MambaFlow: A Mamba-Centric Architecture for End-to-End Optical Flow Estimation

1st Juntian Du
College of Computer Science and Technology
Zhejiang University of Technology
Hangzhou, China
djt@zjut.edu.cn

2nd Yuan Sun
College of Computer Science and Technology
Zhejiang University of Technology
Hangzhou, China
sy02g@zjut.edu.cn

3rd Zhihu Zhou*
College of Computer Science and Technology
Zhejiang University of Technology
Hangzhou, China
zhouzhihu@zjut.edu.cn

4th Pinyi Chen
College of Computer Science and Technology
Zhejiang University of Technology
Hangzhou, China
chenpinyi@zjut.edu.cn

5th Runzhe Zhang
College of Computer Science and Technology
Zhejiang University of Technology
Hangzhou, China
runzhezhang@zjut.edu.cn

6th Keji Mao
College of Computer Science and Technology
Zhejiang University of Technology
Hangzhou, China
maokeji@zjut.edu.cn

***Abstract*—Recently, the Mamba architecture has demonstrated significant successes in various computer vision tasks. However, its application to optical flow estimation remains unexplored. In this paper, we introduce MambaFlow, a novel framework centered around the Mamba design tailored specifically for the end-to-end optical flow estimation. It comprises two key components: (1) PolyMamba, which enhances feature representation through a dual-Mamba architecture, combining a Self-Mamba module for intra-token modeling with a Cross-Mamba module for inter-modality interaction; and (2) PulseMamba, which leverages an Attention Guidance Aggregator (AGA) to adaptively integrate features with dynamically learned weights in contrast to naive concatenation, and then leverages Mamba's recurrent mechanism to iteratively perform autoregressive flow decoding. Extensive experiments demonstrate that MambaFlow achieves remarkable results comparable to mainstream methods on benchmark datasets. Compared to SEA-RAFT, MambaFlow attains higher accuracy on the Sintel benchmark, demonstrating stronger potential for real-time deployment in practical systems.**

***Index Terms*—Optical Flow, Mamba, Computer Vision, Image Motion Analysis.**

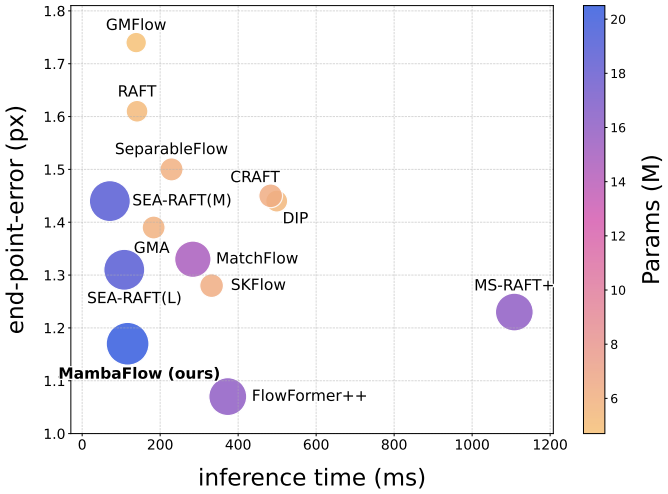

Fig. 1. End-point-error on Sintel (clean) vs. inference time (ms) and model params (M). Compared to state-of-the-art models, our proposed model achieves comparable accuracy while significantly reducing inference time, demonstrating an optimal balance between computational efficiency and performance.

I. Introduction

Optical flow estimation aims to compute pixel-wise motion between consecutive frames and is a fundamental task in computer vision.

Architectures based on Convolutional Neural Networks (CNNs) have achieved substantial progress in optical flow estimation. Early methods such as FlowNet [1] revealed the potential of end-to-end learning for this task, while PWC-Net [2] further improved estimation accuracy through pyramid feature extraction and cost-volume construction. However, these methods still struggled with large displacements and complex motions due to their limited ability in global modeling. To address this issue, RAFT [3] introduced a recurrent iterative refinement mechanism, which significantly improved estimation accuracy. Nevertheless, its repeated update process relies on numerous iterations, resulting in increased inference time and computational overhead. To further enhance global modeling, Transformer-based architectures have been introduced into optical flow estimation [4], [5]. By leveraging attention mechanisms, these methods capture long-range dependencies more effectively, but their quadratic computational complexity severely limits efficiency in practical applications.

Recently, Mamba [6] has achieved broad adoption across

multiple domains [7], [8]. Mamba addresses the limitations of CNNs and Transformers by overcoming local perception constraints and quadratic complexity. With its selective state space mechanism, it captures global information and spatiotemporal dependencies in linear complexity, while its recurrent property makes it well-suited for long-sequence modeling and autoregression.

