DeepLiDARFlow: A Deep Learning Architecture For Scene Flow Estimation Using Monocular Camera and Sparse LiDAR

Rishav[*,1,2] Ramy Battrawy[*,1] René Schuster[1] Oliver Wasenmüller[1,3] Didier Stricker[1,4]

***Abstract*— Scene flow is the dense 3D reconstruction of motion and geometry of a scene. Most state-of-the-art methods use a pair of stereo images as input for full scene reconstruction. These methods depend a lot on the quality of the RGB images and perform poorly in regions with reflective objects, shadows, ill-conditioned light environment and so on. LiDAR measurements are much less sensitive to the aforementioned conditions but LiDAR features are in general unsuitable for matching tasks due to their sparse nature. Hence, using both LiDAR and RGB can potentially overcome the individual disadvantages of each sensor by mutual improvement and yield robust features which can improve the matching process. In this paper, we present DeepLiDARFlow, a novel deep learning architecture which fuses high level RGB and LiDAR features at multiple scales in a monocular setup to predict dense scene flow. Its performance is much better in the critical regions where image-only and LiDAR-only methods are inaccurate. We verify our DeepLiDARFlow using the established data sets KITTI and FlyingThings3D and we show strong robustness compared to several state-of-the-art methods which used other input modalities. The code of our paper is available at https://github.com/dfki-av/DeepLiDARFlow.**

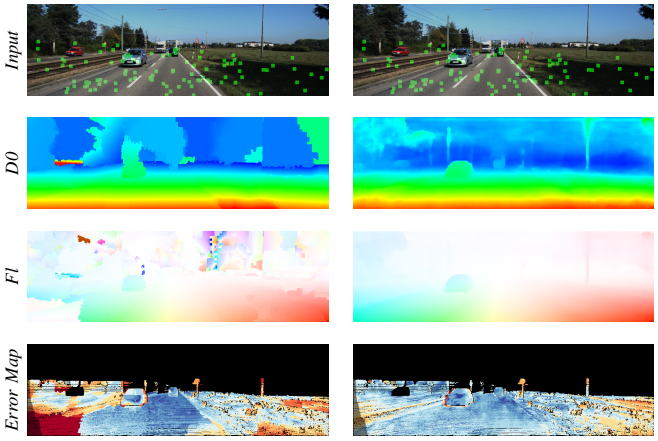
I. Introduction

Robust understanding about 3D geometry and dynamic changes in the environment is very important for autonomous vehicles, robot navigation, advanced driver assistance systems and so on. In this context, scene flow estimation is an essential task which aims to the reconstruct 3D geometry as well as 3D motion of each observed point in the entire scene. Hence, dense scene flow enriches the perceptual information which makes it very useful to increase the reliability of autonomous systems.

Most of the popular scene flow methods use stereo images. But there is an inherent disadvantage with image-based methods because they depend completely on the quality of the image. Therefore, scene flow estimation gets extremely difficult if the images contain insufficient details in some regions due to reflective surfaces, shadows, bad illumination and many more.

LiDAR sensors are much less sensitive to the aforementioned environmental conditions. Thus, they can possibly act as anchor points in the regions where the RGB features are not robust. A problem with LiDAR sensors is that they get expensive as the density of points they provide increases. Hence an ideal method should be able to work with very sparse LiDAR measurements in order to ensure its cost effectiveness. The fusion of high level features of RGB images and robust features of LiDAR measurements for dense matching can potentially result in highly accurate scene flow estimates even under bad environmental conditions. Recently, Battrawy et al. [2] proposed a conventional approach that fuses LiDAR measurements into stereo images for dense scene flow estimation. They show impressive improvement compared to a stereo-only setup, however, their approach is computationally inefficient. Thus, we propose an end-to-end learning-based approach to estimate dense scene flow from sparse LiDAR and RGB images (see Fig. 1). To the best of our knowledge, our DeepLiDARFlow is the first approach that uses the fusion of sparse LiDAR measurements and RGB images in a monocular setup for dense scene flow estimation.

Our DeepLiDARFlow learns high level features of sparse LiDAR measurements and RGB images at multiple scales and fuses them into each other in an end-to-end learning-based fashion. It aims to resolve critical regions of bad illumination, shadows, reflective objects in RGB image and produce robust features for matching. Overall, the main contributions of this work are:



Conventional approach [2] Our DeepLiDARFlow

Fig. 1: We introduce our DeepLiDARFlow, a novel deep learning architecture which fuses a monocular image and the corresponding sparse LiDAR measurements (shown as green spots on input image) for dense scene flow estimation. For very sparse LiDAR (∼100 points), our DeepLiDARFlow outperforms comfortably the conventional scene flow approach (monocular version of [2]) which employs such a fusion.

*Equal contribution
[1]German Research Center for Artificial Intelligence - DFKI, Kaiserslautern, Germany: firstname.lastname@dfki.de
[2]Birla Institute of Technology and Science - BITS Pilani, Pilani, India: f2016108@pilani.bits-pilani.ac.in
[3]University of Applied Sciences Mannheim, Mannheim, Germany: o.wasenmueller@hs-mannheim.de
[4]University of Kaiserslautern - TUK, Kaiserslautern, Germany

- A novel deep learning strategy for dense scene flow estimation by fusing sparse LiDAR and RGB images in a monocular setup.
- A novel multi-scale fusion strategy of RGB and LiDAR features for dense scene flow estimation.
- Exhaustive experiments, showing the superior performance of our DeepLiDARFlow over image-based methods in critical regions with reflective objects, bad illumination, etc.
- Overall competitive and robust results against different state-of-the-art algorithms which use other input modalities.

