Context-Aware Iteration Policy Network for Efficient Optical Flow Estimation

Ri Cheng, Ruian He, Xuhao Jiang, Shili Zhou, Weimin Tan*, Bo Yan*

School of Computer Science, Shanghai Key Laboratory of Intelligent Information Processing, Fudan University
rcheng22@m.fudan.edu.cn, rahe16@fudan.edu.cn, 20110240011@fudan.edu.cn, slzhou19@fudan.edu.cn,
wmtan@fudan.edu.cn, byan@fudan.edu.cn

Abstract

Existing recurrent optical flow estimation networks are computationally expensive since they use a fixed large number of iterations to update the flow field for each sample. An efficient network should skip iterations when the flow improvement is limited. In this paper, we develop a Context-Aware Iteration Policy Network for efficient optical flow estimation, which determines the optimal number of iterations per sample. The policy network achieves this by learning contextual information to realize whether flow improvement is bottlenecked or minimal. On the one hand, we use iteration embedding and historical hidden cell, which include previous iterations information, to convey how flow has changed from previous iterations. On the other hand, we use the incremental loss to make the policy network implicitly perceive the magnitude of optical flow improvement in the subsequent iteration. Furthermore, the computational complexity in our dynamic network is controllable, allowing us to satisfy various resource preferences with a single trained model. Our policy network can be easily integrated into state-of-the-art optical flow networks. Extensive experiments show that our method maintains performance while reducing FLOPs by about 40%/20% for the Sintel/KITTI datasets.

Introduction

Optical flow is a fundamental task that attempts to estimate per-pixel correspondences between video frames. Optical flow models are widely used in applications such as video tracking (Vihlman and Visala 2020), video super-resolution (Chan et al. 2022), video frame interpolation (Kong et al. 2022), and autonomous driving (Capito, Ozguner, and Redmill 2020). Recently, following RAFT (Teed and Deng 2020), the recurrent networks have demonstrated superior performance, because they can optimize the optical flow in a iterative manner. However, they estimate optical flow with a fixed large number of iterations, such as 32 iterations for the Sintel dataset (Butler et al. 2012), which is restrained by limited computational resource during inference. Therefore, exploring efficient optical flow estimation algorithms is urgently needed for practical applications.

To address this problem, we present three representative examples in Figure 2 with two discoveries. (1) There will be a bottleneck in the optical flow network. A bottleneck indicates that the flow network cannot improve the flow after some iterations due to the limited estimation ability of the network. For example, the average Endpoint Error (EPE) improvement of examples B and C fails to improve EPE after the 12th and 10th iterations. (2) Although EPE continues to improve, the magnitude of improvement is different for each sample. For example, example B decreases EPE by 5.71 from the 8th to 12th iteration, while example C only decreases EPE by 0.04 from the 6th to 10th iteration. Therefore, if we encounter resource constraints, we can reduce computational complexity in two ways. The first is to skip the iteration when a bottleneck is encountered, and the second is to skip an iteration where the sample only has a marginal optical flow improvement.

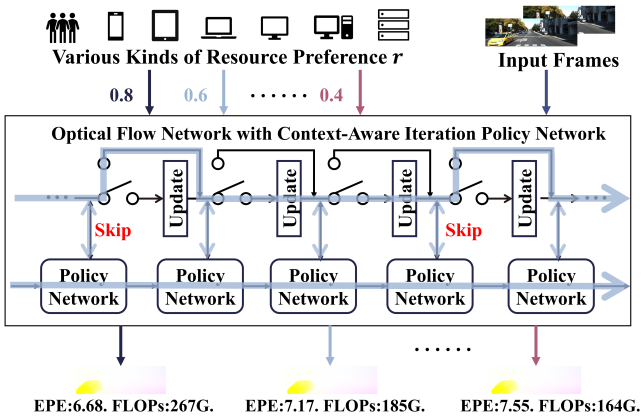


Figure 1: Efficient inference. Our policy network can skip the iteration to determine the optimal number of iterations depending on contextual information. Users are capable of altering the resource preference value r to control computational complexity.

In this paper, we propose the dynamic optical flow network with our proposed Context-Aware Iteration Policy Network, to dynamically determine optimal number of iterations for efficient optical flow estimation. We integrated our proposed policy network into four state-of-the-art backbones. The experiment results show that our dynamic networks can maintain performance while reducing floating

*Corresponding authors: Weimin Tan, Bo Yan.

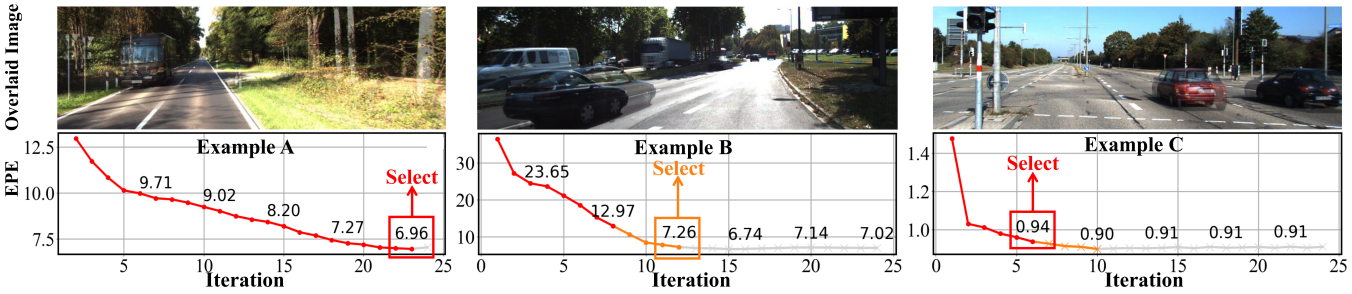


Figure 2: Three examples of changes in EPE as iteration increases.When the network encounters a bottleneck or the EPE improvement is small, we can reduce the computational complexity by skipping iterations. The EPE of examples B and C has a bottleneck after the 12th and 10th iterations. Compared with example A and B, the EPE improvement of example C from the 6th to 10th iteration is small.

point operations (FLOPs) by around 20% and 40% for KITTI (Menze and Geiger 2015) and Sintel-Final (Butler et al. 2012) training datasets. Figure 1 displays the overview of our method. Our iteration policy determines whether to skip the iteration at each time step, and we design the policy network to be lightweight so that it does not introduce a large amount of FLOPs. In addition, users can alter the resource preference value r to control computational complexity based on the availability of various computing resources. We force the computation cost of the recurrent process less than r times the original.

