A GPU-accelerated Deep Stereo-LiDAR Fusion for Real-time High-precision Dense Depth Sensing

Haitao Meng, Chonghao Zhong, Jianfeng Gu, Gang Chen
Sun Yat-sen University, Guang Zhou, China

***Abstract*—Active LiDAR and stereo vision are the most commonly used depth sensing techniques in autonomous vehicles. Each of them alone has weaknesses in terms of density and reliability and thus cannot perform well on all practical scenarios. Recent works use deep neural networks (DNNs) to exploit their complementary properties, achieving a superior depth-sensing. However, these state-of-the-art solutions are not satisfactory on real-time responsiveness due to the high computational complexities of DNNs. In this paper, we present FastFusion, a fast deep stereo-LiDAR fusion framework for real-time high-precision depth estimation. FastFusion provides an efficient two-stage fusion strategy that leverages binary neural network to integrate stereo-LiDAR information as input and use cross-based LiDAR trust aggregation to further fuse the sparse LiDAR measurements in the back-end of stereo matching. More importantly, we present a GPU-based acceleration framework for providing a low latency implementation of FastFusion, gaining both accuracy improvement and real-time responsiveness. In the experiments, we demonstrate the effectiveness and practicability of FastFusion, which obtains a significant speedup over state-of-the-art baselines while achieving comparable accuracy on depth sensing.**

I. Introduction

Reconstructing 3D information is essential for many robotics applications such as robot navigation and autonomous driving. Most of them require the depth map to be dense, reliable, and processed in real time. For example, in autonomous driving, detecting distant objects such as humans requires to be performed timely and accurately using high resolution depth for dynamic obstacle avoidance. To achieve reliable depth sensing for outdoor robots, two techniques are generally utilized, including LiDAR sensors and stereo vision sensors. 3D LiDAR sensors can be considered as practical depth sensing solutions since other sensors such as RGBD cameras perform poorly outdoors with the presence of sunlight. Although LiDAR sensors can provide very reliable depth sensing, it is generally expensive, whilst generate sparse depth and simultaneously cannot acquire color information to cover all relevant objects in a scene. For example, even using high-end HDL-64 LiDAR which costs 75000 dollars, it *only* covers less than 6% of the total depths of image points. As a comparison, stereo vision sensors based on stereo matching algorithms can provide dense depth information with corresponding color information. However, it suffers from high computational complexity and unreliability depth sensing in the regions with repetitive patterns, homogeneous appearance, and occluded objects.

Therefore, it is desirable to perform the stereo-LiDAR fusion for achieving a trustworthy dense depth sensing. To exploit their complementary properties, recent researches [1]–[3] attempt to adopt deep neural networks (DNNs) to learn a proper registration between LiDAR and stereo images, which have shown to yield significant gains in improving accuracy. However, these state-of-the-art solutions are still not satisfactory in terms of their practicability on real-world applications. First, existing DNN-based stereo-LiDAR fusion methods [1], [2] highly rely on end-to-end DNN architectures to achieve the accuracy improvement, which generally tends to over-specialize under the confined training scenarios (e.g. the KITTI dataset [4]) and cause generalization problem on the new unseen scenarios. This similar issue has also been indicated in [5], [6] and further elaborated in our experiment in Section V-B. Second, due to the high computational complexities of DNNs, existing DNN-based stereo-LiDAR fusion methods still suffer from significant latency and require hundreds of milliseconds to process stereo-LiDAR fusion. For example, the approach in [2] takes about 1 second to perform stereo-LiDAR fusion on a high-end GPU TITAN Xp. This is obliviously unacceptable for adaptive obstacle avoidance in autonomous driving which requires performing timely.

