This paper has been accepted for publication at the
IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, 2025. ©IEEE

GG-SSMs: Graph-Generating State Space Models

Nikola Zubić and Davide Scaramuzza
Robotics and Perception Group, University of Zurich, Switzerland

Abstract

State Space Models (SSMs) are powerful tools for modeling sequential data in computer vision and time series analysis domains. However, traditional SSMs are limited by fixed, one-dimensional sequential processing, which restricts their ability to model non-local interactions in high-dimensional data. While methods like Mamba and VMamba introduce selective and flexible scanning strategies, they rely on predetermined paths, which fails to efficiently capture complex dependencies. We introduce Graph-Generating State Space Models (GG-SSMs), a novel framework that overcomes these limitations by dynamically constructing graphs based on feature relationships. Using Chazelle's Minimum Spanning Tree algorithm, GG-SSMs adapt to the inherent data structure, enabling robust feature propagation across dynamically generated graphs and efficiently modeling complex dependencies. We validate GG-SSMs on 11 diverse datasets, including event-based eye-tracking, ImageNet classification, optical flow estimation, and six time series datasets. GG-SSMs achieve state-of-the-art performance across all tasks, surpassing existing methods by significant margins. Specifically, GG-SSM attains a top-1 accuracy of 84.9% on ImageNet, outperforming prior SSMs by 1%, reducing the KITTI-15 error rate to 2.77%, and improving eye-tracking detection rates by up to 0.33% with fewer parameters. These results demonstrate that dynamic scanning based on feature relationships significantly improves SSMs' representational power and efficiency, offering a versatile tool for various applications in computer vision and beyond.

Multimedial Material: For GitHub code, poster and other details visit https://github.com/uzh-rpg/gg_ssms.

1. Introduction

Modeling complex, long-range dependencies in high-dimensional data is a fundamental challenge in computer vision and time series analysis. Accurately capturing these dependencies is crucial for understanding intricate structures and relationships within data, directly impacting the performance of machine learning models in real-world applications.

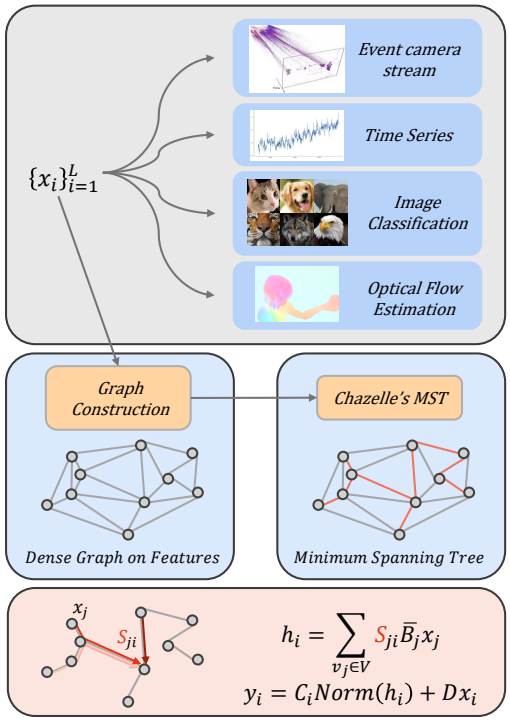

Figure 1. **Illustration of the Graph-Generating State Space Model (GG-SSM).** Given an input feature set $\{\mathbf{x}_i\}_{i=1}^{L}$, we construct a graph based on feature dissimilarities and apply an efficient algorithm to generate a minimum spanning tree $\mathcal{T}$. SSM state propagation is then performed along this tree to obtain improved feature representations.

Traditional models like Convolutional Neural Networks (CNNs) [22] excel at capturing local patterns but struggle with global context due to their limited receptive fields. Transformers [51] address this limitation by employing self-attention mechanisms to model global relationships; however, they suffer from quadratic computational com-

plexity with respect to input size, making them less practical for high-resolution images or long sequences.

State Space Models (SSMs) have emerged as efficient alternatives for sequential data modeling, demonstrating the ability to capture long-range dependencies with linear computational complexity concerning sequence length [20, 21]. SSMs process sequences using state transitions, making them suitable for tasks requiring memory of previous inputs. However, traditional SSMs are constrained by fixed, one-dimensional sequential processing, which limits their ability to model non-local interactions inherent in high-dimensional visual data [36]. This limitation hampers their effectiveness in capturing complex spatial dependencies critical for comprehensive visual understanding.

Recent methods like Mamba [19] and VMamba [30] have attempted to overcome these constraints by introducing selective and flexible scanning strategies. Mamba [19] introduces a selective mechanism to improve context awareness, while VMamba [30] extends this approach with more scanning paths to achieve global context. Despite these advancements, they rely on predetermined 1D scanning trajectories unrolled in a few directions (e.g., on an image grid) and struggle to adapt to the diverse and complex structures found in data. Consequently, *they fail to efficiently capture intricate dependencies and long-range interactions not aligned with the predefined paths.*

The core challenge remains: designing models that can adaptively and efficiently capture complex, non-local dependencies in high-dimensional data without incurring prohibitive computational costs. Fixed scanning strategies cannot accommodate the diverse structures in data. While graph-based models naturally represent complex relationships [27], they often face high computational complexity when constructing and processing large graphs.

This paper introduces **Graph-Generating State Space Models (GG-SSMs)**, a novel framework that overcomes these limitations by dynamically constructing graphs based on feature relationships within the data. By utilizing Chazelle's Minimum Spanning Tree (MST) algorithm [6], which operates with near-linear time complexity, GG-SSMs adapt to the inherent data structure, enabling robust feature propagation across dynamically generated graphs. This approach efficiently models complex, long-range dependencies by moving beyond fixed scanning paths and capturing intrinsic relationships between features.

The key idea behind GG-SSMs is integrating dynamic graph construction into the state space modeling framework, allowing the model to naturally adapt its processing pathways based on the data's structure. GG-SSMs efficiently identify the most significant connections among data elements by constructing an MST over the data features. This dynamic graph is the backbone for state propagation, enabling the model to capture long-range dependencies and complex interactions without significant computational overhead.

Our contributions can be summarized as follows:

1. **Dynamic Graph-Based State Space Modeling**: We propose a method for integrating dynamic graph structures into SSMs, enabling the capture of complex spatial and temporal dependencies in high-dimensional data.

2. **Efficient Computation with MSTs**: Leveraging Chazelle's MST algorithm [6], GG-SSMs construct optimal graphs with minimal computational overhead, ensuring scalability to large datasets and high-resolution inputs.

3. **State-of-the-Art Performance**: GG-SSMs consistently outperform existing methods, including Mamba [19] and VMamba [30], across multiple benchmarks, achieving higher accuracy with fewer parameters and lower computational costs.

We validate GG-SSMs on 11 diverse datasets, covering various computer vision and time series tasks. Specifically:

- On the **ImageNet** classification benchmark [10], GG-SSM achieves a top-1 accuracy of **84.9%**, outperforming prior SSM-based models by **1%**.

- In **optical flow estimation** on the KITTI-15 dataset [18], GG-SSM reduces the error rate to **2.77%**, the lowest reported to date.

- For **event-based eye-tracking** datasets (INI-30 [3] and Event-based LPW [7]), GG-SSM improves detection rates by up to **0.33%** while using fewer parameters.

- Across six renowned **time series datasets**, GG-SSMs achieve superior forecasting accuracy, demonstrating versatility beyond traditional vision tasks.

These results demonstrate that dynamic scanning based on feature relationships significantly improves SSMs' representational power and efficiency, offering a versatile tool for various applications in computer vision and beyond.

