Removing Cost Volumes from Optical Flow Estimators

Simon Kiefhaber[1,2] Stefan Roth[1,2] Simone Schaub-Meyer[1,2]

[1]Department of Computer Science, Technical University of Darmstadt
[2]Hessian Center for AI (hessian.AI)

https://visinf.github.io/recover

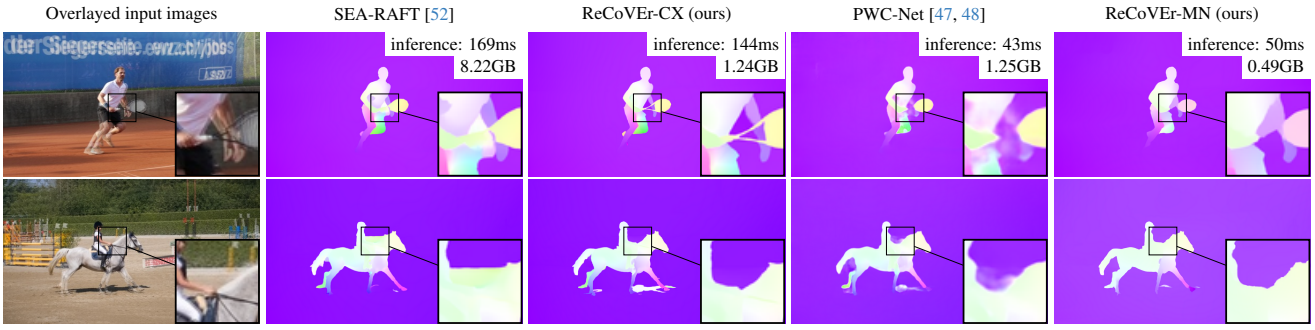

Figure 1. **ReCoVEr.** We propose a method to remove cost volumes from optical flow estimators during training, and thereby, we are able to create fast and accurate optical flow estimators with a significantly reduced memory footprint. Our most accurate model, ReCoVEr-CX, reaches state-of-the-art accuracy while being more efficient w.r.t. inference and memory than SEA-RAFT [52]. Our most efficient model, ReCoVEr-MN, predicts sharper motion boundaries compared to the popular PWC-Net [47, 48], while having comparable efficiency.

Abstract

Cost volumes are used in every modern optical flow estimator, but due to their computational and space complexity, they are often a limiting factor regarding both processing speed and the resolution of input frames. Motivated by our empirical observation that cost volumes lose their importance once all other network parts of, e.g.*, a RAFT-based pipeline have been sufficiently trained, we introduce a training strategy that allows removing the cost volume from optical flow estimators throughout training. This leads to significantly improved inference speed and reduced memory requirements. Using our training strategy, we create three different models covering different compute budgets. Our most accurate model reaches state-of-the-art accuracy while being* $1.2\times$ *faster and having a* $6\times$ *lower memory footprint than comparable models; our fastest model is capable of processing Full HD frames at* $20\,\mathrm{FPS}$ *using only* $500\,\mathrm{MB}$ *of GPU memory.*

1. Introduction

The task of optical flow estimation is to compute the apparent 2D motion between two consecutive frames for each pixel. Optical flow is a core part of many downstream tasks such as video inpainting [11, 27, 58, 62], video frame interpolation [6, 37, 43, 60], and object tracking [59].

Since optical flow essentially amounts to a 2D search problem [55] for every pixel, it is very expensive to compute due to the quadratic nature of the problem. Recent methods for optical flow prediction are usually based on deep learning. However, they require a part of the network to specialize in computing similarities across time. This part is often realized using non-learnable layers that calculate the similarity of the pixels between frames. These layers are often referred to as 'cost volumes,' 'correlation layers,' or, in the context of transformer-based architectures, 'cross-attention.' All these layer types have in common that they measure the similarity between pixels by calculating the cosine similarity, or a closely related measure, between every pixel of one input frame to candidate matching pixels of the other. They all share the problem of quadratic growth in computational and space complexity with input resolution. Due to these layers using very similar computations,

To appear in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Honolulu, Hawai'i, USA, 2025.

we mainly refer to them as *cost volumes* in this work.

The recent SEA-RAFT approach [52] achieves state-of-the-art accuracy while being relatively efficient. Still, the cost volume is responsible for more than half of the computations, as visualized in Fig. 2, significantly dominating the computational cost. The cost volume also heavily influences and limits the maximum input resolution processable by current networks. The memory requirements of many methods increase so rapidly that even processing common Full HD (1920×1080) frames can be problematic due to limited memory, as just the cost volume already requires 4 GB at this resolution, growing to 62 GB when the resolution is doubled. Being able to remove or replace cost volumes is, therefore, a promising direction to significantly improve the efficiency of a wide range of current optical flow estimators.

Inspired by early research on this topic [9, 21], in this work, we analyze the role and benefit of cost volumes for the overall prediction error. Based on our empirical observation that optical flow estimators are less dependent on their cost volume after training if they also have a context encoder for an initial prediction of the flow, we introduce a specific training strategy to adapt optical flow networks during training such that the cost volume is no longer needed at inference time. We propose three different models based on modified RAFT-like [50] architectures, utilizing their two parallel branches: One branch computes features with a cost volume, while the other encodes context information and makes an initial optical flow prediction using an architecture without a cost volume (or similar). Together with our training strategy, we are able to remove the necessity of the cost volume completely during training. Our modified networks reach competitive accuracies, and our most powerful model even reaches state-of-the-art accuracies while being significantly faster in inference compared to previous models with comparable accuracies.

2. Related Work

Optical flow estimation. Many different methods for optical flow estimation have been proposed over the years. Earlier methods often formulated optimization problems that were (approximately) solved to predict optical flow [3, 14, 31, 39]. FlowNet [9] was the first deep learning-based method that reached similar accuracies to classical methods by utilizing CNNs and correlation layers. Following the success of FlowNet, many different CNN-based methods like FlowNet2 [21], PWC-Net [47, 48], and SpyNet [41] were proposed. All of these methods rely on the computation of some type of matching score in the form of either a correlation layer, cost volume, warping, or a combination of the aforementioned methods to determine the matching compatibility of features corresponding to pixels in both input frames. Early research on the FlowNet-S architecture

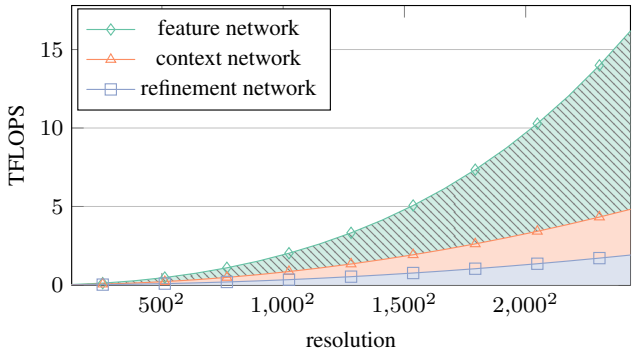

Figure 2. **Analysis of the computational expense.** Cumulative number of floating point operations (FLOPS) required for a single optical flow prediction using SEA-RAFT-M [52] for various input resolutions. The cost volume of SEA-RAFT is part of the *feature network*. In this work, we demonstrate a method to remove the *feature network* during training, thereby eliminating a major part of the required compute operations at inference time.

[9] showed that CNNs without correlation layers are less accurate. We revisit this issue here.

Following the general trend of using vision transformers [10] in computer vision, many transformer-based methods like FlowFormer [17], CroCo-Flow [53], and MemFlow [7] utilize transformers. Still, notably, none of these approaches uses plain vision transformers, containing only self-attentions, but rather include some custom layers, like cost volumes or cross-attention. In contrast to other computer vision tasks, recent papers demonstrated that CNNs can still outperform ViTs for optical flow estimation while also being more efficient [8, 52].