II. Related Work

A. Image-based Scene Flow

Most of the image-based scene flow methods utilize a pair of stereo images at two time steps, like the early variational methods [5], [15], [17], [20]. The improvements brought about by deep convolutional neural networks (CNNs) for various computer vision tasks [13], [19] over traditional approaches, were successfully transferred to dense pixel-wise matching tasks. FlowNet [6] is the first deep learning method developed to predict dense optical flow. SceneFlowNet [16] is the first to use an end-to-end deep learning approach for scene flow using a pair of stereo images. Recently, PWOC-3D [26], DWARF [1] and SENSE [18] propose light weight end-to-end networks which operate with the stereo image setup. DRISF [22] applies piece-wise rigid planes model [29] and employs a combination of deep learning and conventional approaches.

As alternative to the stereo setup, some methods use RGB-D cameras for dense scene flow estimation [11], [14], [25]. Qiao et al. [24] are the first to develop a deep learning method that utilizes RGB-D images for scene flow estimation. However, RGB-D setup performs poorly for outdoor scene flow estimations due to sensor range limitations. Approaches like [27], [30], [31], [32] use the power of multi-task CNNs by poising scene flow estimation from monocular images as a problem of single view depth and optical flow estimation. Mono-SF [4] is a recent method that jointly estimates the 3D structure and motion of the scene by combining multi-view geometry and single-view depth information. Unlike most of these methods, our DeepLiDARFlow solves the scene flow problem as a whole in an end-to-end fashion.

The major problem with purely image-based approaches is their heavy reliance on the image quality. These methods usually perform poorly in critical image regions with poor illumination, shadows or reflective objects. These are the regions where robust measurements from a LiDAR sensor are extremely useful. Our DeepLiDARFlow takes the advantage of these measurements and fuses them into the image domain to improve scene flow estimates.

B. LiDAR-only Scene Flow

Some methods use point clouds to directly estimate scene flow. In this context, FlowNet3D [21] is among the first to propose a neural network architecture which utilize point clouds only. PointFlowNet [3] uses a highly compartmentalized architecture to estimate scene flow from point clouds. HPLFlowNet [12] takes inspiration from Bilateral Convolutional Layers (BCL) [9] that restore structural information from unstructured point clouds. The two major problems with LiDAR-only approaches are the difficulty of matching unstructured data and the inherent low resolution compared to cameras. Our DeepLiDARFlow overcomes the individual disadvantages of each sensor by mutual improvement, hence proposing a novel sensor setup with strong potential for robust and accurate dense scene flow predictions.

C. LiDAR and Image-based Scene Flow

Recently, scene flow estimation in a heterogeneous sensor environment was proposed by LiDAR-Flow [2]. However, this work focuses on considerable dense scene flow improvement over stereo-only approach by using a pair of stereo images and LiDAR measurements. Different from LiDAR-Flow, our DeepLiDARFlow aims to resolve the stereo camera dependency entirely by the fusion of a monocular camera and a LiDAR sensor. This is a much more challenging task, because the LiDAR information can not just be used to resolve ambiguous image cues, but is the only source of 3D information of the scene. Therefore to obtain a dense scene flow result, the sparse LiDAR measurements need to be converted into a dense representation. To the best of our knowledge, DeepLiDARFlow is the first approach that explores to this sensor with monocular setup for dense scene flow.

III. Method

For scene flow estimation, the input of our DeepLiDARFlow is RGB images (I^t, I^{t+1}) and the corresponding LiDAR measurements (D^t, D^{t+1}) at two consecutive time steps t and $t+1$ respectively. Our DeepLiDARFlow fuses the high level features of I^t, I^{t+1} and D^t, D^{t+1} to predict dense scene flow through three main modules: The feature extraction module, the fusion module, and the scene flow module. The following sections describe each module in details.

A. Feature Extraction Module

The feature extraction module consists of four multi scale feature pyramid networks for I^t, I^{t+1}, D^t, and D^{t+1}. PWOC-3D [26] uses a feature pyramid network to extract features with strong semantics and localization at multiple scales. Having features at multiple scales helps in tackling problems like large motion for dense pixel-wise matching. The pyramids of RGB and LiDAR input are similar in structure to the feature pyramid network in PWOC-3D. However, feature extraction from LiDAR data differs in the set of operations and layers we use. In [28] it was shown that regular convolution fails to perform equivalently with varying density or pattern of sparse input. Therefore, sparsity-aware convolution is proposed, which uses a binary sparsity mask for normalization. As a further development,