To resolve the issues above, we present an efficient and fast deep stereo-LiDAR fusion framework, called FastFusion, for real-time high-precision depth estimation. Instead of taking an extremely data-driven approach that adopts end-to-end DNN architectures [1], [2], [7], we aim to design a practical stereo-LiDAR fusion system with an emphasis on its generalizability and low latency. To guarantee the generalizability, we adopt StereoBit [6] as the backbones and present a compact BNN architecture to learn high-level joint feature representations from the incorporated information of sparse LiDAR sensing data with stereo images. Moreover, we propose a new aggregation scheme to fuse the sparse LiDAR measurements in the phase of cost volume aggregation and adaptively refine cost volume optimization for further accuracy improvement. Compared to end-to-end DNN architectures [1], [3], [7], FastFusion can achieve strong generalization ability to the new unseen scenarios, being practical in the real-world applications. In addition, we provide a set of GPU-based schemes to achieve a low latency implementation of FastFusion, gaining both accuracy improvement and real-time responsiveness.

We evaluate our low latency framework on two challenging datasets: KITTI Stereo 2015 dataset [4] and ETH3D [8] dataset. Results show that FastFusion can achieve 2.81% 3-pixel stereo error in 16.8ms with a 20.9KB model, which is an order of

Corresponding author: Gang Chen (email: cheng83@mail.sysu.edu.cn).

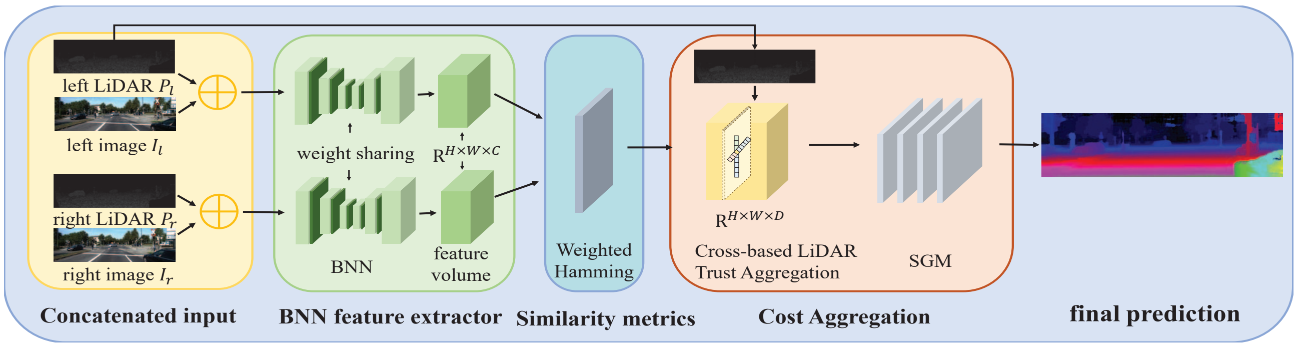


Figure 1. Architecture overview of the proposed FastFusion.

magnitude faster and three orders of magnitude smaller than prior works. In addition, FastFusion is general to the unseen scenarios and is robust to the various input LiDAR density, performing strong practical capability to the real applications.

II. Related Work

Stereo matching has been considered as a promising solution for depth estimation task, and some classical methods such as semi-global matching (SGM) [9] have been widely used but still limited to the insufficient accuracy. Recently, deep learning based stereo matching methods [7], [10], following the inherent paradigm of SGM, bring a significant leap in accuracy for dense depth. However, these end-to-end methods suffer from high computation complexity and ambiguous depth sensing.

Recent studies attempt to fuse the 3D LiDAR data into stereo matching to tackle the problems. Maddern et al. [11] proposed a probabilistic fusion framework to generate depth and uncertainty estimation in real time, but it can not achieve high accuracy. Wang et al. [1] takes the concatenation of LiDAR depth and RGB images as input, and regularize the cost volume optimization depended on LiDAR measurements. Cheng et al. [2] propose an unsupervised deep stereo-LiDAR network that aggregates images and LiDAR points to form a feature volume and performs an iteration method to deal with noise. However, both networks for image input and sparse LiDAR input maintain the original deep learning based computational architectures and cannot meet the real-time requirement. Moreover, Park et al. [3] utilize deep features extracted from LiDAR disparity to refine the output of stereo estimation, which is viewed as rough fusion and fails to make the most of LiDAR facilitation.

