2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

NeuFlow: Real-time, High-accuracy Optical Flow Estimation on Robots Using Edge Devices

Zhiyong Zhang[1], Huaizu Jiang[2], Hanumant Singh[1]

***Abstract*— Real-time high-accuracy optical flow estimation is a crucial component in various applications, including localization and mapping in robotics, object tracking, and activity recognition in computer vision. While recent learning-based optical flow methods have achieved high accuracy, they often come with heavy computation costs. In this paper, we propose a highly efficient optical flow architecture, called NeuFlow, that addresses both high accuracy and computational cost concerns. The architecture follows a global-to-local scheme. Given the features of the input images extracted at different spatial resolutions, global matching is employed to estimate an initial optical flow on the 1/16 resolution, capturing large displacement, which is then refined on the 1/8 resolution with lightweight CNN layers for better accuracy. We evaluate our approach on Jetson Orin Nano and RTX 2080 to demonstrate efficiency improvements across different computing platforms. We achieve a notable 10-80 speedup compared to several state-of-the-art methods, while maintaining comparable accuracy. Our approach achieves around 30 FPS on edge computing platforms, which represents a significant breakthrough in deploying complex computer vision tasks such as SLAM on small robots like drones. The full training and evaluation code is available at https://github.com/neufieldrobotics/NeuFlow.**

I. Introduction

Optical flow estimation, a fundamental task in computer vision [1], plays a pivotal role in various applications such as object tracking [2], [3], motion analysis [4], scene understanding [5], and visual odometry [6], [7]. Optical flow refers to the distribution of apparent velocities of movement of brightness patterns in an image [8], which can result from the relative motion of objects and the viewer [9], [10]. Real-time optical flow estimation, in particular, holds immense significance in scenarios requiring quick and accurate analysis of dynamic scenes [11], ranging from robotics [12] to autonomous driving systems [13], augmented reality [14], and beyond [15].

In recent years, significant advancements have been made in the development of algorithms and techniques aimed at achieving high-accuracy optical flow estimation [16], [17], [18], [19], [20]. Starting from FlowNet [16], learning-based optical flow methods have emerged to learn features for matching instead of relying on hand-crafted features like Lucas-Kanade [21] or SIFT [22], [23]. However, early optical flow methods still suffer from two major problems: large displacement and ambiguity [19]. Recent deep learning methods [18], [24], [19], [25], [26], [27], [20], [28] have made strides in addressing these issues to some extent, at the expense of computation time.

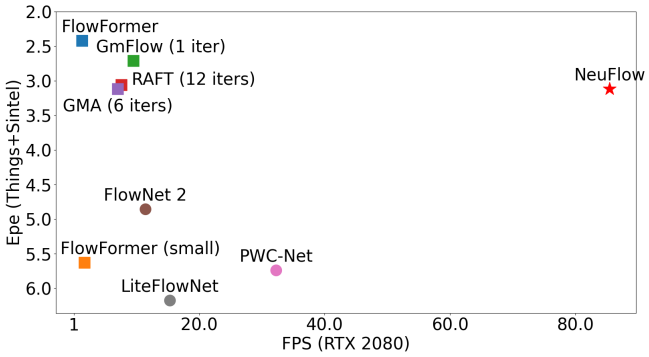

Fig. 1: End point error (EPE) v.s. frame per second (FPS) throughput on a common computing platform (Nvidia RTX 2080). Individual points represents a broad class of optical flow methods. Our algorithm is comparable in accuracy but significantly better (close to an order of magnitude) in terms of its computational complexity. All models trained solely on FlyingThings and FlyingChairs.

Fig. 2: Optical flow results of NeuFlow: on the left is a result from the standard KITTI dataset. On the right are results from a UAS flight overlow-contrast glacier images in the Arctic. Our approach is notable for both computational efficiency and speed as well as accuracy, as shown in Fig. 1.

Early optical flow methods rely on CNNs (Convolutional Neural Networks) and local correlations of image features, which can only capture small-displacement pixel movement due to the restricted range of operation of these techniques [19]. Recent solutions, such as RAFT [18] and GMA [24], use iterative methods to mitigate such problems. Transformer-based approaches, like GmFlow [19], Unimatch [25], GMFlowNet [26] or CRAFT [27], leverage global attention [29] layers to address the issue. However, iterative

[1]Department of Electrical and Computer Engineering, Northeastern University, Boston, MA 02115. zhang.zhiyo@northeastern.edu ha.singh@northeastern.edu [2]Khoury College of Computer Sciences, Northeastern University, Boston, MA 02115. HJ is supported by the National Science Foundation under Award IIS-2310254. h.jiang@northeastern.edu