2. Related Work

2.1. Challenges with Sequential-Only SSMs

Data naturally exhibit multi-dimensional structures with complex spatial or spatiotemporal relationships in many computer vision applications and event-based data processing [66, 67]. Traditional State Space Models (SSMs) process signals in a strictly one-dimensional sequence, which makes them ill-suited to capture non-local interactions. For instance, images and event streams include spatial dependencies that span two or more dimensions, but standard SSMs restrict the flow of information along predetermined scanning paths. While these 1D-sequential SSMs excel in tasks such as NLP, their inability to adapt to more complex data topologies leads to poor performance on high-dimensional or multi-variate inputs. Such limitations are particularly pronounced in tasks that demand a global understanding of spatial patterns, such as image classification,

optical flow, and dynamic vision sensing.

2.2. Approaches for High-Dimensional Visual SSMs

Researchers have proposed specialized architectures for multi-dimensional data to address the shortcomings of sequential-only processing. S4ND [36] pioneered a multi-dimensional approach by stacking independent 1D SSM modules, enabling limited handling of spatial axes. Baron et al. [2] introduced a discrete multi-axial framework (A2S) incorporating a 2D-SSM spatial layer to better capture spatial dependencies. More recent advances explore sophisticated scanning and correlation techniques to improve spatial consistency, such as bidirectional [65], four-way [30], continuous [56], zigzag [23], window-based [24], and topology-based [38, 54] processing. Among these, Mamba [19] and VMamba [30] propose flexible paths or multi-directional scanning to escape strictly linear reading orders. However, such approaches still rely on fixed or predefined trajectories, limiting their adaptability to the diverse and irregular structures inherent to visual data. Additionally, even approaches that consider specialized topologies tend to become computationally heavy and less scalable.

Our framework, **Graph-Generating State Space Models (GG-SSMs)**, tackles these issues by constructing *dynamic* graph structures. Rather than imposing fixed scanning paths, GG-SSMs leverage graph edges derived from feature similarities or dissimilarities to capture long-range interactions *on the fly*. Moreover, an efficient MST-based construction keeps the graph sparse, ensuring scalability and improving the representational power of state propagation. Our experiments show that this improves various vision tasks and time series analyses.

3. Method

In this section, we introduce the *Graph-Generating State Space Models* (GG-SSMs), a novel framework that improves traditional SSMs by dynamically constructing graph topologies to capture complex, long-range dependencies in sequential and high-dimensional data. We begin by revisiting the limitations of conventional SSMs (Sec 3.1) and then detail our proposed method (Sec 3.2), which integrates efficient graph construction algorithms into the SSM framework. Also, we analyze its computational complexity (Sec 3.3).

3.1. Revisiting State Space Models

SSMs are powerful mathematical models for sequential data processing, defined by the evolution of hidden states over time [26, 69]. The discrete-time linear time-invariant SSM is typically formulated as:

$$\begin{aligned} \mathbf{h}[n] &= \bar{\mathbf{A}}\mathbf{h}[n-1] + \bar{\mathbf{B}}\mathbf{x}[n], \\ \mathbf{y}[n] &= \bar{\mathbf{C}}\mathbf{h}[n] + \bar{\mathbf{D}}\mathbf{x}[n], \end{aligned} \tag{1}$$

where $\mathbf{h}[n] \in \mathbb{R}^N$ is the hidden state at time step n, $\mathbf{x}[n] \in \mathbb{R}^D$ is the input, and $\mathbf{y}[n] \in \mathbb{R}^D$ is the output. The matrices $\bar{\mathbf{A}} \in \mathbb{R}^{N\times N}$, $\bar{\mathbf{B}} \in \mathbb{R}^{N\times D}$, $\bar{\mathbf{C}} \in \mathbb{R}^{D\times N}$, and $\bar{\mathbf{D}} \in \mathbb{R}^{D\times D}$ are the model parameters.

While effective for certain applications, traditional SSMs process inputs in a strictly 1D-sequential order [21, 44, 45], limiting their ability to capture complex, non-local dependencies inherent in high-dimensional data such as images [36], language sequences [16, 47], or event-based signals [68]. Handcrafted scanning strategies [24, 30, 56] attempt to address this but often fail to preserve the structural information adequately.

3.2. Graph-Generating State Space Models (GG-SSMs)

To overcome these limitations, we propose GG-SSMs, which dynamically construct graphs based on input data and perform state propagation along these graphs. Therefore, the scanning is not handcrafted, and we believe it represents the best solution to scan with Visual SSMs. Moreover, this approach allows the model to capture long-range dependencies and complex interactions beyond the capabilities of purely 1D-sequential architectures.

3.2.1. Graph Construction

Given an input feature set $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{L}$, where L is the number of elements (e.g., pixels in an image or tokens in a sequence), our goal is to construct a graph that captures the most significant relationships among these elements.

We define a fully connected undirected graph $G = (V, E)$, where each node $v_i \in V$ corresponds to a feature $\mathbf{x}_i$. The edge weight w_{ij} between nodes v_i and v_j is calculated based on a dissimilarity measure $d(\mathbf{x}_i, \mathbf{x}_j)$, such as cosine dissimilarity:

$$w_{ij} = \exp\left(-\frac{\mathbf{x}_i^\top \mathbf{x}_j}{\|\mathbf{x}_i\|\|\mathbf{x}_j\|}\right). \tag{2}$$

To efficiently capture the essential structure of the data without the computational burden of processing a fully connected graph, we construct a minimum spanning tree (MST), denoted as $\mathcal{T} = (V, E_T)$, where $E_T \subseteq E$. The MST retains the most critical edges that connect all nodes with minimal total weight, effectively capturing the core relationships in the data. So, the vertices V represent the pixel or token embeddings, and the weights of the edges E are the dissimilarities between the embeddings that we calculate with the cosine distance metric.

Given such a dense graph, we employ an efficient MST algorithm from Bernard Chazelle [6] with a time complexity of $\mathcal{O}(E\alpha(E, V))$, where α is the inverse Ackermann function, which grows extremely slowly and can be considered nearly constant for all practical purposes. The MST construction operates effectively in linear time for sparse

graphs where $E = \mathcal{O}(L)$. This allows us to get a sparse graph where the edge weights are most similar out of all possible sparse graphs.

3.2.2. State Propagation Along the Graph

With the MST $\mathcal{T}$ constructed, we perform state propagation along its edges to capture long-range dependencies.

Path Weight Computation To aggregate information from all nodes in the tree and express the hidden state $\mathbf{h}_i$ at node v_i in terms of the inputs $\mathbf{x}_j$ at all nodes v_j, we define a *path weight* S_{ji} that quantifies the cumulative effect of the state transitions along the path from node v_j to node v_i. The MST $\mathcal{T}$ has a unique path between any two nodes. Let's denote P_{ji} as the ordered sequence of nodes along the path from node v_j to node v_i. This path consists of nodes $v_{k_1}, v_{k_2}, \ldots, v_{k_n}$, where each v_{k_m} is connected to $v_{k_{m-1}}$ and $v_{k_{m+1}}$ (with $v_{k_0} = v_j$ and $v_{k_{n+1}} = v_i$).