Inspired by Mamba's capabilities for long-sequence modeling and autoregression with linear complexity, we propose MambaFlow, a novel optical flow estimation framework designed to address the challenges in CNN- or Transformer-based methods. As demonstrated in Fig. 1, our method significantly reduces inference time while maintaining remarkably high accuracy.

Our main contributions can be summarized as follows:

- We propose MambaFlow, a novel optical flow estimation architecture built on the Mamba framework both in feature enhancement and flow propagation. To our knowledge, this is the first Mamba-centric architecture designed for end-to-end optical flow estimation. Our method achieves accuracy comparable to state-of-the-art algorithms while significantly reducing inference time, which makes it more suitable for real-world application on resource-constrained devices.
- We introduce the PolyMamba module, which consists of a Self-Mamba, a Cross-Mamba, and a MLP in sequence, to globally enhance feature representation for both the reference image and its cross features with the matching image. As we will demonstrate, it effectively leverages the Mamba's long sequence modeling capabilities while maintaining a linear computational load.
- We design the PulseMamba, which integrates an Attention Guidance Aggregator for adaptive feature fusion and a Mamba-based autoregressive decoder, addressing the computational overload, large displacement and occlusion issues in recurrently refined RNN-like methods. By leveraging the Mamba's powerful autoregressive capability and linear complexity, it offers significant potential to enhance accuracy and speed simultaneously through ablation studies.

II. Related Work

A. Optical Flow Estimation

Optical flow estimation was traditionally formulated as an energy minimization task, with early variational methods like Horn and Schunck [9] relying on brightness constancy and spatial smoothness assumptions. Subsequent optimization-based approaches improved robustness through regularization [10].

In the era of deep learning, following the introduction of FlowNet [1] as the first CNN-based optical flow method, numerous subsequent algorithms such as PWC-Net [2] adopted a coarse-to-fine strategy for optical flow estimation. However, these approaches struggled with fast motions and small displacements due to inaccurate coarse-level guidance. RAFT [3] addressed this via iterative full-field refinement, inspiring variants such as GMA [11], MS-RAFT [12] and SEA-RAFT [13], which introduced modules to better handle occlusion and motion ambiguity. Despite their accuracy, these CNN-based methods require heavy iterative computation and struggle with large displacements and complex motion, limiting their efficiency on resource-constrained devices.

Transformers have emerged as powerful alternatives due to their ability to model global dependencies via self-attention. FlowFormer [4] and CRAFT [14] exploit Transformer-based designs to enhance feature expressiveness and global matching. GMFlow [5] further improves efficiency by formulating optical flow estimation as a one-shot global matching problem. However, the quadratic complexity of attention mechanisms results in high computational and memory costs, limiting deployment on lightweight systems.

B. State Space Models

State space models (SSMs) [15], originally developed in control theory, have been introduced into deep learning for long-range dependency modeling. Mamba [6] enhances SSMs with input-dependent parameterization, achieving competitive performance with improved efficiency. Building on this idea, Vision Mamba [16] and V-Mamba [17] extend SSMs to vision and demonstrate promising results on image classification [16], video understanding [18], and biomedical image segmentation [19].

However, their applicability to optical flow estimation has remained unexplored until recently. Lin et al. [20] pioneered the use of Mamba in the scene flow estimation task, proposing an SSM-based iterative update module to replace the GRU module as a decoder. Nevertheless, it was specifically designed for scene flow with point cloud feature and is not suitable for the generally used backbone optical flow methods.

Inspired by the Mamba's strength of long sequence modeling and autoregression with linear complexity, we propose a Mamba-centric architecture for optical flow estimation. Our goal is to design an end-to-end solution that utilizes Mamba for feature enhancement and flow optimization. These Mamba-centric designs guarantee the accuracy of our method while maintaining its high speed in inference, providing valuable insights for resource-constrained devices.