Cost volumes. FlowNet [9] introduced the concept of a correlation layer in the context of neural networks, where the similarity between a feature and its neighbors from one frame and all features of the other frame are computed. The computational complexity for calculating this layer is $\mathcal{O}(h^2w^2)$, where h and w refer to the height and width of the feature map, respectively. Since this layer is very expensive to compute at higher resolutions, FlowNet does limit the maximum displacement where the correlations are calculated to a fixed distance D, reducing the complexity to $\mathcal{O}(D^2hw)$. The disadvantage of this is that larger motions cannot be captured anymore. To overcome the limited motion range to a certain degree, PWC-Net [47, 48] and other methods [18, 19, 41] utilize image pyramids for a coarse-to-fine estimation where the flow is first estimated at a very low resolution and further upsampled and refined until the target resolution is reached. This allows to capture large motions at lower resolutions, where the corresponding pixel displacements are smaller, even when limiting the maximum displacements considered. RAFT [50] addresses the limitation of motion ranges by introducing a cost volume computed at multiple resolutions without any displace-

ment range limitations. Instead of limiting the displacement range, RAFT limited the maximum resolution of the cost volume to $1/8$ of the input frame resolutions, reducing the computational complexity enough to compute a global cost volume for the full feature map, though at a smaller feature resolution. The multi-scale cost volume is then sampled by a recurrent module that iteratively refines the flow prediction. The idea of globally matching the pixels was adapted and improved by multiple methods like GMA [25], Flow1D [55], FlowFormer [17], CRAFT [46], and SEA-RAFT [52]. The idea was also adopted by ViT-based [10] approaches, where cross-attention is used to compute similar features as a cost volume [53, 56, 57].

Efficient cost volumes. Since cost volumes play a critical role in the overall accuracy of optical flow estimators while also using a significant amount of compute, multiple methods have been introduced to simplify the calculations in more sophisticated ways than the displacement range limitation used in earlier works. Jiang et al. [26] introduced the concept of sparse cost volumes, where a strategy is formulated to identify the top-k best matching pixels from the other frame for each pixel and then only calculating the matching cost for these matches. Flow1D [55] introduced a decomposition of a cost volume into two lower-dimensional cost volumes and, therefore, approximated the solution of the 2D matching problem of optical flow by solving two 1D matching problems, allowing the calculation of optical flow between high-resolution input frames. As an alternative, HCVFlow [61] proposed the calculation of hybrid cost volumes that combine the top-k matching idea of Jiang et al. [26] with the decomposition idea of Flow1D. Instead of pre-calculating the entire cost volume, it is also possible to reduce the memory requirements of cost volumes by on-demand calculation of individual entries of the cost volume [23, 24, 50]. While this approach reduces the memory footprint, it often increases the inference time significantly on commonly used accelerators. Recently, Briedis et al. [4] proposed to combine sparse evaluations of the cost volumes with specialized sampling strategies such that these calculations can be run more efficiently on common accelerators.

Efficient CNNs. Not only the efficiency of cost volumes can be improved, but also the efficiency of CNNs. Over the years, many efficient architectures were proposed [15, 16, 20, 29, 34, 42, 49, 54]. The efficiency improvements in CNNs are often reached by modifying kernel sizes [29, 54], utilizing downsampling [13, 29, 49, 54] or dilated convolutions [34], and modified convolutional operators [15, 16, 42]. We leverage this progress to realize efficient but powerful context networks for computing the initial optical flow.

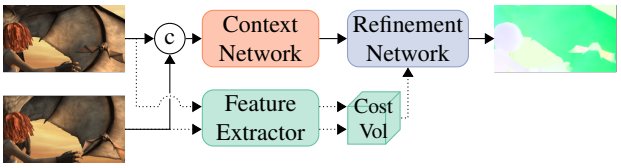

Figure 3. **ReCoVEr architecture overview.** Our method assumes an architecture similar to RAFT [50] where the input frames are processed by a *context network* to obtain an initial flow estimate and context features, and in parallel, the inputs are processed by the *feature network*, consisting of a feature extractor and a cost volume. The outputs of both branches are then combined by the *refinement network* to obtain the optical flow prediction. Our training strategy allows us to cut away the *feature network* during training (dotted path). This increases the computational and memory efficiency of the entire optical flow estimator at inference.

3. Enhancing Optical Flow Estimators

Since the introduction of FlowNet [9] correlation or cost volumes are an integral part of many optical flow estimators. While in the original paper, the benefit of including these volumes over a pipeline without them was not yet really obvious, the subsequent work, FlowNet2 [21], clearly advocates for using them. However, cost volumes now heavily dominate the computational cost, including time and memory of an optical flow estimator, since their complexity is $\mathcal{O}(h^2w^2)$ for frames of size $\mathbf{I} \in \mathbb{R}^{3\times h\times w}$. But since FlowNet, there have been multiple advancements in the building blocks of neural networks, and more powerful CNNs have been developed [12, 29]. This raises the question whether or in what form cost volumes are still needed or whether there are more efficient solutions with the same or even better accuracy.

3.1. Analysis of SOTA optical flow pipelines

We focus our analysis and proposed solution on the RAFT [50] architecture, more precisely SEA-RAFT [52], as it represents the current state of the art w.r.t. endpoint error (EPE) while also being compute efficient. RAFT-based optical flow estimators mainly consist of three building blocks: a *feature network*, a *context network*, and a *refinement network*, as illustrated in Figure 3. The feature network extracts features from each input frame and calculates a cost volume from these extracted features. In parallel to the feature network, the input is also processed by a context network, which usually only uses convolutional layers. In the case of SEA-RAFT, the context network predicts an initial optical flow estimate as well as a feature map from the concatenated input frames. The outputs of the feature and context networks are then used as inputs for the refinement network, which iteratively refines the initial flow estimate by utilizing the cost volume and the additional features predicted by the context network to create the final optical flow prediction. While the feature network and the context net-

work both produce information used for optical flow, they do not interact with each other directly and can individually be removed without breaking the network. However, while the benefit of the context and refinement network has been ablated, the feature network with the cost volume has been treated as given. In the case of SEA-RAFT-M, the EPE on the Spring dataset [33] improves by 0.62 from the initial to the final prediction, but it remains unclear how big the contributions of the cost volume are to this result, in addition to just performing the refinement iterations.

We argue that the extra computation and memory needed for the cost volumes is disproportionally high w.r.t. to the rest of the framework. Figure 2 shows the FLOPS used for each component, and even for very efficient methods like SEA-RAFT, the computation of the cost volume takes 53 % of compute at low input resolutions like 480×320 and 61 % at higher resolutions like 1920×1080. Since the memory complexity grows at least as fast as the compute, this also leads to many methods being unable to even process Full HD (1920×1080) frames on a GPU with 48GB of VRAM.

Based on the high computational cost of cost volumes for an unclear impact on the accuracy, we conclude that having a closer look at the role and importance of the cost volume within the RAFT architecture is a promising direction towards significantly more efficient optical flow estimators.

3.2. ReCoVEr

Our goal is to **Re**move **Co**st **V**olumes from optical flow **E**stimato**r**s (ReCoVEr) while retaining the accuracy.

The naive approach, *i.e*. having no cost volume and only using the context (here, a ResNet-based optical flow estimator) and refinement network (here, a convolutional GRU run for 4 refinement iterations), indeed, does lead to significantly worse results (*cf*. Tab. 1). Motivated by the observation of FlowNet2 [21] that just modifying datasets and training schedules can lead to significant improvements, we analyze various strategies of reducing the contribution of the feature network during training. Specifically, we test *(i)* the option of fading out the contribution of the cost volume by adding a dropout layer between the feature and refinement network with an increasing drop rate; *(ii)* removing the entire feature network with the cost volume after a certain number of training steps (referred to as *cut-off*). As we can see in Tab. 1, both strategies lead to similar results but are significantly better than the refinement network that never had access to the cost volume. On the one hand, this empirically shows that the refinement module benefits from having access to the features from the cost volume when iteratively refining the flow, but on the other hand, this also hints that by just varying the training scheme, we can decrease the dependence of the refinement module on the cost volume for inference drastically.