The path weight S_{ji} is then defined as the product of the state transition matrices $\bar{\mathbf{A}}_{k_m}$ associated with the nodes v_{k_m} along the path P_{ji}:

$$S_{ji} = \prod_{m=1}^{n} \bar{\mathbf{A}}_{k_m}, \tag{3}$$

where k_m indexes the nodes along the path from v_j to v_i and $\bar{\mathbf{A}}_{k_m}$ is the state transition matrix associated with node v_{k_m}. The product is ordered from v_j to v_i. Using this path weight, we can express the hidden state $\mathbf{h}_i$ at node v_i as an aggregated sum of contributions from all nodes $v_j \in V$:

$$\mathbf{h}_i = \sum_{v_j \in V} S_{ji} \bar{\mathbf{B}}_j \mathbf{x}_j. \tag{4}$$

In this formulation, the term $\bar{\mathbf{B}}_j \mathbf{x}_j$ represents the initial contribution from node v_j. The path weight S_{ji} modulates this contribution as it propagates along the path from v_j to v_i through the state transition matrices $\bar{\mathbf{A}}_{k_m}$. Finally, for each feature in the sequence, y_i can be formulated as:

$$y_i = C_i \,\mathrm{Norm}(h_i) + D\, x_i, \tag{5}$$

where y_i, h_i, and x_i represent individual elements in the sequences $\{y_i\}_{i=1}^{L}$, $\{h_i\}_{i=1}^{L}$, and $\{x_i\}_{i=1}^{L}$, respectively.

3.3. Computational Complexity of GG-SSMs

The GG-SSM achieves an efficient computational complexity of $\mathcal{O}(L)$, where L is the number of nodes in the graph, by structuring computations around an MST. Constructed with Chazelle's algorithm [6], this MST guarantees exactly $L-1$ edges, ensuring sparsity and a unique path between any two nodes, which prevents redundant calculations. GG-SSM propagates state information from the leaves to the root in the forward pass. Each node performs fixed operations: initializing its state based on the input, aggregating states from child nodes, and updating its hidden state. By processing each node and edge only once through either breadth-first or depth-first traversal, the forward pass efficiently captures hierarchical dependencies with a total complexity of $\mathcal{O}(L)$.

In the backward pass, gradients are propagated from the root back down to the leaves, where each node performs constant-time operations to compute and aggregate gradients with respect to both inputs and parameters. Intermediate results, such as partial states and gradients, are stored through dynamic programming, avoiding recalculations and resolving each dependency exactly once. This storage and reuse of intermediate values maintain the linear complexity. Additionally, the model relies exclusively on local information from each node's immediate neighbors, eliminating the need for global operations. Since MST construction itself runs in $\mathcal{O}(L)$ for sparse graphs due to the nearly constant inverse Ackermann function α, GG-SSM's overall complexity remains $\mathcal{O}(L)$, making it scalable and computationally efficient even for large datasets with complex dependency structures. Generation of MST is implemented in CUDA and is very fast, and one forward pass is in time approximately the same as the Mamba's forward pass [19].

4. Experiments

4.1. Event-Based Eye Tracking

LPW Dataset The LPW event-based dataset [7] contains 66 high-quality, 20-second videos of eye regions, stored as event sequences in `.h5` files. Each event is represented as $e_i = (x_i, y_i, t_i, p_i)$, where (x_i, y_i) denotes pixel location, t_i is the timestamp, and $p_i \in \{+1, -1\}$ represents the brightness change polarity. To aggregate events into frames $V(x, y)$ per pixel, we use a constant time-bin count with a 4.4 ms window ΔT [17], aligning with the frame rate of the source RGB dataset [49] for precise synchronization with ground truth labels. The original frames at 640×480 are resized to the DAVIS240 resolution of 240×180 [4] and further downsampled to 80×60, yielding a synthetic dataset with 11k event-based frames across 22 videos to reduce computational load.

While previous approaches leveraged deep CNNs [14, 39] for pupil detection in RGB datasets, the sparsity of event-based frames presents challenges for traditional per-frame predictions. We address this issue using GG-SSMs, which capture both spatial and temporal dependencies within event-based data. Unlike standard LSTM methods [7], GG-SSMs leverage graph-based spatial processing alongside where the same GG-SSM block is used to model temporal features, achieving a balance of high performance with minimal parameters and FLOPs, as shown in Table 1.

The GG-SSM efficiently models the complex relationships among events by dynamically constructing graphs on images (in the spatial domain) and aggregating these fea-

Table 1. Comparison of detection rates (%) for pupil tracking on the event-based LPW dataset. The best results are in **bold**.

Model	Params (M)	FLOPs (G)	p_3	p_5	p_{10}
CNN [14]	0.40	18.4	57.80	77.40	91.40
ConvLSTM [41]	0.42	42.61	88.70	97.10	99.40
CB-ConvLSTM [7]	0.42	9.00	88.50	96.70	99.20
TemporalResNet [15]	0.28	11.0	84.10	93.50	98.20
VMamba+Mamba [19, 30]	0.35	8.60	89.00	98.00	99.30
GG-SSM (Ours)	**0.22**	**8.01**	**89.33**	**98.89**	**99.50**

tures in the temporal domain. So, GG-SSM can serve as an efficient processor of features not only for spatial (image-like) data but also for temporal data. By propagating information along the MST, the model effectively captures long-range dependencies, improving prediction accuracy even in sparse frames. The accuracy of pupil detection is evaluated based on the detection rate within p pixels (p_3, p_5, p_{10}), representing Euclidean distances of 3, 5, and 10 pixels between the predicted and ground truth pupil centers. We compare our model with baseline models, including VMamba+Mamba [30]. VMamba is used for spatial (image) processing, and Mamba for temporal processing. GG-SSM achieves superior detection rates of **89.33%**, **98.89%**, and **99.50%** for pixel distances p_3, p_5, and p_{10} from the ground truth, respectively. With 0.22 million parameters and 8.01 GFLOPs, GG-SSM surpasses VMamba+Mamba detection rates while being more energy-efficient. GG-SSM also outperforms Change-Based ConvLSTM (CB-ConvLSTM) [7], which applies change-based convolutions to address sparsity, as well as other baseline models such as ConvLSTM [41] and CNNs [14].

INI-30 dataset Unlike previous datasets [1, 63] that focus on gaze tracking with fixed head positions and lower-resolution sensors, Ini-30 [3] provides high-resolution event data captured in unconstrained, "in-the-wild" settings. The dataset was collected using two DVXplorer event cameras (640 $\times$ 480 pixels) mounted on a glasses frame, one for each eye, allowing natural head and eye movements without restrictive setups. The dataset comprises 30 recordings, each containing variable durations ranging from 14.64 to 193.8 seconds and labels per recording ranging from 475 to 1.848. This variability introduces diverse challenges regarding event density and temporal dynamics, making Ini-30 a comprehensive benchmark for event-based eye tracking.

We evaluated GG-SSM on Ini-30 and compared its performance against the 3ET model [7], Retina [3], and VMamba+Mamba [19, 30] using Centroid Error, which measures the Euclidean distance between predicted and true pupil centers. As shown in Table 2, GG-SSM achieves the lowest centroid error on Ini-30, outperforming all other models and highlighting its robustness in real-world scenarios. On the Synthetic LPW dataset, based on RGB-camera data [49], GG-SSM again delivers the lowest error, demonstrating its adaptability across different data sources.

Table 2. Centroid error comparison on the validation set. Lower values indicate better accuracy.

Dataset	3ET [7]	Retina [3]	VMamba+Mamba	GG-SSM (Ours)
Ini-30	4.48 ($\pm$ 1.94)	3.24 ($\pm$ 0.79)	3.42 ($\pm$ 0.89)	**3.11 ($\pm$ 0.80)**
Synthetic LPW	5.33 ($\pm$ 1.59)	6.46 ($\pm$ 2.49)	6.58 ($\pm$ 2.33)	**5.11 ($\pm$ 0.98)**

Table 3. Comparison of model complexity. Lower values indicate more efficient models.