III. Method

A. Preliminaries

State space models (SSMs) map an input sequence $x_t \in \mathbb{R}^d$ to an output $y_t \in \mathbb{R}^d$ via a hidden state $h_t \in \mathbb{R}^N$:

$$\begin{aligned} h_t &= \overline{\mathbf{A}} h_{t-1} + \overline{\mathbf{B}} x_t, \\ y_t &= \mathbf{C} h_t. \end{aligned} \tag{1}$$

Mamba extends SSMs by making the parameters input-dependent, enabling dynamic state transitions while retaining the efficiency of parallel scan:

$$\begin{aligned} &\mathbf{B}_i = S_B(x_i), \qquad \mathbf{C}_i = S_C(x_i), \\ &\mathbf{\Delta}_i = \mathrm{softplus}\big(S_\Delta(x_i)\big). \end{aligned} \tag{2}$$

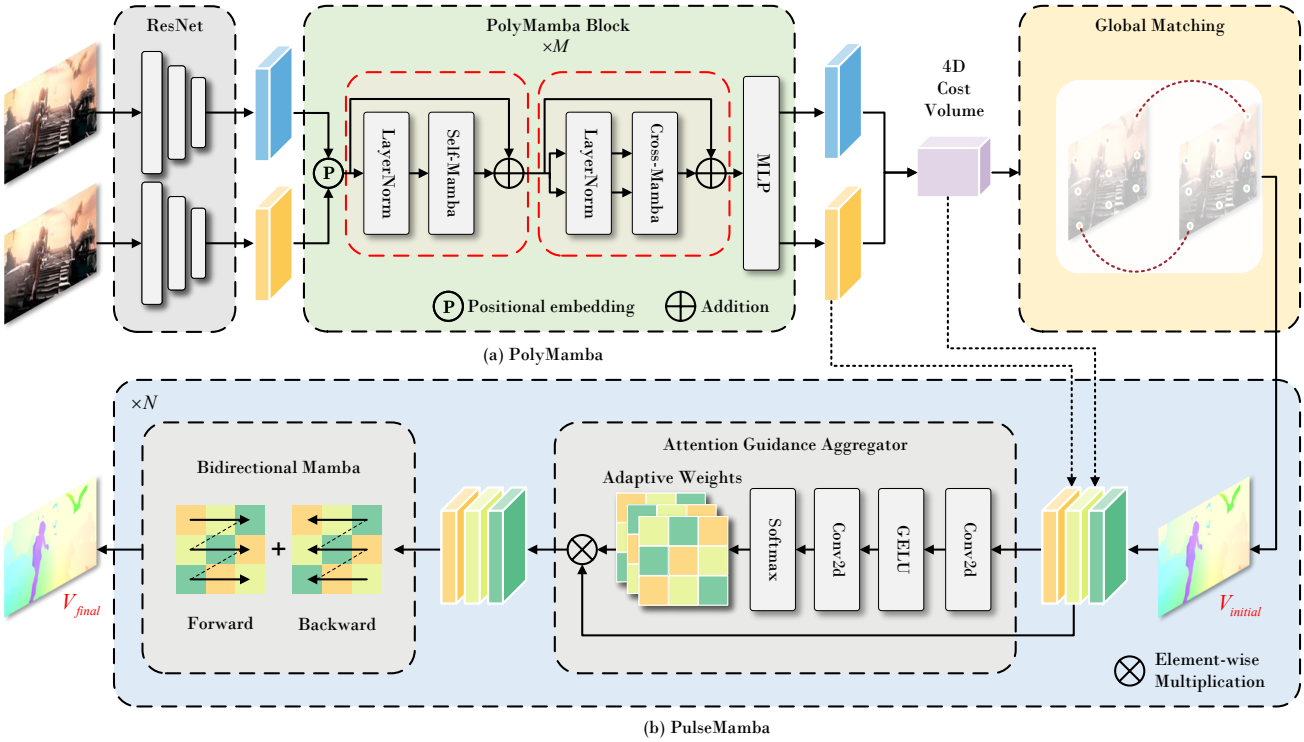

Fig. 2. The overall network architecture of our MambaFlow.

This selective, input-conditioned parametrization allows the model to capture global dependencies with (near-)linear complexity and naturally supports autoregressive processing.

B. Overall Architecture

As shown in Fig. 2, given two frames $\boldsymbol{I}_1$ and $\boldsymbol{I}_2$, a shared-weight encoder extracts dense features $\boldsymbol{F}_1, \boldsymbol{F}_2$. PolyMamba enhances them, where Self-Mamba models intra-frame dependencies and Cross-Mamba injects cross-frame cues, and a lightweight MLP fuses local and global evidence, yielding $\boldsymbol{F}_q$ and $\boldsymbol{F}_v$. A global matching stage computes all-pairs similarities and produces the initial flow $\mathbf{V}_{\text{initial}}$. PulseMamba then refines $\mathbf{V}_{\text{initial}}$ using correlations and context via attention-guided aggregation and Mamba-based iterative decoding, producing the final estimate $\mathbf{V}_{\text{final}}$.

