AnyFlow: Arbitrary Scale Optical Flow with Implicit Neural Representation

Hyunyoung Jung[1*] Zhuo Hui[2] Lei Luo[2] Haitao Yang[2]
Feng Liu[2†] Sungjoo Yoo[1] Rakesh Ranjan[2] Denis Demandolx[2]

[1]Seoul National University [2]Meta Reality Labs

Abstract

To apply optical flow in practice, it is often necessary to resize the input to smaller dimensions in order to reduce computational costs. However, downsizing inputs makes the estimation more challenging because objects and motion ranges become smaller. Even though recent approaches have demonstrated high-quality flow estimation, they tend to fail to accurately model small objects and precise boundaries when the input resolution is lowered, restricting their applicability to high-resolution inputs. In this paper, we introduce AnyFlow, a robust network that estimates accurate flow from images of various resolutions. By representing optical flow as a continuous coordinate-based representation, AnyFlow generates outputs at arbitrary scales from low-resolution inputs, demonstrating superior performance over prior works in capturing tiny objects with detail preservation on a wide range of scenes. We establish a new state-of-the-art performance of cross-dataset generalization on the KITTI dataset, while achieving comparable accuracy on the online benchmarks to other SOTA methods.

1. Introduction

Optical flow seeks to estimate per-pixel correspondences, characterized as the horizontal and vertical shift, from a pair of images. Specifically, it aims to identify the correspondence across pixels in different images, which is at the heart of numerous computer vision tasks such as video denoising [4, 33, 70], action recognition [55, 60, 62] and object tracking [11, 26, 46]. This is particularly challenging due to the fact that the scene geometry and object motion are combined into a single observation, making the inverse problem of estimating motion highly ill-posed.

A common assumption for enabling computationally tractable optical flow is to explicitly account for small motion in local neighborhood and incorporate additional prior to constrain the solution space, such as using total variation

[*]This work was done during the author's internship at Meta.
[†]Affiliated with Meta at the time of this work.

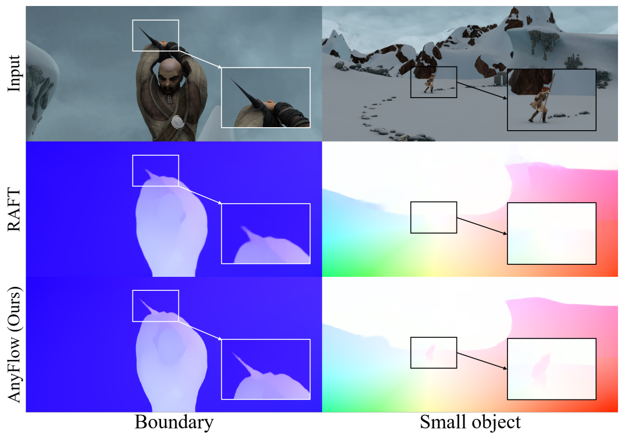

Figure 1. Predicting 50% downsized images on Sintel [5], we compare AnyFlow to RAFT [57]. AnyFlow shows clearer boundaries, accurate shapes, and better detection of small objects.

prior [15, 71] and smooth prior [2, 3, 51]. While effective, these techniques are significantly restricted by the assumption and do not generalize well for real-life scenes with sophisticated geometry and motions.

An alternative approach is motivated by the success of deep neural networks. Different from using handcrafted priors, learning-based methods [14, 67, 74] design end-to-end networks to regress the motion field. Sun et al. [53] use the coarse-to-fine approach to characterize the pixel-to-pixel mapping and several methods [20, 22, 44, 67] are developed along this line. The drawback of these approaches is that the accuracy of flow estimation is not only limited by the sampling but also has a critical dependence on a good initial solution because the pixel correspondence is fundamentally ambiguous. Tedd and Deng [57] resort to iterative update module on top of the correlation volumes, allowing better estimates in tiny and fast-moving objects. Indeed the success of this model relies on having a correlation volume that is high quality in characterizing the feature correspondence for sub-pixel level. Inspired by its success, follow-up methods [23, 25, 36, 50, 52] have further improved it in multiple ways. One popular direction is to introduce attention mechanisms [59] for efficiency [64, 72], occlusion handling [24], and large displacements [49]. As attention has been found to be effective in estimating long-range dependencies, Xu et al. [65] and Huang et al. [19] adopt Vision Transformers [34] to perform global matching and demonstrate their effectiveness. However, these methods naturally become less effective for low-resolution images, where inaccuracy is introduced when computing the correlation volumes and the iterative refinements. They tend to fail in accurately modeling small objects and precise boundary when the input resolution is lowered. This inherently limits the applicability for mobile devices in particular, where resizing the input to small size is often necessary to reduce the computational cost.

In this paper, we propose AnyFlow, a novel method that is agnostic to the resolution of images. The key insight behind AnyFlow is the use of implicit neural representation (INR) for flow estimation. INRs, such as LIIF [8], have been demonstrated to be effective for image super-resolution, as they model an image as a continuous function without limiting the output to a fixed resolution. This enables image generation at arbitrary scales. Inspired by this capability, we design a continuous coordinate-based flow upsampler to infer flow at any desired scale. In addition, we propose a novel warping scheme utilizing multi-scale feature maps together with dynamic correlation lookup to generalize well on diverse shapes of input. As the proposed methods produce a synergistic effect, we demonstrate superior performance for tiny objects and detail preservation on a wide range of scenes involving complex geometry and motions. Specifically, we verify the robustness for the input with low resolution. Fig. 1 showcases our technique against RAFT [57]. Our contributions are summarized as follows:

- We introduce AnyFlow, a novel network to produce high quality flow estimation for arbitrary size of image.
- We present the methods to utilize multi-scale feature maps and search optimal correspondence within predicted range, enabling robust estimation in wide ranges of motion types.
- We demonstrate strong performance on cross-dataset generalization by achieving more than 25% of error reduction with only 0.1M additional parameters from the baseline and rank 1st on the KITTI dataset.

Together, our contributions provide, for the first time, an approach for *arbitrary* size of input, which significantly improves the quality for low-resolution images and extends the applicability for portable devices.

2. Related Work

Physical based flow method. Estimating motion from images has been a long-standing goal in computer vision. Under the common assumption that the brightness across consecutive images remains constant, it has been addressed by utilizing a smooth prior [2, 3, 17, 51] and noise estimates by incorporating variational prior [15, 71]. However, the induced physical-based priors are inherently limited in their ability to provide precise approximation to the real-world.