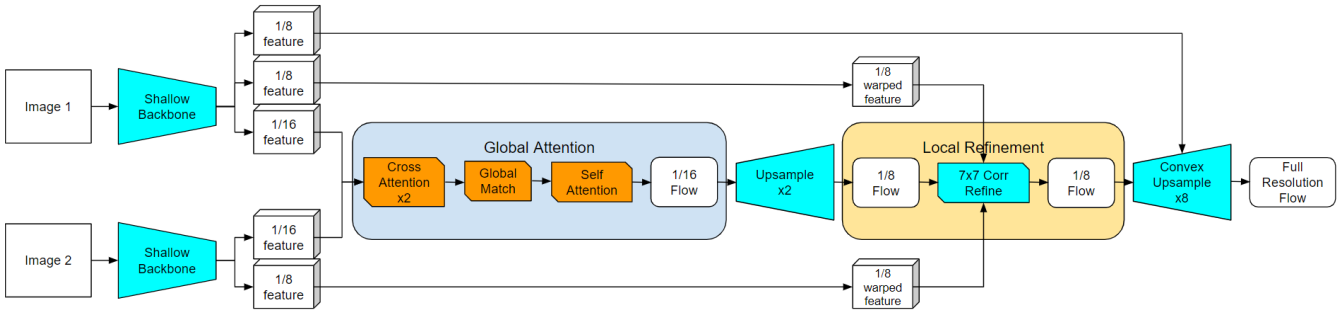

Fig. 3: NeuFlow Architecture: We begins with a shallow CNN backbone. The backbone outputs feature vectors at 1/8 and 1/16 scale for both images. The feature vectors at 1/16 scale are then fed into two cross-attention layers for global matching. The resulting flow is passed into a self-attention layer for flow propagation based on feature self-similarity. Subsequently, the flow is upsampled to obtain 1/8 resolution flow. We wrap the 1/8 features with the flow and perform local refinement within a 7x7 window. The refined 1/8 flow is then upsampled to obtain full-resolution flow using a convex upsampling module, which additionally requires 1/8 features from image one.

methods necessitate numerous iterations to estimate large-displacement optical flow [18] and global attention computes correlations between each pair pixels across two images, both resulting in significant computational costs [27].

Another challenge is ambiguity, which includes occlusions and textureless regions [30], is typically addressed by aggregating pixels that likely belong to the same object [24]. Early optical flow methods, constrained by the limited receptive field of CNNs and the local correlation range, struggle to address these challenges in a global manner [19]. Transformer-based models with self-attention can indeed address ambiguity problems to some extent by leveraging global feature aggregation. However, they also entail high computational costs, even when working on a 1/8 resolution of the image rather than the full resolution [27].

In this paper, we propose a novel optical flow model, called NeuFlow, for *real-time* optical flow estimation on edge devices while ensuring *high accuracy*. As shown in Fig. 2, NeuFlow runs at 30fps on a Jetson Orin Nano to process images with the resolution of 512×384. Specifically, we first use different lightweight CNNs (Convolutional Neural Networks) to encode the input images at different scales of image pyramids. They are enhanced by cross-attention to share information between the input images. Global matching is then adopted at a lower image scale (1/16) to capture large displacement with small computation burden, which is refined by a self-attention module to improve estimation in the ambiguous regions. The initial optical flow is further processed at the 1/8 scale with CNN layers for local refinement. It runs much faster than global matching and thus is designed to work on a higher spatial resolution. Finally, full-resolution optical flow is obtained through a convex upsampling module.

We conduct experiments on standard benchmarks, training solely on the FlyingChairs and FlyingThings datasets, and evaluate on both the FlyingThings and Sintel datasets for full-resolution flow. Fig. 1 shows the end-point error versus frames per second on an RTX 2080. We achieve comparable accuracy to the latest optical flow methods, including RAFT, GMFlow, and GMA, while being 10 faster. FlowFormer achieves the highest accuracy but is 70 slower than our method.

Our main contribution is an optical flow system. We make design choices that ensure real-time inference on edge devices without postprocessing (*e.g.*, compression, pruning) and high accuracy at the same time. Our code and model weights have been publicly released. By sharing NeuFlow with the community, we believe it will empower the next generation of robotic SLAM, visual SLAM and visual inertial odometry applications on UAS and other SWaP-C contained robotic vehicles.

II. Related Work

FlowNet [16] was the first end-to-end convolutional network for optical flow estimation, proposing two variants: FlowNetS and FlowNetC, along with the synthetic FlyingChairs dataset for end-to-end training and benchmarking. An improved version, FlowNet 2.0 [17], fused cascaded FlowNets with a small displacement module and decreased the estimation error by more than 50%.

Following FlowNet [16], researchers sought lightweight optical flow methods. SPyNet [31] computed optical flow by combining a classical spatial-pyramid formulation, offering a model 96% smaller than FlowNet in terms of model parameters. PWC-Net [32] was 17 times smaller in size than the FlowNet 2 [17] model. LiteFlowNet [33] presented an alternative network with comparable results to FlowNet 2 [17] while being 30 times smaller in model size and 1.36 times faster in running speed. LiteFlowNet 2 [34] improved optical flow accuracy on each dataset by around 20% while being 2.2 times faster. LiteFlowNet 3 [35] further improved flow accuracy by exploring local flow consistency. VCN [36] utilizes volumetric encoder-decoder architectures to efficiently capture large receptive fields, reducing computation and parameters while preserving accuracy. DCFlow [37] estimates accurate optical flow through direct cost volume