C. PolyMamba

Self-Mamba. Before entering the Mamba block, we inject explicit spatial position information. Specifically, we flatten each feature map into a sequence and add a learnable 2D positional embedding $\boldsymbol{E}_{\text{pos}} \in \mathbb{R}^{HW \times D}$:

$$\tilde{\boldsymbol{F}}_i = \text{Flatten}(\boldsymbol{F}_i) + \boldsymbol{E}_{\text{pos}}, \quad i \in \{1, 2\}. \tag{3}$$

We then process $\tilde{\boldsymbol{F}}_1$ and $\tilde{\boldsymbol{F}}_2$ with a weight-sharing Mamba block to capture long-range dependencies within each frame. A bidirectional pass aggregates forward and backward states:

$$\begin{aligned} \boldsymbol{F}_i^{\text{self}} &= \text{Self-Forward}(\tilde{\boldsymbol{F}}_i) + \text{Self-Backward}(\tilde{\boldsymbol{F}}_i), \\ & i \in \{1, 2\}. \end{aligned} \tag{4}$$

Cross-Mamba. We implement cross-frame interaction by conditioning the SSM parameters of one stream on the other. Given $\boldsymbol{F}_1^{\text{self}}$ and $\boldsymbol{F}_2^{\text{self}}$, we first construct two complementary features for parameter generation. From the target stream, we extract a content feature that preserves local spatial structure by applying a depth-wise convolution, followed by the SiLU nonlinearity. The depth-wise design aggregates local context with low computational overhead. From the reference stream, we obtain a modulation feature through a lightweight projection, which performs a point-wise linear transformation to align channel dimensions. We then concatenate the two features and use a linear mapping, to predict the input-conditioned SSM parameters $(\mathbf{\Delta}, \mathbf{B}, \mathbf{C})$:

$$\begin{aligned} \mathbf{z}_1 &= \text{SiLU}\left(\text{DWConv}(\boldsymbol{F}_1^{\text{self}})\right), \\ \mathbf{z}_2 &= \text{Proj}\left(\boldsymbol{F}_2^{\text{self}}\right), \\ [\mathbf{\Delta}, \mathbf{B}, \mathbf{C}] &= \text{Linear}\left([\mathbf{z}_1; \mathbf{z}_2]\right). \end{aligned} \tag{5}$$

With these parameters, SelectiveScan updates the hidden state of the first stream; an output projection with a residual connection then yields the cross-attended representation:

$$\begin{aligned} \boldsymbol{y}_{1\leftarrow 2} &= \text{SelectiveScan}\left(\mathbf{\Delta}, \mathbf{B}, \mathbf{C};\ \boldsymbol{F}_1^{\text{self}}\right), \\ \boldsymbol{F}_{1\leftarrow 2}^{\text{cross}} &= \boldsymbol{F}_1^{\text{self}} + \text{Proj}\left(\boldsymbol{y}_{1\leftarrow 2}\right). \end{aligned} \tag{6}$$

We mirror this operation by swapping the two streams to obtain $\boldsymbol{F}_{2\leftarrow 1}^{\text{cross}}$. To preserve sequence symmetry, we apply forward and backward scans and sum the outputs. Finally, stacking the Self-Mamba block and the Cross-Mamba block, followed by

TABLE I

QUANTITATIVE RESULTS ON SINTEL AND KITTI. ALL MODELS FOLLOW THE UNIFIED C+T+S+K+H TRAINING RECIPE; "EXTRA DATA" NOTES ANY ADDITIONAL PRETRAINING. EVALUATION USES BATCH SIZE 1 WITH 540×960 INPUTS ON AN RTX 3090.

Extra Data	Method		Sintel		KITTI	Inference Cost	
			Clean↓	Final↓	Fl-all↓	Param	Time
	RAFT [3]	ECCV'20	1.61	2.86	5.10	5.3M	140.7ms
	GMA [11]	ICCV'21	1.39	2.47	5.15	5.9M	183.3ms
	DIP [21]	CVPR'22	1.44	2.83	4.21	5.4M	498.9ms
	CRAFT [14]	CVPR'22	1.45	2.42	4.79	6.3M	483.4ms
	GMFlow [5]	CVPR'22	1.74	2.90	9.32	4.7M	138.5ms
	SKFlow [22]	NeurIPS'22	1.28	2.23	4.85	6.3M	331.9ms
	MatchFlow [23]	CVPR'23	1.33	2.64	4.72	14.8M	283.8ms
	MS-RAFT+ [12]	IJCV'24	1.23	2.68	**4.15**	16.0M	1108.2ms
	MambaFlow (ours)		**1.17**	**2.10**	5.33	20.5M	**116.5ms**
CroCo-Pretrain	CroCoFlow [24]	ICCV'23	1.09	2.44	3.64	437.4M	6422.0ms
YouTube-VOS	FlowFormer++ [25]	CVPR'23	1.07	**1.94**	4.52	16.1M	373.4ms
VIPER	CCMR+ [26]	WACV'24	1.07	2.10	3.86	-	-
TartanAir	SEA-RAFT(M) [13]	ECCV'24	1.44	2.86	4.64	18.8M	**71.0ms**
TartanAir	SEA-RAFT(L) [13]	ECCV'24	1.31	2.60	4.30	18.8M	108.0ms
Kubric-NK	DPFlow [27]	CVPR'25	**1.04**	1.97	**3.56**	-	-