However, it is challenging for the policy network to decide whether to skip iteration or not, since the features in each iteration lack the global iteration information including the process and magnitude of optical flow improvement. Therefore, we must provide extra information to the policy network, so we propose to make the policy network rely on contextual information to decide. Specifically, historical hidden cell and iteration embedding are fed into the policy network to provide previous iterations information. Historical hidden cell, which comes from the previous policy network, contains the information about preceding decision and flow change information, and iteration embedding provides iteration progress information.

For future iterations information, we use the incremental loss to force the policy network to predict how much the optical flow will improve in the subsequent iteration. Both previous and future iterations information help the policy to realize whether the iteration network hits a bottleneck and the magnitude of optical flow improvement. By exploiting this contextual information, the iteration policy network can determine whether to skip the iteration or not.

This paper mainly has the following contributions:

- This paper presents an efficient optical flow estimation method that reduces computational complexity by determining the optimal number of iterations for each sample. The proposed policy network is also controllable, and it only needs a single trained network to deal with different computational resource situations.
- The proposed context-aware policy network can determine whether the flow improvement is limited. We propose using historical hidden cell, iteration embedding and incremental loss to help the policy network estimate whether the flow improvement encounters the bottleneck and the magnitude of the improvement.
- Our policy network can be seamlessly integrated into contemporary optical flow architecture. The experiments show that our dynamic network can maintain the performance while reducing FLOPs by about 40%/20% for Sintel/KITTI datasets.

Related Work

Optical Flow Estimation. Optical flow aims to find pixel-wise correspondences between two video frames. The traditional method models optical flow as an optimization problem, and they try to improve regularizations (Black and Anandan 1993; Ranftl, Bredies, and Pock 2014; Sun, Roth, and Black 2014) and energy terms (Zach, Pock, and Bischof 2007) to maximize the visual similarity between image pairs.

Recent efforts on optical flow have primarily relied on deep neural networks. FlowNet (Dosovitskiy et al. 2015) first proposed the prototype of a CNN-based optical flow model. After that, a series of well-designed works were proposed, such as FlowNet2.0 (Ilg et al. 2017), SpyNet (Ranjan and Black 2017), and PWC-Net (Sun et al. 2018, 2020). Then, the field made significant progress when RAFT (Teed and Deng 2020) proposed a new recurrent optical flow network to estimate optical flow. Based on this breakthrough architecture, many recurrent networks (Jiang et al. 2021b; Luo et al. 2022b; Sui et al. 2022; Xu et al. 2021; Zhang et al. 2021; Zheng et al. 2022; Zhou et al. 2023) have been proposed. For example, GMA (Jiang et al. 2021a) suggested combining global motion to solve the problem of estimating occlusion, and KPA-Flow (Luo et al. 2022a) designed kernel patch attention to deal with the local relationships of optical flow. FlowFormer (Huang et al. 2022) implemented a transformer structure in the optical flow network to capture long-range relations. However, these methods do not focus on efficient inference and have heavy computational complexity due to their large number of iterations during inference. For example, the iteration number is 32 and 24 for Sintel (Butler et al. 2012) and KITTI datasets (Menze and Geiger 2015), respectively.

Dynamic inference. As summarized by Han *et al.* (Han et al. 2022), dynamic inference networks have the advantages of efficiency, representation power, and interpretability since they can adapt the network structures during inference. Inference has become more efficient in recent years with the help of sparse convolution (Habibian et al. 2021; Parger et al. 2022; Wang et al. 2021; Xie et al. 2020; Yang, Huang, and Wang 2022), early exiting strategies (Bolukbasi et al. 2017; Huang et al. 2018; Wang et al. 2022; Xing et al. 2020), and inference path selection (Choi et al. 2021; Ding et al. 2021; Kong et al. 2021; Liu et al. 2022). For example, SMSR (Wang et al. 2021) and QueryDet (Yang, Huang, and Wang 2022) only use convolutions in important image areas and lower the computational cost of the unimportant region in super-resolution and object recognition tasks. FrameExit (Ghodrati, Bejnordi, and Habibian 2021) enables us to lower computational costs for video recognition by processing fewer frames for simpler videos. RBQE (Xing et al. 2020) employs a faster process to remove minor artifacts for efficient compressed image enhancement. ClassSR (Kong et al. 2021) and MADA (Choi et al. 2021) achieve efficient inference using a policy network to select different inference paths for each patch in super-resolution and video frame interpolation tasks.

Methodology

Statistical Observation

In this subsection, we illustrate our statistic observation regarding the flow results of Sintel-Final and KITTI training datasets. These observations inspire us to design a more efficient dynamic optical flow network. We obtain the EPE value of each iteration step using RAFT (Teed and Deng 2020). Similar to recent works (Teed and Deng 2020; Huang et al. 2022; Luo et al. 2022a), the total iteration number of Sintel-Final and KITTI is 32 and 24, respectively. Then we count the minimum number of iteration steps required to be within 0.01 of the best EPE for each sample and present the percentage of each iteration step in Figure 3.

We can observe that approximately 67.1% and 43.0% of the samples within the Final and KITTI datasets, respectively, achieve almost the same results to the best EPE within the first 15 iterations. This shows that the optical flow network hits a bottleneck, leading to ΔEPE close to zero. Therefore, we can skip the iterations to reduce computational complexity if the network encounters a bottleneck or the magnitude of EPE improvement is small. Since the information regarding the improvement of EPE is contained in previous iterations, we will incorporate the features of previous iterations into our policy network. In addition, we can make the policy network perceive the magnitude of improvement in the future iteration to help it decide whether to enter the subsequent iteration.