However, the current EPE of this preliminary attempt is slightly worse than the original baseline, although being significantly faster ($2.5\times$) and more memory efficient ($8.8\times$). In the following, we further improve the training strategy and show that with the right choice of context network and training strategy, the EPE can be further reduced, leading to overall state-of-the-art results.

	Sintel (val.)		Spring	FLOPS	memory
	Clean	Final			
SEA-RAFT [52]	**(0.43)**	**(0.58)**	**0.54**	<u>4.40T</u>	<u>8.21GB</u>
no cost volume	0.93	1.06	0.86		
fade-out	0.81	0.92	<u>0.68</u>	**1.74T**	**0.93GB**
cut-off	<u>0.80</u>	<u>0.91</u>	<u>0.68</u>		

Table 1. **Ablation of cost volume contributions**. Comparison between the endpoint errors (EPE) on the Sintel validation set [5] and the Spring training set [33] when training a ResNet-based optical flow estimator without any cost volume compared to ResNet-based estimator where the cost volume is slowly faded out by increasing Dropout [45] to 100% over time, and one where the cost volume is cut off after a fixed number of training iterations. For completeness, we also show the accuracies achievable by an unmodified SEA-RAFT model. However, given that it was also trained on the Sintel validation set, we put these numbers in parentheses. The FLOPS and memory reported refer to the fully trained networks in inference mode on an input pair from the Spring dataset. **Bold** and <u>underlined</u> values indicate the best and second best results.

ReCoVEr training strategy. To benefit from the cost volume during training but enable its removal during inference, we propose the following training strategy. Based on our analysis in Tab. 1, we know that we can stop using the cost volume by fading out or completely cutting off the feature extraction branch after a certain number of training iterations, but not from the beginning. Therefore, we start training all parts of our optical flow estimator without any modifications to make sure that the training is stable and useful weights are learned for each part of our network. Since we have found no significant difference between fading out the cost volume compared to cutting it off at a fixed step, *cf*. Tab. 1, we use the cut-off strategy and stop calculating the result of the feature network after a certain number of training iterations; we remove the parts of the refinement module that receive the cost volume as an input. Afterward, we continue training the shrunk-down network until it fully converges on the training dataset(s).

We mostly follow the training protocol of SEA-RAFT [52], where each model is trained in multiple stages. In the first stage, TartanAir [51], which consists of mostly rigid motions, is used, followed by training on the relatively simplistic motions and objects in FlyingChairs [9]. Afterwards, the network is trained on FlyingThings [32]. For the final training stage, a combined dataset is created, denoted as TSKH, consisting of FlyingThings [32], the

Cut-off	Sintel (val.)		Spring
	Clean	Final	
Never	**(0.43)**	**(0.58)**	**0.54**
TartanAir	0.93	1.06	0.86
FlyingChairs	0.80	0.91	0.70
FlyingThings	0.80	0.91	0.68
TSKH	0.81	0.93	0.71

Table 2. **Analysis of training strategy.** Evaluation of the end-point error (EPE) of a ResNet-based model after completing the entire training schedule. The cost volume is removed starting at the dataset mentioned in the "cut-off" column. The "Never" row denotes a model where the cost volume was never removed. Note that the training split of Sintel is part of the training, while Spring is not seen during training and, therefore, better shows the generalization ability of each model.

FlowNet training split [9] of Sintel [5], KITTI [35, 36], and HD1K [28]. We decided to use this training protocol because it includes many different datasets, showcasing a wide variety of motions. Ablation studies in RAFT [50] and SEA-RAFT have already shown that each part of this training stage improves the accuracies of their resulting models.

As this training strategy involves changing the datasets during training multiple times, a natural choice for the cut-off point is between swapping out the training datasets. Table 2 shows that by just varying the cut-off point, we can influence the EPE on Spring by 0.18 pixels, and we conclude that it is best to remove the cost volume before the training on FlyingThings starts.

ReCoVEr backbones. Until this point, we only evaluated the ResNet-34 context network proposed by SEA-RAFT, but in principle, every neural network that is capable of processing input frames and regressing dense features can be used as a context network. Currently, many optical flow methods utilize smaller ResNets as context networks [22, 25, 50, 52] because they offer a good trade-off between accuracy and compute complexity. Since we can remove the cost volume (during finetuning and inference), which is responsible for the largest amount of compute, this allows for the usage of more complex networks like ConvNeXt [29] while still being faster than today's methods.

Specifically, we explore three different architectures for the context networks while keeping the feature and refinement network as proposed by SEA-RAFT. All of our context networks take stacked input frames as input and return feature maps at $1/8$ of the input resolution. Our *ReCoVEr-RN* model uses the first three residual blocks of a ResNet-34 as the context network. Thereby, the resulting network is equivalent to the SEA-RAFT-M architecture. Our second model, *ReCoVEr-CX*, is based on a ConvNeXt-t [29] where we replace the last two downsampling-convolutions by convolutions of the same kernel size with stride 1 to create feature maps at the required spatial dimensions. The third model, *ReCoVEr-MN*, utilizes a slightly modified version of the MobileNetV3-L [15] encoder. We replaced the stride of the 13 th block by 1 to prevent the encoder from downsampling our features to $1/16$ th of the input resolution.

Context network	Sintel (val.)		Spring	FLOPS	memory
	Clean	Final			
MobileNetV3-L	0.81	0.90	0.99	**0.86T**	**0.49GB**
ResNet-34	0.80	0.91	0.68	1.74T	0.93GB
ConvNeXt-t	**0.36**	**0.42**	**0.51**	2.65T	1.24GB

Table 3. **Analysis of different backbones.** A comparison of EPEs reached by different context networks shows that the ConvNeXt-based network performs best on Sintel and Spring when all models are trained on the same data. The accuracies of ResNet and MobileNetV3 are almost identical on Sintel. The FLOPS and memory reported refer to an input frame resolution of 1920×1080.

As shown in Tab. 3, our three proposed models offer different trade-offs: ConvNeXt is known to be very accurate, but expensive to compute [29], MobileNetV3 is a backbone with a very small memory footprint to optimize it for mobile devices, but compared to other CNNs its accuracy is limited [15], and ResNet generalizes very well across different computer vision tasks and usually offers a good trade-off between accuracy and computational efficiency [12].

4. Results

To evaluate the performance of our models in more detail, we compare our networks against state-of-the-art methods like SEA-RAFT [52] and more specialized versions of RAFT [50], like GMA [25] and GMFlow [56].

Training details. We use the mixture of Laplace loss function introduced by SEA-RAFT for all of our trainings and a linear one-cycle learning rate scheduler [44] with 6k iterations of warmup time and an AdamW optimizer [30]. For a fair comparison, all of our trainings are done on 8 NVIDIA RTX 6000 Ada (48 GB) GPUs for the exact same number of iterations, and we do not utilize any form of early stopping or model selection from the different states obtained during the training. We always report the accuracies of the models obtained after the last training step.

Evaluation details. Comparing the accuracies of existing optical flow methods in a comparable and fair setting is tricky as training and evaluation datasets have changed over time. Newer approaches, like SEA-RAFT, pre-train on the rigid motion of the TartanAir dataset before training on more traditional methods as this was shown to improve accuracies [52]. Further, datasets like Sintel [5] and KITTI [35, 36] do not have an official validation split, and

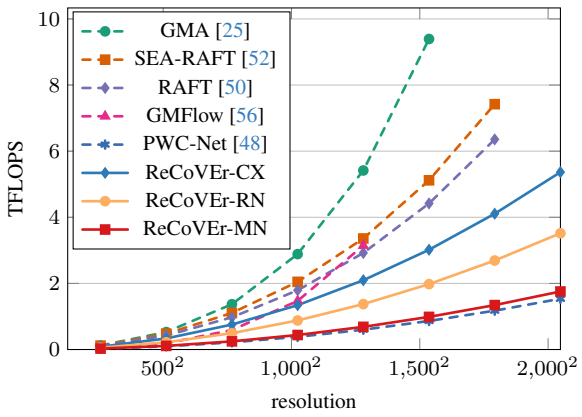

Figure 4. **Comparison of the required number of floating point operations (FLOPS)** for representative optical flow estimators and our models at various resolutions. Missing data points are due to out-of-memory errors.