Method	MAC Operations	Parameters
3ET [7]	107M	418k
Retina [3]	3.03M	63k
VMamba+Mamba [19, 30]	4.52M	92k
GG-SSM (Ours)	**3.01M**	**62k**

Table 3 provides a comparison of model complexity in terms of Multiply-Accumulate Operations (MACs) and parameter counts. GG-SSM achieves significantly reduced complexity compared to 3ET [7], with only 3.01M MACs and 62k parameters, making it highly efficient for real-time applications. Importantly, GG-SSM also surpasses VMamba+Mamba [19, 30], our best SSM competitor, in both accuracy and efficiency.

4.2. Time Series

In this section, we evaluate the performance of our GG-SSM on six real-world time series forecasting datasets, comparing it with state-of-the-art models. The datasets used in our experiments are:

- **Exchange**: Daily exchange rates of eight countries from 1990 to 2016.
- **Weather**: Meteorological data collected every 10 minutes from the Weather Station of the Max Planck Biogeochemistry Institute in 2020, containing 21 indicators.
- **Solar-Energy**: Solar power records from 137 photovoltaic plants in Alabama in 2006, sampled every 10 minutes.
- **ETTh2**: Electricity Transformer Temperature dataset, including load and oil temperature data collected hourly over two years.
- **Traffic**: Hourly road occupancy rates from 862 sensors in San Francisco Bay area freeways from January 2015 to December 2016.
- **ETTm2**: Another subset of the Electricity Transformer Temperature dataset, collected every 15 minutes.

We compare GG-SSM with nine representative and state-of-the-art forecasting models. The models include **S-Mamba** [52], which is an SSM-based model that integrates the Mamba Variate-Correlated Fusion Layer, and **iTransformer** [29], a transformer-based model that initially an-

alyzes time series data for each variate individually and then integrates across variates. **RLinear** [28] is a linear model utilizing reversible normalization and channel independence, while **PatchTST** [37] segments time series into patches for input tokens, using channel-independent embeddings for efficient representation learning. **Crossformer** [62] employs a cross-attention mechanism to capture long-term dependencies. The **TiDE** [9] model is an encoder-decoder structure based on multi-layer perceptrons (MLPs), and **TimesNet** [53] leverages TimesBlock to transform 1D time series into 2D tensors, capturing both intra-period and inter-period variations. **DLinear** [58] is a simple one-layer linear model with a decomposition architecture, and **FEDformer** [59] is a frequency-enhanced transformer that utilizes sparse representations in bases such as the Fourier transform.

We conduct experiments with the lookback length $L = 96$ and forecast lengths $T = 96, 192, 336$, and 720. The evaluation metrics are Mean Squared Error (MSE) and Mean Absolute Error (MAE). The results are presented in Table 4. The best results are highlighted in **bold**, and the second best are underlined.

4.2.1. Analysis of Results

Our GG-SSM consistently achieves the best accuracy, as can be seen from the comprehensive results presented in Table 4. These datasets are characterized by many variates exhibiting strong inter-variable correlations and pronounced periodic patterns. By modeling the intrinsic relationships among variates, GG-SSM improves representation learning and forecasting accuracy, demonstrating its superiority in handling datasets with rich inter-variable interactions.

On the **ETTh2** and **ETTm2** datasets, which contain fewer variates and exhibit weaker periodicity, GG-SSM outperforms all the other models. This showcases the robustness of GG-SSM in handling datasets where variate correlations are less pronounced. The model's ability to dynamically adjust the graph topology allows it to adapt to varying data characteristics without significant loss in forecasting accuracy.

Compared to the previous best SSM-based model, S-Mamba [52], GG-SSM demonstrates consistent improvements across all datasets and forecast horizons. While S-Mamba [52] employs the Mamba [19] Variate-Correlated Fusion Layer to capture inter-variate relationships, GG-SSM further augments this capability through graph-based state propagation. Additionally, GG-SSM outperforms other transformer-based models such as iTransformer [29], PatchTST [37], and Crossformer [62], highlighting the benefits of incorporating graph structures within SSMs. The superior performance of GG-SSM underscores the effectiveness of its design in capturing complex dependencies inherent in multivariate time series data.

4.3. Object Classification

In this subsection, we evaluate the performance of our proposed GG-SSM on the ImageNet-1K [10] dataset for object classification. We adopt the training protocols from VMamba [30], ensuring a fair comparison with existing state-of-the-art models, especially SSM-based.

Training Setup We closely follow the hyperparameter settings and experimental configurations of VMamba [30] to train our GG-SSM models. Specifically, only for this task, we employ the H200 GPU for training, which involves training for 200 epochs with a cosine learning rate decay. The models are optimized using the AdamW optimizer [33] with an initial learning rate of 1×10^{-3} and a weight decay of 0.05. We utilize data augmentation techniques such as RandAugment [8], MixUp [60], and CutMix [57] to improve generalization and prevent overfitting.

Results Our GG-SSM models outperform previous architectures across all scales. At the **Tiny** scale, GG-SSM-T achieves a top-1 accuracy of **83.6%**, surpassing VMamba-T by **1%**, Swin-T by **2.3%**, and DeiT-S by a significant **3.8%**. This demonstrates the efficacy of our graph-based approach in capturing intricate spatial dependencies within images. At the **Small** and **Base** scales, GG-SSM-S and GG-SSM-B achieve top-1 accuracies of **84.4%** and **84.9%**, respectively, scoring as best and second best model. Notably, GG-SSM-B outperforms VMamba-B by **1%** and Swin-B [31] by **1.4%**, highlighting the scalability of our approach. Compared to SSM-based models like S4ND-Conv-T [36] and Vim-S [65], GG-SSM demonstrates a clear performance advantage. At the Tiny scale, GG-SSM-T outperforms S4ND-Conv-T by **1.4%** in top-1 accuracy while having a smaller parameter count and fewer FLOPs. This underscores the effectiveness of our graph-based state space modeling over conventional SSM approaches. Furthermore, when compared to transformer-based models, GG-SSM consistently achieves higher accuracy with similar or fewer computational resources. This highlights the potential of integrating graph structures within SSMs to capture complex dependencies in visual data more effectively than traditional self-attention mechanisms.

Computational Efficiency GG-SSM maintains competitive computational efficiency. Despite achieving higher accuracy, GG-SSM-T requires fewer FLOPs (4.4G) than VMamba-T (4.9G) and fewer parameters. Similarly, GG-SSM-S and GG-SSM-B have reduced FLOPs and parameter counts relative to their VMamba counterparts. This efficiency is attributed to our model's linear computational complexity, resulting from using Chazelle's MST algorithm

Table 4. Forecasting results on six datasets. The best results are in **bold**, and the second best are underlined.