All of these DNN-based methods could not fulfill the requirements of gaining both real-time and accurate depth estimation. An appropriate solution for fusing LiDAR measurements and stereo matching algorithms is still lacking. To address this issue, our work focus on the efficient fusion strategy to obtain accurate and robust depth prediction in the real-time manner.

III. TECHNICAL APPROACH

The pipeline of our proposed framework is shown in Fig. 1. Rectified left image I_l and right image I_r as well as corresponding projected sparse LiDAR map P_l and P_r were concatenated along the channel dimension, feeding to the system as input. Then with our binary feature extractor, two kinds of information are jointly represented as high-level bit-wise feature vector. Along with a weighted hamming distance similarity metric, an initial matching cost volume for each candidate disparity can be obtained. In the following aggregation procedure, cross-based LiDAR trust aggregation and SGM will optimize the cost volume and extract the disparity map. With sophisticated post-processing, the framework could generate a well-performed disparity map.

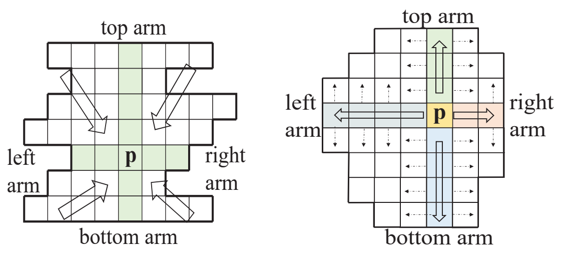
In this section, we will first outline our BNN model design. Then, we will introduce a simple aggregation method that takes benefit from sparse LiDAR data to gain accuracy improvement with almost no extra computing burden.

A. Network Design

Binary neural network is an extremely quantized network that float-point weights were represented as +1 or -1 to achieve the largest compression ratio. However, directly imposing binary quantitation to a float-point CNN will experience a severe accuracy drop due to the poor representation capacity of BNN network [12]. Here we refer to the StereoBit [6] network architecture and its weighted hamming distance similarity metric to produce the initial matching cost. With encoding the LiDAR measurement information, we could extract more accurate and robust feature than that merely depend on photometric appearance. Similar to [6], we did not construct an end-to-end architecture for prediction. Instead, the BNN only function as a preliminary matching likelihood generator, such that we could further integrate LiDAR information in the following aggregation procedure to boost the performance on accuracy.

B. Cross-based LiDAR Trust Aggregation

Above we obtained a matching cost volume which indicates the prediction on every candidate disparity. However, directly outputting a most likely disparity map according to the cost volume is not competitive with modern stereo algorithms. It is important to exploit different forms of cost aggregation to derive high quality depth estimation. In this section, we introduce a matching cost re-updating aggregation strategy for each LiDAR available pixel and its adjacent field to maximize utilize LiDAR measurements. Moreover, this strategy is also simple to implement. With our efficient fusion scheme proposed in IV-C, we could improve the estimated disparity map with neglectable extra overhead.



(a) Cross-based local support aggregation (b) Proposed cross-based LiDAR trust aggregation

Figure 2. Comparison of two aggregation approach. p is the anchor point.

Previous work has shown that building a local support region to aggregate initial matching costs can efficiently obtain a well-performed disparity map [10], [13]. As shown in Fig. 2(a), the core principle of conducting local region aggregation for one pixel is to seek the support from its neighbour pixels in a given size-constrained window. However, this kind of algorithmic idea is incompetent for stereo-LiDAR fusion task, since the LiDAR map already provide sparse accurate depth. In stereo-LiDAR fusion task, radiating the information from these sparse LiDAR points is required, as indicated in Fig. 2(b). Updating the obtained corresponding disparity matching cost to zero according to the LiDAR map is a simple fusion strategy. While this straightforward altering gains little improvement when very sparse cost updated. Since these updated matching costs will be seen as outliers by aggregation procedure. The strong disagreement from original stereo costs will force the aggregation process to ignore or overlap these updates.

