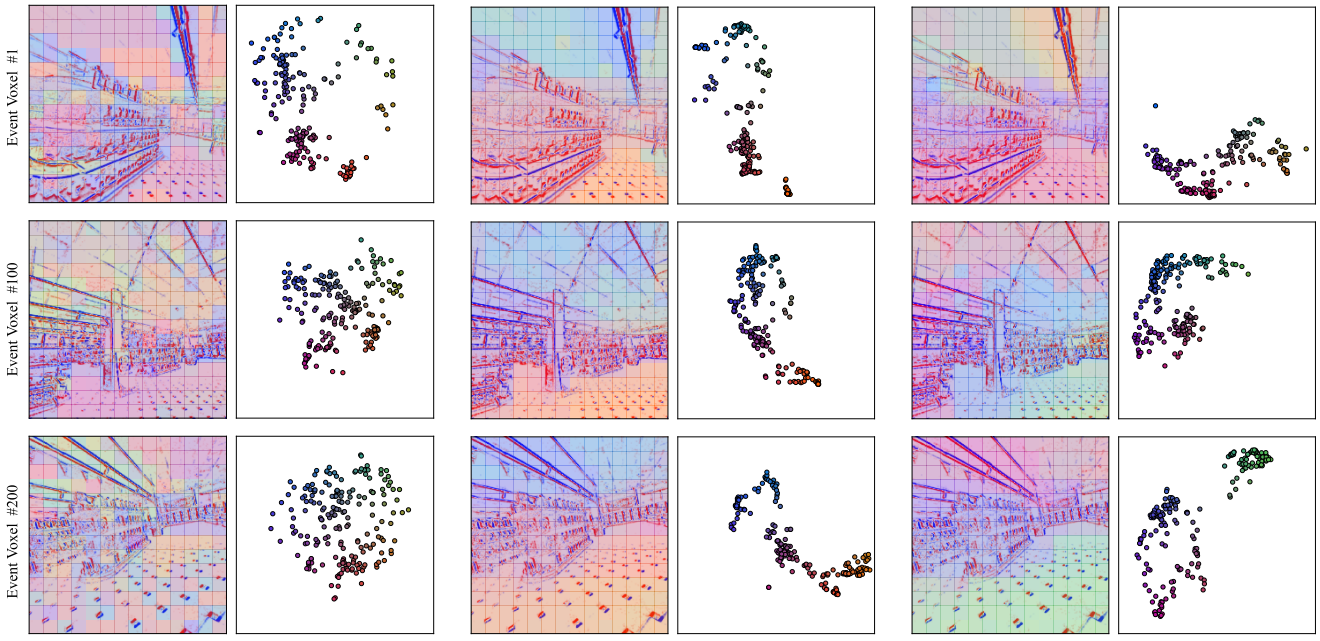
loss. A dynamic schedule, starting with a higher value and gradually decreasing it, could improve convergence by initially allowing smoother distributions before sharpening.

- **Teacher momentum schedule.** Similar to BYOL-style frameworks, currently, the teacher momentum is constant. A progressive schedule, beginning with lower momentum and increasing it as training stabilizes, may yield more stable teacher representations.
- **Loss parameter tuning.** The contrastive loss includes several tunable hyperparameters, such as the importance of different types of negatives, Gaussian mask sigmas, and the earlier-mentioned temperature. Systematic exploration of these could uncover better trade-offs.
- **Prediction setup.** Future experiments should explore alternative past–future voxel configurations (beyond the current "5pred3") and systematically compare parallel versus sequential prediction. In addition, a curriculum strategy, first predicting fewer future steps from more past steps (e.g., "6pred2"), then progressively increasing difficulty ("5pred3", "4pred4", etc.), could help stabilize training and encourage stronger temporal reasoning.
- **Recurrency setup.** Currently, a ConvGRU is applied only to the output of the FPN network. A different setup, more similar to TESPEC, would be to apply a ConvGRU on the output of every stage from the Swin-T backbone, and discard the FPN network altogether.
- **Augmentation strategy.** DPC advocates frame-wise augmentations to reduce the risk of the model exploiting optical-flow shortcuts. Although our approach uses sequence-wise augmentations and does not appear to implicitly learn optical flow (as reflected by non-superior flow performance), evaluating framewise augmentation schemes remains a relevant direction to improve semantic learning.
- **Cluster-level objective.** ECDDP exhibits temporally stable semantic clusters, likely aided by its cluster-based objective, also indicated by its ablations. Our current loss operates solely at the patch level. Adding a complementary cluster-level objective could enhance semantic grouping and promote more consistent long-range structure.



(a) Random initialization. (b) ECDDP pretrained. (c) SPICE pretrained.

Figure 18. **t-SNE [61] projection of Swin-T/7 stage-3 patch embeddings.** Colors are matched between projected points and their corresponding patches in the event frame. Both ECDDP and SPICE exhibit more coherent semantic structure than random initialization, with ECDDP showing notably higher temporal consistency in cluster locations.



(a) Random initialization. (b) ECDDP pretrained. (c) SPICE pretrained.

Figure 19. **UMAP [45] projection of Swin-T/7 stage-3 patch embeddings.** Colors are matched between projected points and their corresponding patches in the event frame. Both ECDDP and SPICE exhibit more coherent semantic structure than random initialization, with ECDDP showing notably higher temporal consistency in cluster locations.

F.2. Proposed Future Ablations

While this work already includes ablations on the main components of SPICE, several additional studies could provide further insights and possibly improve performance:

- **Event representation.** Because of some defects, we proposed to use a slightly different event representation. Though initial experiments showed that our event representation was superior, we have not done a direct one-to-one comparison of the different representations under the exact same settings. As such, we propose it as a future ablation.
- **Loss composition.** The current loss includes only a contrastive alignment objective. An additional MSE term has been implemented, weighted by a parameter α, to combine the contrastive loss with an L2 alignment. Evaluating this formulation could clarify whether mixing objectives improves representation quality. Early exploratory tests suggested that adding an L2 term did not help, so we did not pursue it further, but given the many changes to the model since then, revisiting this variant may be informative.
- **Input normalization.** The event representations are presently used in raw form. Other works (e.g., TESPEC [47]) report gains when normalizing input representations, suggesting this as a promising modification to test.
- **Similarity metric.** The loss currently employs a dot-product similarity following DPC [25]. Replacing it with cosine similarity could be explored; preliminary tests suggested the dot product may perform better, but a full evaluation on the final model remains open.
- **Batch size and negatives.** Earlier ablations indicated that batch negatives contribute positively but are not critical. Systematically varying the batch size would clarify how the number of negatives interacts with our normalized version of the contrastive loss.