Overview of the Proposed Architecture

Given the source image I_1 and the target image I_2, the task of optical flow estimation attempts to estimate a per-pixel displacement field between them. Recurrent optical flow networks using deep learning have shown outstanding

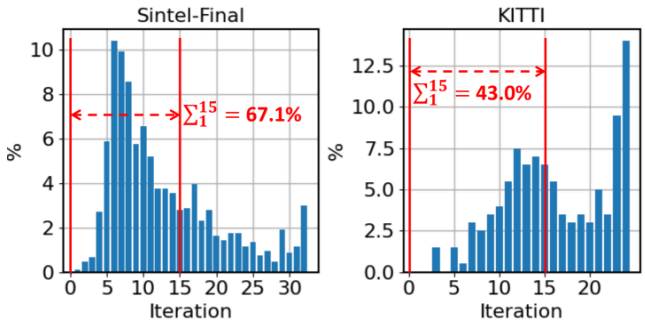


Figure 3: Statistical observation. X-axis denotes the minimum number of iteration steps to achieve the near best EPE ($||EPE_x - EPE_{best}|| < 0.01$). The Y-axis denotes the percentage of samples that achieve the near EPE at the step.

performance in recent years, and their structures typically include encoders for features and contexts, as well as an update operator. As shown in Figure 4, the update operator takes the feature ϕ_0 and a 4D cost volume C as input and iteratively updates the flow T times. The frame features ϕ_0 come from the output of feature and context encoders, and the 4D cost volume is constructed from these frame features ϕ_0. In each iteration, the update operator generates a refined feature ϕ_t and optical flow f_t. This recurrent update manner is similar to the steps of an optimization algorithm, and the optical flow estimation procedure is as follows:

$$\begin{aligned} &\phi_0, C = \boldsymbol{Encoder}(I_1, I_2), \\ &\phi_t, f_t = \boldsymbol{Update}(\phi_{t-1}, f_{t-1}, C), t \in \{1, 2, ..., T\}. \end{aligned} \tag{1}$$

The flow field f_0 is initialized to 0 everywhere. Finally, the absolute distance between the ground truth flow f_{gt} and the entire sequence of predictions $f_1, ..., f_t$ is used, which can be expressed as:

$$\mathcal{L}_{flow}(f_{gt}, \{f_t\}_1^T) = \sum_{t=1}^{T} \omega^{T-t} \left\| f_{gt} - f_t \right\|_1 \tag{2}$$

where ω is set to 0.8.

Our context-aware policy network can decide whether to skip the next iteration after each iteration performed by the update operator. We use the iteration mask p_t to indicate whether to skip or not , i.e., 0 for skip and 1 for enter. To make the mask learnable, we use the Gumbel softmax trick (Jang, Gu, and Poole 2017) to predict p_t from the feature $P_t \in \mathbb{R}^{2\times1\times1}$ that is outputted by the last convolution in the policy network, which we express as follows:

$$p_t = \frac{\exp\left(\left(P_t[0] + G_t[0]\right)/\tau\right)}{\sum_{i=0}^{1} \exp\left(\left(P_t[i] + G_t[i]\right)/\tau\right)}, \tag{3}$$

where G_t is the Gumbel noise tensor which follows a Gumbel(0, 1) distribution, and τ is the temperature parameter. During inference, the network skips the update operator if $P_0 < P_1$ unless it enters the update operator.

Figure 1 shows part of the inference procedure, where we see that the update operator does not bring any computational complexity if we skip the update. However, the update operator will execute all T iterations during the training

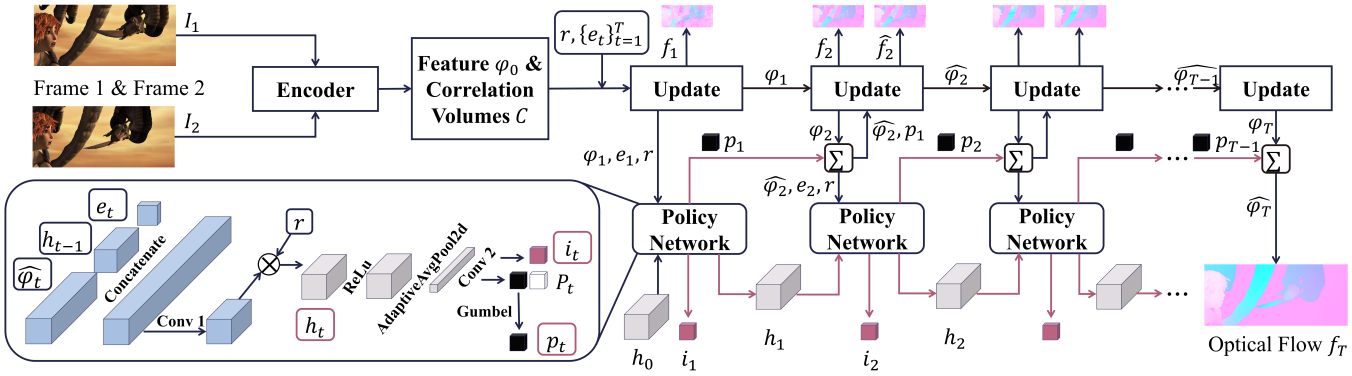


Figure 4: The architecture of the proposed dynamic optical flow network with the proposed context-aware iteration policy network. $\sum$ represents the aggregation described in Equation 4, and we omit the aggregation for $\hat{f}_t$ in this figure. $\times$ denotes multiplication.