a lightweight MLP for local and global fusion, produces the feature maps $\boldsymbol{F}_q, \boldsymbol{F}_v \in \mathbb{R}^{H\times W\times D}$ used for matching. Since we modulate SSM parameters instead of forming pairwise attention maps, the computation scales linearly with sequence length and channels while injecting inter-frame alignment cues.

D. Global Matching

We compute global similarities between all pixel pairs in $\boldsymbol{F}_q$ and $\boldsymbol{F}_v$, followed by a softmax to obtain a matching distribution:

$$\mathbf{M} = \mathrm{softmax}\left(\frac{\boldsymbol{F}_q \boldsymbol{F}_v^\top}{\sqrt{D}}\right). \tag{7}$$

Let $\mathbf{G} \in \mathbb{R}^{H\times W\times 2}$ be the pixel grid in the target image. The initial flow is the offset between matched coordinates and $\mathbf{G}$:

$$\mathbf{V}_{\text{initial}} = \mathbf{M}\,\mathbf{G} - \mathbf{G}. \tag{8}$$

E. PulseMamba

Attention Guidance Aggregator (AGA). To replace the uniform weighting of naive concatenation, AGA assigns spatial weights to motion, context, and hidden features, then aggregates them by a weighted sum. Given motion features $\boldsymbol{M}$ (from a motion encoder over $\mathbf{V}_{\text{initial}}$ and correlations), context $\boldsymbol{F}_q$, and the current hidden state $\boldsymbol{h}$, we form

$$\boldsymbol{x}_{\text{AGA}} = \sum_{f\in\{M,F_q,h\}} \boldsymbol{A}_f \odot \boldsymbol{f}, \tag{9}$$

where $\boldsymbol{A}_f$ are attention maps produced from projected inputs.

Mamba-based iterative decoding. PulseMamba refines the flow in N steps using Mamba's recurrence:

$$\begin{aligned} \boldsymbol{h}^{(i)} &= \mathrm{Mamba}\left(\boldsymbol{x}_{\text{AGA}}^{(i-1)}\right), \\ \Delta\mathbf{V}^{(i)} &= \mathrm{FlowHead}\left(\boldsymbol{h}^{(i)}\right), \\ \mathbf{V}^{(i)} &= \mathbf{V}^{(i-1)} + \Delta\mathbf{V}^{(i)}, \end{aligned} \tag{10}$$

where $\mathrm{FlowHead}$ is a lightweight flow regression head that maps the hidden state $\boldsymbol{h}^{(i)}$ to a residual flow update $\Delta\mathbf{V}^{(i)}$ at the current resolution. We initialize $\mathbf{V}^{(0)} = \mathbf{V}_{\text{initial}}$, and take $\mathbf{V}_{\text{final}} = \mathbf{V}^{(N)}$ as the final estimate. By combining attention-guided fusion with linear-time state updates, PulseMamba improves robustness in occluded or textureless regions while keeping computation efficient.

IV. EXPERIMENTS

A. Experimental Setup

We pre-train on FlyingChairs (Chairs) [1] and FlyingThings3D (Things) [28], then fine-tune on the unified mixed set C+T+S+K+H (Chairs, Things, Sintel [29], KITTI-2015 [30], HD1K [31]), and finally fine-tune on the KITTI-2015 training split and report results on the official online benchmarks. The model is implemented in PyTorch and trained on 8 NVIDIA GeForce RTX 3090 GPUs using AdamW: 100k iterations on Chairs (batch size 16, learning rate 4×10^{-4}), 200k on Things (batch size 8, learning rate 2×10^{-4}), followed by 200k on Sintel with the same hyperparameters, and a final 100k on KITTI-2015 with batch size and learning rate unchanged. We report end-point error (EPE) on Sintel and the KITTI *Fl-all* outlier rate (error >3 pixels or relative error $>5\%$); to analyze large motions, we additionally compute s_{40+}, the EPE over pixels whose ground-truth flow magnitude exceeds 40 pixels.