Dataset	Horizon	GG-SSM		S-Mamba		iTransformer		RLinear		PatchTST		Crossformer		TiDE		TimesNet		DLinear	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Exchange	96	0.0878	0.2073	**0.0860**	0.2070	**0.0860**	0.2060	0.0930	0.2170	0.0880	**0.2050**	0.2560	0.3670	0.0940	0.2180	0.1070	0.2340	0.0880	0.2180
	192	0.1813	0.3029	0.1770	**0.2990**	0.1770	**0.2990**	0.1840	0.3070	**0.1760**	**0.2990**	0.4700	0.5090	0.1840	0.3070	0.2260	0.3440	**0.1760**	0.3150
	336	0.3318	0.4172	0.3320	0.4180	0.3310	0.4170	0.3510	0.4320	**0.3010**	**0.3970**	1.2680	0.8830	0.3490	0.4310	0.3670	0.4480	0.3130	0.4270
	720	0.8518	0.6950	0.8470	**0.6910**	0.8470	**0.6910**	0.8860	0.7140	0.9010	0.7140	1.7670	1.0680	0.8520	0.6980	0.9640	0.7460	**0.8390**	0.6950
	Avg	0.3632	0.4056	0.3600	**0.4030**	0.3600	**0.4030**	0.3780	0.4170	0.3670	0.4040	0.9400	0.7070	0.3700	0.4130	0.4160	0.4430	**0.3540**	0.4140
Weather	96	**0.1473**	**0.1956**	0.1650	0.2100	0.1740	0.2140	0.1920	0.2320	0.1770	0.2180	0.1580	0.2300	0.2020	0.2610	0.1720	0.2200	0.1960	0.2550
	192	**0.1918**	**0.2405**	0.2140	0.2520	0.2210	0.2540	0.2400	0.2710	0.2250	0.2590	0.2060	0.2770	0.2420	0.2980	0.2190	0.2610	0.2370	0.2960
	336	**0.2458**	**0.2815**	0.2740	0.2970	0.2780	0.2960	0.2920	0.3070	0.2780	0.2970	0.2720	0.3350	0.2870	0.3350	0.2800	0.3060	0.2830	0.3350
	720	**0.3151**	**0.3315**	0.3500	0.3450	0.3580	0.3470	0.3640	0.3530	0.3540	0.3480	0.3980	0.4180	0.3510	0.3860	0.3650	0.3590	0.3450	0.3810
	Avg	**0.2250**	**0.2623**	0.2510	0.2760	0.2580	0.2780	0.2720	0.2910	0.2590	0.2810	0.2590	0.3150	0.2710	0.3200	0.2590	0.2870	0.2650	0.3170
Solar-Energy	96	**0.1644**	**0.2297**	0.2050	0.2440	0.2030	0.2370	0.3220	0.3390	0.2340	0.2860	0.3100	0.3310	0.3120	0.3990	0.2500	0.2920	0.2900	0.3780
	192	**0.1769**	**0.2396**	0.2370	0.2700	0.2330	0.2610	0.3590	0.3560	0.2670	0.3100	0.7340	0.7250	0.3390	0.4160	0.2960	0.3180	0.3200	0.3980
	336	**0.1911**	**0.2525**	0.2580	0.2880	0.2480	0.2730	0.3970	0.3690	0.2900	0.3150	0.7500	0.7350	0.3680	0.4300	0.3190	0.3300	0.3530	0.4150
	720	**0.2003**	**0.2602**	0.2600	0.2880	0.2490	0.2750	0.3970	0.3560	0.2890	0.3170	0.7690	0.7650	0.3700	0.4250	0.3380	0.3370	0.3560	0.4130
	Avg	**0.1832**	**0.2455**	0.2400	0.2730	0.2330	0.2620	0.3690	0.3560	0.2700	0.3070	0.6410	0.6390	0.3470	0.4170	0.3010	0.3190	0.3300	0.4010
ETTh2	96	**0.2823**	0.3472	0.2960	0.3480	0.2970	0.3490	0.2880	**0.3380**	0.3020	0.3480	0.7450	0.5840	0.4000	0.4400	0.3400	0.3740	0.3330	0.3870
	192	**0.3534**	**0.3982**	0.3760	0.3960	0.3800	0.4000	0.3740	0.3900	0.3880	0.4000	0.8770	0.6560	0.5280	0.5090	0.4020	0.4140	0.4770	0.4760
	336	**0.3722**	**0.4163**	0.4240	0.4310	0.4280	0.4320	0.4150	0.4260	0.4260	0.4330	1.0430	0.7310	0.6430	0.5710	0.4520	0.4520	0.5940	0.5410
	720	**0.4039**	**0.4434**	0.4260	0.4440	0.4270	0.4450	0.4200	0.4400	0.4310	0.4460	1.1040	0.7630	0.8740	0.6790	0.4620	0.4680	0.8310	0.6570
	Avg	**0.3529**	0.4013	0.3810	0.4050	0.3830	0.4070	0.3740	**0.3980**	0.3870	0.4070	0.9420	0.6840	0.6110	0.5500	0.4140	0.4270	0.5590	0.5150
Traffic	96	**0.3486**	**0.2487**	0.3820	0.2610	0.3950	0.2680	0.6490	0.3890	0.4620	0.2950	0.5220	0.2900	0.8050	0.4930	0.5930	0.3210	0.6500	0.3960
	192	**0.3585**	**0.2554**	0.3960	0.2670	0.4170	0.2760	0.6010	0.3660	0.4660	0.2960	0.5300	0.2930	0.7560	0.4740	0.6170	0.3360	0.5980	0.3700
	336	**0.3725**	**0.2626**	0.4170	0.2760	0.4330	0.2830	0.6090	0.3690	0.4820	0.3040	0.5580	0.3050	0.7620	0.4770	0.6290	0.3360	0.6050	0.3730
	720	**0.4167**	**0.2855**	0.4600	0.3000	0.4670	0.3020	0.6470	0.3870	0.5140	0.3220	0.5890	0.3280	0.7190	0.4490	0.6400	0.3500	0.6450	0.3940
	Avg	**0.3741**	**0.2631**	0.4140	0.2760	0.4280	0.2820	0.6260	0.3780	0.4810	0.3040	0.5500	0.3040	0.7600	0.4730	0.6200	0.3360	0.6250	0.3830
ETTm2	96	**0.1725**	0.2633	0.1790	0.2630	0.1800	0.2640	0.1820	0.2650	0.1750	**0.2590**	0.2870	0.3660	0.2070	0.3050	0.1870	0.2670	0.1930	0.2920
	192	**0.2373**	0.3105	0.2500	0.3090	0.2500	0.3090	0.2460	0.3040	0.2410	**0.3020**	0.4140	0.4920	0.2900	0.3640	0.2490	0.3090	0.2840	0.3620
	336	**0.2826**	0.3448	0.3120	0.3490	0.3110	0.3480	0.3070	**0.3420**	0.3050	0.3430	0.5970	0.5420	0.3770	0.4220	0.3210	0.3510	0.3690	0.4270
	720	**0.3596**	**0.3928**	0.4110	0.4060	0.4120	0.4070	0.4070	0.3980	0.4020	0.4000	1.7300	1.0420	0.5580	0.5240	0.4080	0.4030	0.5540	0.5220
	Avg	**0.2630**	0.3278	0.2880	0.3320	0.2880	0.3320	0.2860	0.3270	0.2810	**0.3260**	0.7570	0.6100	0.3580	0.4040	0.2910	0.3330	0.3500	0.4010

Table 5. Comparison of classification accuracy, model size, and computational cost on the ImageNet-1K dataset. All models are evaluated with an input image size of 224×224. The best results are in **bold**, and the second best are underlined.

Model	Params (M)	FLOPs (G)	Top-1 Acc. (%)
Transformer-Based Models			
DeiT-T [50]	22	4.6	79.8
DeiT-B [50]	86	17.5	81.8
Swin-T [31]	28	4.5	81.3
Swin-B [31]	88	15.4	83.5
HiViT-T [61]	**19**	4.6	82.1
HiViT-B [61]	66	15.9	83.8
Convolutional Neural Networks			
ConvNeXt-T [32]	29	4.5	82.1
ConvNeXt-B [32]	89	15.4	83.8
State Space Models (SSM-Based)			
S4ND-Conv-T [36]	30	–	82.2
Vim-T [65]	26	–	80.5
VMamba-T [30]	30	4.9	82.6
GG-SSM-T (Ours)	28	**4.4**	83.6
VMamba-S [30]	50	8.7	83.6
GG-SSM-S (Ours)	49	6.6	84.4
S4ND-ViT-B [36]	89	–	80.4
VMamba-B [30]	89	15.4	83.9
GG-SSM-B (Ours)	87	14.1	**84.9**

for graph construction and localized computations in state propagation.