To promote the effective utilization of the sparse LiDAR prior, the urgent request is to design an effective information fusion mechanism. Inspired by the way in which [13] allows pixels to influence each other. We propose cross-based LiDAR trust aggregation to achieve adaptive stereo-LiDAR fusion. Rather than building local regions for every pixel and conduct aggregation for all candidate disparity, our method only requires a small amount of updating on specific disparity of pixels according to the LiDAR prior. Then, the following aggregation process will automatically promote the support of LiDAR prior to their neighbours. With this optimized manner, a depth accuracy improvement can be obtained.

Let us consider a matching cost volume $C_b \in \mathbb{R}^{H \times W \times D}$, where H and W denote the height and width respectively, D denote the maximum disparity. A triplet (x, y, d) represents a cost volume coordinate, where x and y are along H and W axes corresponding to the position (x, y) in the input image as well as LiDAR map, d is the coordinate in D axis. In order to transmit the LiDAR information at the position $p^*(x^*, y^*)$ to its neighbours, we build four arms upright cross p^* with direction $X = \{\leftarrow, \rightarrow, \uparrow, \downarrow\}$ in the input image. The length of each direction path $\{xl, xr, xu, xd\}$ is limited by L. Take xl path for example, the biggest range L_{xl} can be calculated with:

$$L_{xl} = \max_{l \in [1,L)} \left(l \cdot \prod_{j \in [0,l]} \delta(p^*, p_j) \right) \tag{1}$$

where $\delta(p^*, p_j)$ is the indicator function evaluating absolute image intensity difference between anchor point p^* and compared point p_j. Let maximum image intensity difference toleration is τ and the intensity of the input image is I. The $\delta(p^*, p_j)$ is defined as:

$$\delta(p^*, p_j) = \begin{cases} 1 & \text{if } (|I(p^*) - I(p_j)|) < \tau \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

Let P denote the disparity in LiDAR map, and l denote the distance from the pixel p_c to its nearest LiDAR point p^* in the direction of xl. Thus, we could perform following updating to the cost volume C_b :

$$C_c(p_c, P(p^*)) = \begin{cases} 0 & \text{if } l < L_{xl} \text{ and } \delta(p^*, p_c) = 1 \\ C_b(p_c, P(p^*)) & \text{otherwise,} \end{cases} \tag{3}$$

Above, we have detailed the cross-based LiDAR trust aggregation for an available LiDAR point in a single direction. To achieve the best performance, these processes should be conducted on every LiDAR measurement point among four directions.

IV. Parallel optimization

In this section, we show how to construct bit-manipulation capability in modern GPUs. To achieve this goal, we employ a set of systematic view optimizations for efficient BNN network inference computing. Moreover, we introduce a low latency calculation scheme for weighted hamming distance to avoid bank conflicts. Finally, we give a highly efficient implementation optimization for proposed cross-based LiDAR trust aggregation, such that this aggregation can be adopted in our framework to gain accuracy improvement without introducing notable computing burden. As shown in Tab. I, by these powerful techniques, our stereo estimation framework could generate accurate disparity prediction with significantly lower latency than state-of-the-art counterparts.

A. Network Optimization

In BNNs, weights are binarized to reduce memory footprint and access burden. To accelerate the computing, logical operations (e.g. *xor*, *popcount*) are used to replace arithmetic operation (e.g. float-point multiplication) [14]. In our work, we follow the optimization schemes in [6] to exploit the computing power of GPU. We implement HWC order to achieve naive locality support to the bit-packing. Then bit-wise logical computing can be implemented in an efficient way, deriving 32× fewer memory accesses and 6× faster computing comparing to float point multiplication. Furthermore, to reduce expensive memory access operations in convolutional network computing, a multiply layers integration technique is adopted, which fuses the binary convolutional layer, scale layer, batch normalization layer and binary activation layer into a stand-alone layer. Due to limited space, we refer the interested reader to [6] for details.