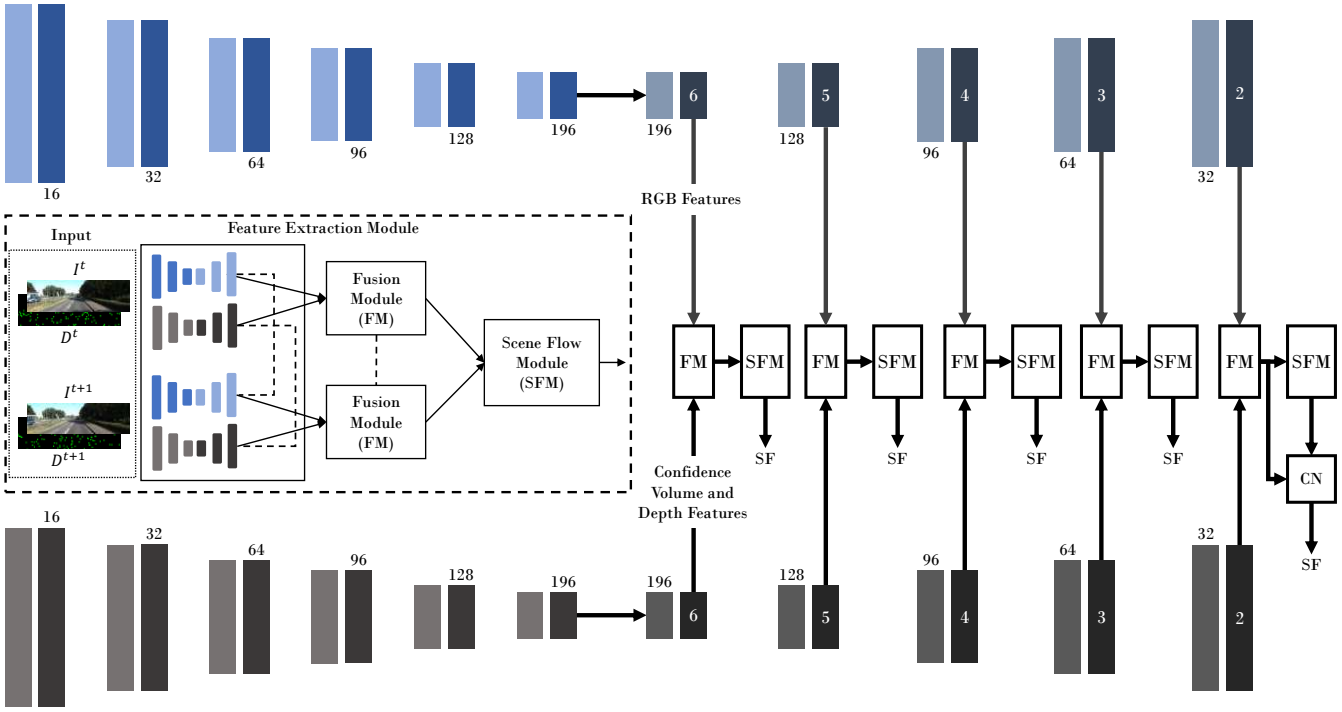


Fig. 2: Detailed architecture of our DeepLiDARFlow. Residual connections from the feature pyramid are omitted for clarity. RGB features from the RGB pyramid and confidence volume along with the depth features from the confidence pyramid are sent to Fusion Module (FM). The fused features are then sent to the Scene Flow Module (SFM). The numbers 2 to 6 denote the levels which are used for multi-scale prediction. The output of level 2 is refined in the context network (CN) to form the final scene flow. The number of channels are same for blocks of similar size.

Eldesokey et al. [8] propose a confidence convolution which uses differentiable confidence volumes to indicate sparsity and the reliability of the densification. We use the same concept of confidence convolution [8], max-confidence pooling (for down-sampling), and nearest neighbor up-sampling to account for the sparse nature of LiDAR measurements during feature extraction. The resolution of features is halved at each level of the pyramid and each level consists of two convolutions. All pyramids have 6 levels, hence the final map is of $\frac{1}{64}$ resolution of the original input. Afterwards, the features are successively decoded and up-sampled until $\frac{1}{4}$ of the input resolution is reached again. The final features at a particular level l are denoted by i_l^t, i_l^{t+1}, d_l^t, and d_l^{t+1} for RGB and LiDAR input at the two time steps respectively. Feature pyramids for either the two RGB images or the two LiDAR measurements share their weights. Fig. 2 presents more details of the feature extraction module.

B. Fusion Module

The fusion of heterogeneous RGB and depth information is an important part of our approach. On the one hand, the depth information from the LiDAR feature is supposed to refine the image features to improve dense matching. On the other hand, dense RGB information is used to guide the densification of the sparse LiDAR measurements to obtain a dense depth representation. Previous work [7], [8] experimented with early and late fusion, of which the late fusion strategy was shown to perform better. Our DeepLiDARFlow builds on this finding and extends the late fusion of high level RGB and LiDAR features into a multi scale late fusion and prediction strategy. With increasing level l, d_l^t (and d_l^{t+1}) get more and more dense and semantically strong, but there is only little structural information depending on the density of the LiDAR input. The RGB features i_l^t and i_l^{t+1} are rich in structural information. The fusion module is responsible for the combination of structured RGB and unstructured LiDAR features to produce high level features for matching. These features combine the robustness and accuracy of LiDAR measurements and the rich textural and structural information from RGB images. The features at a particular level l (i_l^t and d_l^t, same for i_l^{t+1} and d_l^{t+1}) are concatenated along the channel dimension into a feature volume which then goes through several convolutions while maintaining the input spatial dimensions (see Fig. 3). The fusion is performed for levels $l = 6$ till $l = 2$, i.e. there is continuous fusion during the top down branch of the feature pyramids. The fusion modules for the two time steps t and $t+1$ share their weights. The fused features at a level l are denoted as f_l^t and f_l^{t+1} (the features of the reference frame and the next frame respectively). These features are then sent to the scene flow module for final prediction of dense scene flow on each scale.