B. Comparison with State-of-the-Arts

As shown in Table I, under the unified C+T+S+K+H setting and without any extra data, MambaFlow achieves **1.17** EPE on Sintel Clean and **2.10** on Sintel Final, while delivering the fastest inference among methods trained under the same protocol. It improves over MS-RAFT+ in both accuracy and runtime, reducing latency by **9.5**× (116.5,ms vs. 1108.2,ms) while also lowering the Sintel Clean error (1.17 vs. 1.23). Against SKFlow, another strong competitor under this setting, MambaFlow further lowers the Sintel Final error (2.10 vs. 2.23) and runs substantially faster (116.5,ms vs. 331.9,ms).

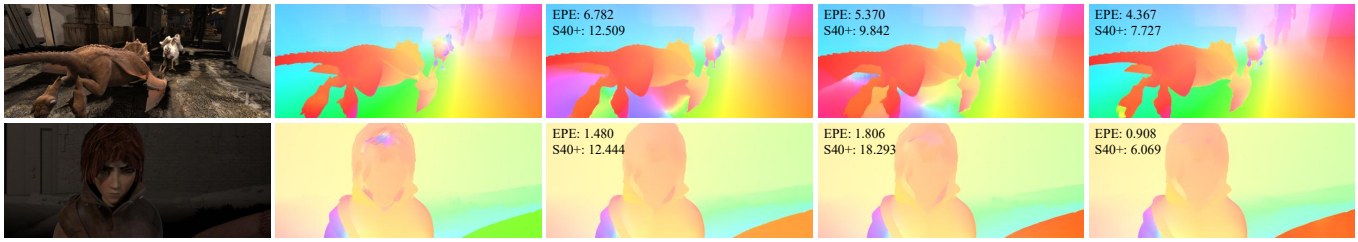

Fig. 3. Visual results on Sintel test set.

TABLE II
ABLATIONS ON THE COMPONENTS OF POLYMAMBA

Setup	Things (val)	Sintel (train)		Param
	clean	clean	final	(M)
full	**2.22**	**1.15**	**3.05**	20.5
w/o self	2.40	1.24	3.12	18.2
w/o cross	3.84	2.00	4.61	14.7
w/o MLP	2.46	1.25	3.16	17.5
w/o pos	2.29	1.18	3.11	20.5

TABLE III
ABLATIONS ON THE NUMBER OF POLYMAMBA BLOCKS

Blocks	Sintel (train)		KITTI (train)		Param (M)
	Clean	Final	EPE	F1-all	
4	1.20	3.21	7.54	22.28	16.5
6	1.16	3.17	7.27	20.81	18.5
8	**1.15**	**3.05**	**6.76**	**20.50**	**20.5**
10	1.14	3.11	6.71	20.61	22.5
12	1.15	2.96	6.45	18.90	24.5

TABLE IV
ABLATIONS ON PULSEMAMBA DESIGNS

PulseMamba iterations	use AGA	Sintel (train, clean)		
		all	matched	unmatched
0	–	1.96	0.68	15.75
1	✓	0.62	0.38	3.22
	✗	0.72	0.45	3.58
2	✓	0.56	0.34	2.99
	✗	0.60	0.36	3.16
3	✓	0.55	0.33	2.93
	✗	0.57	0.35	2.91

TABLE V
ABLATIONS ON BIDIRECTIONAL MAMBA IN POLYMAMBA AND PULSEMAMBA.

Module	Bi-scan	Sintel (train)	
		Clean	Final
PolyMamba	✓	1.15	3.05
	✗	1.34	3.21
PulseMamba	✓	1.15	3.05
	✗	1.20	3.13

For methods with extra data, SEA-RAFT(L) is a strong efficiency baseline in the fast-inference regime. MambaFlow offers similar inference time while achieving lower Sintel errors (1.17 vs. 1.31 on Clean and 2.10 vs. 2.60 on Final; 116.5,ms vs. 108.0,ms) despite SEA-RAFT(L) using TartanAir pretraining. Large-scale pretrained models (e.g., FlowFormer++, CroCoFlow) deliver strong accuracy but incur much higher latency, MambaFlow remains competitive at markedly lower runtime. Fig. 3 presents visual results on the Sintel test set.

Overall, MambaFlow offers a strong accuracy-efficiency trade-off, achieving low-latency inference while remaining competitive against extra-data pretrained methods.