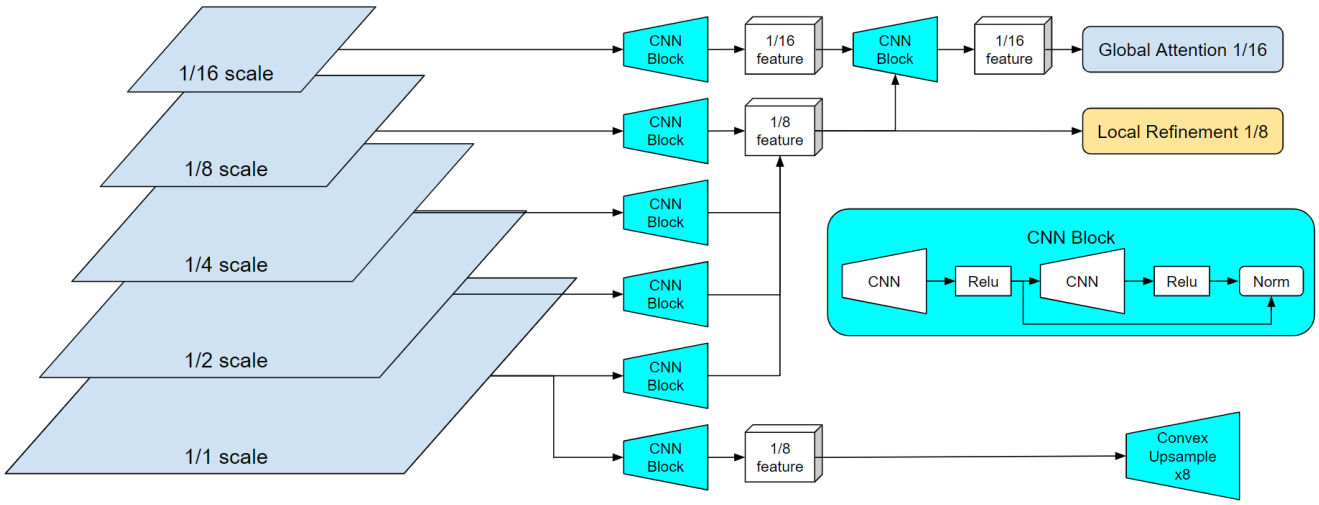

Fig. 4: NeuFlow Shallow CNN Backbone: Initially, we downsample the image into different scales, ranging from 1/1 scale to 1/16 scale. Subsequently, we extract features using a CNN block. The feature vectors at 1/1, 1/2, 1/4, and 1/8 scales are concatenated into a single 1/8 feature vector. Then, another CNN block is employed to merge the 1/8 feature vector with the 1/16 feature vector, resulting in a 1/16 feature vector. The 1/16 feature vector is utilized for global attention, while the 1/8 feature vector is employed for local refinement. The CNN block consists solely of two CNN layers along with activation functions and normalization layers. The kernel size and stride of the CNN layers depend on the input and output dimensions upstream and downstream of the network. An additional 1/8 feature is extracted from the full-resolution image to perform convex upsampling.

processing. DCVNet [38], a novel approach that combines dilated cost volumes and 3D convolutions, has demonstrated real-time inference on 1080ti GPU while achieving comparable accuracy to other optical flow estimation approaches.

More recently, RAFT [18] used recurrent all-pairs field transforms to achieve strong cross-dataset generalization as well as high efficiency in inference time, training speed, and parameter count. GMA [24] used global motion aggregation to help resolve ambiguities caused by occlusions. GmFlow [19] reformulated optical flow as a global matching problem to achieve both high accuracy and efficiency. CRAFT [27] used a cross-attentional flow transformer to revitalize the correlation volume computation. FlowFormer [20], [28] introduced a transformer-based neural network architecture for learning optical flow and has achieved new state-of-the-art performance. Many works [39], [40], [41], [42], [43], [44], [45] are also proposed to either reduce the computational costs or improve the flow accuracy.

III. Proposed Approach: NeuFlow

We introduce NeuFlow, a global-to-local architecture for optical flow that achieves both high accuracy and efficiency. An illustration of NeuFlow's architecture is shown in Fig. 3. Initially, a shallow CNN backbone extracts low-level features from a multi-scale image pyramid. Next, global cross-attention and self-attention are applied at the 1/16 scale to address the challenges of large displacement. Subsequently, local refinement is conducted at the 1/8 scale to yield high-accuracy optical flow. Convex upsampling is then employed to generate full-resolution flow.

A. Shallow CNN Backbone

While most optical flow methods employ a relatively deep CNN backbone for feature extraction, we believe that high-level, semantical encoding of input images is not necessary for optical flow tasks. Instead, sufficient low-level features are more crucial. Thus, in our approach, we employ a simple CNN block to directly extract features at multiple scales of the images, as depicted in Fig. 4.

Simple CNN blocks are used to extract features from various scales of the input images. Each block comprises only two CNN layers, activation functions, and normalization. This design prioritizes the extraction of a large number of low-level features directly from the image. High-level CNN is only employed to merge features from different scales.

B. Global Cross-Attention

Similar to GMFlow [19], we utilize Transformers [29] to implement global cross-attention. This mechanism takes features from image one as the query and features from image two as both key and value. This mechanism enhances the distinctiveness of matching features and reduces the similarity of unmatched features.

Global matching is then applied to find corresponding features. Unlike local regression-based optical flow methods, this approach does not restrict the range of flow between image pairs. Consequently, it performs well on datasets with large pixel displacement, such as FlyingThings [46], and exhibits greater stability in real-world scenarios, including fast-moving camera situations.