Learning based flow method. Deep neural networks have shown unprecedented power in solving ill-posed problems. In the context of optical flow, Dosovitskiy et al. [14] first train a network end-to-end, utilizing correlations between deep feature vectors. To capture large motions, coarse-to-fine methods [16,20,53,68,74] estimate flow from low resolution and refine at high resolution. Stacked networks in series with warping [21] and pyramidal feature networks [44, 67] are adopted to refine flow field. Since the initial flow is estimated at the coarse stage with low-resolution, they mainly suffer from difficulties in detecting small and fast-moving objects. Recently, Tedd and Deng [57] extend the idea by first introducing the recurrent layer to refine a single flow field, obviating the need for multi-resolution search on feature matching. Based on its success, further improvements were made by leveraging patch-wise estimation [36, 76], sparsity [25], graph reasoning [37] and development of training strategies [23, 50, 52]. One of the recent popular directions is to utilize attention mechanism [59] to improve efficiency [64, 72], resolve occlusion [24], and capture large displacements [49]. Since self-attention proves useful in estimating long-range dependency, vision transformer [34] was adopted for optical flow [19, 65] to perform global matching and demonstrated effectiveness. While these methods enable high-quality estimation, the prediction for tiny objects is still a challenge due to the fact that their underlying architectures estimate the flow in fixed accuracy, making it sensitive to resolution changes. In contrast, our method processes images with arbitrary size without affecting the quality in flow estimation.

High-resolution flow from low-resolution image. High-resolution (HR) optical flow from low-resolution (LR) images is desired for various tasks. For example, in video super-resolution, temporal consistency of HR frames is crucial and optical flow [6, 56, 58] is required to achieve it. Sajjadi et al. [45] use a recurrent approach to reconstruct HR frame, while maintaining temporal consistency by bilinearly interpolating LR flow. Wang et al. [61] propose a coarse-to-fine network with motion compensation to infer HR flows from LR images. However, the interpolation incurs artifacts and CNN-based coarse-to-fine approach limits the resolution to fixed scale. In addition, such task requires precise flow estimation from LR images, since inaccurate motion leads to severe distortion in HR reconstruction. We demonstrate robust estimation from LR images and resize

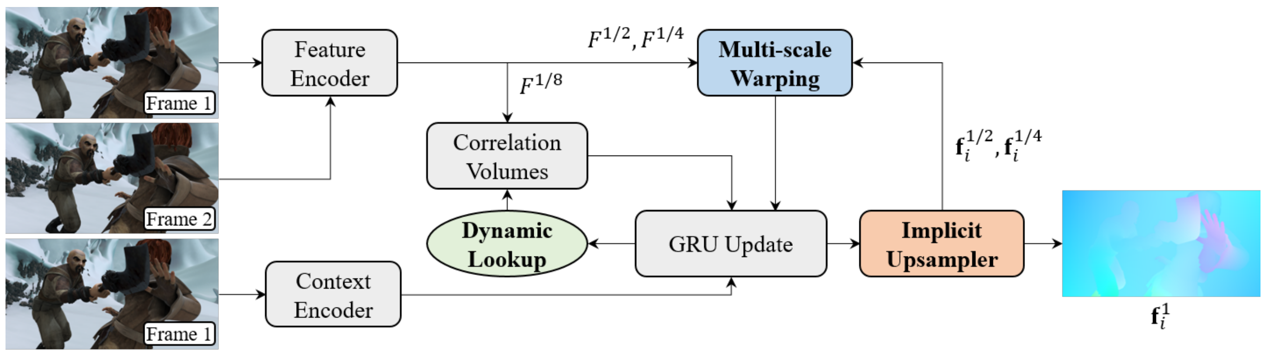

Figure 2. **Overall architecture of AnyFlow.** AnyFlow is built upon the RAFT [57] framework: feature extraction, all-pairs correlation volumes, and GRU update, as depicted by the gray blocks. After building the correlation volumes, the network performs a dynamic lookup for sampling values within an input-dependent local grid. The neural implicit flow upsampler rescales the output flow to arbitrary scales and generates multi-scale optical flows ($\mathbf{f}_i^{1/2}, \mathbf{f}_i^{1/4}$) for the multi-scale feature warping module, which in turn warps feature maps from the feature encoder. The GRU Update combines these elements with its output feature M to produce a residual flow $\Delta\mathbf{f} \in \mathbb{R}^{H/8\times W/8\times 2}$.

flow into arbitrary resolution, extending our applicability to such downstream task.

Implicit neural representation. Implicit neural representation (INR), parameterized as an MLP, maps a spatial coordinate to output signal. It has been widely adopted to represent learning-based 3D geometry [1, 7, 32, 40, 41, 43, 63, 69, 73] and several studies explore it in 2D tasks as well such as image and shape generation [10, 27], image fitting [48] and semantic segmentation [18]. Recently, for super-resolution [8, 9, 66], Chen et al. [8] propose LIIF that learns continuous image representation from local latent code. Different from the previous methods [13, 29–31] that upsample image into a fixed set of scales, LIIF models an image as a continuous function and the output is not limited to the fixed resolution, enabling image generation at arbitrary scales. Based on it, Chen et al. [9] design an implicit network to learn intermediate motion fields and maintain temporal consistency for continuous video super-resolution. However, the motion network does not perform precise estimation and has difficulty handling large motion. Different from them, we re-design LIIF as a novel flow upsampler and integrate it into the existing optical flow framework [57], thereby enabling us to generate optical flow in arbitrary scales with robust estimation in diverse scenarios.

3. Arbitrary Scale Optical Flow

Problem statement. Given a pair of input images, I_1 and $I_2 \in \mathbb{R}^{H\times W\times 3}$, optical flow seeks to solve per-pixel motion fields $\mathbf{f} \in \mathbb{R}^{H\times W\times 2}$, which account for horizontal and vertical motion. Our goal is to estimate the flow as an implicit function of an input image pair for arbitrary scales as

$$\mathbf{f}^s = \mathcal{I}(I_1, I_2),$$

where $\mathbf{f}^s \in \mathbb{R}^{sH\times sW\times 2}$ for scale factor s, and $\mathcal{I}$ denotes the underlying function we expect to model via the network.