C. Runtime Analysis.

End-to-end latency is dominated by iteration loops, high-resolution feature processing, and interaction complexity rather than parameter count, so MambaFlow can be slightly larger yet faster. RAFT-style methods spend most of their time in many recurrent refinement iterations, while MS-RAFT+ further adds a U-Net multi-scale coarse-to-fine pipeline with repeated refinement across scales. Transformer-based flow models typically incur quadratic token interactions and heavier memory traffic at high resolution. In contrast, MambaFlow uses linear-time state-space scans for cross-frame interaction and achieves accurate refinement with only a few steps, leading to lower latency in practice.

D. Ablation Analysis

We ablate four aspects: (a) the components of PolyMamba, (b) the number of PolyMamba blocks, (c) the PulseMamba decoder (iterations and AGA), and (d) the use of bidirectional scanning in PolyMamba and PulseMamba. As shown in Table II, Cross-Mamba provides the largest gain by modeling inter-frame relations; the self branch and positional encoding offer complementary improvements, and removing the MLP degrades accuracy. As shown in Table III, increasing depth improves accuracy up to eight blocks, which gives the best accuracy–size trade-off. As shown in Table IV, adding the Mamba-based decoder is effective, with one iteration reducing overall EPE from 1.96 to 0.62; further iterations bring smaller but consistent gains, and AGA consistently outperforms naive

concatenation. We therefore adopt two iterations with AGA by default. Finally, Table V further validates the importance of bidirectional scanning, which yields consistent gains for both PolyMamba and PulseMamba.

V. Conclusion

As Mamba continues to make significant progress in the field of computer vision, this paper explores for the first time to leverage it to estimate optical flow in an end-to-end paradigm. We suggest a Mamba-centric optical flow method, MambaFlow, which achieves a good balance between accuracy and performance compared to state-of-the-art methods. We attribute this to the Mamba's capabilities in global modeling and autoregression, as well as its linear complexity, which offers greater potential for real-world deployment on resource-constrained devices, and provides valuable insights that may influence ongoing and future developments in this rapidly evolving field of optical flow.

While MambaFlow achieves results comparable to some state-of-the-art methods, there is still room for improvement, as indicated by the KITTI results in Table I. The model's generalization may be limited when there is a substantial gap between training and test data, and its performance under challenging conditions such as motion blur, fog, or high-resolution images remains unexplored. We plan to pretrain on large-scale datasets like TartanAir [32] and VIPER [33] to address these issues. Additionally, with the emergence of many Mamba variants [34], we see potential to further refine the architecture and core Mamba-based components for improved accuracy and real-world applicability.

Acknowledgment

This research is supported by "Pioneer" and "Leading Goose" R&D Program of Zhejiang Province under No. 2026C02A1019, 2026C02A1222, 2026C02A1018; Zhejiang Provincial Natural Science Foundation of China under Grant No: LTGG23F020002; Zhejiang University of Technology School-enterprise Cooperation Project under Grant No: KYY-HX-20250798,KYY-HX-20230493, KYY-HX-20250381, KYY-HX-20250423; Medical and Health Science Program of Zhejiang Province, Grant No: WKJ-ZJ-26092.