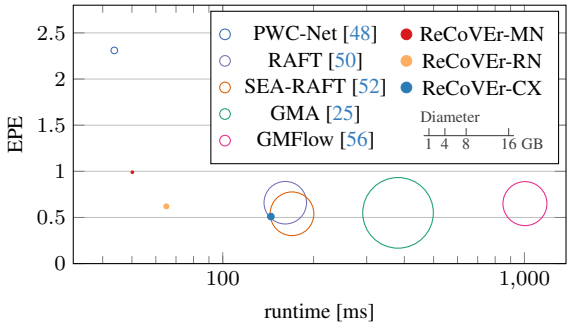

Figure 5. **Trade-offs** between runtime, memory usage, and accuracies for different models on Spring [33] in full resolution.

consequently, the validation splits used by different methods differ, making a fair comparison on these datasets impossible since some methods have seen the evaluation data during training. To overcome this, we evaluate all methods on the Spring dataset [33] without ever training any method on this dataset. The Spring dataset also has additional annotated regions in each frame with attributes like level of detail or rigidity of motions, allowing for further insights. We use the endpoint error (EPE) [1, 38] for evaluating accuracy.

We use a batch size of 1 and FP32 precision for measuring runtime and memory allocations. We average the runtime over 100 iterations for all runtime measurements and perform 10 warmup iterations that are not part of the measurements. The resolution reported in our figures is directly used as input sizes for all methods without applying additional down- and upsampling strategies for a fairer comparison of methods, as these kinds of strategies are not model-specific and could be applied for every model. We show the effects of using these strategies in the supplemental materials.

We compare our proposed models to GMA [25], GMFlow [56], and SEA-RAFT [52], because all of them are

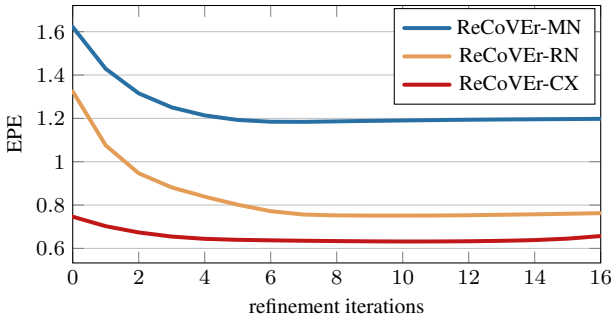

Figure 6. **Evaluation of the effect of varying the number of refinement iterations** in our models on the resulting accuracy on every 20th training frame of Spring. In the case of 0 iterations, the refinement network is not used, and the flow prediction of the context network is upsampled and used as a prediction.

Method	Middlebury [2]	Monkaa [32]		Spring [33]
		Clean	Final	
SEA-RAFT [52]	0.34	**1.69**	**2.41**	0.54
ReCoVEr-MN	0.60	2.88	3.03	0.99
ReCoVEr-RN	0.36	2.16	2.77	0.62
ReCoVEr-CX	**0.33**	1.71	2.46	**0.51**

Table 4. **Out-of-domain evaluations** of EPE on datasets that were not used during training without any finetuning.

extensions of RAFT where each method tackles a different shortcoming of RAFT. To clearly show the improvement of each of these methods over RAFT, we also included the original RAFT [50] model. Additionally, we include PWC-Net [47, 48] in our comparisons as it is very fast and memory efficient and, therefore, commonly used in downstream tasks as a building block of other models.

4.1. Quantitative results

Analysis of refinement. Since the number of iterations in the refinement network is a hyperparameter, we also evaluate different numbers of iterations, as shown in Fig. 6. The best accuracy for all of our models is reached between 7 and 9 iterations on our subset of Spring. Since the changes after 4 iterations are minor for all of our models except for ResNet, we use only 4 iterations for our MobileNetV3 and ConvNeXt-based models for all subsequent experiments and 8 iterations for the ResNet-based model because the runtime tends to increase linearly with the number of refinement iterations [52]. The number of refinement iterations can be increased without retraining the model.

Computational comparison. We compare the computational requirements of our models to other representative methods in Figs. 4 and 5. While the complexity of our three models differs quite a lot, they still require fewer operations than most other existing methods. Unsurprisingly,

Method	Sintel	TartanAir	total	low-detail	high-detail	matched	unmatched	rigid	non-rigid	s0-10	s10-40	s40+	time[ms]	memory[GB]
PWC-Net [47, 48]	✗	✗	2.31	2.27	10.84	2.13	11.28	2.10	4.45	1.83	3.16	15.87	**43.72**	1.25
RAFT [50]	✓	✗	0.66	0.61	9.66	0.54	6.55	0.24	4.91	0.18	2.85	10.83	161.19	8.00
GMFlow [56]	✓	✗	0.65	0.61	8.93	0.54	5.99	0.41	3.04	0.36	1.36	**8.28**	1004.23	8.28
GMA [25]	✓	✗	0.55	0.50	11.35	0.45	**5.97**	**0.20**	4.13	**0.16**	1.82	10.33	381.21	13.29
SEA-RAFT [52]	✓	✓	0.54	0.50	**8.76**	0.40	7.25	0.31	**2.90**	0.20	**0.95**	10.56	169.72	8.22
ReCoVEr-MN	✓	✓	0.99	0.93	14.36	0.82	9.48	0.54	5.57	0.28	1.95	21.96	50.16	**0.49**
ReCoVEr-RN	✓	✓	0.62	0.59	9.48	0.49	7.77	0.34	3.60	0.18	1.24	13.78	65.03	0.93
ReCoVEr-CX	✓	✓	**0.51**	**0.47**	9.10	**0.39**	6.51	0.24	3.28	**0.16**	1.08	10.80	144.41	1.24

Table 5. **Comparison of the EPE** of different methods on the different annotated regions of the Spring training split. None of the methods were fine-tuned on Spring, and all of them were at least trained on FlyingChairs and FlyingThings, but since the training schedules changed over time, we marked the methods that were trained using additional data from TartanAir or Sintel. All evaluated methods received the full-resolution frames as inputs without any resizing. All our models are faster (inference) and smaller than SEA-RAFT.

our ReCoVEr-CX model requires the most operations, followed by ReCoVEr-RN and ReCoVEr-MN. As can be seen, the number of computations for almost all existing methods tends to rise very quickly. This is due to the growing size of their cost volumes (*cf*. Sec. 4.1), and since our models do not have any cost volumes after training, their growth in complexity is less steep, which allows for higher resolution images being processed. The only model we evaluated that requires fewer computations than any of our models is PWC-Net. Its requirements are close to our fastest model, ReCoVEr-MN, but as we show in Tab. 5 and Fig. 5, our model is much more accurate at roughly the same cost. We observe similar trends for the inference time and memory requirements of the different models as for the number of floating-point operations. Further breakdowns of the different complexities can be found in the supplemental material.

Generalization capabilities. To quantify the generalization capabilities of our proposed methods, we evaluate SEA-RAFT and our methods on three different datasets that were not used during training. As shown in Tab. 4, we find no significant differences between SEA-RAFT and our best method regarding the achieved accuracies.