Overview. We design AnyFlow based on the general stages of RAFT [57]. We first extract multi-scale feature maps through the feature encoder, compute correlation volumes and update the flow field by accumulating the residual flow. Built upon it, we propose Neural Implicit Flow Upsampler (Sec. 3.1) to rescale the accumulated flow into arbitrary scale, including restoring the original resolution from downscaled inputs. Our proposed Multi-scale Feature Warping module (Sec. 3.2) enables utilization of high-resolution representations for flow prediction. Finally, we propose Dynamic Lookup (Sec. 3.3) to search correspondence from the correlation volumes depending on input images. The overall architecture is illustrated in Fig. 2 and we describe the methods in detail in the following sections.

3.1. Neural Implicit Flow Upsampler

Iterative refinement. Following RAFT, the network first produces $1/8$-sized residual flow $\Delta\mathbf{f} \in \mathbb{R}^{H/8\times W/8\times 2}$ and accumulates it by GRU iterations. After i^{th} update, the accumulated flow $\mathbf{f}_i \in \mathbb{R}^{H/8\times W/8\times 2}$ is computed as: $\mathbf{f}_i = \Delta\mathbf{f}_i + \mathbf{f}_{i-1}$. Note that, in the following sections, the flow *without superscript*, i.e., $\mathbf{f}_i$, always denotes the accumulated flow in $1/8$ size of the input resolution .

Local implicit image function. We utilize LIIF [8] to generate an arbitrary scale of output from a fixed input signal. In the LIIF representation, output signal o at 2D continuous coordinate $x = (u, v)$ is estimated from a decoding function f_θ, parameterized as an MLP as: $o = f_\theta(z, x)$. Here, z is a feature vector sampled from the 2D feature map M at location x, and M is extracted from input signal I through encoding function $\mathcal{E}$. That is, the decoding function f_θ maps each coordinate x to the output signal, depending on the feature encoding z.

Upsample in arbitrary scale. We first extract 2D feature map $M \in \mathbb{R}^{H/8 \times W/8 \times C}$ from the input pair, I_1 and I_2.

$$M = \mathcal{E}(I_1, I_2) \tag{1}$$

The encoding function $\mathcal{E}$ is composed of modules ranging from the feature encoder to the GRU, as described in Fig. 2, with M representing the GRU's hidden state.

Similar to RAFT, we represent the motion of each pixel in the upsampled resolution as a convex combination of 3×3 local neighbors in the coarse flow, $\mathbf{f}_i \in \mathbb{R}^{H/8 \times W/8 \times 2}$. The convex weights, referred to as *mask*, are symbolized by $\mathcal{O}$ and defined at a continuous coordinate x_q as:

$$\mathcal{O}(x_q) = f_\theta(z^*, x_q - v^*, \psi(x_q - v^*)) \tag{2}$$

, where z^* represents the nearest feature vector from x_q in M, and v^* corresponds to the coordinate of z^*. To learn high-frequency signals, we apply positional encoding [18, 66] ψ to the relative position, $x_q - v^*$. The inputs are then fed into the decoding function f_θ, parameterized as an MLP.

The output, $\mathcal{O}(x_q)$, represents a $3 \times 3 \times n^2$-dimensional vector where the first two dimensions (3×3) correspond to the convex weights of 3×3 neighbors in the coarse resolution. n is a fixed hyperparameter, indicating that we generate an $n \times n$ local patch in the higher resolution with a single query at x_q.

To upsample $\mathbf{f}_i$ into our desired resolution, $H_o \times W_o$, we first evenly sample the query points x_q from a continuous 2D grid. The horizontal and vertical ranges of the grid are within the size of M. Since a single query upsamples a single pixel of flow to an $n \times n$ patch, we sample $\frac{H_o}{n} \cdot \frac{W_o}{n}$ numbers of the query points and feed them into f_θ to produce $\mathcal{O}$, which upsample $\mathbf{f}_i$ into $H_o \times W_o$ resolution.

Unlike RAFT, which relies on a fixed shape of the mask to achieve only $8-$times upsampling, we offer greater flexibility by allowing for general resizing through the adjustment of the number of samples.

3.2. Multi-scale Feature Warping

RAFT's capacity to capture small objects is limited by its reliance on $1/8$-scaled intermediate features. Moreover, the intricacy of 4D correlation volumes hinders its capacity to benefit from high-resolution features. To address this, we compute a partial cost volume through warping to effectively utilize high-resolution feature maps.

Following feature extraction, we obtain multi-scale feature maps F^s, where $s \in \{1/2, 1/4\}$, from each input frame, as well as a pair of $F^{1/8}$ maps for calculating correlation volumes. Given a pair of feature maps, F_1^s and F_2^s, we inversely warp F_2^s using the upsampled flow prediction, $\mathbf{f}_i^s$, which is derived by re-scaling the accumulated flow $\mathbf{f}_i$ with the implicit upsampler. We then concatenate F_1^s and the warped one $\mathcal{W}(F_2^s, \mathbf{f}_i^s)$, followed by applying a 1×1 convolution to reduce the channel dimension.

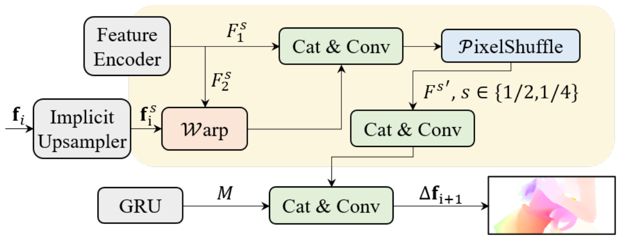

Figure 3. **Illustration of the multi-scale feature warping.** To utilize high-resolution representation, multi-scale feature maps, F^s, $s \in \{1/2, 1/4\}$, are combined with GRU feature, M. The combined features are used to predict the residual flow, $\Delta\mathbf{f}$.

Note that we directly upsample intermediate flow by reusing the implicit upsampler, removing the need for pyramidal architectures [20, 22, 44, 53, 67] and extra parameters.

Downsample by PixelShuffle. Our objective is to integrate the outputs from the prior step with the GRU feature M and employ them to estimate the residual flow. First, we must ensure that the spatial dimensions align, given that the outputs possess spatial dimensions of $sH \times sW$, while M is at a $1/8$ scale. To achieve this, we apply the PixelShuffle [47] (denoted by $\mathcal{P}$) to downscale the dimensions to match those of M and concatenate the local neighborhoods along the channel dimensions. This approach allows us to fully exploit spatial information while simultaneously lowering the resolution.