To give meta-guidance to the fusion module, the confidence volumes of the LiDAR features are concatenated with

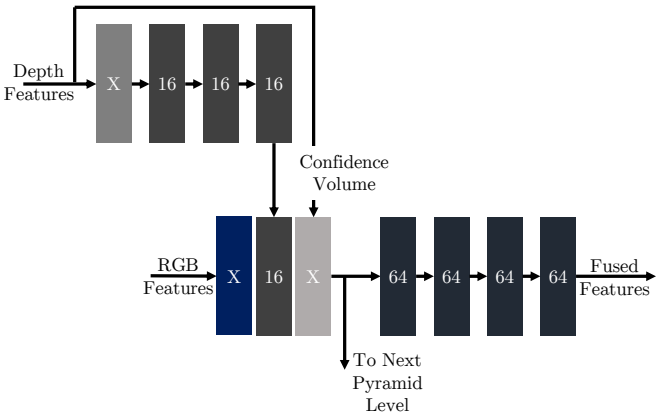


Fig. 3: The fusion module at a particular scale level. The depth features obtained from the confidence pyramid go through a series of convolutions as a preprocessing step before finally being used for fusion. X denotes the number of channels in the feature pyramid for that specific level. The fused features go through a series of convolutions to ensure proper mixing of the two heterogeneous information.

the (raw) RGB features before fusion (see Fig. 2), at each scale ($l = 6$ to $l = 2$). Since the confidence is a reliability measure of the depth features, this way, the fusion module is more flexible in how the two heterogeneous feature maps are fused. The improvement brought about the concatenation of confidence maps is proved with the help of an ablation study discussed in Section IV-C.

C. Scene Flow Module

The scene flow module (Fig. 4) is the final component of DeepLiDARFlow. At any particular level l ($l = 2$ to $l = 6$), it comprises of a warping layer, a cost volume layer, and the scene flow estimator. The blocks mentioned above differ from the ones used in PWOC-3D [26] in the following aspects. The input to this module are f_l^t and f_l^{t+1}, i.e., the output features from the respective fusion modules. Only a single 2-dimensional warping operation is needed, which warps f_l^{t+1} towards f_l^t to form w_l^{t+1}. w_l^{t+1} and f_l^t are fed to the cost volume layer, which computes a 2D cost volume (denoted as c_l) in the same way as PWOC-3D [26]. c_l, w_l^{t+1}, and f_l^t are then concatenated and given as input to the scene flow estimator which predicts the final, dense 4D scene flow at level l. When the final level $l = 2$ is reached, the dense prediction is further refined with a residual prediction from the context network. The context network gets f_l^{t+1}, f_l^t and the last level features from the scene flow estimator as input. Fig. 4 gives a schematic view of the entire module. Note that at $l = 6$, i.e. the lowest resolution, there is no previous flow estimate. Instead the initial flow is assumed to be zero, resulting in no warping, i.e. $w_6^{t+1} = f_6^{t+1}$.

IV. Experiments

We conduct several experiments to verify the results of our DeepLiDARFlow. Firstly, we verify our design decisions. Secondly, we show the robustness of our DeepLiDARFlow compared to state-of-the approaches and we investigate finally the performance over different sparsity levels of the LiDAR, compared to a conventional approach.

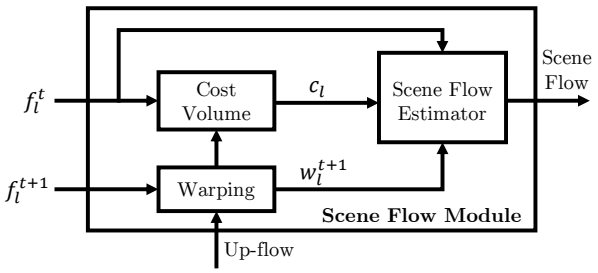


Fig. 4: The scene flow module. f_l^t ,f_l^{t+1} are the fused features of the frames at time t and $t+1$. Up-flow denotes the optical flow estimate from the previous level. This module further comprises of the warping layer, the cost volume layer and the scene flow estimator. The output of the last level is further refined by a context network as in [26].

A. Data Sets and Evaluation Metrics

Data Sets: Since the prime objective of this work is to predict scene flow for autonomous systems, KITTI [10] is a direct choice. The train set of KITTI consists of 200 consecutive frames with ground truth of scene flow aligned to a reference frame at time step t and represented as optical flow components (Fl) and disparity maps (D0) and (D1) for a consecutive time steps t and $t+1$ respectively. The disparity maps are generated using a high resolution LiDAR sensor and projected into image coordinate as disparity maps by using the calibration parameters. We de-warp the LiDAR frame of time step $t + 1$ to mimic the real capture of the second LiDAR frame same as in [2]. However, the established LiDAR frames are insufficient for training a deep neural network, hence, for all our experiments, we first pre-train our DeepLiDARFlow on FlyingThings3D (FT3D) [23] and then fine tune it on KITTI [10]. We split the train set of KITTI into training and validation splits and we conduct the same validation frames to the stat-of-the art methods mentioned in Section IV-D for a fair comparison.