Full Res (960540)	Things (val, clean)	Things (val, final)	RTX 2080 (s)	Jetson Orin Nano (s)	Batch Size (8G)	Params
FlowFormer	3.488	**2.774**	0.834	N/A	2	16.17M
FlowFormer (small)	9.773	8.745	0.661	N/A	2	6.18M
GMFlow (1 iter)	**3.271**	3.123	0.115	1.000	8	4.68M
RAFT (12 iters)	4.122	3.775	0.142	0.878	8	5.26M
GMA (6 iters)	4.396	3.785	0.145	0.936	4	5.88M
NeuFlow	3.846	3.828	**0.013**	**0.097**	**42**	**3.85M**
Full Res (1024436)	Sintel (train, clean)	Sintel (train, final)	RTX 2080 (s)	Jetson Orin Nano (s)	Batch Size (8G)	Params
FlowFormer	**1.004**	**2.401**	0.715	N/A	2	16.17M
FlowFormer (small)	1.324	2.679	0.534	N/A	2	6.18M
GMFlow (1 iter)	1.495	2.955	0.097	0.820	10	4.68M
RAFT (12 iters)	1.548	2.791	0.124	0.760	8	5.26M
GMA (6 iters)	1.423	2.866	0.141	0.747	8	5.88M
NeuFlow	1.660	3.126	**0.011**	**0.084**	**64**	**3.85M**

TABLE I: This table compares latest optical flow methods when outputting full-resolution flow, all models are trained with FlyingThings and FlyingChairs: FlowFormer achieves the highest accuracy but is 70 times slower than NeuFlow. GmFlow achieves 20% higher accuracy than NeuFlow on the FlyingThings dataset and demonstrates similar accuracy on Sintel; however, NeuFlow is 10 times faster. Compared to RAFT (12 iters) and GMA (6 iters), NeuFlow achieves comparable accuracy on both the FlyingThings and Sintel datasets, while being 12 times faster. Additionally, the batch size indicates that NeuFlow consumes less memory.

1/8 Res (12066)	Things (val, clean)	Things (val, final)	RTX 2080 (s)	Jetson Orin Nano (s)	Batch Size (8G)
FlowFormer	0.463	**0.394**	0.819	N/A	2
FlowFormer (small)	1.244	1.111	0.647	N/A	2
GMFlow (1 iter)	**0.434**	0.411	0.114	0.994	8
RAFT (12 iters)	0.574	0.527	0.136	0.830	10
GMA (6 iters)	0.608	0.528	0.142	0.914	6
NeuFlow	0.525	0.518	**0.010**	**0.078**	**56**
1/8 Res (12854)	Sintel (train, clean)	Sintel (train, final)	RTX 2080 (s)	Jetson Orin Nano (s)	Batch Size (8G)
FlowFormer	**0.145**	**0.313**	0.700	N/A	2
FlowFormer (small)	0.195	0.355	0.548	N/A	2
GMFlow (1 iter)	0.188	0.367	0.096	0.816	10
RAFT (12 iters)	0.217	0.365	0.118	0.747	14
GMA (6 iters)	0.198	0.370	0.139	0.733	8
NeuFlow	0.220	0.394	**0.008**	**0.068**	**72**

TABLE II: This table compares latest optical flow methods when outputting 1/8 resolution flow, all models are trained with FlyingThings and FlyingChairs: NeuFlow is optimized for higher accuracy and efficiency at 1/8 resolution flow. We achieve significantly higher accuracy than RAFT (12 iters) and GMA (6 iters) on the FlyingThings dataset. Additionally, NeuFlow is 80 times faster than FlowFormer and 12 times faster than GmFlow, RAFT, and GMA on both GPU platforms.

However, global attention tends to be significantly slow as it computes correlations between all pixels in the image. Due to this heavy computational load, many transformer-based optical flow methods operate at a lower resolution (1/8), but it remains too slow. In our approach, we implement global cross-attention on 1/16 resolution features and stack 2 layers of it. Additionally, we apply Flash-attention [47] for slight speed improvement.

C. Flow Self-Attention

The cross-attention mechanism for pixel matching operates under the assumption that all matching pixels are visible in the image pair. However, this assumption is not always accurate. Out-of-boundary pixels and occluded pixels can violate this assumption, posing a significant ambiguity challenge in optical flow estimation. To address this issue, we incorporate global self-attention on features to globally assess the similarity of pixels. This allows us to propagate unseen flows based on such similarity. Additionally, the implementation of this process can be optimized using flash-attention for improved speed [19], [24].

D. Local Refinement

As the cross-attention at the 1/16 scale of the image has already established the global correspondence of pixels, we focus on local refinement at a larger scale, specifically 1/8 in our case. Initially, we warp the features of image two using the flow computed at 1/16 scale, ensuring that matching pixels in the image pair are located nearby within a small range. To determine the best matching pixel within this range, we compute the local correlation of each pixel in image one with the nearby 7x7 pixels on the warped image two. The feature vector and the estimated coarse flow are also incorporated and fed into deep CNN layers to estimate the delta flow at the 1/8 scale.