Following this, we apply an additional convolution to the PixelShuffle outputs, $F^{1/2\prime}$ and $F^{1/4\prime}$, and use them to estimate the residual flow for the subsequent iteration, $\Delta\mathbf{f}_{i+1}$, in conjunction with M. We outline the entire process in Eqns. 3-5 and Fig. 3.

$$\Delta\mathbf{f}_{i+1} = \text{Conv}([M, \text{Conv}_{1\times1}([F^{1/2\prime}, F^{1/4\prime}])]), \tag{3}$$

$$\text{where } F^{1/2\prime} = \mathcal{P}(\text{Conv}_{1\times1}(F_1^{1/2}, \mathcal{W}(F_2^{1/2}, \mathbf{f}_i^{1/2}))), \tag{4}$$

$$\text{and } F^{1/4\prime} = \mathcal{P}(\text{Conv}_{1\times1}(F_1^{1/4}, \mathcal{W}(F_2^{1/4}, \mathbf{f}_i^{1/4}))) \tag{5}$$

3.3. Dynamic Lookup with Region Encoding

Fixed lookup in RAFT. RAFT first maps each pixel $\mathbf{x}$ in I_1 to its estimated correspondence $\mathbf{x}'$ in I_2 with the current flow $\mathbf{f}_i$, as $\mathbf{x}' = \mathbf{x} + \mathbf{f}_i$. It defines a local grid around $\mathbf{x}'$ as *the set of integer offsets* within a radius r and bilinearly

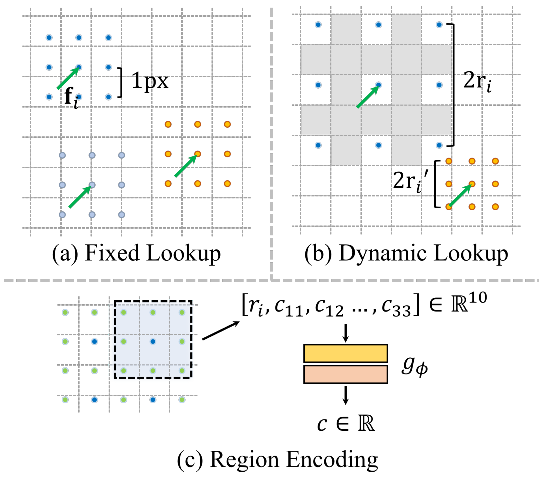

Figure 4. **Comparisons of lookup strategies.** (a) Fixed lookup in RAFT defines local grid in a fixed size and each grid point has 1px apart. (b) Dynamic lookup controls size of the local grid while using the same number of samples. When the grid size increases, there exist blindspots (gray area) between samples. (c) Auxiliary points (green) are defined between each real point (blue). 3×3 auxiliary region where each real point is the center are fed into the MLP (g_θ) to encode regional correlation.

samples the values of the local grid from each grid location of correlation volume. The radius r is set as a hyperparameter; hence, the single r is applied to every pixel in every GRU iteration. Here, the length of each side of the local grid is defined as $2r$ and the number of samples in each grid is $(2r+1)^2$. The samples from each local grid are concatenated along the channel dimension and fed into the GRU. We call this way *Fixed Lookup*, illustrated in Fig. 4 (a).[1]

Dynamic lookup. We can benefit from various sizes of the local grid. When r increases, large displacements can be captured well within each local grid. Small r is beneficial for capturing small motions accurately by concentrating on a small area. To generalize well on diverse shapes of inputs and ranges of motions, we dynamically predict the lookup range and update it for the correlation sampling.

To this end, the network predicts r as the pixel-wise output while keeping the number of grid points without change, which is updated by the GRU iteration. To be specific, the network predicts residual radius Δr, and accumulates it as $r_i = \Delta r_i + r_{i-1}$ in the same manner as the flow updates. We use the same number of samples in each grid as in the fixed lookup and the initial radius r_0 is set as a hyperparameter.

Region encoding. As r_i is updated, the length of the local grid can exceed the sample number in each side, making the sample interval larger than 1px. In such cases, sampling on the fixed number of points incurs blindspots (gray regions in Fig. 4 (b)), where the correlation values are not sampled to compute the output. To alleviate this problem, we introduce region encoding. We first define auxiliary points (green points in Fig. 4 (c)) between the existing points of each grid. We set a 3×3 auxiliary local region on each existing grid location, where the original grid location (blue points in Fig. 4 (c)) becomes the center of the auxiliary local region.

We define the encoding function g_ϕ, parameterized as an MLP, to learn the mapping from each auxiliary local region to a single correlation value. We also feed r_i as additional input to make the output depend on the area of respective region. After obtaining the correlation values with the region encoding, the values of each local grid are then concatenated along the channel dimension. In this way, we can utilize more sample points and encode regional information to remove the blindspots, without increasing the dimension of the correlation feature maps.

3.4. Multi-scale Training.

During training, we randomly downsample input RGB pairs with a probability of p. The scale factor is sampled from a uniform distribution for height and width, respectively. AnyFlow takes the downsampled input and generates output in the original size. We supervise it with RAFT loss [57], which minimizes the L1 distance between ground-truth and the predictions from multiple iterations.

$$\mathcal{L} = \sum_{i=1}^{N} \gamma^{N-i} \|\mathbf{f}_{gt} - \mathbf{f}_i^1\|_1 \tag{6}$$

4. Experiments

Following the previous works [24, 49, 57, 65], we first pretrain AnyFlow on the synthetic datasets. We train for 120K iterations on FlyingChairs [14] and for another 240k iterations on FlyingThings [38] with batch size of 16. We evaluate the pretrained model on the training set of Sintel [5] and KITTI-2015 [39] to compare cross-dataset generalization performance. Subsequently, we perform finetuning on the training set of Sintel combined with FlyingThings, KITTI, and HD1K [28] datasets for 120k iterations with batch size of 8. Then, the model is evaluated on the Sintel online benchmark. We apply warm start strategy [24, 49, 57] for it. Finally, we perform additional finetuning on the KITTI for 100k iterations and evaluate the results on the KITTI online benchmark.

Implementation Details. We implement AnyFlow in PyTorch [42]. During training, we use the AdamW [35] optimizer with one-cycle learning rate policy and gradient clip-

[1]For simplicity, in the figure, we assume 3×3 local grid.