Comparison to other methods. When comparing our models in Tab. 5 to other representative methods that were proposed over the years, we find that our largest model, ReCoVEr-CX, is the best or second-best model for almost all accuracy measures, and overall best regarding total EPE. When comparing the accuracy of ReCoVEr-CX to SEA-RAFT for motions that are larger than 40 pixels, we can see that our EPE is only 0.24 higher for these regions, which equals less than $0.6\,\%$ of the entire motion magnitude. This shows that even though our networks do not have cost volumes anymore, modern convolutional networks are capable of having mostly the same accuracies for small and large motions, which differs from findings in the early works of FlowNet2 [21] and FlowNet [9], showing the benefit of recent CNN innovations. While ReCoVEr-MN and ReCoVEr-RN do not reach state-of-the-art performance, they are competitive with existing models while being faster. *E.g*., ReCoVEr-RN has an accuracy that is very similar to that of RAFT, while being $2.5\times$ faster and using less than $1/8$ th of the memory. ReCoVEr-MN is almost as fast as PWC-Net but more than twice as accurate. Generally speaking, w.r.t. to comparable computational budget, our ReCoVEr training strategy achieves much more accurate flow predictions than existing methods.

4.2. Qualitative results

Quantitatively evaluating optical flow on natural frames captured at high resolutions is impossible as no annotated, high-resolution dataset is currently available. In Fig. 7, we therefore qualitatively compare the predictions of SEA-RAFT to the predictions of our models for two natural sequences from the DAVIS [40] dataset captured at a resolution of 1920×1080. We find that both architectures using ResNet as a context network, ReCoVEr-RN and SEA-RAFT, have comparable prediction qualities and very similar failure cases where, *e.g*., the motion of the bike wheel is predicted very similarly. In contrast, ReCoVEr-CX shows much sharper motion boundaries and is better at separating moving foreground objects from the background motion, especially visible at the wheels and the clear separation of the foreground and background motion, while still being faster and requiring less memory than SEA-RAFT. The overall quality of ReCoVEr-MN prediction is noticeably lower than the others, but given the very low compute requirements, the overall quality of the results is still good, and when comparing it to the predictions of PWC-Net in Fig. 1, ReCoVEr-MN shows an overall higher prediction quality at comparably low runtimes.

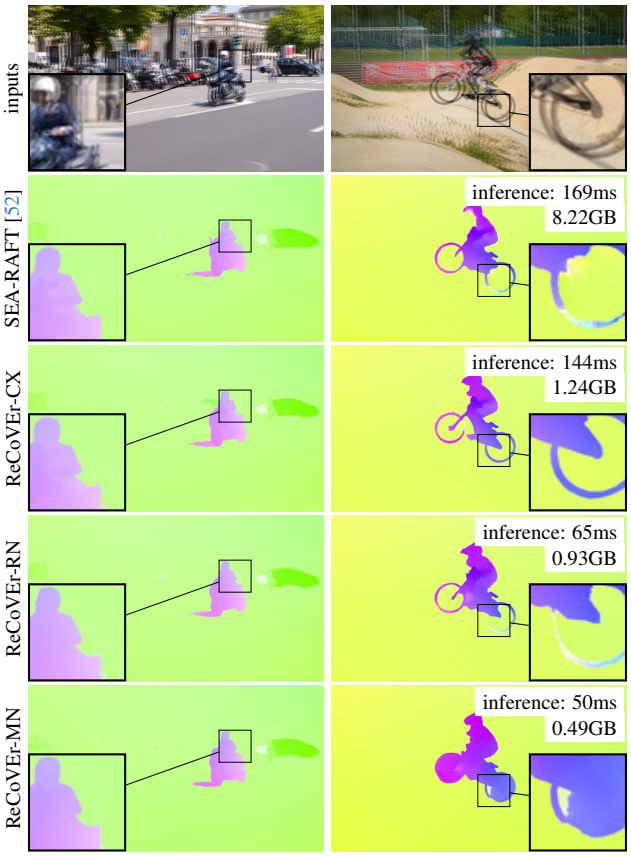

Figure 7. **Qualitative comparison** of our method compared to SEA-RAFT on Full HD frames from the DAVIS dataset [40]. Note that our models use only between $1/6$ to $1/20$th of the memory required by SEA-RAFT and are 1.2 to 3.4× faster.

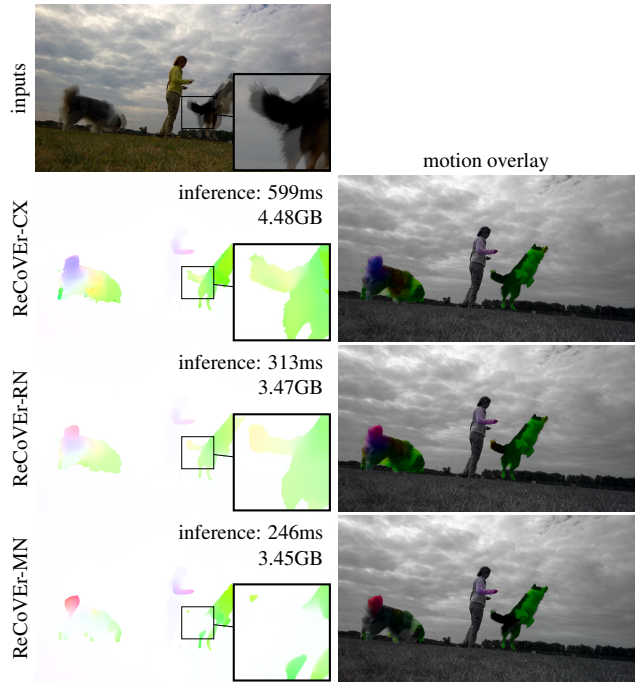

Figure 8. **Qualitative result** for the prediction on 4K frames from the DAVIS dataset [40]. Overlaying the prediction with the frames shows that the predictions are well-aligned with the input frames. The motion overlay shows a blend between the prediction and input frame where the color channel in LCh color space is taken from the optical flow visualization while the other channels are taken from the input frame.

Even though our models are only trained on frames up to a resolution of 960×432, we find that our models generalize to much higher resolutions, as can be seen in Fig. 8 where we predict the optical flow on an input at a resolution of 3840×2160 pixels. Similar to our findings for Fig. 7, we observe the highest prediction quality for ReCoVEr-CX. ReCoVEr-MN shows some significant shortcomings when processing these high-resolution images, as can be seen by the holes in the prediction where no foreground motion is detected due to a lack of texture in that region.

4.3. Limitations

As shown in our quantitative and qualitative results, our fastest architecture, ReCoVEr-MN, starts failing for larger motions. The qualitative results show that ReCoVEr-MN in this case either fails to detect motions entirely or predicts motions going in the wrong direction. Further, the computational advantages of MobileNetV3 over ResNet start to diminish at high resolutions. Therefore, we conclude that the ReCoVEr-MN architecture should only be used for lower-resolution inputs in settings where a low memory footprint is the most important criterion. Further, we only explore the effect of replacing the context network and do not evaluate the effect other refinement network architectures can have on the performance. Figure 6 even shows that our refinement yields barely any increase in accuracy for ReCoVEr-CX and, therefore, removing the refinement module offers possibilities for further improvements.

5. Conclusion

In this work, we have analyzed and re-evaluated the role of cost volumes in optical flow estimators, taking into account the progress made in modern convolution network backbones and the availability of current datasets. We found that cost volumes are necessary initially during training, but with the right training strategy, they are not needed anymore during inference. To demonstrate optical flow methods without cost volumes, we introduced a simple yet highly effective training strategy where the cost volume is removed during the training process. We utilized this training strategy to create three different models that can cover a wide range of applications covering state-of-the-art accuracies, low compute times, and low memory footprints.

Acknowledgments. This work was funded by the Hessian Ministry of Science and the Arts (HMWK) through the project "The Third Wave of Artificial Intelligence – 3AI". The work was further supported by the Deutsche Forschungsgemeinschaft (German Research Foundation, DFG) – project number 529680848 and under Germany's Excellence Strategy (EXC 3057/1 "Reasonable Artificial Intelligence", Project No. 533677015). Stefan Roth acknowledges support by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 866008).