B. Weighted Hamming Distance Optimization

In our efficient stereo estimation framework, we use weighted hamming distance as a feature similarity metric for its discrimination capability. As float point weight is introduced

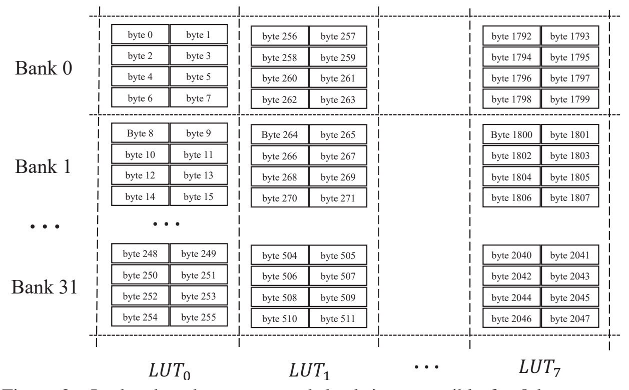


Figure 3. In the shared memory, each bank is responsible for 8 bytes query index data of LUT_i.

to each channel, the calculating of weighted hamming distance could be much slower than the mere hamming distance computation. A reasonable solution is to construct a lookup table (LUT) and make use of shared memory to accelerate computing. However, due to the mechanisms of the lookup table, massive parallel threads in a block may access the same bank of shared memory simultaneously. This unbalance access, which is called bank conflict, will cause drastic pipeline stalls, resulting in significant performance loss. To tackle this issue, we conduct a well-designed implementation on GPU for a lookup table with a bank conflict avoiding scheme. By doing so, we could achieve almost the same throughput as solely bit-wise hamming distance computing.

When calculating the weighted hamming distance, each bit is given a float-point weight. To obtain the weighted distance, we build a lookup table for two 64-bit feature vectors obtained from the BNN feature extractor. We divide each input vector into 8 segments in which bit-wise operation XNOR can be performed to obtain a byte-type index. Each segment enumerates all possibilities of weighted hamming distance for a byte-type query index, such that the similarity can be obtained by summing up these 8 segments indexing values.

Since the weights of hamming distance will remain unchanged once the training is done, it can be loaded into the global memory in the initial stage. In the LUT optimization, each item in one segment is presented in 8-bit (i.e. 1 byte). Thus each segment requires a 256-bytes query index. For all segments, the total size of LUT should be $256 \times 8 = 2048$ bytes (2 KB). For the memory access of LUT, since LUT is constant and the size of LUT is only 2 KB, we make use of shared memory to store LUT. To avoid bank conflicts, bank bandwidth of GPU is set to 64 bits, such that a 256-bytes query index of one segment in LUT could be loaded within the same depth of 32 memory banks, as shown in the Fig. 3. Due to the broadcast mechanism, simultaneous access by a warp does not generate bank conflict between two threads that access any sub-byte within the same bank [15]. By adopting this optimization approach, a fast and efficient similarity metric can be achieved.

C. Aggregation Optimization

Aiming to achieve the real-time responsiveness, an efficient implementation for cross-based aggregation is very important. However, even with well-design choices, a stand-alone aggregation procedure still takes tens of millisecond [10], [13] which is unacceptable and may play a performance bottleneck of the stereo estimation framework. To tackle this issues, we present a highly efficient optimization of cross-based LiDAR trust aggregation which merges the computing and memory accesses into the SGM procedure. By adopting this aggregation strategy, we could obtain an accuracy improvement without introducing notable latency.

To be more explicit to state the details, we first provide three observations for cross-based LiDAR trust aggregation: (1) There is no dependence on the arm in different directions; (2) Selected pixels which fulfill the restrictions in Eqn. (3) only need one-time cost updating. The updating value is constant (i.e. minimum value in matching cost range); (3) Only a small part of pixels are key points and need to be updated. From these three observations, it is possible to accelerate the cross-based LiDAR trust aggregation by launching massive parallel threads. While to step further to decrease the latency of our framework, we explore a nearly zero cost aggregation approach.