Training	C + T				C + T + S + K + H			
Method	Sintel (train)		KITTI (train)		Sintel (test)		KITTI (test)	Params
	Clean	Final	F1-epe	F1-all	Clean	Final	F1-all	(M)
PWC-Net [53]	2.55	3.93	10.35	33.7	-	-	-	9.4
RAFT [57]	1.43	2.71	5.04	17.4	1.61	2.86	5.10	5.3
Flow1D [64]	1.98	3.27	6.69	22.95	2.24	3.81	6.27	5.7
SCV [25]	1.29	2.95	6.80	19.3	1.72	3.60	6.17	5.3
GMA [24]	1.30	2.74	4.69	17.1	1.39	2.47	5.15	5.9
Separable Flow [72]	1.30	2.59	4.60	15.9	1.50	2.67	4.64	6.0
AGFlow [37]	1.31	2.69	4.82	17.0	1.43	2.47	4.89	5.6
DIP [76]	1.30	2.82	4.29	13.73	1.44	2.83	**4.21**	5.4
CRAFT [49]	1.27	2.79	4.88	17.5	1.45	2.42	4.79	6.3
GMFlowNet [75]	1.14	2.71	4.24	15.4	1.39	2.65	4.79	9.3
GMFLow [65]	1.08	2.48	7.77	23.4	1.74	2.90	-	4.7
SKFLow [54]	1.22	2.46	4.27	15.5	1.28	2.23	4.84	6.3
FlowFormer (small) [19]	1.20	2.64	4.57	16.62	-	-	-	6.2
FlowFormer [19]	**0.64**	**1.5**	4.09	14.72	**1.16**	**2.09**	4.68	18.2
AnyFlow (*dynamic*)	1.17	2.58	**3.95**	**13.01**	1.23	2.44	4.41	5.4
AnyFlow (*R.E.*)	1.10 / 1.05	2.52	**3.76**	**12.44**	1.26	2.63	4.64	5.4
AnyFlow + GMA [24]	1.16	2.62	**4.05**	13.74	1.21	2.46	-	6.0

Table 1. **Quantitative comparisons with recent state-of-the-arts.** The left half results (C+T) are for cross-dataset generalization. All methods are evaluated on train set of Sintel and KITTI dataset after they are trained on FlyingChairs and FlyingThings (C+T). The right half results are fine-tuned on the target datasets and evaluated on the online benchmark. *dynamic* means the proposed dynamic lookup and *R.E.* denotes the region encoding. We describe the best results in **bold** and the second best results in underline.

ping [57]. We use 12 flow updates for training, 32 for Sintel evaluation and 24 for KITTI evaluation. For the implicit flow upsampler, we set n as 4. We initialize $r_0 = 4$ for dynamic lookup and $r_0 = 6$ for region encoding, respectively. During the multi-scale warping, we estimate $\mathbf{f}_{1/2}$ by the flow upsampler as mentioned in Sec. 3.2, and we compute $\mathbf{f}_{1/4}$ by applying average pooling to $\mathbf{f}_{1/2}$. Further details are described in the supplementary material.

4.1. Comparison with State-of-the-arts

Cross-dataset generalization. In the left half of Table 1, we evaluate the results on the training set of Sintel [5] and KITTI [39] to compare how each method can generalize on the target datasets. All methods are trained on the FlyingChairs [14] and FlyingThings [38].

On the Sintel clean, AnyFlow (*R.E.*) achieves 1.10 EPE, reducing the error of the baseline, RAFT, by 23%. We achieve the largest improvement among the recent RAFT-based methods [24, 37, 49, 54, 64, 72], even though we use only 0.1M more parameters than RAFT. GMFlow [65] and FlowFormer [19] adopt Transformer [34] for global matching to capture large displacements and achieve the best results. Instead, AnyFlow captures large motions by dynamic control of correlation lookup and achieves competitive performance. With warm-start strategy [57], AnyFlow (*R.E.*) achieves the second best result, 1.05 EPE. On the KITTI, AnyFlow outperforms every previous method by large margin in both metrics, which improves the baseline by more than 25%. As the KITTI dataset contains relatively smaller motions compared to the Sintel, the proposed way of utilizing high-resolution features is effective to capture them. In addition, AnyFlow updates lookup range depending on input frames and generalizes well on the real-world dataset, KITTI, even though the training is done only on the synthetic datasets.

Test set benchmark. In the right half of Table 1, we evaluate the results on the test set of Sintel and KITTI datasets. AnyFlow (*dynamic*) achieves the second best result (1.23 EPE) on Sintel clean with large improvements over RAFT. Since our modifications to the baseline are orthogonal to others, AnyFlow is compatible with previous works. We combine GMA [24] with AnyFlow (AnyFlow + GMA) to demonstrate AnyFlow can further benefit from them. Several previous methods [19, 49, 54] also adopt GMA to resolve occlusion. AnyFlow achieves 1.21 EPE on Sintel clean after leveraging it.

On the KITTI dataset, AnyFlow (*dynamic*) achieves 4.41 F1-all and ranks second among the previous studies. Note that we do not compare with the methods [23, 50] that have no difference in architecture from our baseline, other than training strategy and datasets. Even though DIP [76] shows better result on this metric, we outperform it on all the other metrics. This result further demonstrates that AnyFlow generalizes well in real-world scenarios.

Overall, AnyFlow performs well in general and is the best choice when considering diverse scenarios: cross-dataset generalization, test set results and number of parameters. Even though FlowFormer [19] performs best on Sintel, AnyFlow is better on the real-world dataset, KITTI, with three times less parameters (18.2M vs 5.4M). Also, FlowFormer requires ImageNet [12] pretraining to achieve good accuracy. We also report the results of FlowFormer (small) that uses less parameters without ImageNet training. We show better results on cross-dataset generalization.

4.2. Ablation Study

In Table 2, we conduct ablation studies to examine the impact of each component on the overall performance. The model is trained on synthetic datasets (C+T) and evaluated on the training sets of Sintel and KITTI.

In the upper section, we evaluate various strategies for multi-scale feature warping. MS *bilin.* employs bilinear upsampling on $\mathbf{f}_i$ to derive $\mathbf{f}_i^{1/2}$ and $\mathbf{f}_i^{1/4}$ for warping, while MS calculates intermediate flows using the implicit upsampler, as described in Sec. 3.2.

MS *bilin.* exhibits significant improvements over w/o MS across all metrics, and MS further enhances the results. The findings reveal that employing high-resolution features effectively increases accuracy, while the proposed methods contribute additional synergistic effects.

Essentially, the accumulated flow $\mathbf{f}_i$ is not directly supervised by ground-truth during training, but the flow after convex upsampling, i.e., $\mathbf{f}_i^1$, contributes to the loss computation. As a result, neither the accumulated flow nor the bilinearly upsampled one provides precise estimations.