References

[1] Simon Baker, Daniel Scharstein, J. P. Lewis, Stefan Roth, Michael J. Black, and Richard Szeliski. A database and evaluation methodology for optical flow. Technical Report MSR-TR-2009-179, Microsoft Research, Redmond, Washington, 2009. 6

[2] Simon Baker, Daniel Scharstein, J. P. Lewis, Stefan Roth, Michael J. Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *IJCV*, 92(1):1–31, 2011. 6

[3] Michael J. Black and P. Anandan. Robust dynamic motion estimation over time. In *CVPR*, pages 296–302, 1991. 2

[4] Karlis Martins Briedis, Markus Gross, and Christopher Schroers. Efficient correlation volume sampling for ultra-high-resolution optical flow estimation. *CoRR*, abs/2505.16942, 2025. 3

[5] Daniel J. Butler, Jonas Wulff, Garrett B. Stanley, and Michael J. Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, pages 611–625, 2012. 4, 5, I, II, III

[6] Jiong Dong, Kaoru Ota, and Mianxiong Dong. Video frame interpolation: A comprehensive survey. *ACM Multimedia*, pages 78:1–78:31, 2023. 1

[7] Qiaole Dong and Yanwei Fu. MemFlow: Optical flow estimation and prediction with memory. In *CVPR*, pages 19068–19078, 2024. 2

[8] Qiaole Dong, Chenjie Cao, and Yanwei Fu. Rethinking optical flow from geometric matching consistent perspective. In *CVPR*, pages 1337–1347, 2023. 2

[9] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Häusser, Caner Hazırbaş, Vladimir Golkov, Patrick v. d. Smagt, Daniel Cremers, and Thomas Brox. FlowNet: Learning optical flow with convolutional networks. In *ICCV*, pages 2758–2766, 2015. 2, 3, 4, 5, 7

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 2, 3

[11] Chen Gao, Ayush Saraf, Jia-Bin Huang, and Johannes Kopf. Flow-edge guided video completion. In *ECCV*, pages 713–729, 2020. 1

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 3, 5

[13] Robin Hesse, Simone Schaub-Meyer, and Stefan Roth. Content-adaptive downsampling in convolutional neural networks. In *CVPRW*, pages 4543–4552, 2023. 3

[14] Berthold K. P. Horn and Brian G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1–3):185–203, 1981. 2

[15] Andrew Howard, Ruoming Pang, Hartwig Adam, Quoc V. Le, Mark Sandler, Bo Chen, Weijun Wang, Liang-Chieh Chen, Mingxing Tan, Grace Chu, Vijay Vasudevan, and Yukun Zhu. Searching for MobileNetV3. In *ICCV*, pages 1314–1324, 2019. 3, 5

[16] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. MobileNets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017. 3

[17] Zhaoyang Huang, Xiaoyu Shi, Chao Zhang, Qiang Wang, Ka Chun Cheung, Hongwei Qin, Jifeng Dai, and Hongsheng Li. FlowFormer: A transformer architecture for optical flow. In *ECCV*, pages 668–685, 2022. 2, 3

[18] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. LiteFlowNet: A lightweight convolutional neural network for optical flow estimation. In *CVPR*, pages 8981–8989, 2018. 2

[19] Junhwa Hur and Stefan Roth. Iterative residual refinement for joint optical flow and occlusion estimation. In *CVPR*, pages 5747–5756, 2019. 2

[20] Forrest N. Iandola, Matthew W. Moskewicz, Khalid Ashraf, Song Han, William J. Dally, and Kurt Keutzer. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <1MB model size. *CoRR*, abs/1602.07360, 2016. 3

[21] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. FlowNet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*, pages 1647–1655, 2017. 2, 3, 4, 7

[22] Azin Jahedi, Lukas Mehl, Marc Rivinius, and Andrés Bruhn. Multi-Scale RAFT: Combining hierarchical concepts for learning-based optical flow estimation. In *ICIP*, pages 1236–1240, 2022. 5

[23] Azin Jahedi, Maximilian Luz, Marc Rivinius, and Andrés Bruhn. CCMR: high resolution optical flow estimation via coarse-to-fine context-guided motion reasoning. In *WACV*, pages 6885–6894, 2024. 3

[24] Azin Jahedi, Maximilian Luz, Marc Rivinius, Lukas Mehl, and Andrés Bruhn. MS-RAFT+: high resolution multi-scale RAFT. *IJCV*, 132(5):1835–1856, 2024. 3

[25] Shihao Jiang, Dylan Campbell, Yao Lu, Hongdong Li, and Richard I. Hartley. Learning to estimate hidden motions with global motion aggregation. In *ICCV*, pages 9752–9761, 2021. 3, 5, 6, 7, II

[26] Shihao Jiang, Yao Lu, Hongdong Li, and Richard Hartley. Learning optical flow from a few matches. In *CVPR*, pages 16592–16600, 2021. 3

[27] Lei Ke, Yu-Wing Tai, and Chi-Keung Tang. Occlusion-aware video object inpainting. In *ICCV*, pages 14448–14458, 2021. 1

[28] Daniel Kondermann, Rahul Nair, Katrin Honauer, Karsten Krispin, Jonas Andrulis, Alexander Brock, Burkhard Güssefeld, Mohsen Rahimimoghaddam, Sabine Hofmann, Claus Brenner, and Bernd Jähne. The HCI benchmark suite: Stereo and flow ground truth with uncertainties for urban autonomous driving. In *CVPRW*, pages 19–28, 2016. 5

[29] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A ConvNet for the 2020s. In *CVPR*, pages 11966–11976, 2022. 3, 5

[30] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 5

[31] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI*, pages 674–679, 1981. 2

[32] 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. In *CVPR*, pages 4040–4048, 2016. 4, 6, II

[33] Lukas Mehl, Jenny Schmalfuss, Azin Jahedi, Yaroslava Nalivayko, and Andrés Bruhn. Spring: A high-resolution high-detail dataset and benchmark for scene flow, optical flow and stereo. In *CVPR*, pages 4981–4991, 2023. 4, 6, I, II, III

[34] Sachin Mehta, Mohammad Rastegari, Anat Caspi, Linda G. Shapiro, and Hannaneh Hajishirzi. ESPNet: Efficient spatial pyramid of dilated convolutions for semantic segmentation. In *ECCV*, pages 561–580, 2018. 3

[35] Moritz Menze, Christian Heipke, and Andreas Geiger. Joint 3D estimation of vehicles and scene flow. In *ISPRS Workshop on Image Sequence Analysis (ISA)*, 2015. 5, I, II, III

[36] Moritz Menze, Christian Heipke, and Andreas Geiger. Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS)*, 2018. 5, I, II, III

[37] Simon Niklaus and Feng Liu. Softmax splatting for video frame interpolation. In *CVPR*, pages 5436–5445, 2020. 1

[38] Michael Otte and Hans-Hellmut Nagel. Optical flow estimation: Advances and comparisons. In *ECCV*, pages 51–60, 1994. 6

[39] Nils Papenberg, Andrés Bruhn, Thomas Brox, Stephan Didas, and Joachim Weickert. Highly accurate optic flow computation with theoretically justified warping. *IJCV*, 67(2): 141–158, 2006. 2

[40] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alexander Sorkine-Hornung, and Luc Van Gool. The 2017 DAVIS challenge on video object segmentation. *CoRR*, abs/1704.00675, 2017. 7, 8, I, II, III

[41] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *CVPR*, 2017. 2

[42] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. MobileNetV2: Inverted residuals and linear bottlenecks. In *CVPR*, pages 4510–4520, 2018. 3

[43] Hyeonjun Sim, Jihyong Oh, and Munchurl Kim. XVFI: eXtreme video frame interpolation. In *ICCV*, pages 14469–14478, 2021. 1

[44] Leslie N. Smith and Nicholay Topin. Super-Convergence: Very fast training of residual networks using large learning rates. *CoRR*, abs/1708.07120, 2017. 5

[45] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 15 (1):1929–1958, 2014. 4

[46] Xiuchao Sui, Shaohua Li, Xue Geng, Yan Wu, Xinxing Xu, Yong Liu, Rick Siow Mong Goh, and Hongyuan Zhu. CRAFT: Cross-attentional flow transformer for robust optical flow. In *CVPR*, pages 17581–17590, 2022. 3