Cross-based LiDAR trust aggregation can be simplified as four separate single direction pixel census. Therefore, we merge this computing process to the following SGM aggregation which access pixels successively among one direction in the same pattern. In SGM computing, the original matching cost can be updated under the restriction of cross-based LiDAR trust aggregation beforehand. The only memory overhead we introduce here is the access of input image and corresponding projected LiDAR map. With the massive parallel threads scheme in semi-global matching [9], coalesced memory access is achieved to reduce the loading latency. As shown in Tab. I, the extra overhead introduced by our aggregation procedure of the whole system is only 0.2ms.

By integrating two aggregation procedure, we only require one time pixel access for one direction. The matching cost can be updated with our LiDAR prior information fusion strategy and gain accuracy improvement with neglectable extra computing.

V. Experiment

We develop FastFusion framework with CUDA programming. To evaluate the effectiveness of proposed optimizations and schemes, we test FastFusion on two different stereo datasets to verify the performance in terms of accuracy, responsiveness and generalization. In addition, we conduct a robustness experiment by inputting vary density and misalignment LiDAR map to demonstrate the practice power of our framework. For the sake of fairness, we re-implement all compared algorithms on NVIDIA TITAN Xp GPU and test them with the same input to guarantee all results are comparable.

A. Experiment Setup.

We evaluated the performance of FastFusion in two challenging datasets: KITTI Stereo 2015 [4] and ETH3D [8] with several state-of-the-art stereo estimation algorithms. For KITTI Stereo 2015 dataset [4], it is one of the most famous stereo dataset and benchmark. It contains 200 pairs RGB images with

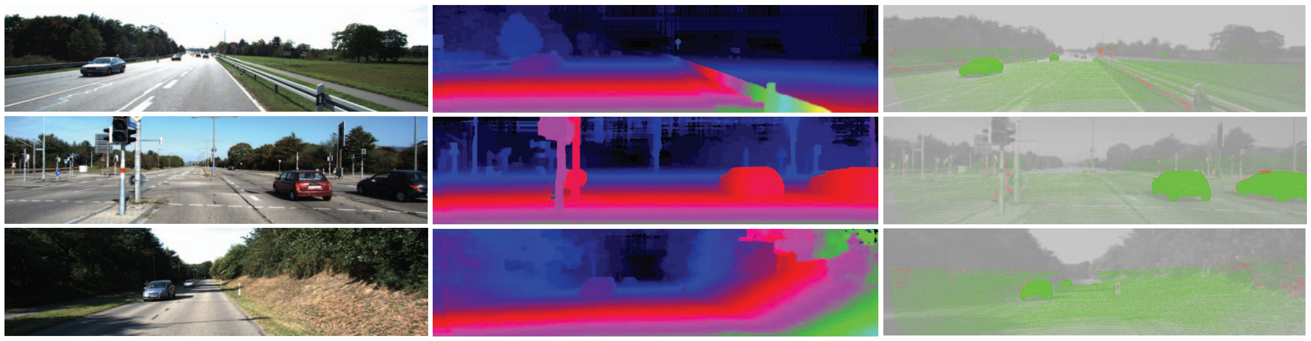


Figure 4. Qualitative examples. From left to right, Left column: left RGB input; Middle column: depth map generated by our framework; Right column: error map, we overlap ground truth points with the input image, green and red points indicate the correct and bad prediction respectively.

ground truth disparity for training purpose and another 200 pairs RGB images for testing purpose. We follow the identical approach as previous works [1], [2], [11] to trace 141 image pairs from KITTI raw dataset that have corresponding frames in the KITTI 2015 dataset and reported their results on this subset. We also evaluate our FastFusion on the ETH3D dataset which is a stereo dataset and benchmark, covering both indoor and outdoor scenes. To test the generalization power, we did not train model in this dataset. Instead, we employ identical models and hyper-parameters setting with the experiment in KITTI Stereo 2015. Since all these methods trained in road scene dataset (i.e. KITTI dataset), ETH3D, which contains different scene images, could function as novel scenarios to test whether methods could reconstruct 3D information precisely.