phase to make the network trainable, as shown in Figure 4. We design the refined features and flow as an aggregation of previous and current outputs using the iteration mask. The aggregation is defined as follows:

$$\begin{aligned} \hat{\phi}_t &= \phi_t \times p_{t-1} + \hat{\phi}_{t-1} \times (1 - p_{t-1}), \\ \hat{f}_t &= f_t \times p_{t-1} + \hat{f}_{t-1} \times (1 - p_{t-1}), \end{aligned} \tag{4}$$

where $t \in \{2, 3, ..., T\}$. If $p_{t-1} \to 0$, the network skips the iteration, since $\hat{\phi}_t$ and $\hat{f}_t$ become $\hat{\phi}_{t-1}$ and $\hat{f}_{t-1}$. If $p_{t-1} \to 1$, the network enters the iteration, since $\hat{\phi}_t$ and $\hat{f}_t$ is ϕ_t and f_t. Thus, the input of next update is $\hat{\phi}_t$, $\hat{f}_t$ and C:

$$\phi_{t+1}, f_{t+1} = \boldsymbol{Update}(\hat{\phi}_t, \hat{f}_t, C), t \in \{2, 3, ..., T-1\}. \tag{5}$$

Iteration Policy Network

Controllable Network: Our proposed dynamic optical flow network is user-controllable, allowing users to control its computational complexity based on their available computational resources. We input the resource preference value r into the policy network to achieve this. The policy network can condition on this resource preference value to determine whether to skip the subsequent iterations. The smaller the resource preference value is, the greater the likelihood of skipping the next iteration, and vice versa. To introduce user resource preferences into the network, we multiply the output features of the first convolution in the policy network by the value r. We then constraint the policy network output the iteration mask p_t based on r through the resource preference loss, which is expressed as follows:

$$\mathcal{L}_{res} = max(0, \frac{1}{T-1} \sum_{t=1}^{T-1} p_t - r). \tag{6}$$

$\mathcal{L}_{res}$ constraints the average p_t should be smaller than resource preference r. As a result, we force the computational cost of the recurrent process to less than r times the original.

Previous Iterations Information ('PI'): Resource preference r enables controllability of our dynamic network. However, the policy network is unable to accurately predict whether the flow improvement is limited using the features $\hat{\phi}_t$ in the iteration. Based on the statistical observation discussed above, we input the previous iterations information from the history hidden cell h_{t-1} and iteration embedding e_t into our policy network, as shown in Figure 4. The encoding function for e_t is $e_t = \{\sin\left(2^i \pi t\right), \cos\left(2^i \pi t\right)\}_{i=0}^{2}$. h_{t-1} and e_t contains the information about flow change and progress information, assisting the policy network in determining whether the flow improvement has encountered a bottleneck.

Specifically, iteration embedding indicates the iteration progress information, which informs the policy network the position and order of the current iteration. In addition, the historical hidden cell is the condensed features of flow, iteration embedding and iteration mask in the previous iteration, so it introduces the previous policy decision and flow change information to policy network. By convolving the previous decision information in h_{t-1} with the current feature $\hat{\phi}_t$ and e_t, the policy network can determine whether the optical flow has been improved in the current iteration. The entire process of obtaining h_t and p_t is defined as follows:

$$\begin{aligned} h_t &= Conv_1(Concatenate\{\hat{\phi}_t, h_{t-1}, e_t\}) \times r, \\ P_t, i_t &= Conv_2(AdaptiveAvgPool(ReLu(h_t))), \\ p_t &= Gumbel(P_t), \end{aligned} \tag{7}$$

where $t \in \{1, 2, ..., T-1\}$, and h_0 is initialized to 0 everywhere. We obtain the history hidden cell h_t by multiplying the resource preference value r with the output of first convolution. Then, we input the h_t through a ReLu activation to enhance the ability of network representation and an AdaptiveAvgPool operator for the fusion of global feature. Finally, we output the P_t and i_t by the second convolution, and we obtain p_t by inputting P_t into the Gumbel softmax. i_t is the predicted magnitude of flow improvement, which we will describe how it works later. The computational cost of the policy network is negligible compared with the update operator since the FLOPs of the policy network is less than 1% of the update operator in RAFT. The procedure of policy

Method	Sintel-Clean (train)			Sintel-Final (train)			KITTI-15 (train)			
	EPE	FLOPs(G)	Time(s)	EPE	FLOPs(G)	Time(s)	EPE	F1-all	FLOPs(G)	Time(s)
C+T Training Data										
RAFT	1.48	730	0.12	2.67	730	0.12	5.04	17.5	595	0.09
DRAFT	1.48	**406(-44%)**	**0.09**	2.67	**502(-31%)**	**0.10**	5.06	17.5	**473(-21%)**	**0.09**
GMA	1.31	813	0.15	2.75	813	0.15	4.48	16.9	660	0.13
DGMA	1.32	**420(-48%)**	**0.09**	2.75	**429(-47%)**	**0.09**	4.51	16.9	**541(-18%)**	**0.12**
FlowFormer*	0.94	974	0.31	2.33	974	0.31	4.10	14.5	496	0.19
DFlowFormer*	0.94	**572(-41%)**	**0.21**	2.33	**496(-49%)**	**0.20**	4.11	14.5	**403(-19%)**	**0.17**
KPA-Flow	1.22	824	0.26	2.48	824	0.26	4.24	15.7	672	0.20
DKPA-Flow	1.22	**411(-50%)**	**0.16**	2.48	**416(-49%)**	**0.16**	4.25	15.7	**552(-18%)**	**0.18**
C+T+S/K+(H) Training Data										
RAFT	(0.77)	730	0.12	(1.22)	730	0.12	(0.63)	(1.5)	595	0.09
DRAFT	(0.77)	**283(-61%)**	**0.06**	(1.22)	**301(-59%)**	**0.07**	(0.63)	(1.5)	**248(-58%)**	**0.05**
GMA	(0.63)	813	0.15	(1.05)	813	0.15	(0.58)	(1.3)	660	0.13
DGMA	(0.63)	**367(-55%)**	**0.08**	(1.06)	**383(-53%)**	**0.09**	(0.58)	(1.3)	**308(-53%)**	**0.07**
FlowFormer*	(0.41)	974	0.31	(0.61)	974	0.31	(0.54)	(1.1)	496	0.19
DFlowFormer*	(0.41)	**422(-57%)**	**0.18**	(0.60)	**432(-56%)**	**0.18**	(0.54)	(1.1)	**264(-47%)**	**0.12**
KPA	(0.62)	824	0.26	(1.05)	824	0.26	(0.54)	(1.1)	672	0.20
DKPA	(0.62)	**412(-50%)**	**0.16**	(1.06)	**420(-49%)**	**0.16**	(0.54)	(1.1)	**286(-57%)**	**0.11**