Discussion The superior performance of GG-SSM is attributed to its ability to dynamically construct graphs that effectively capture the most significant relationships among image patches. By performing state propagation along the MST, our model excels in modeling long-range dependencies and complex spatial interactions, which are crucial for object classification tasks. Moreover, using the MST ensures that the graph remains sparse, facilitating efficient computation without sacrificing the richness of the relational information captured. This allows GG-SSM to surpass the limitations of traditional sequential or grid-based models, providing a more powerful representation of the input data.

4.4. Optical Flow Estimation

Generalization Performance. Following previous works, we first evaluate the generalization performance of **GG-SSM** model on the Sintel [5] and KITTI-15 [18] datasets, after being trained on FlyingChairs [13] and FlyingThings3D [35] datasets. The results are summarized in Table 6. Our GG-SSM achieves state-of-the-art zero-shot performance on both challenging datasets, outperforming existing methods, including those that utilize multi-frame (*MF*) inputs. Specifically, GG-SSM attains an end-point error (EPE) of **0.89** on the Sintel [5] clean pass and **1.90** on the final pass, surpassing previous best results. On KITTI-15 [18], our model achieves an Fl-epe of **3.72** and Fl-all of **13.7**, demonstrating its superior ability to generalize to unseen data.

Finetuning Evaluation. We further assess the performance of GG-SSM after finetuning on the Sintel and KITTI

Table 6. Generalization performance of optical flow estimation on Sintel [5] and KITTI-15 [18] after training on FlyingChairs [13] and FlyingThings3D [35] datasets. *MF* indicates methods using multi-frame inputs for optical flow estimation. The best results are in **bold**, and the second-best results are underlined.

Model	Sintel		KITTI-15	
	Clean	Final	Fl-epe	Fl-all
RAFT [48]	1.43	2.71	5.04	17.4
GMA [25]	1.30	2.74	4.69	17.1
GMFlow [55]	1.08	2.48	7.77	23.4
GMFlowNet [64]	1.14	2.71	4.24	15.4
SKFlow [46]	1.22	2.46	4.27	15.5
MatchFlow [12]	1.03	2.45	4.08	15.6
FlowFormer++ [43]	0.90	2.30	3.93	14.1
TransFlow$^{(MF)}$ [34]	0.93	2.33	3.98	14.4
VideoFlow-BOF$^{(MF)}$ [42]	1.03	2.19	3.96	15.3
VideoFlow-MOF$^{(MF)}$ [42]	1.18	2.56	3.89	14.2
MemFlow $^{(MF)}$ [11]	0.93	2.08	3.88	**13.7**
GG-SSM (Ours)$^{(MF)}$	**0.89**	**1.90**	**3.72**	**13.7**

datasets. The results are presented in Table 7. Our GG-SSM achieves an EPE of **0.97** on Sintel clean pass and **1.58** on final pass, setting new state-of-the-art results. On KITTI-15, GG-SSM attains an Fl-all score of **2.77**, significantly outperforming all previous methods, including those using multi-frame inputs.

Table 7. Optical flow finetuning evaluation on the public benchmarks. *MF* indicates methods using multi-frame inputs for optical flow estimation. * denotes methods using RAFT's multi-frame warm-start strategy on Sintel. The best results are in **bold**, and the second-best results are underlined.

Model	Sintel		KITTI-15
	Clean	Final	Fl-all
RAFT* [48]	1.61	2.86	5.10
GMA* [25]	1.39	2.47	5.15
GMFlow [55]	1.74	2.90	9.32
GMFlowNet [64]	1.39	2.65	4.79
SKFlow* [46]	1.28	2.23	4.84
MatchFlow* [12]	1.16	2.37	4.63
FlowFormer++ [43]	1.07	1.94	4.52
PWC-Fusion$^{(MF)}$ [40]	3.43	4.57	7.17
TransFlow$^{(MF)}$ [34]	1.06	2.08	4.32
VideoFlow-BOF$^{(MF)}$ [42]	1.01	1.71	4.44
VideoFlow-MOF$^{(MF)}$ [42]	0.99	1.65	3.65
VideoFlow-MOF$^{(MF)}$ (online) [42]	-	-	4.08
MemFlow$^{(MF)}$ [11]	1.05	1.91	4.10
GG-SSM (Ours)$^{(MF)}$	**0.97**	**1.58**	**2.77**

Remarkably, GG-SSM achieves the lowest Fl-all score of **2.77** on KITTI-15, outperforming the previous best method VideoFlow-MOF [42], which reports an Fl-all of 3.65. On Sintel, our model sets new records on both clean and final passes. These results highlight the effectiveness of GG-SSM in capturing fine-grained motion details and its robustness across different datasets.

5. Ablation Study

We conducted an ablation to compare different MST algorithms for graph construction: Chazelle's MST [6], Kruskal's, and Prim's. Table 8 summarizes the results on both ImageNet (Tiny scale) and a widely used time-series benchmark (ETTh2).

Table 8. Ablation of MST algorithms. Accuracy/performance is nearly identical, but Chazelle's MST consistently offers lower runtime, especially as the dataset size L increases.

Algorithm	ImageNet (Tiny)		ETTh2 (Horizon=192)	
	Top-1 (%)	Time (s/epoch)	MSE	Time (s/epoch)
Kruskal	83.5	1.00×	0.356	1.00×
Prim	83.4	0.98×	0.358	0.95×
Chazelle	83.6	0.90×	0.353	0.88×

All three algorithms yield comparable model accuracy (within $\pm 0.1\%$) but differ in computational efficiency. In particular, Chazelle's MST runs faster in practice for larger L, matching its near-linear theoretical time complexity.

6. Conclusion

Our GG-SSM framework effectively integrates dynamic graph structures into SSMs, enabling the capture of complex, long-range dependencies that are difficult to model with traditional sequential architectures. Chazelle's MST [6] algorithm ensures computational efficiency, making our approach suitable for large-scale applications in computer vision, event-based processing, and time series analysis. By leveraging the inherent structure in data through graph-based modeling, GG-SSMs open new avenues for advanced representation learning, potentially benefiting a wide range of domains requiring complex interaction modeling.

7. Acknowledgment

This work was supported by the European Research Council (ERC) under grant agreement No. 864042 (AGILEFLIGHT). We also thank the Swiss National Supercomputing Center through the Swiss AI Initiative for granting access to the Alps supercomputer to perform some of the experiments presented in this work.

GG-SSMs: Graph-Generating State Space Models

Supplementary Material

8. Chazelle's MST Algorithm

In this section, we provide a concise overview of *Chazelle's MST algorithm* [6] and why it is instrumental to our proposed Graph-Generating State Space Models (GG-SSMs). Although multiple efficient minimum spanning tree (MST) algorithms exist (e.g., Kruskal's, Prim's, or Borůvka's), Chazelle's algorithm is particularly interesting due to its near-linear time complexity in the general graph setting.

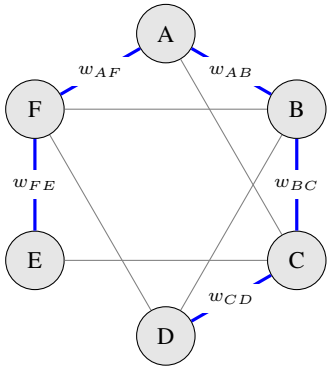

Figure 2. **Chazelle's MST Overview**. Soft heaps allow near-linear sorting of edges. MST edges (in blue) form a spanning structure with no cycles, connecting all vertices using the smallest weights w.

8.1. Core Idea and Time Complexity

Chazelle's MST algorithm belongs to the family of *soft heap* approaches. Its most prominent feature is achieving a runtime of $\mathcal{O}(E\,\alpha(E,V))$, where

- V is the number of vertices in the graph,
- E is the number of edges in the graph,
- $\alpha(\cdot,\cdot)$ is the *inverse Ackermann function*, a function that grows extremely slowly (much more slowly than $\log\log n$).