In contrast, our proposed implicit upsampler performs convex upsampling for any size, including $s \in \{1/2, 1/4\}$. This approach allows for accurate estimation of intermediate flows while preserving high-frequency details, making it more appropriate for warping. We compare the visualizations in Fig. 5.

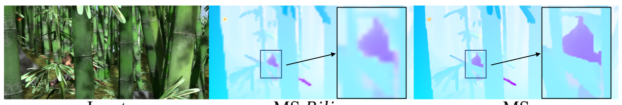

Figure 5. Comparison of upsampling strategies for multi-scale feature warping. We produce $\mathbf{f}_i^{1/2}$ by bilinear interpolation (MS *Bilin.*) and the proposed implicit upsampler (MS), respectively.

The middle section reports the effect of the neural implicit flow upsampler (NIFU). We use the original RAFT upsampler for w/o NIFU. Note that we adopt MS *bilin.* in this ablation, since the complete MS warping leverages the implicit upsampler. As shown in the table, we obtain large improvements especially on the KITTI. Owing to the continuous representation, we can estimate precise boundaries and adopt multi-scale training to train our network under diverse motion ranges. Since it consists of MLP, it produces better results with less parameters. Not only improving the quantitative performance, the implicit upsampler also produces synergistic effect with the MS warping strategy and enables flow generation in arbitrary scales. Further ablations and the effects of multi-scale training can be found in our supplementary material.

Method	Sintel		KITTI		Param
	Clean	Final	F1-epe	F1-all	
w/o MS	1.29	2.72	4.02	13.91	5.04
MS *bilin.*	1.16	2.55	3.81	12.82	5.39
MS	**1.10**	**2.52**	**3.76**	**12.44**	5.39
w/o NIFU	1.24	**2.53**	4.47	14.39	5.62
NIFU	**1.16**	2.55	**3.81**	**12.82**	5.39
Fixed	1.21	2.68	4.41	14.28	5.38
Dynamic	1.17	2.58	3.95	13.01	5.39
Region E.	**1.10**	**2.52**	**3.76**	**12.44**	5.39

Table 2. **Ablation experiments.** The models are trained on the synthetic datasets (C+T), and evaluated on Sintel and KITTI.

In the last section, we compare the lookup strategies. Since the dynamic lookup controls the lookup ranges depending on the input images, it can focus on more optimal candidates to compute accurate flow. It shows large improvements especially on KITTI, which demonstrates its effectiveness for real-world generalization. The region encoding (Region E.) further improves the result in all metrics. As shown in Table 1, on the other hand, AnyFlow (*dynamic*) shows better results on the test set.

4.3. Arbitrary Scale Optical Flow Estimation

Robustness on downsampled images. In Table 3, we compare robustness to downsampling with the recent methods. We feed input to the networks as downsampled images by multiple scales from 50% to 90%. The produced outputs are restored to the original shape and used to compute EPE between ground-truth. All models are trained on the synthetic datasets (C + T) and evaluated on Sintel and KITTI.

On the Sintel dataset in Table 3 (a), AnyFlow shows only 1% of degradation up to 80% of downsampling. Up to 70% of downsampling, which is a typical range of downsampling ratio in practice, the EPEs increase by 0.21 for RAFT, 0.32 for GMA, and 0.28 for GMFlow. AnyFlow shows only 0.1 error increase, which demonstrates its ro-

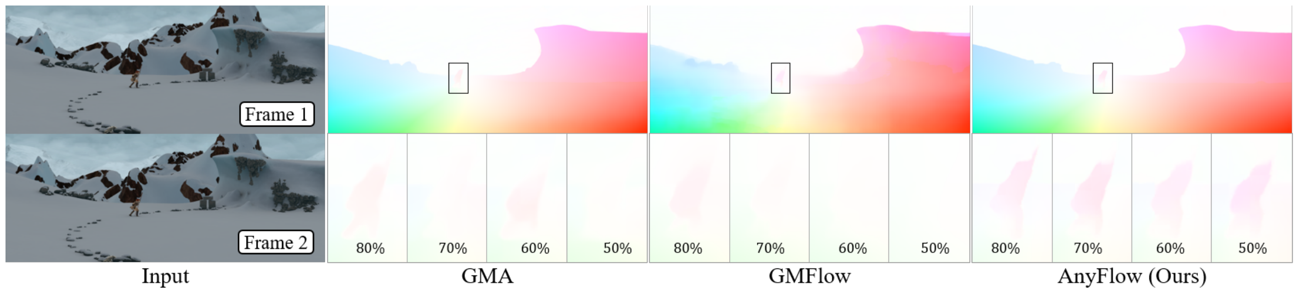

Figure 6. Qualitative comparison on Sintel test images against GMA [24] and GMFlow [65]. The top row is from input frames without resize and the bottom row showcases the close-up for the corresponding results by different down-scale ratios, indicated by the numbers.

Method	100%	90%	80%	70%	60%	50%
RAFT [57]	1.47	1.51	1.60	1.68	1.81	1.92
GMA [24]	1.31	1.43	1.45	1.63	1.71	1.92
GMFlow [65]	**1.08**	1.18	1.17	1.36	1.71	2.35
AnyFlow	1.10	**1.12**	**1.11**	**1.20**	**1.31**	**1.41**

(a) EPE on Sintel (clean)

Method	100%	90%	80%	70%	60%	50%
RAFT [57]	5.04	5.22	5.39	6.17	6.61	7.50
GMA [24]	4.48	4.60	4.79	5.21	5.67	6.25
GMFlow [65]	7.49	7.81	8.76	9.63	9.17	10.41
AnyFlow	**3.76**	**3.80**	**3.92**	**3.97**	**4.28**	**4.88**

(b) EPE on KITTI

Table 3. Quantitative comparisons of performance with downsampled images on Sintel and KITTI datasets.

bustness in practical scenarios where downsampling is used. In the more extreme cases, i.e., 60% and 50% of downsampling, other methods severely suffer from accuracy loss. In both datasets, AnyFlow achieves competitive accuracy even with 50% downsampled images, which is still better than the original results (100%) of the baseline, RAFT.

In Fig. 1 and Fig. 6, we qualitatively compare the predictions. Since the metric is averaged over every pixel, it does not reflect well how each method can detect small objects under downsampled images. As shown in the figures, recent methods [24, 57, 65] lose ability to capture small objects since they are dealing with low-resolution intermediate features. On the other hand, AnyFlow restores more clear boundary and captures small objects well with images of any sizes. Thanks to the proposed multi-scale warping and dynamic control of the correlation range, AnyFlow performs more robust estimation on diverse resolutions.