[47] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *CVPR*, pages 8934–8943, 2018. 1, 2, 6, 7, II

[48] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Models matter, so does training: An empirical study of CNNs for optical flow estimation. *TPAMI*, 42(6):1408–1423, 2020. 1, 2, 6, 7, II

[49] Mingxing Tan and Quoc V. Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In *ICML*, pages 6105–6114, 2019. 3

[50] Zachary Teed and Jia Deng. RAFT: Recurrent all-pairs field transforms for optical flow. In *ECCV*, pages 402–419, 2020. 2, 3, 5, 6, 7, I, II

[51] Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian A. Scherer. TartanAir: A dataset to push the limits of visual SLAM. In *IROS*, pages 4909–4916, 2020. 4

[52] Yihan Wang, Lahav Lipson, and Jia Deng. SEA-RAFT: Simple, efficient, accurate RAFT for optical flow. In *ECCV*, pages 36–54, 2024. 1, 2, 3, 4, 5, 6, 7, 8, I, II, III

[53] Philippe Weinzaepfel, Thomas Lucas, Vincent Leroy, Yohann Cabon, Vaibhav Arora, Romain Brégier, Gabriela Csurka, Leonid Antsfeld, Boris Chidlovskii, and Jérôme Revaud. CroCo v2: Improved cross-view completion pre-training for stereo matching and optical flow. In *ICCV*, pages 17923–17934, 2023. 2, 3

[54] Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, pages 5987–5995, 2017. 3

[55] Haofei Xu, Jiaolong Yang, Jianfei Cai, Juyong Zhang, and Xin Tong. High-resolution optical flow from 1D attention and correlation. In *ICCV*, pages 10478–10487, 2021. 1, 3

[56] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. GMFlow: Learning optical flow via global matching. In *CVPR*, pages 8111–8120, 2022. 3, 5, 6, 7, II

[57] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, Fisher Yu, Dacheng Tao, and Andreas Geiger. Unifying flow, stereo and depth estimation. *TPAMI*, 45(11):13941–13958, 2023. 3

[58] Rui Xu, Xiaoxiao Li, Bolei Zhou, and Chen Change Loy. Deep flow-guided video inpainting. In *CVPR*, pages 3723–3732, 2019. 1

[59] Mufeng Yao, Jiaqi Wang, Jinlong Peng, Mingmin Chi, and Chao Liu. FOLT: Fast multiple object tracking from uav-captured videos based on optical flow. In *ACM*, pages 3375–3383, 2023. 1

[60] Guozhen Zhang, Yuhan Zhu, Haonan Wang, Youxin Chen, Gangshan Wu, and Limin Wang. Extracting motion and ap-

pearance via inter-frame attention for efficient video frame interpolation. In *CVPR*, pages 5682–5692, 2023. 1

[61] Yang Zhao, Gangwei Xu, and Gang Wu. Hybrid cost volume for memory-efficient optical flow. In *ACM Multimedia*, pages 8740–8749, 2024. 3

[62] Shangchen Zhou, Chongyi Li, Kelvin C. K. Chan, and Chen Change Loy. ProPainter: Improving propagation and transformer for video inpainting. In *ICCV*, pages 10443–10452, 2023. 1

Removing Cost Volumes from Optical Flow Estimators

Supplementary Material

A.1. Complexity

In the main paper, we only refer to FLOPS as a measure of complexity. However, a reduction of FLOPS does not necessarily lead to a reduction in compute time on currently available accelerators, mostly due to memory alignment issues. Since we only remove entire parts of the networks and do not introduce sparsity or similar, we find that the reduction in FLOPS is proportional to the reduction in runtime. For completeness, we show the runtimes for different resolutions in Fig. A.1.

Another limiting factor is often the amount of memory required for a single prediction. We evaluate the memory footprint of our method in Fig. A.2, and due to the missing cost volumes, we find a significant reduction in memory footprint. However, technically RAFT-style architectures never require every value of the cost volume to be available at the same time since the refinement network can only sample a certain number of values from the cost volume per iteration. Therefore, the required values could be calculated only when they are requested from the refinement module. This was also noticed and implemented by Teed and Deng in the original implementation of RAFT, but the disadvantage of this approach is a significant slowdown in processing speed, and therefore, we do not consider this approach in our work.

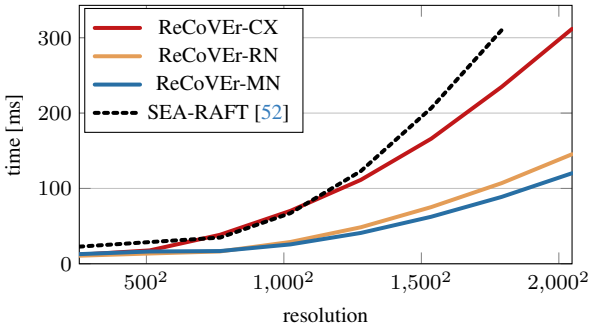

Figure A.1. **Runtime** of our methods and SEA-RAFT for different input resolutions.

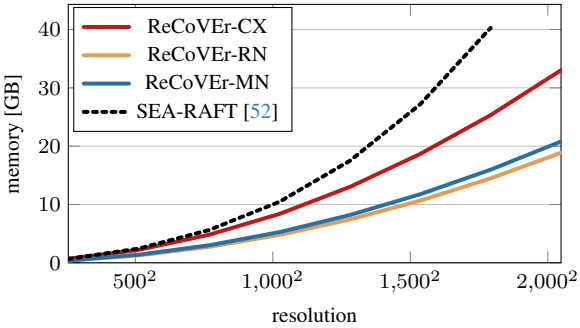

Figure A.2. **Memory** requirements of our methods and SEA-RAFT for different input resolutions.

A.2. Downsample-upsample strategies

Downsampling the inputs and bilinearly upsampling the resulting optical flow is another method to reduce the memory footprint and inference time, and is, *e.g.*, applied by SEA-RAFT on Full-HD frames to increase the computational efficiency for higher resolution inputs [52]. In our work, we did not apply this orthogonal strategy as it can be applied theoretically to all optical flow methods. Table A.1 shows that when evaluating on the Spring dataset, most methods achieve even higher accuracies using the downsample-upsample strategy, but the results on Sintel and Monkaa clearly show that the increase in accuracy is not persistent between datasets.

A.3. Additional technical details

Refinement network. The refinement network is implemented such that the sampling from the cost volume during the refinement can return an all-zero tensor instead of actually sampling from the cost volume. This simplifies the implementation of cutting away the feature network because by returning only zeros, the weights of the first layer of the refinement network that deal with this part of the input are not used since all of them are multiplied by zero. Theoretically, removing the affected weights from this layer completely would be possible. Still, this way of implementing the process is much easier, and the number of calculations needed to process the zero tensor is negligible compared to all other computations necessary for the optical flow prediction.

Training protocol. As described in Sec. 3.2, we mostly follow the training protocol proposed by SEA-RAFT [52]. The only difference is the composition of the TSKH dataset, where we do not include the validation split of Sintel. This allows us to fairly evaluate our methods on parts of the Sintel dataset for our analysis.

A.4. Qualitative examples

More qualitative examples, including samples from Sintel [5], Spring [33], and natural images taken from KITTI [35, 36] and DAVIS [40] can be found in Figs. A.3 and A.4.