Table 1: Quantitative comparison on Sintel and KITTI 2015 training datasets. ↓EPE/↓F1-all/↓FLOPs(G)/↓Time(s) are used for evaluation. 'C+T' refers to results that are trained on Chairs (Dosovitskiy et al. 2015) and Things (Mayer et al. 2016) datasets. "S/K(+H)" refers to methods fine-tuned on Sintel (Butler et al. 2012), KITTI (Menze and Geiger 2015), and some on HD1K (Kondermann et al. 2016) datasets. The red text denotes the best result, and parentheses indicate the training results. $*$ denotes that FlowFormer should forward four times to obtain the optical flow, and here we show the FLOPs and time for one forward. As described in their paper, transformers are sensitive to image size, so they crop the image into four pieces and feed it into the model four times in their code.

network can be summarized as follows:

$$h_t, p_t, i_t = \boldsymbol{Policy}(\hat{\phi}_t, h_{t-1}, e_t, r), t \in \{1, 2, ..., T-1\}, \tag{8}$$

where $\hat{\phi}_1$ is set to ϕ_0.

Future Iterations Information ('FI'): The magnitude of flow improvement in subsequent iterations is important for the policy network to make decisions based on the statistical observation discussed above. The efficient networks should dedicate computational resources to iterations where samples can have a significant optical flow improvement. Therefore, we use an incremental loss to make our policy network predict the flow improvement i_t in subsequent iterations. The incremental loss $\mathcal{L}_{incre}$ is expressed as follows:

$$\mathcal{L}_{incre} = \sum_{t=1}^{T-1} \left\| \|f_{gt} - \hat{f}_t\|_1 - \|f_{gt} - f_{t+1}\|_1 - i_t \right\|_1, \tag{9}$$

where $\hat{f}_t$ is the output flow of the current iteration after the aggregate operation, as described in Equation 4, and f_{t+1} is the output flow of the next iteration before the aggregate operation. Predicting future improvements is difficult, but our network acquires the information about the approximate magnitude improvements in optical flow. As shown in Figure 4, since the predictions of iteration mask p_t and incremental improvement i_t are based on the same features, the estimation of p_t can implicitly refer to information about the magnitude of optical flow improvement.

Overall Loss

We summarize the training loss described above as follows:

$$\mathcal{L}_{overall} = \mathcal{L}_{flow}(f_{gt}, \{\hat{f}_t\}_1^T) + \lambda_{res}\mathcal{L}_{res} + \lambda_{incre}\mathcal{L}_{incre}, \tag{10}$$

where λ_{res} and λ_{incre} are the weight for $\mathcal{L}_{res}$ and $\mathcal{L}_{incre}$.

Experiment

Our iteration policy network and training strategy can seamlessly integrate into contemporary optical flow architecture, and the policy network is FLOPs-efficient. In this work, we select four state-of-the-art methods, including RAFT (Teed and Deng 2020), GMA (Jiang et al. 2021a), FlowFormer (Huang et al. 2022), and KPA (Luo et al. 2022a), as our backbone. They are recurrent-based deep learning optical flow networks, which iteratively update the flow field through an update operator.

Evaluation Metrics

We measure average Endpoint Error (EPE) and the percentage of optical flow outliers over all pixels (F1-all) for Sintel (Butler et al. 2012) and KITTI (Menze and Geiger 2015) datasets. $EPE = \|f_{gt} - f_t\|_2$, and *F1-all*$= \frac{EPE>3 \text{ and } EPE/\|f_t\|>0.05}{\#\text{ valid pixels}}$. Outlier is the pixel satisfying $EPE > 3$ and $EPE/\|f_t\| > 0.05$. Each method was evaluated on an NVIDIA GeForce RTX 3090 GPU to measure the inference speed per sample. Additionally, we count

floating point operations (FLOPs) to determine the computational complexity by running RAFT, GMA, and KPA-Flow at a resolution of 1248×376 for Sintel and 1024×440 for KITTI, respectively. For FlowFormer, the resolution for KITTI training, KITTI test, and Sintel datasets is 720×376, 1242×432, and 960×432, as they provide in the code.

Implementation Details

The iteration policy network is implemented as shown in Figure 4, which FLOPs is less than 1% of the update operator in RAFT. We used the pre-trained network to initialize the network and fixed the network except for our policy network during training.

Specifically, we initialized the network parameter for RAFT and KPA-Flow with the pre-trained network trained on the FlyingThings (Mayer et al. 2016) dataset and then trained them for 40k iterations. This model was used to evaluate the Sintel and KITTI training datasets. To obtain models for the Sintel and KITTI test datasets, we initialized network parameters with the pre-trained network trained on Sintel, KITTI, and HD1K (Kondermann et al. 2016). Then, we trained the policy network parameters using 40k and 20k iterations for Sintel and KITTI, respectively. For GMA and FlowFormer, we trained models for Sintel and KITTI training datasets and Sintel test datasets using 50k iterations. In addition, like recent works, the total testing iterations for Sintel and KITTI are 32 and 24, while the training iteration is 12. The weights λ_{res} and λ_{incre} in the overall loss (Equation 10) are set to 50 and 1. r is randomly sampled from 0.2 $\sim$ 1.0. The learning rate is the same with their codes.

Policy Network with Existing Flow Networks

Using our proposed iteration policy network, RAFT, GMA, FlowFormer, and KPA-Flow are denoted as DRAFT, DGMA, DFlowFormer, and DKPA-Flow, respectively. Table 1 and Table 2 show that our dynamic network can maintain performance while achieving lower computational cost, with approximately 40%/20% reduction in FLOPs for Sintel/KITTI datasets. For example, Table 1 displays that the F1-all metric of all our dynamic networks is consistent with the origin backbone for the KITTI datasets. However, the FLOPs is reduced by around 20% and 50% for our dynamic networks trained on C+T and C+T+S/K+(H), respectively. In addition, our dynamic networks maintain the performance on Sintel-Final, but our models (DRAFT, DGMA, DFlowFormer, DKPA-Flow) reduce the FLOPs by 43%/43%/23%/35%, as shown in the Table 2.