Generating high-resolution optical flow. In Fig. 7, we compare the results of upsampling at higher resolutions ($\times 2$, $\times 3$) using different methods. To compare with a super-resolution method, we first convert the flow output to a 3-channel RGB and feed it into the pre-trained LIIF network [8]. For both interpolation and super-resolution, we initially estimate the output at the original scale, $\mathbf{f}_i^1$, and then upsample it. On the other hand, the implicit upsampler produces high-resolution flow, $\mathbf{f}_i^2$ and $\mathbf{f}_i^3$, directly from the accumulated flow, $\mathbf{f}_i$. As illustrated in the figure, interpolation and LIIF introduce blocking artifacts because they use low-resolution flow ($\times 1$) as input, amplifying noise and artifacts as resolution increases. In contrast, we train the network end-to-end, allowing the implicit upsampler to directly generate high-resolution flow from the latent space. This results in clearer boundaries and fewer artifacts.

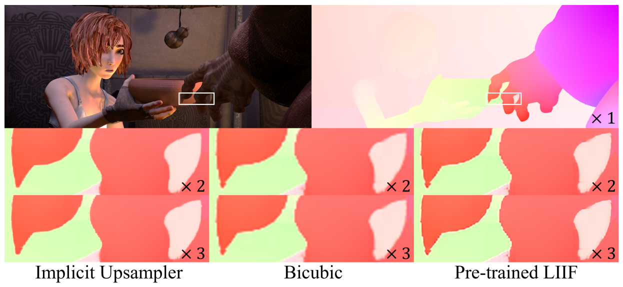

Figure 7. We evaluate our method (left) on flow upsampling against bicubic (middle) and results by using pre-trained LIIF [8] (right). Compared to both methods, our results produce sharper object boundaries without introducing artifacts.

5. Conclusion

We have proposed AnyFlow, a new method that models optical flow as a continuous coordinate-based representation, enabling the generation of outputs at arbitrary scales. With the novel warping module and dynamic lookup strategy, AnyFlow exhibits robust performance across various motion types and input resolutions. In particular, it substantially enhances estimation quality for low-resolution images and small objects while preserving intricate details. To our knowledge, AnyFlow is the first approach designed to provide robust estimations for low-resolution inputs, which extends its applicability to portable devices.

References

[1] Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *ICCV*, 2021. 3

[2] M.J. Black and P. Anandan. A framework for the robust estimation of optical flow. In *ICCV*, 1993. 1, 2

[3] Andres Bruhn, Joachim Weickert, and Christoph Schnörr. Lucas/kanade meets horn/schunck: Combining local and global optic flow methods. In *IJCV*, 2005. 1, 2

[4] Antoni Buades, Jose-Luis Lisani, and Marko Miladinović. Patch-based video denoising with optical flow estimation. In *IEEE TIP*, 2016. 1

[5] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, 2012. 1, 5, 6

[6] Jose Caballero, Christian Ledig, Andrew Aitken, Alejandro Acosta, Johannes Totz, Zehan Wang, and Wenzhe Shi. Real-time video super-resolution with spatio-temporal networks and motion compensation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4778–4787, 2017. 2

[7] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *ICCV*, 2021. 3

[8] Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *CVPR*, pages 8628–8638, 2021. 2, 3, 8

[9] Zeyuan Chen, Yinbo Chen, Jingwen Liu, Xingqian Xu, Vidit Goel, Zhangyang Wang, Humphrey Shi, and Xiaolong Wang. Videoinr: Learning video implicit neural representation for
continuous space-time super-resolution. In *CVPR*, 2022. 3

[10] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *CVPR*, 2019. 3

[11] Douglas Decarlo and Dimitris Metaxas. Optical flow constraints on deformable models with applications to face tracking. In *IJCV*, 2000. 1

[12] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 7

[13] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. In *IEEE TPAMI*, 2015. 3

[14] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *ICCV*, 2015. 1, 2, 5, 6

[15] Marius Drulea and Sergiu Nedevschi. Total variation regularization of local-global optical flow. In *International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2011. 1, 2

[16] Markus Hofinger, Samuel Rota Bulò, Lorenzo Porzi, Arno Knapitsch, Thomas Pock, and Peter Kontschieder. Improving optical flow on a pyramid level. In *ECCV*, 2020. 2

[17] Berthold KP Horn and Brian G Schunck. Determining optical flow. In *Artificial intelligence*, 1981. 2

[18] Hanzhe Hu, Yinbo Chen, Jiarui Xu, Shubhankar Borse, Hong Cai, Fatih Porikli, and Xiaolong Wang. Learning implicit feature alignment function for semantic segmentation. In *ECCV*, 2022. 3, 4

[19] 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*, 2022. 2, 6, 7

[20] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. Lite-flownet: A lightweight convolutional neural network for optical flow estimation. In *CVPR*, 2018. 1, 2, 4

[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*, 2017. 2

[22] Azin Jahedi, Lukas Mehl, Marc Rivinius, and Andrés Bruhn. Multi-scale raft: Combining hierarchical concepts for learning-based optical flow estimation. In *ICIP*, 2022. 1, 4

[23] Jisoo Jeong, Jamie Menjay Lin, Fatih Porikli, and Nojun Kwak. Imposing consistency for optical flow estimation. In *CVPR*, 2022. 1, 2, 6

[24] Shihao Jiang, Dylan Campbell, Yao Lu, Hongdong Li, and Richard Hartley. Learning to estimate hidden motions with global motion aggregation. In *ICCV*, 2021. 2, 5, 6, 8

[25] Shihao Jiang, Yao Lu, Hongdong Li, and Richard Hartley. Learning optical flow from a few matches. In *CVPR*, pages 16592–16600, 2021. 1, 2, 6

[26] Kiran Kale, Sushant Pawar, and Pravin Dhulekar. Moving object tracking using optical flow and motion vector estimation. In *2015 4th international conference on reliability, infocom technologies and optimization (ICRITO)(trends and future directions)*, pages 1–6. IEEE, 2015. 1

[27] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. In *NeurIPS*, 2021. 3

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

[29] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Deep laplacian pyramid networks for fast and accurate super-resolution. In *CVPR*, 2017. 3