We train our siamese BNN on the KITTI Completion dataset [16] which contains 93 thousand depth maps with corresponding raw LiDAR scans and RGB images covering 29 road scenes. To avoid frames similar to the selected subset involved in our training set, we follow [1] to exclude the scene sequences containing any 141 image pairs in our testing set, which remained us 34 thousand pairs. Considering that our BNN model is a shallow architecture, also the sequences in each scene are successive, we only utilize roughly 1% of stereo pairs, resulting in a total of 277 stereo image pairs for training.

B. Quality Evaluation.

KITTI Stereo 2015 To evaluate the performance of our framework with state-of-the-art methods, we first conduct an experiment on the subset of KITTI Stereo 2015 dataset. We randomly sample 15 % of ground truth as LiDAR input, which corresponding to the average 2.83% pixels of the whole image associated with LiDAR. Then, we evaluate algorithms with the remained 85% of ground truth. The types of input, metrics of accuracy, runtime latency, and model size of FastFusion in comparison with five state-of-the-art methods are summarized in Tab. I. Comparing to the state-of-the-art methods, the accuracy of our framework outperforms MC-CNN-fast [10] and CCVNorm [1] by a large margin. Although FastFusion is slightly inferior to PSMNet [7] and LidarStereoNet [2], it is worth noting that FastFusion only takes 16.8ms with the 20.9KB model, which is an order of magnitude faster and three orders of magnitude smaller than its counterparts. When

Table I
Performance comparison with various algorithms. We adopt the same metric as KITTI online benchmark that the pixel percentage of disparity error bigger than 3. For a fair comparison, we train stereo input methods with KITTI 2012 stereo dataset and evaluate them on selected 141 image pairs. As for fusion methods, we follow the open-source implementation and adopt the same setting of the corresponding paper to train the model.

Method	Input	Error	Runtime(ms)	Model Size
MC-CNN-fast [10]	Stereo	3.66%	480	1.5MB
PSMNet [7]	Stereo	2.73%	420	21MB
StereoBit [6]	Stereo	4.32%	**16.6**	**18.6KB**
LidarStereoNet [2]	Stereo+LiDAR	**2.18%**	960	21.3MB
CCVNorm [1]	Stereo+LiDAR	4.44%	1490	23.7MB
Ours	Stereo+LiDAR	2.81%	16.8	20.9KB

comparing the FastFusion with StereoBit [6], we can observe a 1.51% accuracy improvement with only 0.2ms extra computing cost. Considering that StereoBit [6] is the backbone of our work, this performance boost is strong evidence to demonstrate the effectiveness of the proposed cross-based LiDAR trust aggregation and its efficient implementation. Qualitative examples of the disparity map generated by our proposed FastFusion can be seen in Fig. 4. According to these examples, it is clear that by our efficient stereo-LiDAR fusion scheme, dense 3D information could be accurately achieved.

ETH3D. For a generalization capability test, we conduct an experiment on another challenging stereo dataset: ETH3D [8]. Since the ground truth provided in ETH3D is much denser than KITTI Stereo 2015 dataset. We only randomly sample 5% of ground truth to maintain approximate 3% image pixels corresponding to LiDAR points. From the Tab. II, we could draw the following observations: (1) The Proposed FastFusion surpass LidarStereoNet [2] and PSMNet [7] and achieves the best performance among all accuracy metrics with only 15.9ms computing latency. It means that FastFusion can adapt much better to new unseen environments; (2) Three end-to-end DNN methods: PSMNet [7], LidarStereoNet [2] and CCVNorm [1] can properly perceive accurate depth information when treating similar scenarios to the training data (e.g. KITTI Stereo 2015). However, they gain poor performance in the unseen dataset, showing their over specialization to the confined scenarios; (3) Though ETH3D is a novel scene dataset for FastFusion, it still gains significant accuracy improvement over StereoBit [6] with

Table II
Results of various methods on ETH3D dataset. We use the same setting as the evaluation in KITTI stereo 2015 dataset and collect data with ETH3D development kit. The Noc denotes a 2-pixel error percentage in the non-occluded region. The All denotes 2-pixel error percentage in all-region.