Method	downsample	Sintel (val.) [5]		Monkaa [32]		Spring [33]	time[ms]	memory[GB]
		Clean	Final	Clean	Final			
PWC-Net [47, 48]	✗	3.22	3.66	4.19	4.56	2.31	43.72	1.25
	✓	5.27	5.95	5.16	5.34	3.83	12.30	0.36
RAFT [50]	✗	(0.74)	(1.19)	1.99	2.92	0.66	161.19	8.00
	✓	(1.38)	(2.07)	1.98	2.59	0.48	28.04	0.56
GMFlow [56]	✗	(0.76)	(1.11)	3.01	2.97	0.65	1004.23	8.28
	✓	(1.98)	(2.55)	2.08	3.59	0.95	99.44	1.47
GMA [25]	✗	(0.62)	(1.06)	1.75	2.63	0.55	381.21	13.29
	✓	(1.27)	(1.95)	1.90	2.57	0.46	46.88	0.92
SEA-RAFT [52]	✗	(0.43)	(0.58)	**1.69**	2.41	0.54	169.72	8.22
	✓	(1.22)	(2.06)	1.73	**2.09**	**0.41**	28.66	0.66
ReCoVEr-MN	✗	0.81	0.90	2.88	3.03	0.99	50.16	0.49
	✓	2.64	2.82	3.61	3.93	0.79	**12.12**	**0.28**
ReCoVEr-RN	✗	0.72	0.84	2.16	2.77	0.62	65.03	0.93
	✓	1.85	2.33	2.52	3.05	0.57	13.46	0.30
ReCoVEr-CX	✗	**0.36**	**0.42**	1.71	2.46	0.51	144.41	1.24
	✓	1.34	1.71	2.15	2.70	0.50	29.89	0.43

Table A.1. **Effect of downsampling** by a factor of 2× and bilinearly upsampling the resulting optical flow on different datasets. The time and memory refer to input frames of size 1920 × 1080. For completeness, we also show the accuracies achievable by methods that were also trained on the Sintel validation set, and we put these numbers in parentheses.

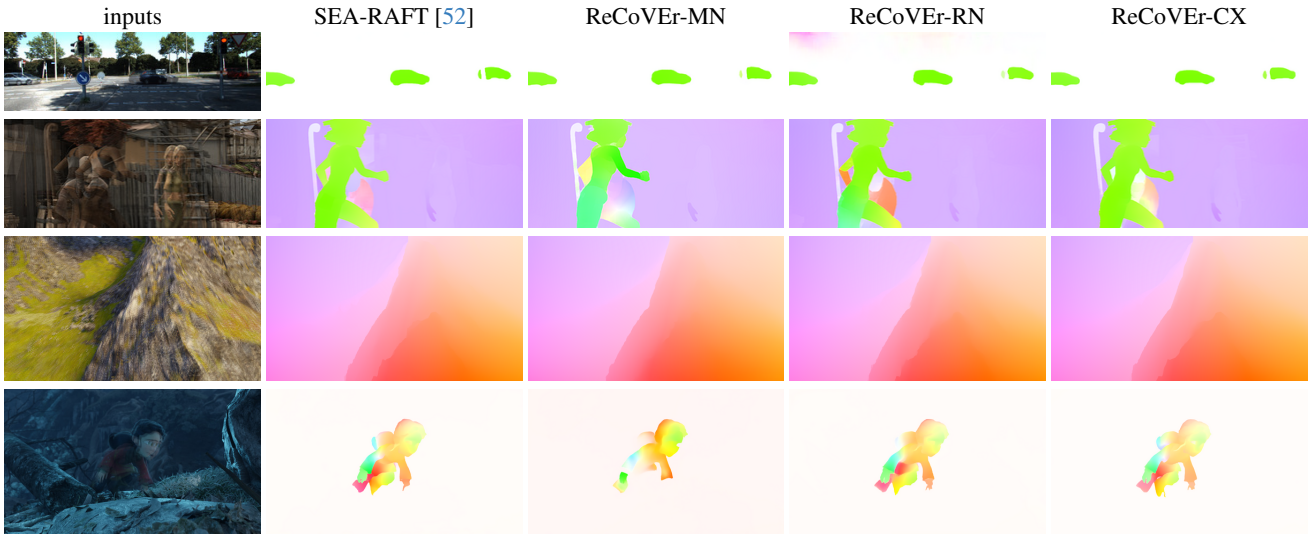

Figure A.3. More qualitative examples on various frames taken from DAVIS [40], KITTI [35, 36], Sintel [5], and Spring [33].

.

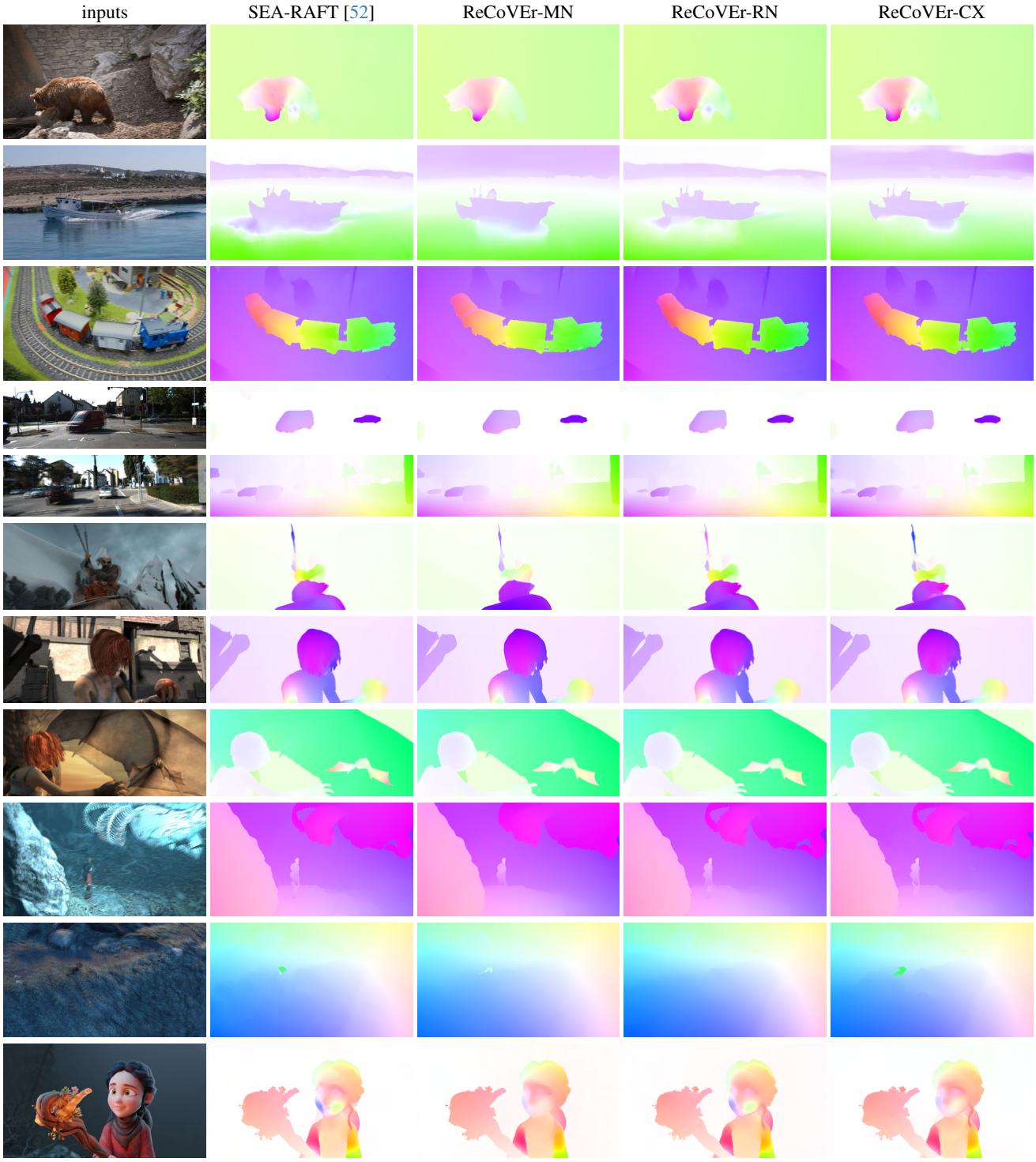

Figure A.4. More qualitative examples on various frames taken from DAVIS [40], KITTI [35, 36], Sintel [5], and Spring [33].