Evaluation Metrics: We split our metrics into two categories: *Dense scene flow evaluation* – We compute the average KITTI outlier rate for scene flow (SF) and its components (i.e. (D0), (D1) and (Fl)) over all pixels for which the endpoint error is > 3 pixels and the relative error is > 5 % compared to the ground truth. Additionally, the euclidean distance (the endpoint error (SF-EPE)) is computed over all scene flow components. *Sparse scene flow evaluation* – Same thresholds as in dense evaluation are used to compute the outlier rate of optical (Fl) and we also compute the endpoint error for 2D optical flow (Fl-EPE) but only for the sparse input. In addition to these metrics, we consider 3D space metrics by projecting the input points and the measured displacement of scene flow as well as the ground truth into 3D Cartesian points. The average outlier rate of scene flow (SF-3D) is computed over all 3D input points

whose euclidean distance to ground truth (endpoint error (SF-EPE-3D)) is > 0.3 meters and the relative error is > 10 %.

B. Implementation and Training

Since FT3D and KITTI data sets have dense disparity maps, we use a uniform random sampling strategy to sample disparity points. Most real LiDAR sensors have some inherent amount of noise and to mimic this characteristic, some noise is simulated into the sampled depth points during training and fine tuning. Additionally, we apply the same data augmentation as in [6] for the RGB input. For training our architecture, we use the hyper-parameters and a multi-level losses as in [26]. Noticeable, when trained with a fixed number of LiDAR points, the accuracy of DeepLiDARFlow is deteriorated a lot when testing with differently dense LiDAR input which is not a desirable characteristic. To overcome this problem, we generalize our model across different sparsity levels of LiDAR (i.e. resolutions) by varying the number of LiDAR samples on the fly (points are varied from 0.2% to 20% of full density) for both pre-training and fine-tuning. Using this strategy, our DeepLiDARFlow is able to achieve an almost constant accuracy for a wide range of LiDAR points.

C. Design Decisions

Our DeepLiDARFlow handles simultaneously the densification of LiDAR features to predict the dense scene flow. In this context, there are several questions that may come up, *do we really need confidence convolution? Do we need to concatenate the confidences during fusion?*. To answer these questions, we conduct an experiment where the LiDAR pyramid uses regular convolution layers (i.e. no confidence convolution) and the results with this case are worse than when using confidence convolutions as shown in Table I. In the fusion module, confidence volumes are concatenated to the RGB and LiDAR features. These confidence volumes act as meta-guidance to the fusion module, this also improves the final results as shown in Table I.

D. Robustness and Comparison to State-of-the-Art

Since our DeepLiDARFlow claims that a fusion approach can yield robust estimates as compared to image-only and LiDAR-only approaches, we compare its performance to several state-of-the-art methods which utilize different input modalities. We verify the run time in milliseconds (ms) of our method compared to other methods using a GeForce GTX 1080 Ti.

TABLE I: Ablation study on various design choices for our DeepLiDARFlow. We test all variants 5000 points as LiDAR input on our test splits of FT3D and KITTI.

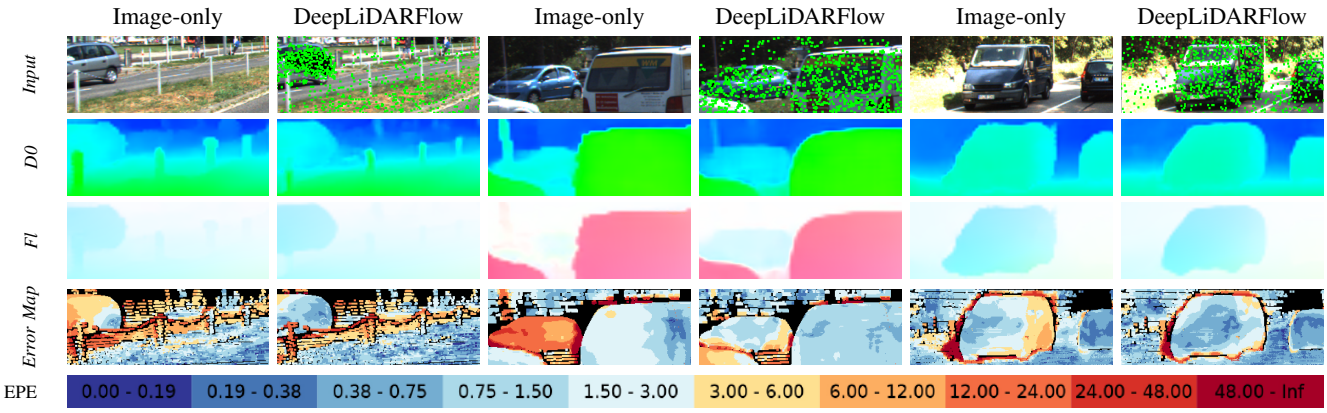
Confidence Convolution	Confidence Concatenation	FT3D [23]		KITTI [10]	
		SF	SF-EPE	SF	SF-EPE
✗	✗	30.97	8.77	16.33	3.75
✓	✗	21.83	8.70	16.31	4.05
✓	✓	**20.20**	**7.64**	**13.77**	**3.67**

One of the main advantages of using LiDAR as an input for scene flow methods is its robustness. Image-based methods rely heavily on the quality of the image and hence often fail in regions of the image which contain mirror-like reflections, ill-conditioned environment etc. For the qualitative results, we visualize multiple examples in Fig. 5 to verify the robustness, the strong localization and the superior performance of our DeepLiDARFlow compared to image-only and LiDAR-only methods.