References

[1] A. Dosovitskiy, P. Fischer, E. Ilg *et al.*, "Flownet: Learning optical flow with convolutional networks," in *ICCV*, 2015, pp. 2758–2766.
[2] D. Sun, X. Yang, M.-Y. Liu *et al.*, "Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume," in *CVPR*, 2018, pp. 8934–8943.
[3] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," in *ECCV*. Springer, 2020, pp. 402–419.
[4] Z. Huang, X. Shi, C. Zhang, Q. Wang, K. C. Cheung, H. Qin, J. Dai, and H. Li, "Flowformer: A transformer architecture for optical flow," in *European conference on computer vision*. Springer, 2022, pp. 668–685.
[5] H. Xu, J. Zhang, J. Cai *et al.*, "Gmflow: Learning optical flow via global matching," in *CVPR*, 2022, pp. 8121–8130.
[6] A. Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces," *arXiv preprint arXiv:2312.00752*, 2023.
[7] D. Wu, Y. Wang, X. Wu *et al.*, "Cross-attention inspired selective state space models for target sound extraction," in *ICASSP*. IEEE, 2025, pp. 1–5.
[8] K. Li, X. Li, Y. Wang *et al.*, "Videomamba: State space model for efficient video understanding," in *ECCV*. Springer, 2024, pp. 237–255.
[9] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.
[10] A. Bruhn, J. Weickert, and C. Schnörr, "Lucas/kanade meets horn/schunck: Combining local and global optic flow methods," *International journal of computer vision*, vol. 61, pp. 211–231, 2005.
[11] S. Jiang, D. Campbell, Y. Lu *et al.*, "Learning to estimate hidden motions with global motion aggregation," in *ICCV*, 2021, pp. 9772–9781.
[12] A. Jahedi, M. Luz, M. Rivinius *et al.*, "Ms-raft+: High resolution multi-scale raft," *IJCV*, vol. 132, no. 5, pp. 1835–1856, 2024.
[13] Y. Wang, L. Lipson, and J. Deng, "Sea-raft: Simple, efficient, accurate raft for optical flow," in *ECCV*. Springer, 2025, pp. 36–54.
[14] X. Sui, S. Li, X. Geng *et al.*, "Craft: Cross-attentional flow transformer for robust optical flow," in *CVPR*, 2022, pp. 17 602–17 611.
[15] A. Gu, K. Goel, and C. Ré, "Efficiently modeling long sequences with structured state spaces," 2022. [Online]. Available: https://arxiv.org/abs/2111.00396
[16] L. Zhu, B. Liao, Q. Zhang, X. Wang, W. Liu, and X. Wang, "Vision mamba: Efficient visual representation learning with bidirectional state space model," *arXiv preprint arXiv:2401.09417*, 2024.
[17] Y. Liu, Y. Tian, Y. Zhao *et al.*, "Vmamba: Visual state space model," *NeurIPS*, vol. 37, pp. 103 031–103 063, 2024.
[18] K. Li, X. Li, Y. Wang, Y. He, Y. Wang, L. Wang, and Y. Qiao, "Videomamba: State space model for efficient video understanding," in *ECCV*. Springer, 2025, pp. 237–255.
[19] Z. Xing, T. Ye, Y. Yang *et al.*, "Segmamba: Long-range sequential modeling mamba for 3d medical image segmentation," in *MICCAI*. Springer, 2024, pp. 578–588.
[20] M. Lin, G. Xu, Y. Wang, X. Wang, and X. Yang, "Flowmamba: Learning point cloud scene flow with global motion propagation," *arXiv preprint arXiv:2412.17366*, 2024.
[21] Z. Zheng, N. Nie, Z. Ling *et al.*, "Dip: Deep inverse patchmatch for high-resolution optical flow," in *CVPR*, 2022, pp. 8925–8934.
[22] S. Sun, Y. Chen, Y. Zhu *et al.*, "Skflow: Learning optical flow with super kernels," *NeurIPS*, vol. 35, pp. 11 313–11 326, 2022.
[23] Q. Dong, C. Cao, and Y. Fu, "Rethinking optical flow from geometric matching consistent perspective," in *CVPR*, 2023, pp. 1337–1347.
[24] P. Weinzaepfel, T. Lucas, V. Leroy *et al.*, "Croco v2: Improved cross-view completion pre-training for stereo matching and optical flow," in *ICCV*, 2023, pp. 17 969–17 980.
[25] X. Shi, Z. Huang, D. Li *et al.*, "Flowformer++: Masked cost volume autoencoding for pretraining optical flow estimation," in *CVPR*, 2023, pp. 1599–1610.
[26] A. Jahedi, M. Luz, M. Rivinius *et al.*, "Ccmr: High resolution optical flow estimation via coarse-to-fine context-guided motion reasoning," in *WACV*, 2024, pp. 6899–6908.
[27] H. Morimitsu, X. Zhu, R. M. Cesar *et al.*, "Dpflow: Adaptive optical flow estimation with a dual-pyramid framework," in *CVPR*, 2025, pp. 17 810–17 820.
[28] N. Mayer, E. Ilg, P. Hausser *et al.*, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *CVPR*, 2016, pp. 4040–4048.
[29] D. J. Butler, J. Wulff, G. B. Stanley *et al.*, "A naturalistic open source movie for optical flow evaluation," in *ECCV*. Springer, 2012, pp. 611–625.
[30] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," in *CVPR*, 2015, pp. 3061–3070.
[31] D. Kondermann, R. Nair, K. Honauer *et al.*, "The hci benchmark suite: Stereo and flow ground truth with uncertainties for urban autonomous driving," in *CVPRW*, 2016, pp. 19–28.
[32] W. Wang, D. Zhu, X. Wang *et al.*, "Tartanair: A dataset to push the limits of visual slam. in 2020 ieee," in *IROS*, pp. 4909–4916.
[33] S. R. Richter, Z. Hayder, and V. Koltun, "Playing for benchmarks," in *ICCV*, 2017, pp. 2213–2222.
[34] H. Zhang, Y. Zhu, D. Wang *et al.*, "A survey on visual mamba," *Applied Sciences*, vol. 14, no. 13, p. 5683, 2024.