For any practical input size (e.g., up to millions of edges), $\alpha(E,V)$ remains a small constant (typically ≤ 4). Therefore, the runtime is effectively *linear* for all real-world purposes.

Algorithmic Outline. At a high level, Chazelle's algorithm proceeds by maintaining a specialized priority queue known as a *soft heap* to handle edge weight comparisons and merges. It selectively *corrupts* (or perturbs) a small fraction of keys but guarantees a sufficiently accurate ordering to recover the MST. The soft heap structure allows various operations like insertions, extractions, and merges to be executed in approximately constant amortized time, modulo the very slowly growing α factor.

The algorithm can be outlined in three major steps:

1. **Sort or partition the edges** using the soft-heap structure such that they can be processed in a non-decreasing order of weights.
2. **Merge edge sets** while extracting edge candidates for the MST. These extractions remain almost linear because each edge is either integrated into the MST structure or discarded.
3. **Selective Corruption & Verification**: Since the soft heap may slightly perturb edge weights, a verification step ensures that these corrupted weights do not impact the MST correctness. With high probability, only a small fraction of edges require re-checking.

By the end, the edges forming the MST are collected using a Union-Find data structure (or a similar disjoint set data structure), combining efficiency with theoretical guarantees of correctness.

8.2. Practical Implications for GG-SSMs

In our GG-SSM framework, each layer requires constructing an MST on feature embeddings (e.g., pixel or token embeddings) to identify critical connections before performing state propagation. Since the number of edges E can be large in dense graphs, employing Chazelle's MST algorithm with its near-linear time complexity is particularly appealing:

- **Scalability:** For a graph of L vertices (e.g., L embeddings), we can handle $\mathcal{O}(L^2)$ potential edges in worst-case dense settings or employ faster approximate methods in sparser representations. Either way, Chazelle's $\mathcal{O}(E\,\alpha(E,V))$ ensures minimal overhead when E is proportional to L or $L\log L$.
- **Uniqueness of MST Paths:** MST ensures exactly $L-1$ edges and a *unique path* between any two nodes. This property is crucial for our GG-SSMs, as it cleanly defines the path by which hidden states propagate. It further limits redundant computations and fosters efficient parameter sharing in state updates.

8.3. Intuition and Benefits

Intuitively, MST-based graph construction identifies the *closest* (or most similar) neighbors by selecting edges of the smallest weight (lowest dissimilarity). This yields a minimal set of edges connecting all nodes, providing a concise but effective skeleton to diffuse signals across the entire feature set. Consequently, even high-dimensional data with long-range dependencies can be efficiently processed with minimal redundancy. The slow-growing $\alpha(\cdot,\cdot)$ factor en-

sures that the overhead of constructing and maintaining this structure remains negligible for practical dataset sizes.

References

[1] Anastasios Nikolas Angelopoulos, Julien N. P. Martel, Amit Kohli, Jörg Conradt, and Gordon Wetzstein. Event-based near-eye gaze tracking beyond 10,000 hz. *IEEE Trans. Vis. Comput. Graph.*, 27:2577–2586, 2020. 5

[2] Ethan Baron, Itamar Zimerman, and Lior Wolf. A 2-dimensional state space layer for spatial inductive bias. In *ICLR*, 2024. 3

[3] Pietro Bonazzi, Sizhen Bian, Giovanni Lippolis, Yawei Li, Sadique Sheik, and Michele Magno. Retina : Low-power eye tracking with event camera and spiking hardware. *CVPRW*, pages 5684–5692, 2023. 2, 5

[4] Christian Brandli, Raphael Berner, Minhao Yang, Shih-Chii Liu, and Tobi Delbruck. A 240 × 180 130 db 3 µs latency global shutter spatiotemporal vision sensor. *IEEE J. Solid-State Circuits*, 49:2333–2341, 2014. 4

[5] Daniel J. Butler, Jonas Wulff, Garrett B. Stanley, and Michael J. Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, 2012. 7, 8

[6] Bernard Chazelle. A minimum spanning tree algorithm with inverse-ackermann type complexity. *J. ACM*, 47(6): 1028–1047, 2000. 2, 3, 4, 8, 1

[7] Qinyu Chen, Zuowen Wang, Shih-Chii Liu, and Chang Gao. 3et: Efficient event-based eye tracking using a change-based convlstm network. *IEEE Biomed. Circuits Syst. Conf.*, pages 1–5, 2023. 2, 4, 5

[8] Ekin Dogus Cubuk, Barret Zoph, Jon Shlens, and Quoc Le. Randaugment: Practical automated data augmentation with a reduced search space. In *NeurIPS*, pages 18613–18624, 2020. 6

[9] Abhimanyu Das, Weihao Kong, Andrew B. Leach, Shaan Mathur, Rajat Sen, and Rose Yu. Long-term forecasting with tide: Time-series dense encoder. *ArXiv*, abs/2304.08424, 2023. 6

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, K. Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. *CVPR*, pages 248–255, 2009. 2, 6

[11] Qiaole Dong and Yanwei Fu. Memflow: Optical flow estimation and prediction with memory. *CVPR*, pages 19068–19078, 2024. 8

[12] Qiaole Dong, Chenjie Cao, and Yanwei Fu. Rethinking optical flow from geometric matching consistent perspective. *CVPR*, pages 1337–1347, 2023. 8

[13] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Häusser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. *ICCV*, pages 2758–2766, 2015. 7, 8

[14] Shaharam Eivazi, Thiago Santini, Alireza Keshavarzi, Thomas C. Kübler, and Andrea Mazzei. Improving real-time cnn-based pupil detection through domain-specific data augmentation. *Proc. ACM Symp. Eye Track. Res. Appl.*, 2019. 4, 5

[15] Christoph Feichtenhofer, Axel Pinz, and Richard P. Wildes. Temporal residual networks for dynamic scene recognition. In *CVPR*, 2017. 5

[16] Daniel Y. Fu, Tri Dao, Khaled K. Saab, Armin W. Thomas, Atri Rudra, and Christopher Ré. Hungry Hungry Hippos: Towards language modeling with state space models. In *ICLR*, 2023. 3

[17] Guillermo Gallego, Tobi Delbrück, G. Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew J. Davison, Jörg Conradt, Kostas Daniilidis, and Davide Scaramuzza. Event-based vision: A survey. *IEEE TPAMI*, 44:154–180, 2019. 4

[18] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. *CVPR*, pages 3354–3361, 2012. 2, 7, 8

[19] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *Conf. Lang. Model.*, 2023. 2, 3, 4, 5, 6

[20] Albert Gu, Isys Johnson, Karan Goel, Khaled Saab, Tri Dao, Atri Rudra, and Christopher Ré. Combining recurrent, convolutional, and continuous-time models with linear state-space layers. *NeurIPS*, 34, 2021. 2

[21] Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *ICLR*, 2022. 2, 3

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1

[23] Vincent Tao Hu, Stefan Andreas Baumann, Ming Gui, Olga Grebenkova, Pingchuan Ma, Johannes Fischer, and Björn Ommer. Zigma: A dit-style zigzag mamba diffusion model. In *ECCV*, 2024. 3

[24] Tao Huang, Xiaohuan Pei, Shan You, Fei Wang, Chen Qian, and Chang Xu. Localmamba: Visual state space model with windowed selective scan. *arXiv preprint arXiv:2403.09338*, 2024. 3

[25] Shihao Jiang, Dylan Campbell, Yao Lu, Hongdong Li, and Richard I. Hartley. Learning to estimate hidden motions with global motion aggregation. *ICCV*, pages 9752–9761, 2021. 8