In addition, we display the visualization results of EPE map in Figure 5, illustrating that our dynamic models require fewer FLOPs to estimate the optical flow, which is comparable or better to utilizing the original backbone.

Ablation Study

For each ablation model, we increase its FLOPs to exceed our DRAFT or DFlowFormer to verify the effectiveness of our proposed approaches.

Controllable Computational Complexity. Users can change their resource preference r to control the number

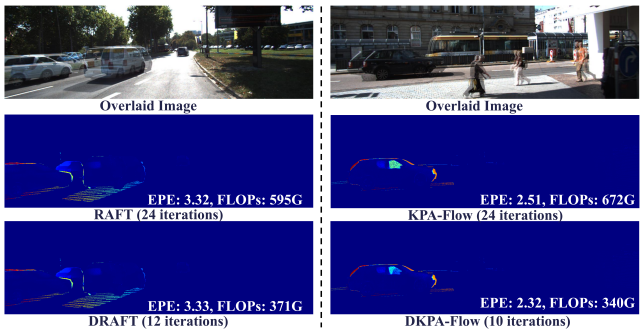


Figure 5: Qualitative comparison on the KITTI-train dataset. In the EPE map, the blue color is better, and the red is worse.

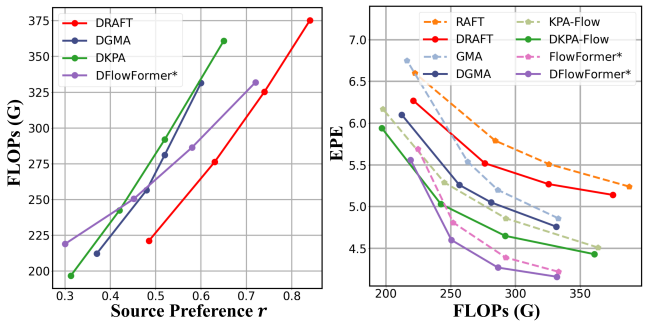


Figure 6: Ablation study of the controllable computational complexity on KITTI. We plot the graph of FLOPs changing with source preference r and the graph of EPE changing with FLOPs.

of FLOPs, which is an excellent property of our dynamic network. For example, Figure 6 illustrates that when we increase the r from 0.5 to 0.8, the FLOPs(G) of DRAFT also increases from 225 to 375. In addition, FLOPs and r are close to a proportional relationship, which is convenient for users to control FLOPs.

Figure 6 also shows that EPE improves with FLOPs, and our dynamic networks outperform the original backbone when their FLOPs is close. For example, when the FLOPs is around 325G, the EPE of our DRAFT is 5.27, which is better than 5.51 EPE of RAFT.

Ablation Study of $\mathcal{L}_{res}$. $\mathcal{L}_{res}$ in Equation 6 uses the $max(0, x)$, forcing the computational cost of the recurrent process to be less than r times the original. We implement an ablation model denoted as '-$\mathcal{L}_1$,' which uses absolute distance $\mathcal{L}_1$ instead of $max(0, x)$. Specifically, $\mathcal{L}_1$ we used is: $\mathcal{L}_1 = \|\frac{1}{T-1}\sum_{t=1}^{T-1} p_t - r\|_1$. Table 3 shows that '-$\mathcal{L}_1$' ablation models perform poorly. For example, DRAFT reduces the EPE of Clean/Final/KITTI by 0.17/0.23/0.32 compared to DRAFT-$\mathcal{L}_1$.

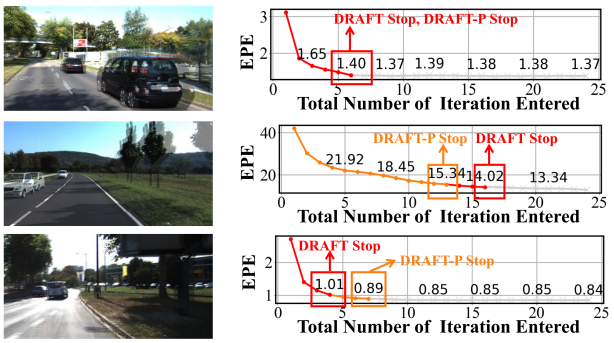
Ablation Study of Contextual Information. We implement two ablation models, '-B' denotes the policy network does not rely on contextual information, and '-P' denotes the policy network only utilizes previous iterations information. Table 3 shows the '-B' ablation model performs worst among all models. For example, DRAFT-B

Method	Sintel-Clean (test)			Sintel-Final (test)			KITTI-15 (test)		
	EPE	FLOPs(G)	Time(s)	EPE	FLOPs(G)	Time(s)	F1-all	FLOPs(G)	Time(s)
RAFT	1.94	730	0.12	3.18	730	0.12	5.10	595	0.09
DRAFT	1.93	**411(-44%)**	**0.09**	3.20	**412(-43%)**	**0.09**	5.16	**465(-22%)**	**0.09**
GMA	1.39	813	0.15	2.47	813	0.15	5.15	660	0.13
DGMA	1.43	**469(-42%)**	**0.10**	2.53	**465(-43%)**	**0.10**	5.18	**507(-23%)**	**0.10**
FlowFormer*	1.15	974	0.31	2.18	974	0.31	4.70	808	0.29
DFlowFormer*	1.17	**627(-36%)**	**0.24**	2.21	**752(-23%)**	**0.26**	4.74	**689(-15%)**	**0.26**
KPA-Flow	1.35	824	0.26	2.27	824	0.26	4.67	672	0.20
DKPA-Flow	1.35	**527(-36%)**	**0.21**	2.29	**533(-35%)**	**0.21**	4.69	**519(-23%)**	**0.18**

Table 2: Quantitative comparison on Sintel and KITTI test datasets. All models are trained on C+T+S/K+(H) training data.