E. Upsampling Module

Similar to the latest optical flow methods like GmFlow and RAFT, we adopt a scheme that estimates optical flow at 1/8 resolution and then upsamples the flow to full resolution. The upsampling module resembles RAFT's approach, where each pixel of the high-resolution flow field is determined as

the convex combination of its 9 coarse resolution neighbors using weights predicted by the network. However, instead of utilizing features employed for matching at 1/16 scale and 1/8 scale, we directly extract features from the original image using a simple CNN block, as illustrated in Fig. 4. This approach allows us to obtain feature maps with finer details, thereby sightly enhancing the accuracy of the full-resolution flow, albeit at the expense of additional computation time.

IV. Experiments

A. Training and Evaluation Datasets

The common optical flow training process is typically divided into three stages: Stage one involves training on the FlyingChairs dataset, followed by stage two, which entails training on the FlyingThings dataset. Stage three involves training on a mixed dataset comprising Sintel [48], Kitti [49], and HD1K. We follow the same procedure and compare the results of stage two (FlyingThings), utilizing the training and evaluation code derived from FlowNet 2.

Since stage two only involves training on FlyingChairs and FlyingThings train set, we validate the model on both the FlyingThings test set and the Sintel training set. The FlyingThings dataset presents challenges due to its large displacement, which tests the model's ability to estimate fast-moving optical flow. Meanwhile, validation on the Sintel dataset helps demonstrate the model's cross-dataset generalization, as it has not been trained on data from the same domain.

B. Comparision with Latest Optical Flow Methods

We begin by comparing our method to several state-of-the-art optical flow methods renowned for their superior accuracy: RAFT, GMA, GmFlow, and FlowFormer (Table. I). Unfortunately, due to its computational demands, CRAFT could not be evaluated on our RTX 2080. Our comparison focuses on accuracy differences across the Sintel and FlyingThings datasets, as well as computation time on both the RTX 2080 and the edge computing platform Jetson Orin Nano, considering image sizes of 1024436 for Sintel and 960540 for FlyingThings. The inference batch size is also measured on an 8GB GPU memory to assess the memory usage of different models.

Among these methods, FlowFormer achieves the highest accuracy. However, it is significantly slower than our approach, being approximately 70 times slower on the RTX 2080. Although FlowFormer offers a smaller model version, it fails to perform adequately on the large displacement dataset FlyingThings and remains approximately 60 times slower than our method. Unfortunately, FlowFormer couldn't run on the Jetson Orin Nano as it caused the machine to crash.

GmFlow exhibits slightly better accuracy than our method on datasets with large displacement (FlyingThings) due to its global attention mechanism operating on a 1/8 image scale, whereas ours operates on a 1/16 scale. Specifically, GmFlow shows a 20% accuracy improvement over our method on the FlyingThings dataset and comparable performance on the Sintel dataset. However, our method excels in efficiency, being roughly 10 times faster than GmFlow on both GPU platforms across various resolutions.

Both RAFT and GMA utilize multiple iterations to refine optical flow estimates. We consider 12 iterations for RAFT and 6 iterations for GMA, as they achieve similar accuracy compared to our method at such iterations. Notably, on the FlyingThings dataset, characterized by large displacement, our method outperforms both RAFT and GMA on clean sets and performs comparably on final sets. On the Sintel dataset, where pixel movement is relatively smaller, our method achieves similar accuracy. These results underscore the effectiveness of our global attention mechanism in handling large displacement scenarios, while also exhibiting a 12 times speedup over RAFT and GMA on both GPUs at various resolutions. To ensure fair comparison of computation time, we disable mixed precision computation.

C. Comparison on 1/8-resolution Flow

In certain applications such as SLAM, it's common practice to use 1/8 image resolution instead of full resolution to reduce computational burden for downstream operations. Recent optical flow methods adopt the same approach to obtain 1/8 resolution flow and employ an upsampling module to provide full resolution flow. Our approach follows this scheme and is specifically optimized for 1/8 image resolution to meet real-time requirements on edge computing platforms.

Table. II illustrates the comparison results. Compared to full resolution, our approach achieves relatively higher accuracy with less computation time. For example, on the FlyingThings dataset, we significantly outperform RAFT (12 iters) and GMA (6 iters) on 1/8 resolution flow, whereas our performance advantage is not as pronounced on full resolution flow.

D. Comparision with Local Regression-based Optical Flow Methods

We then compared our approach with local regression-based CNN optical flow methods, encompassing popular methods such as FlowNet 2, PWC-Net, and the LiteFlownet series (LiteFlownet 1, LiteFlownet 2, LiteFlownet 3) (Table. III). Due to the limited receptive field of CNNs and the constrained range of local correlation, none of these methods performed adequately on the large displacement dataset FlyingThings, despite being trained on it. Our approach also consistently outperforms these local regression optical flow methods on Sintel dataset, demonstrating an approximate 40% accuracy improvement.