[30] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, 2017. 3

[31] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *CVPRW*, 2017. 3

[32] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *ICCV*, 2021. 3

[33] Ce Liu and William T Freeman. A high-quality video denoising algorithm based on reliable motion estimation. In *ECCV*, 2010. 1

[34] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. 2, 6

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

[36] Ao Luo, Fan Yang, Xin Li, and Shuaicheng Liu. Learning optical flow with kernel patch attention. In *CVPR*, pages 8906–8915, June 2022. 1, 2

[37] Ao Luo, Fan Yang, Kunming Luo, Xin Li, Haoqiang Fan, and Shuaicheng Liu. Learning optical flow with adaptive graph reasoning. In *AAAI*, 2022. 2, 6

[38] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016. 5, 6

[39] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *CVPR*, 2015. 5, 6

[40] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 3

[41] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *ICCV*, 2021. 3

[42] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 5

[43] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *CVPR*, 2021. 3

[44] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *CVPR*, 2017. 1, 2, 4

[45] Mehdi SM Sajjadi, Raviteja Vemulapalli, and Matthew Brown. Frame-recurrent video super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6626–6634, 2018. 2

[46] Loren Arthur Schwarz, Artashes Mkhitaryan, Diana Mateus, and Nassir Navab. Human skeleton tracking from depth data using geodesic distances and optical flow. In *Image and Vision Computing*, 2012. 1

[47] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*, 2016. 4

[48] Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *NeurIPS*, 2020. 3

[49] Xiuchao Sui, Shaohua Li, Xue Geng, Yan Wu, Xinxing Xu, Yong Liu, Rick Siow Mong Goh, and Hongyuan Zhu. Craft: Cross-attentional flow transformers for robust optical flow. In *CVPR*, 2022. 2, 5, 6

[50] Deqing Sun, Charles Herrmann, Fitsum Reda, Michael Rubinstein, David J. Fleet, and William T Freeman. Disentangling architecture and training for optical flow. In *ECCV*, 2022. 1, 2, 6

[51] Deqing Sun, Stefan Roth, and Michael Black. A quantitative analysis of current practices in optical flow estimation and the principles behind them. In *IJCV*, 2014. 1, 2

[52] Deqing Sun, Daniel Vlasic, Charles Herrmann, Varun Jampani, Michael Krainin, Huiwen Chang, Ramin Zabih, William T Freeman, and Ce Liu. Autoflow: Learning a better training set for optical flow. In *CVPR*, 2021. 1, 2

[53] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *CVPR*, 2018. 1, 2, 4, 6

[54] Shangkun Sun, Yuanqi Chen, Yu Zhu, Guodong Guo, and Ge Li. Skflow: Learning optical flow with super kernels. In *arXiv preprint arXiv:2205.14623*, 2022. 6

[55] Shuyang Sun, Zhanghui Kuang, Lu Sheng, Wanli Ouyang, and Wei Zhang. Optical flow guided feature: A fast and robust motion representation for video action recognition. In *CVPR*, 2018. 1

[56] Xin Tao, Hongyun Gao, Renjie Liao, Jue Wang, and Jiaya Jia. Detail-revealing deep video super-resolution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4472–4480, 2017. 2

[57] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020. 1, 2, 3, 5, 6, 8

[58] Zhigang Tu, Hongyan Li, Wei Xie, Yuanzhong Liu, Shifu Zhang, Baoxin Li, and Junsong Yuan. Optical flow for video super-resolution: A survey. In *arXiv preprint arXiv:2203.10462*, 2022. 2

[59] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2

[60] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *Proceedings of the IEEE international conference on computer vision*, pages 3551–3558, 2013. 1

[61] Longguang Wang, Yulan Guo, Li Liu, Zaiping Lin, Xinpu Deng, and Wei An. Deep video super-resolution using hr optical flow estimation. In *IEEE TIP*, 2020. 2

[62] Lei Wang, Piotr Koniusz, and Du Q Huynh. Hallucinating idt descriptors and i3d optical flow features for action recognition with cnns. In *ICCV*, 2019. 1

[63] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. NeRF−−: Neural radiance fields without known camera parameters. In *arXiv preprint arXiv:2102.07064*, 2021. 3

[64] Haofei Xu, Jiaolong Yang, Jianfei Cai, Juyong Zhang, and Xin Tong. High-resolution optical flow from 1d attention and correlation. In *ICCV*, 2021. 2, 6

[65] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. Gmflow: Learning optical flow via global matching. In *CVPR*, 2022. 2, 5, 6, 8

[66] Xingqian Xu, Zhangyang Wang, and Humphrey Shi. Ultrasr: Spatial encoding is a missing key for implicit image function-based arbitrary-scale super-resolution. In *arXiv preprint arXiv:2103.12716*, 2021. 3, 4

[67] Gengshan Yang and Deva Ramanan. Volumetric correspondence networks for optical flow. In *NeurIPS*, 2019. 1, 2, 4

[68] Zhichao Yin, Trevor Darrell, and Fisher Yu. Hierarchical discrete distribution decomposition for match density estimation. In *CVPR*, 2019. 2

[69] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *CVPR*, 2021. 3

[70] Songhyun Yu, Bumjun Park, Junwoo Park, and Jechang Jeong. Joint learning of blind video denoising and optical flow estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 500–501, 2020. 1

[71] Christopher Zach, Thomas Pock, and Horst Bischof. A duality based approach for realtime tv-l1 optical flow. In *Pattern Recognition*, 2007. 1, 2

[72] Feihu Zhang, Oliver J. Woodford, Victor Adrian Prisacariu, and Philip H.S. Torr. Separable flow: Learning motion cost volumes for optical flow estimation. In *ICCV*, 2021. 2, 6

[73] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. In *arXiv preprint arXiv:2010.07492*, 2020. 3

[74] Shengyu Zhao, Yilun Sheng, Yue Dong, Eric I-Chao Chang, and Yan Xu. Maskflownet: Asymmetric feature matching with learnable occlusion mask. In *CVPR*, 2020. 1, 2

[75] Shiyu Zhao, Long Zhao, Zhixing Zhang, Enyu Zhou, and Dimitris Metaxas. Global matching with overlapping attention for optical flow estimation. In *CVPR*, 2022. 6

[76] Zihua Zheng, Ni Nie, Zhi Ling, Pengfei Xiong, Jiangyu Liu, Hao Wang, and Jiankun Li. Dip: Deep inverse patchmatch for high-resolution optical flow. In *CVPR*, 2022. 2, 6, 7