Method	$\mathcal{L}_{res}$	Information		Clean	Final	KITTI	
		PI	FI	EPE	EPE	EPE	F1-all
DRAFT-$\mathcal{L}_1$/DFlowFormer-$\mathcal{L}_1$	-	✓	✓	1.73/1.15	2.97/2.60	5.84/4.30	19.2/15.1
DRAFT-B/DFlowFormer-B	✓	-	-	1.81/1.20	2.99/2.65	6.97/4.84	25.7/19.4
DRAFT-P/DFlowFormer-P	✓	✓	-	1.58/1.07	2.77/2.51	5.66/4.23	19.0/15.2
DRAFT/DFlowFormer	✓	✓	✓	**1.56/1.03**	**2.74/2.43**	**5.52/4.21**	**19.0/15.1**

Table 3: Quantitative comparison of the $\mathcal{L}_{res}$ and contextual information ablation study. '-$\mathcal{L}_1$' denotes the model uses absolute distance to control the computational cost. Previous iterations information ('PI') includes historical hidden cell h_{t-1} and iteration embedding e_t in the policy network as described in Eqa.(7). Future iterations information ('FI') is the flow improvement i_t in subsequent iterations estimated by the network and constrained by an incremental loss $\mathcal{L}_{incre}$.



Overlaid Image **EPE-Iterations Entered Figure**

Figure 7: Three visualization examples of the contextual information ablation study on KITTI.

has 0.25/0.22/1.45 more EPE than DRAFT for Clean/Final/KITTI. We find that the optimal number of iterations predicted by the '-B' model is either extremely small or extremely large, indicating that the network is incapable of making accurate predictions in the absence of contextual information. The '-P' ablation model helps decide by feeding historical hidden cell and iteration embedding into the policy network, but its performance is still inferior to DRAFT or DFlowFormer, which exploits contextual information. For example, Clean/Final/KITTI EPE is reduced by 0.02/0.03/0.14 when comparing DRAFT to DRAFT-P.

To further understand how contextual information works, we provide three visualization examples in Figure 7. By incorporating previous information into our policy network, the policy network knows whether the flow improvement has hit a bottleneck. As shown in the first example of Figure 7, DRAFT and DRAFT-P only enter iteration 6 times since the improvement of optical flow after that point is negligible.

From the second and third examples in Figure 7(b), we can observe that DRAFT skips 3 more iterations for the third example, so EPE increases the EPE by 0.12 compared to DRAFT-P. However, DRAFT reduces 1.32 EPE for the second example by entering 3 more iterations. These observations indicate that by using incremental loss to help the policy network perceive future information, the policy network can assign more iterations to samples with a large improvement in subsequent iterations.

Conclusion

This paper proposes a novel context-aware iteration policy network for efficient optical flow estimation. The policy network determines whether the flow improvement hit a bottleneck based on the context information. The ablation study shows the controllability and effectiveness of our FLOPs control strategy and the usefulness of contextual information. Extensive experiments demonstrate that our network maintains the performance with a 40%/20% reduction in FLOPs for the Sintel/KITTI datasets.

Acknowledgments

This work is supported by NSFC (GrantNo.: U2001209 and 62372117) and Natural Science Foundation of Shanghai (21ZR1406600).

References

Black, M.; and Anandan, P. 1993. A framework for the robust estimation of optical flow. In *1993 (4th) International Conference on Computer Vision*, 231–236.

Bolukbasi, T.; Wang, J.; Dekel, O.; and Saligrama, V. 2017. Adaptive Neural Networks for Efficient Inference. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, 527–536.

Butler, D. J.; Wulff, J.; Stanley, G. B.; and Black, M. J. 2012. A Naturalistic Open Source Movie for Optical Flow Evaluation. In *Computer Vision – ECCV 2012*, 611–625. Berlin, Heidelberg.

Capito, L.; Ozguner, U.; and Redmill, K. 2020. Optical Flow based Visual Potential Field for Autonomous Driving. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, 885–891.

Chan, K. C.; Zhou, S.; Xu, X.; and Loy, C. C. 2022. BasicVSR++: Improving Video Super-Resolution With Enhanced Propagation and Alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 5972–5981.

Choi, M.; Lee, S.; Kim, H.; and Lee, K. M. 2021. Motion-Aware Dynamic Architecture for Efficient Frame Interpolation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 13839–13848.

Ding, T.; Liang, L.; Zhu, Z.; and Zharkov, I. 2021. CDFI: Compression-Driven Network Design for Frame Interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 8001–8011.

Dosovitskiy, A.; Fischer, P.; Ilg, E.; Hausser, P.; Hazirbas, C.; Golkov, V.; van der Smagt, P.; Cremers, D.; and Brox, T. 2015. FlowNet: Learning Optical Flow With Convolutional Networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Ghodrati, A.; Bejnordi, B. E.; and Habibian, A. 2021. FrameExit: Conditional Early Exiting for Efficient Video Recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 15608–15618.

Habibian, A.; Abati, D.; Cohen, T. S.; and Bejnordi, B. E. 2021. Skip-Convolutions for Efficient Video Processing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2695–2704.

Han, Y.; Huang, G.; Song, S.; Yang, L.; Wang, H.; and Wang, Y. 2022. Dynamic Neural Networks: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11): 7436–7456.

Huang, G.; Chen, D.; Li, T.; Wu, F.; van der Maaten, L.; and Weinberger, K. Q. 2018. Multi-scale dense networks for resource efficient image classification.

Huang, Z.; Shi, X.; Zhang, C.; Wang, Q.; Cheung, K. C.; Qin, H.; Dai, J.; and Li, H. 2022. FlowFormer: A Transformer Architecture for Optical Flow. In *Computer Vision – ECCV 2022*, 668–685. Cham.

Ilg, E.; Mayer, N.; Saikia, T.; Keuper, M.; Dosovitskiy, A.; and Brox, T. 2017. FlowNet 2.0: Evolution of Optical Flow Estimation With Deep Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jang, E.; Gu, S.; and Poole, B. 2017. Categorical Reparameterization with Gumbel-Softmax. In *International Conference on Learning Representations*.