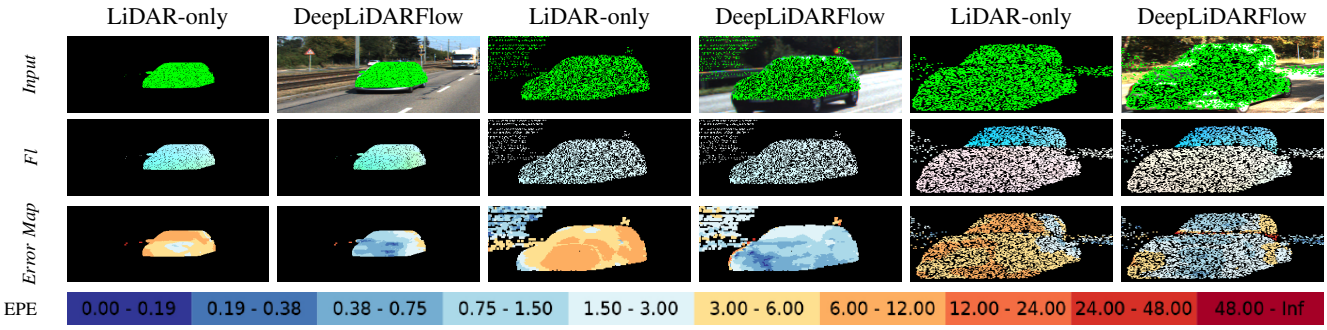
Comparison with Image-only Method: Our DeepLiDARFlow uses concepts like pyramids, warping, occlusion and cost volume. PWOC-3D [26] also uses similar concepts but with a pair of stereo images as an input. Our DeepLiDARFlow is able to obtain good performance for a wide range of sparsity levels with an optimum of just 5000 LiDAR points. For this input density, our DeepLiDARFlow is compared to PWOC-3D on KITTI [10] and FT3D [23] as shown in Table II. Moreover, we visualize the robustness and the localization in three examples as shown in Fig. 5a. These examples present occlusions, reflective surfaces and shadows which are often challenging examples for any of image-only approaches. In these areas, our DeepLiDARFlow has the capability to resolve them and to result in more accurate scene flow estimations.

Comparison with LiDAR-only Method: HPLFlowNet [12] utilizes LiDAR scans represented as 3D Cartesian coordinates (i.e. a point cloud) at two time steps to estimate scene flow. Since they use sparse points, we perform a sparse evaluation on KITTI using 8192 of LiDAR points (proposed sparse level in HPLFlowNet) to compare between HPLFlowNet and our DeepLiDARFlow. Since HPLFlowNet is originally evaluated without including ground surface, we present the results for HPLFlowNet once by excluding the ground surface and once with the ground surface included. However, we include the ground surface in our DeepLiDARFlow but evaluate the same sparse locations of LiDAR measurements as in HPLFlowNet only. Note that DeepLiDARFlow produces a dense result (w.r.t the image resolution) independent of the input density. Our DeepLiDARFlow outperforms comfortably HPLFlowNet over (Fl) and (Fl-EPE) metrics and achieves comparable results to HPLFlowNet in terms of the 3D metrics (SF-3D and SF-EPE-3D) as shown in Table III. The qualitative results show as well our superior accuracy compared to [12] (see Fig. 5b).

Comparison with Image plus LiDAR Method: The only other method available in literature which uses the fusion of LiDAR and RGB images is LiDAR-Flow [2] but it operates in a stereo setup and utilizes both stereo images and the corresponding LiDAR measurements. Therefore, it has much more information given as input than our DeepLiDARFlow which uses only monocular images and the corresponding LiDAR measurements. For having a fair comparison, we adopt a monocular setup with LiDAR measurements (called MonoLiDAR-Flow). To this end, we firstly densify the sparse LiDAR input by using the edge-preserving interpolation algorithm described in [2]. Secondly,



(a) Image-only vs. our DeepLiDARFlow



(b) LiDAR-only vs. our DeepLiDARFlow (ground surface removed in outputs and error maps)

Fig. 5: Our DeepLiDARFlow presents high robustness using LiDAR features and the rich textural information of RGB features. Here we compare some results from DeepLiDARFlow against an image-only approach [26] and a LiDAR-only approach [12]. DeepLiDARFlow shows superior performance in regions of bad illumination compared to the image-only approach, and overcomes the problem of unstructured point clouds yielding a result of much higher resolution, compared to LiDAR-only.

TABLE II: Comparison of scene flow results for PWOC-3D [26], LiDAR-Flow [2], MonoLiDAR-Flow (monocular version of LiDAR-Flow), and DeepLiDARFlow on the test splits of KITTI [10] and FT3D [23]. LiDAR methods are evaluated with an input of 5000 depth measurements.