[26] Rudolf E. Kálmán. A new approach to linear filtering and prediction problems. In *ASME J. Basic Eng.*, 1960. 3

[27] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017. 2

[28] Zhe Li, Shiyi Qi, Yiduo Li, and Zenglin Xu. Revisiting long-term time series forecasting: An investigation on linear mapping. *ArXiv*, 2023. 6

[29] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. *ICLR*, 2023. 5, 6

[30] Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, and Yunfan Liu. Vmamba: Visual state space model. *NeurIPS*, 2024. 2, 3, 5, 6, 7

[31] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin trans-

former: Hierarchical vision transformer using shifted windows. *ICCV*, pages 9992–10002, 2021. 6, 7

[32] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *CVPR*, 2022. 7

[33] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2017. 6

[34] Yawen Lu, Qifan Wang, Siqi Ma, Tong Geng, Victor Y. Chen, Huaijin Chen, and Dongfang Liu. Transflow: Transformer as flow learner. *CVPR*, pages 18063–18073, 2023. 8

[35] Nikolaus Mayer, Eddy Ilg, Philip Häusser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. *CVPR*, pages 4040–4048, 2015. 7, 8

[36] Eric Nguyen, Karan Goel, Albert Gu, Gordon W. Downs, Preey Shah, Tri Dao, Stephen A. Baccus, and Christopher Ré. S4nd: Modeling images and videos as multidimensional signals using state spaces. *NeurIPS*, 35, 2022. 2, 3, 6, 7

[37] Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *ICLR*, 2023. 6

[38] Yuta Oshima, Shohei Taniguchi, Masahiro Suzuki, and Yutaka Matsuo. Ssm meets video diffusion models: Efficient long-term video generation with structured state spaces, 2024. 3

[39] Wei-Liang Ou, Tzu-Ling Kuo, Chin-Chieh Chang, and Chih-Peng Fan. Deep-learning-based pupil center detection and tracking technology for visible-light wearable gaze tracking devices. *Applied Sciences*, 2021. 4

[40] Zhile Ren, Orazio Gallo, Deqing Sun, Ming-Hsuan Yang, Erik B. Sudderth, and Jan Kautz. A fusion approach for multi-frame optical flow estimation. *IEEE/CVF Winter Conf. Appl. Comput. Vis.*, pages 2077–2086, 2019. 8

[41] Xingjian Shi, Zhourong Chen, Hao Wang, D. Y. Yeung, Wai-Kin Wong, and Wang chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *NeurIPS*, 2015. 5

[42] Xiaoyu Shi, Zhaoyang Huang, Weikang Bian, Dasong Li, Manyuan Zhang, Ka Chun Cheung, S. See, Hongwei Qin, Jifeng Dai, and Hongsheng Li. Videoflow: Exploiting temporal cues for multi-frame optical flow estimation. *ICCV*, pages 12435–12446, 2023. 8

[43] Xiaoyu Shi, Zhaoyang Huang, Dasong Li, Manyuan Zhang, Ka Chun Cheung, S. See, Hongwei Qin, Jifeng Dai, and Hongsheng Li. Flowformer++: Masked cost volume autoencoding for pretraining optical flow estimation. *CVPR*, pages 1599–1610, 2023. 8

[44] Jimmy T.H. Smith, Andrew Warrington, and Scott Linderman. Simplified state space layers for sequence modeling. In *ICLR*, 2023. 3

[45] Taylan Soydan, Nikola Zubić, Nico Messikommer, Siddhartha Mishra, and Davide Scaramuzza. S7: Selective and simplified state space layers for sequence modeling, 2024. 3

[46] Shangkun Sun, Yuanqi Chen, Y. Zhu, Guodong Guo, and Gezhong Li. Skflow: Learning optical flow with super kernels. *ArXiv*, 2022. 8

[47] Jamba Team. Jamba-1.5: Hybrid transformer-mamba models at scale. *ArXiv*, abs/2408.12570, 2024. 3

[48] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020. 8

[49] Marc Tonsen, Xucong Zhang, Yusuke Sugano, and Andreas Bulling. Labelled pupils in the wild: a dataset for studying pupil detection in unconstrained environments. *Proc. ACM Symp. Eye Track. Res. Appl.*, 2015. 4, 5

[50] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herv'e J'egou. Training data-efficient image transformers & distillation through attention. In *Int. Conf. Mach. Learn.*, 2020. 7

[51] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 1

[52] Zihan Wang, Fanheng Kong, Shi Feng, Ming Wang, Han Zhao, Daling Wang, and Yifei Zhang. Is mamba effective for time series forecasting? *ArXiv*, abs/2403.11144, 2024. 5, 6

[53] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *ICLR*, 2023. 6

[54] Yicheng Xiao, Lin Song, Shaoli Huang, Jiangshan Wang, Siyu Song, Yixiao Ge, Xiu Li, and Ying Shan. Grootvl: Tree topology is all you need in state space model. *NeurIPS*, abs/2406.02395, 2024. 3

[55] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. Gmflow: Learning optical flow via global matching. *CVPR*, pages 8111–8120, 2022. 8

[56] Chenhongyi Yang, Zehui Chen, Miguel Espinosa, Linus Ericsson, Zhenyu Wang, Jiaming Liu, and Elliot J. Crowley. Plainmamba: Improving non-hierarchical mamba in visual recognition, 2024. 3

[57] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Young Joon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. *ICCV*, pages 6022–6031, 2019. 6

[58] Ailing Zeng, Mu-Hwa Chen, L. Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *AAAI*, 2022. 6

[59] Ailing Zeng, Mu-Hwa Chen, L. Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *AAAI*, 2022. 6

[60] Hongyi Zhang, Moustapha Cissé, Yann Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *ICLR*, 2018. 6

[61] Xiaosong Zhang, Yunjie Tian, Wei Huang, Qixiang Ye, Qi Dai, Lingxi Xie, and Qi Tian. Hivit: Hierarchical vision transformer meets masked image modeling. *ArXiv*, abs/2205.14949, 2022. 7

[62] Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *ICLR*, 2023. 6

[63] Guangrong Zhao, Yurun Yang, Jingwei Liu, Ning Chen, Yiran Shen, Hongkai Wen, and Guohao Lan. Ev-eye: Rethinking high-frequency eye tracking through the lenses of event cameras. In *NeurIPS*, 2023. 5

[64] Shiyu Zhao, Long Zhao, Zhixing Zhang, Enyu Zhou, and Dimitris N. Metaxas. Global matching with overlapping attention for optical flow estimation. *CVPR*, pages 17571–17580, 2022. 8

[65] Lianghui Zhu, Bencheng Liao, Qian Zhang, Xinlong Wang, Wenyu Liu, and Xinggang Wang. Vision mamba: Efficient visual representation learning with bidirectional state space model. In *Int. Conf. Mach. Learn.*, 2024. 3, 6, 7

[66] Nikola Zubić and Pietro Lio. An effective loss function for generating 3d models from single 2d image without rendering. In *Artif. Intell. Appl. Innov.*, 2021. 2

[67] Nikola Zubić, Daniel Gehrig, Mathias Gehrig, and Davide Scaramuzza. From chaos comes order: Ordering event representations for object recognition and detection. In *ICCV*, pages 12846–12856, 2023. 2

[68] Nikola Zubic, Mathias Gehrig, and Davide Scaramuzza. State space models for event cameras. In *CVPR*, pages 5819–5828, 2024. 3

[69] Nikola Zubić, Federico Soldá, Aurelio Sulser, and Davide Scaramuzza. Limits of deep learning: Sequence modeling through the lens of complexity theory, 2024. 3