Jiang, S.; Campbell, D.; Lu, Y.; Li, H.; and Hartley, R. 2021a. Learning To Estimate Hidden Motions With Global Motion Aggregation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 9772–9781.

Jiang, S.; Lu, Y.; Li, H.; and Hartley, R. 2021b. Learning Optical Flow From a Few Matches. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 16592–16600.

Kondermann, D.; Nair, R.; Honauer, K.; Krispin, K.; Andrulis, J.; Brock, A.; Gussefeld, B.; Rahimimoghaddam, M.; Hofmann, S.; Brenner, C.; and Jahne, B. 2016. The HCI Benchmark Suite: Stereo and Flow Ground Truth With Uncertainties for Urban Autonomous Driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Kong, L.; Jiang, B.; Luo, D.; Chu, W.; Huang, X.; Tai, Y.; Wang, C.; and Yang, J. 2022. IFRNet: Intermediate Feature Refine Network for Efficient Frame Interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 1969–1978.

Kong, X.; Zhao, H.; Qiao, Y.; and Dong, C. 2021. ClassSR: A General Framework to Accelerate Super-Resolution Networks by Data Characteristic. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 12016–12025.

Liu, Z.; Wang, Y.; Han, K.; Ma, S.; and Gao, W. 2022. Instance-Aware Dynamic Neural Network Quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 12434–12443.

Luo, A.; Yang, F.; Li, X.; and Liu, S. 2022a. Learning Optical Flow With Kernel Patch Attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 8906–8915.

Luo, A.; Yang, F.; Luo, K.; Li, X.; Fan, H.; and Liu, S. 2022b. Learning Optical Flow with Adaptive Graph Reasoning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(2): 1890–1898.

Mayer, N.; Ilg, E.; Hausser, P.; Fischer, P.; Cremers, D.; Dosovitskiy, A.; and Brox, T. 2016. A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Menze, M.; and Geiger, A. 2015. Object Scene Flow for Autonomous Vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Parger, M.; Tang, C.; Twigg, C. D.; Keskin, C.; Wang, R.; and Steinberger, M. 2022. DeltaCNN: End-to-End CNN Inference of Sparse Frame Differences in Videos. In *Proceed-*

ings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 12497–12506.

Ranftl, R.; Bredies, K.; and Pock, T. 2014. Non-local Total Generalized Variation for Optical Flow Estimation. In Fleet, D.; Pajdla, T.; Schiele, B.; and Tuytelaars, T., eds., *Computer Vision – ECCV 2014*, 439–454. Cham.

Ranjan, A.; and Black, M. J. 2017. Optical Flow Estimation Using a Spatial Pyramid Network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Sui, X.; Li, S.; Geng, X.; Wu, Y.; Xu, X.; Liu, Y.; Goh, R.; and Zhu, H. 2022. CRAFT: Cross-Attentional Flow Transformer for Robust Optical Flow. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 17602–17611.

Sun, D.; Roth, S.; and Black, M. 2014. A Quantitative Analysis of Current Practices in Optical Flow Estimation and The Principles Behind Them. *International Journal of Computer Vision*, 106: 115–137.

Sun, D.; Yang, X.; Liu, M.-Y.; and Kautz, J. 2018. PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Sun, D.; Yang, X.; Liu, M.-Y.; and Kautz, J. 2020. Models Matter, So Does Training: An Empirical Study of CNNs for Optical Flow Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(6): 1408–1423.

Teed, Z.; and Deng, J. 2020. RAFT: Recurrent All-Pairs Field Transforms for Optical Flow. In *Computer Vision – ECCV 2020*, 402–419. Cham: Springer International Publishing.

Vihlman, M.; and Visala, A. 2020. Optical Flow in Deep Visual Tracking. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(07): 12112–12119.

Wang, L.; Dong, X.; Wang, Y.; Ying, X.; Lin, Z.; An, W.; and Guo, Y. 2021. Exploring Sparsity in Image Super-Resolution for Efficient Inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 4917–4926.

Wang, S.; Liu, J.; Chen, K.; Li, X.; Lu, M.; and Guo, Y. 2022. Adaptive Patch Exiting for Scalable Single Image Super-Resolution. 292–307. Berlin, Heidelberg.

Xie, Z.; Zhang, Z.; Zhu, X.; Huang, G.; and Lin, S. 2020. Spatially Adaptive Inference with Stochastic Feature Sampling and Interpolation. In *Computer Vision – ECCV 2020*, 531–548. Cham.

Xing, Q.; Xu, M.; Li, T.; and Guan, Z. 2020. Early Exit or Not: Resource-Efficient Blind Quality Enhancement for Compressed Images. In *Computer Vision – ECCV 2020*, 275–292.

Xu, H.; Yang, J.; Cai, J.; Zhang, J.; and Tong, X. 2021. High-Resolution Optical Flow From 1D Attention and Correlation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 10498–10507.

Yang, C.; Huang, Z.; and Wang, N. 2022. QueryDet: Cascaded Sparse Query for Accelerating High-Resolution Small Object Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 13668–13677.

Zach, C.; Pock, T.; and Bischof, H. 2007. A Duality Based Approach for Realtime TV-L1 Optical Flow. In Hamprecht, F. A.; Schnörr, C.; and Jähne, B., eds., *Pattern Recognition*, 214–223. Berlin, Heidelberg: Springer Berlin Heidelberg.

Zhang, F.; Woodford, O. J.; Prisacariu, V. A.; and Torr, P. H. 2021. Separable Flow: Learning Motion Cost Volumes for Optical Flow Estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 10807–10817.

Zheng, Z.; Nie, N.; Ling, Z.; Xiong, P.; Liu, J.; Wang, H.; and Li, J. 2022. DIP: Deep Inverse Patchmatch for High-Resolution Optical Flow. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 8925–8934.

Zhou, S.; He, R.; Tan, W.; and Yan, B. 2023. SAMFlow: Eliminating Any Fragmentation in Optical Flow with Segment Anything Model. arXiv:2307.16586.