Method	**Modality**	**KITTI** [10]					**FT3D** [23]					**Time (ms)**
		D0	**D1**	**Fl**	**SF**	**SF-EPE**	**D0**	**D1**	**Fl**	**SF**	**SF-EPE**	
PWOC-3D [26]	Stereo-Only	4.07	6.1	10.29	12.24	**3.15**	8.04	9.30	16.64	19.30	**6.97**	**130**
LiDAR-Flow [2]	Stereo + LiDAR	**2.30**	**5.03**	**8.46**	**9.33**	4.67	**3.69**	**6.48**	**15.10**	**16.00**	29.97	65900
MonoLiDAR-Flow	Monocular + LiDAR	**2.10**	**6.55**	13.37	14.11	7.31	**4.04**	**5.80**	**15.04**	**16.02**	24.29	34700
Our DeepLiDARFlow	Monocular + LiDAR	4.18	7.33	**11.26**	**13.77**	**3.64**	6.13	7.75	18.51	20.34	**6.87**	**310**

TABLE III: Sparse evaluation of DeepLiDARFlow and HPLFlowNet [12] with and without ground surface on KITTI. When grounds are included, our DeepLiDARFlow outperforms HPLFlowNet significantly over all terms. HPLFlowNet is able to outperform our DeepLiDARFlow only in terms of 3D metrics (i.e. SF-EPE-3D and SF-3D) when the ground surface is removed. Even then, our DeepLiDARFlow has better performance in terms of optical flow estimation.

Method	**Modality**	**Ground Surface**	**Output Density**	**KITTI** [10]				**Time (ms)**
				Fl-EPE	**Fl**	**SF-EPE-3D**	**SF-3D**	
HPLFlowNet [12]	LiDAR-Only	excluded	$\sim$ 2 %	5.94	47.54	**0.14**	**12.79**	**301**
HPLFlowNet [12]	LiDAR-Only	included	$\sim$ 2 %	9.77	71.62	0.27	33.84	**301**
Our DeepLiDARFlow	Monocular + LiDAR	included	**100.0 %**	**2.89**	**11.74**	0.15	13.27	310

we dissolve the stereo images in LiDAR-Flow pipeline [2] to adopt only monocular setup with LiDAR. The chart in Fig. 6 compares MonoLiDAR-Flow, LiDAR-Flow and our DeepLiDARFlow with varying LiDAR densities on KITTI. LiDAR-Flow outperforms MonoLiDAR-Flow and our DeepLiDARFlow in terms of scene flow outliers (SF

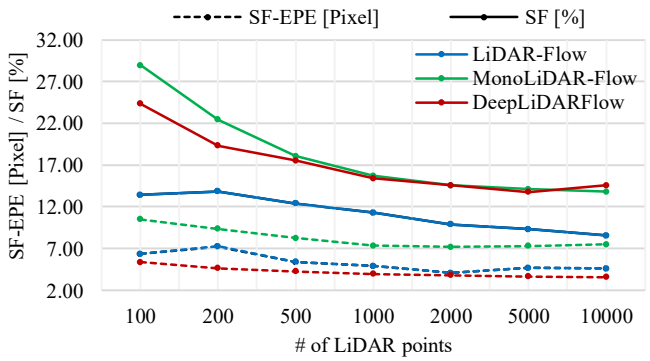


Fig. 6: Our DeepLiDARFlow comparison against other fusion-based approaches in terms of scene flow outliers and scene flow endpoint error with varying number of LiDAR points. Dotted lines show the trend for endpoint error (SF-EPE) in pixels and the solid lines represent the outlier rate (SF) in percent. LiDAR-Flow [2] outperforms all methods in terms of outlier rate (SF) since it exploits both LiDAR and RGB information in a stereo setup. Our DeepLiDARFlow performs better than its direct competitor MonoLiDAR-Flow for very sparse input. In terms of endpoint error, our DeepLiDARFlow comfortably outperforms all other methods.

[%]), a probable reason being the large amount of extra information it has due to the presence of a second camera view. Our DeepLiDARFlow outperforms MonoLiDAR-Flow for very sparse inputs, even for denser inputs; our DeepLiDARFlow results in an equivalent outliers rate compared to MonoLiDAR-Flow but our DeepLiDARFlow operates at a much higher speed than MonoLiDAR-Flow. In terms of (SF-EPE [px]), our DeepLiDARFlow outperforms LiDAR-Flow and MonoLiDAR-Flow consistently for all input densities. Table II presents a detailed comparison of these methods with all other metrics, when evaluated with a constant number of points (5000 points). Our DeepLiDARFlow performs as good as these methods (and better on several metrics) while operating at a much higher speed. As qualitative comparison, we visualize an example in Fig. 1 which shows strong localization and robust scene flow estimation compared to MonoLiDAR-Flow approach using $\sim$100 points.

V. Conclusion

In this paper, we presented our DeepLiDARFlow – a novel deep learning architecture which takes monocular images and the corresponding sparse LiDAR measurements as input, employs a multi-scale late fusion of LiDAR and RGB features, and predicts dense scene flow. In critical regions which contain difficulties like reflective surfaces, ill conditioned environment, shadows, and more, our DeepLiDARFlow shows superior performance over image-only methods. Moreover, we provided a robust localization compared to an image-only approach as well as a conventional approach. Compared to a LiDAR-only approach, we achieved a superior accuracy for scene flow estimation. Our method obtained competitive performance on the challenging KITTI and FlyingThings3D data sets with very sparse LiDAR input ($<$ 1000 points) and almost constant accuracy with different levels of input density.