Local regression-based optical flow methods generally offer faster speeds compared to the latest high-accuracy optical flow methods. To ensure a fair comparison of computation time with our approach, we opted to use their PyTorch implementations instead of Caffe. The results reveal that PWC-Net emerges as the fastest method among these local regression approaches. However, it also exhibits the lowest accuracy and remains approximately 3 times slower than our

Full Res (960540)	Things (val, clean)	Things (val, final)	RTX 2080 (s)
FlowNet 2 (pytorch)	6.782	6.774	0.091
PWC-Net (pytorch)	8.098	8.168	0.033
LiteFlowNet (pytorch)	9.033	9.018	0.072
NeuFlow	**3.846**	**3.828**	**0.013**
Full Res (1024436)	Sintel (train, clean)	Sintel (train, final)	RTX 2080 (s)
FlowNet 2 (pytorch)	2.222	3.639	0.085
PWC-Net (pytorch)	2.643	4.060	0.029
LiteFlowNet (pytorch)	2.588	4.058	0.059
NeuFlow	**1.660**	**3.126**	**0.011**

TABLE III: This table compares NeuFlow with local regression-based optical flow methods, all models are trained with FlyingThings and FlyingChairs: NeuFlow consistently demonstrates a significant advantage in both accuracy and efficiency on both the FlyingThings and Sintel datasets. Inference time is measured using PyTorch implementations of each model.

Full Res (960540)	Things (val, clean)	Things (val, final)	RTX 2080 (s)
LiteFlowNet 2	10.395	10.205	N/A
LiteFlowNet 3 (pytorch)	9.856	9.692	0.050
NeuFlow	**4.044**	**4.025**	**0.013**
Full Res (1024436)	Sintel (train, clean)	Sintel (val, final)	RTX 2080 (s)
LiteFlowNet 2	1.559	1.944	N/A
LiteFlowNet 3 (pytorch)	1.451	1.920	0.042
NeuFlow	**0.987**	**1.294**	**0.011**

TABLE IV: This table compares NeuFlow with LiteFlowNet 2 and 3. As these models do not provide models trained solely on the C+T dataset, we compare them with models trained with mixed datasets. NeuFlow consistently demonstrates a significant advantage in both accuracy and efficiency on both the FlyingThings and Sintel datasets.

approach on the RTX 2080. FlowNet 2, despite its better accuracy, operates around 8 times slower than our approach.

The LiteFlowNet series also lags behind in speed, with a factor of 4 to 6 slower than ours. Only LiteFlowNet 1 provides a model trained specifically with the FlyingThings dataset (Stage 2), whereas LiteFlowNet 2 and 3 offer models trained with mixed datasets (Stage 3). Additionally, Table. IV illustrates that LiteFlowNet 2 and 3 fail to perform adequately on the large displacement dataset (FlyingThings), even though they were trained on it. We trained our model on mixed datasets (Stage 3) for a fair comparison, and we achieved a significant advantage in both accuracy and computation time on the RTX 2080. While we couldn't find the PyTorch implementation for LiteFlowNet 2, published data suggests that its speed is comparable to that of PWC-Net. Unfortunately, we failed to build these methods on Jetson Orin Nano due to its support for only the latest PyTorch version.

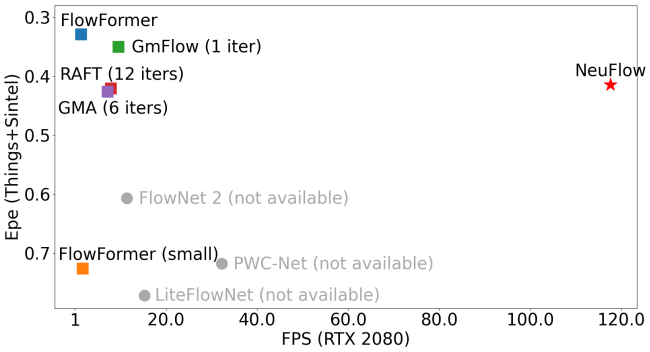

Fig. 5: End point error (EPE) v.s. frame per second (FPS) on Nvidia RTX 2080 while outputting 1/8 resolution flow. All models trained solely on FlyingThings and FlyingChairs. NeuFlow is optimized for accuracy and efficiency at 1/8 resolution, thus we gain more advantage compared to full resolution flow.

E. Overall Comparison

We plot the end-point error (EPE) versus frames per second (FPS) throughput on Nvidia RTX 2080 (see Fig. 1). Each point represents an optical flow method. All models are trained with FlyingChairs and FlyingThings datasets. EPE is measured by averaging EPE values of Things Clean, Things Final, Sintel Clean, and Sintel Final equally. Inference time is measured by averaging the running time on Things images (960540) and Sintel images (1024436). We observe that NeuFlow achieves comparable accuracy to the latest optical flow methods while being 10-70 faster. Compared to local regression-based methods, we have significant advantages in both accuracy and efficiency. Since we optimize for 1/8 resolution flow, we gain more speed advantage and accuracy compared to full resolution (see Fig. 5) Since local regression-based optical flow methods do not adhere to the same scheme of high accuracy 1/8 flow followed by an 8x upsampling module, resulting in lower accuracy for 1/8 resolution flow, we have omitted them from this plot.

F. Inference Time on Jetson Orin Nano

As our ultimate goal is to develop an optical flow method capable of real-time performance on edge computing platforms, we measured the inference frames per second (FPS) on Jetson Orin Nano at various image resolutions (Table. V). For applications utilizing 1/8 resolution optical flow, such as

Frames per second (FPS)	Things 1 (960540)	Sintel 1 (1024436)	Chairs 1 (512x384)
Inference on 2 frames	10.3	11.9	25.0
Inference on 1 frame	11.9	13.9	29.9
Frames per second (FPS)	Things 1/8 (12066)	Sintel 1/8 (12854)	Chairs 1/8 (6448)
Inference on 2 frames	12.8	14.7	29.4
Inference on 1 frame	15.4	17.9	36.4

TABLE V: NeuFlow achieves real-time performance on Jetson Orin Nano for specific image resolutions, with 1/8 resolution flow offering faster inference times. In image stream processing, only one frame needs backbone computation, as the other frame is already computed in the preceding one.