Acknowledgment

This work was partially funded by the Federal Ministry of Education and Research Germany in the project VIDETE (01IW18002).

References

[1] F. Aleotti, M. Poggi, F. Tosi, and S. Mattoccia, "Learning end-to-end scene flow by distilling single tasks knowledge," in *Conference on Artificial Intelligence (AAAI)*, 2020.

[2] R. Battrawy, R. Schuster, O. Wasenmüller, Q. Rao, and D. Stricker, "Lidar-flow: Dense scene flow estimation from sparse lidar and stereo images," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

[3] A. Behl, D. Paschalidou, S. Donné, and A. Geiger, "Pointflownet: Learning representations for rigid motion estimation from point clouds," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[4] F. Brickwedde, S. Abraham, and R. Mester, "Mono-sf: Multi-view geometry meets single-view depth for monocular scene flow estimation of dynamic traffic scenes," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[5] J. Čech, J. Sanchez-Riera, and R. Horaud, "Scene flow estimation by growing correspondence seeds," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[6] 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 *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[7] A. Eitel, J. T. Springenberg, L. Spinello, M. Riedmiller, and W. Burgard, "Multimodal deep learning for robust rgb-d object recognition," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[8] A. Eldesokey, M. Felsberg, and F. S. Khan, "Confidence propagation through cnns for guided sparse depth regression," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2019.

[9] R. Gadde, V. Jampani, M. Kiefel, D. Kappler, and P. V. Gehler, "Superpixel convolutional networks using bilateral inceptions," in *European Conference on Computer Vision (ECCV)*, 2016.

[10] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[11] J.-M. Gottfried, J. Fehr, and C. S. Garbe, "Computing range flow from multi-modal kinect data," in *International Symposium on Visual Computing (ISVC)*, 2011.

[12] X. Gu, Y. Wang, C. Wu, Y. J. Lee, and P. Wang, "Hplflownet: Hierarchical permutohedral lattice flownet for scene flow estimation on large-scale point clouds," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[14] E. Herbst, X. Ren, and D. Fox, "Rgb-d flow: Dense 3-d motion estimation using color and depth," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[15] F. Huguet and F. Devernay, "A variational method for scene flow estimation from stereo sequences," in *IEEE International Conference on Computer Vision (ICCV)*, 2007.

[16] E. Ilg, T. Saikia, M. Keuper, and T. Brox, "Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation," in *European Conference on Computer Vision (ECCV)*, 2018.

[17] M. Isard and J. MacCormick, "Dense motion and disparity estimation via loopy belief propagation," in *Asian Conference on Computer Vision (ACCV)*, 2006.

[18] H. Jiang, D. Sun, V. Jampani, Z. Lv, E. Learned-Miller, and J. Kautz, "Sense: A shared encoder network for scene-flow estimation," in *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[20] R. Li and S. Sclaroff, "Multi-scale 3d scene flow from binocular stereo sequences," *Computer Vision and Image Understanding (CVIU)*, 2008.

[21] X. Liu, C. R. Qi, and L. J. Guibas, "Flownet3d: Learning scene flow in 3d point clouds," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[22] W.-C. Ma, S. Wang, R. Hu, Y. Xiong, and R. Urtasun, "Deep rigid instance scene flow," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[23] 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 *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[24] Y.-L. Qiao, L. Gao, Y. Lai, F.-L. Zhang, M.-Z. Yuan, and S. Xia, "Sf-net: Learning scene flow from rgb-d images with cnns," in *British Machine Vision Conference (BMVC)*, 2018.

[25] J. Quiroga, T. Brox, F. Devernay, and J. Crowley, "Dense semi-rigid scene flow estimation from rgbd images," in *European Conference on Computer Vision (ECCV)*, 2014.

[26] R. Saxena, R. Schuster, O. Wasenmüller, and D. Stricker, "Pwoc-3d: Deep occlusion-aware end-to-end scene flow estimation," *IEEE International Conference on Intelligent Vehicles Symposium (IV)*, 2019.

[27] Q. Teng, Y. Chen, and C. Huang, "Occlusion-aware unsupervised learning of monocular depth, optical flow and camera pose with geometric constraints," *Future Internet*, 2018.

[28] J. Uhrig, N. Schneider, L. Schneider, U. Franke, T. Brox, and A. Geiger, "Sparsity invariant cnns," in *International Conference on 3D Vision (3DV)*, 2017.

[29] C. Vogel, K. Schindler, and S. Roth, "Piecewise rigid scene flow," in *IEEE International Conference on Computer Vision (ICCV)*, 2013.

[30] Z. Yang, P. Wang, Y. Wang, W. Xu, and R. Nevatia, "Every pixel counts: Unsupervised geometry learning with holistic 3d motion understanding," in *European Conference on Computer Vision (ECCV)*, 2018.

[31] Z. Yin and J. Shi, "Geonet: Unsupervised learning of dense depth, optical flow and camera pose," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[32] Y. Zou, Z. Luo, and J.-B. Huang, "Df-net: Unsupervised joint learning of depth and flow using cross-task consistency," in *European Conference on Computer Vision (ECCV)*, 2018.