SLAM, we also measured the FPS of it. Since most vision applications process image streams and estimate optical flow on continuous frames, the backbone of the previous frame has already been computed, which can be utilized for the optical flow estimation of the next continuous two frames. Therefore, we also measured the FPS when feeding only one frame into the backbone neural network. The results show that we are able to achieve around 30 FPS on smaller images when outputting full resolution optical flow, and 36 FPS on larger images when outputting 1/8 resolution flow.

V. Conclusions and Future Work

In this paper, we proposed a novel optical flow architecture called NeuFlow, which enables real-time optical flow estimation on edge computing platforms like Jetson Orin Nano. NeuFlow is 10-80 faster than the latest optical flow methods, with comparable accuracy on both the FlyingThings and Sintel datasets. Therefore, NeuFlow ensures better performance across various use cases. We have released the code and model weights of NeuFlow (https://github.com/neufieldrobotics/NeuFlow) to allow the community full access to use, modify and experiment with as they see fit.

However, we also recognize that sacrificing some computation time for higher accuracy may be necessary for certain users. Conversely, further improvement in efficiency is also possible. Thus, many options are there to extend the model and achieve higher accuracy or higher efficiency, which we leave as future work.

Higher Accuracy. Extending the model in various ways can be effective, such as expanding the feature dimension (ours is 90), stacking more cross-attention layers, increasing the depth of the CNN backbone, or adding CNN layers in the local refinement step. Additionally, iterative refinement is also an option to improve accuracy.

Local refinement can also be applied at 1/16 resolution to refine flow at this lower resolution, which can propagate improved flow to higher resolutions. Moreover, global attention and local refinement can be utilized at higher resolutions. For instance, similar to GmFlow, one can perform global cross-attention at 1/8 resolution and refine at 1/4 resolution, promising improved accuracy across all resolution flows.

Higher Efficiency. Our approach consistently employs native CNN architectures, yet there are several more efficient CNN architectures available that can further enhance efficiency. For instance, MobileNets [50], [51], [52] leverage depthwise separable convolutions to construct lightweight deep neural networks, while ShuffleNet [53] utilizes pointwise group convolution and channel shuffle techniques to reduce computation costs while maintaining accuracy.

Other techniques, such as NVIDIA TensorRT, offer low latency and high throughput for optimized runtime performance. Mixed precision techniques, using 16-bit or lower precisions during training and inference, can significantly speed up inference and reduce memory usage on modern GPUs. Network pruning is also effective in reducing the size of heavy networks by removing redundant parameters while maintaining comparable accuracy.

References

[1] D. Fortun, P. Bouthemy, and C. Kervrann, "Optical flow modeling and computation: A survey," *Computer Vision and Image Understanding*, vol. 134, pp. 1–21, 2015.

[2] J. Shin, S. Kim, S. Kang, S.-W. Lee, J. Paik, B. Abidi, and M. Abidi, "Optical flow-based real-time object tracking using non-prior training active feature model," *Real-time imaging*, vol. 11, no. 3, pp. 204–218, 2005.

[3] K. Kale, S. Pawar, and P. 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)*. IEEE, 2015, pp. 1–6.

[4] J. K. Aggarwal and Q. Cai, "Human motion analysis: A review," *Computer vision and image understanding*, vol. 73, no. 3, pp. 428–440, 1999.

[5] I. Saleemi, L. Hartung, and M. Shah, "Scene understanding by statistical modeling of motion patterns," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 2069–2076.

[6] Z. Teed and J. Deng, "Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras," *Advances in neural information processing systems*, vol. 34, pp. 16 558–16 569, 2021.

[7] P. Muller and A. Savakis, "Flowdometry: An optical flow and deep learning based approach to visual odometry," in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2017, pp. 624–631.

[8] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.

[9] J. J. Gibson, "The perception of the visual world." 1950.

[10] ——, "The senses considered as perceptual systems." 1966.

[11] Y. Liu and J. Miura, "Rdmo-slam: Real-time visual slam for dynamic environments using semantic label prediction with optical flow," *Ieee Access*, vol. 9, pp. 106 981–106 997, 2021.

[12] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger, "Elasticfusion: Real-time dense slam and light source estimation," *The International Journal of Robotics Research*, vol. 35, no. 14, pp. 1697–1716, 2016.

[13] A. Behl, O. Hosseini Jafari, S. Karthik Mustikovela, H. Abu Alhaija, C. Rother, and A. Geiger, "Bounding boxes, segmentations and object coordinates: How important is recognition for 3d scene flow estimation in autonomous driving scenarios?" in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2574–2583.

[14] P. Jain, J. Manweiler, and R. Roy Choudhury, "Overlay: Practical mobile augmented reality," in *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*, 2015, pp. 331–344.

[15] H. Chao, Y. Gu, and M. Napolitano, "A survey of optical flow techniques for robotics navigation applications," *Journal of Intelligent & Robotic Systems*, vol. 73, pp. 361–372, 2014.

[16] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2758–2766.

[17] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2462–2470.

[18] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer, 2020, pp. 402–419.

[19] H. Xu, J. Zhang, J. Cai, H. Rezatofighi, and D. Tao, "Gmflow: Learning optical flow via global matching," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 8121–8130.

[20] Z. Huang, X. Shi, C. Zhang, Q. Wang, K. C. Cheung, H. Qin, J. Dai, and H. Li, "Flowformer: A transformer architecture for optical flow," in *European conference on computer vision*. Springer, 2022, pp. 668–685.

[21] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *IJCAI'81: 7th international joint conference on Artificial intelligence*, vol. 2, 1981, pp. 674–679.

[22] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, pp. 91–110, 2004.

[23] C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. T. Freeman, "Sift flow: Dense correspondence across different scenes," in *Computer Vision–ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part III 10*. Springer, 2008, pp. 28–42.

[24] S. Jiang, D. Campbell, Y. Lu, H. Li, and R. Hartley, "Learning to estimate hidden motions with global motion aggregation," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 9772–9781.

[25] H. Xu, J. Zhang, J. Cai, H. Rezatofighi, F. Yu, D. Tao, and A. Geiger, "Unifying flow, stereo and depth estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[26] S. Zhao, L. Zhao, Z. Zhang, E. Zhou, and D. Metaxas, "Global matching with overlapping attention for optical flow estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 17 592–17 601.

[27] X. Sui, S. Li, X. Geng, Y. Wu, X. Xu, Y. Liu, R. Goh, and H. Zhu, "Craft: Cross-attentional flow transformer for robust optical flow," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, 2022, pp. 17 602–17 611.

[28] X. Shi, Z. Huang, D. Li, M. Zhang, K. C. Cheung, S. See, H. Qin, J. Dai, and H. Li, "Flowformer++: Masked cost volume autoencoding for pretraining optical flow estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 1599–1610.

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[30] Z. Tu, W. Xie, D. Zhang, R. Poppe, R. C. Veltkamp, B. Li, and J. Yuan, "A survey of variational and cnn-based optical flow techniques," *Signal Processing: Image Communication*, vol. 72, pp. 9–24, 2019.

[31] A. Ranjan and M. J. Black, "Optical flow estimation using a spatial pyramid network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4161–4170.

[32] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8934–8943.

[33] T.-W. Hui, X. Tang, and C. C. Loy, "Liteflownet: A lightweight convolutional neural network for optical flow estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8981–8989.

[34] ——, "A lightweight optical flow cnn—revisiting data fidelity and regularization," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 8, pp. 2555–2569, 2020.

[35] T.-W. Hui and C. C. Loy, "Liteflownet3: Resolving correspondence ambiguity for more accurate optical flow estimation," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*. Springer, 2020, pp. 169–184.

[36] G. Yang and D. Ramanan, "Volumetric correspondence networks for optical flow," *Advances in neural information processing systems*, vol. 32, 2019.

[37] J. Xu, R. Ranftl, and V. Koltun, "Accurate optical flow via direct cost volume processing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1289–1297.

[38] H. Jiang and E. Learned-Miller, "Dcvnet: Dilated cost volume networks for fast optical flow," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 5150–5157.

[39] H. Xu, J. Yang, J. Cai, J. Zhang, and X. Tong, "High-resolution optical flow from 1d attention and correlation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 498–10 507.

[40] S. Jiang, Y. Lu, H. Li, and R. Hartley, "Learning optical flow from a few matches," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 16 592–16 600.

[41] F. Zhang, O. J. Woodford, V. A. Prisacariu, and P. H. Torr, "Separable flow: Learning motion cost volumes for optical flow estimation," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 807–10 817.

[42] M. Hofinger, S. R. Bulò, L. Porzi, A. Knapitsch, T. Pock, and P. Kontschieder, "Improving optical flow on a pyramid level," in *European Conference on Computer Vision*. Springer, 2020, pp. 770–786.

[43] J. Hur and S. Roth, "Iterative residual refinement for joint optical flow and occlusion estimation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 5754–5763.

[44] J. Wang, Y. Zhong, Y. Dai, K. Zhang, P. Ji, and H. Li, "Displacement-invariant matching cost learning for accurate optical flow estimation," *Advances in Neural Information Processing Systems*, vol. 33, pp. 15 220–15 231, 2020.

[45] P. Truong, M. Danelljan, and R. Timofte, "Glu-net: Global-local universal network for dense flow and correspondences," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 6258–6268.

[46] N. Mayer, E. Ilg, P. Hausser, 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4040–4048.

[47] T. Dao, D. Fu, S. Ermon, A. Rudra, and C. Ré, "Flashattention: Fast and memory-efficient exact attention with io-awareness," *Advances in Neural Information Processing Systems*, vol. 35, pp. 16 344–16 359, 2022.

[48] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part VI 12*. Springer, 2012, pp. 611–625.

[49] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3061–3070.

[50] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[51] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[52] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan *et al.*, "Searching for mobilenetv3," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1314–1324.

[53] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6848–6856.