Method	Input	Noc	All	Runtime(ms)
MC-CNN-fast [10]	Stereo	1.40%	1.70%	430
PSMNet [7]	Stereo	29.26%	30.87%	340
StereoBit [6]	Stereo	2.48%	3.40%	**15.7**
LidarStereoNet [2]	Stereo+LiDAR	6.59%	7.15 %	750
CCVNorm [1]	Stereo+LiDAR	28.39%	29.66%	1340
Ours	Stereo+LiDAR	**0.33%**	**0.74%**	15.9

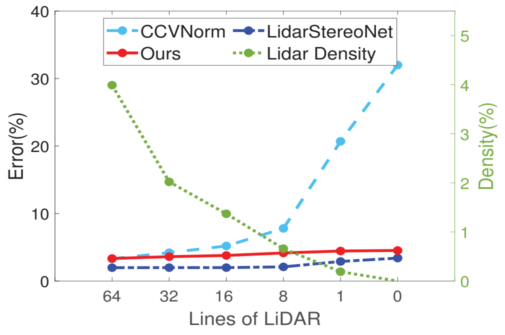


Figure 5. Robustness evaluation to different lines misaligned LiDAR input. We illustrate the performance (left) of three stereo-LiDAR fusion methods and input LiDAR density of various lines (right).

small computing overhead, demonstrating that our proposed aggregation scheme capable of adapting to different application scenarios.

C. Robustness to LiDAR input

To further validate the robustness of FastFusion, we conduct an experiment on KITTI Stereo 2015 with the accumulated raw LiDAR map. Due to the moving of the capture platform, boundaries and thin objects may not perfectly align with input images. In addition, the appearance of rolling shutter effect caused unreliable disparity reference. To be more challenging, we make the input LiDAR even more sparser. The density of raw LiDAR was uniformly sampled to simulate 64 lines, 32 lines, 16 lines, 8 lines, 1 line and 0 lines LiDAR sensor, corresponding to 3.99%, 2.02%, 1.37%, 0.66%, 0.19% and 0% image pixels having depth. Fig. 5 summarized the accuracy performance of various methods with respect to different LiDAR sparsity. As shown in the figure, FastFusion achieves similar accuracy performance to CCVNorm [1] with equivalent 64 lines LiDAR input. However, with the decrease of LiDAR density, an apparent drop trend of accuracy happened to CCVNorm [1]. In contrast, the proposed FastFusion still capable of reconstructing depth with little degradation. LidarStereoNet [2] obtains the best performance among varying LiDAR lines input. However, as Tab. I illustrates, LidarStereoNet [2] achieves that with significant longer latency and bigger model size. It takes 960 ms to generate a depth map with a 21.3 MB model. In contrast, FastFusion is 57× faster and 1019× memory efficient than LidarStereoNet [2]. It is obviously more practical for FastFusion to be deployed in real applications. It is also worth noting that even the density of LiDAR input is zero, FastFusion can still function normally to obtain the precise and dense depth map, which reveals its strong robustness.

VI. Conclusion

In this paper, we propose an efficient stereo-LiDAR fusion framework to generate accurate dense depth map. We use a compact BNN architecture to extract robust feature representation of both LiDAR input and stereo images jointly. Then, a novel cost aggregation approach is provided to further utilize LiDAR measurements, imposing adaptive refinement to the cost volume. To guarantee the responsiveness of our framework, we provide a set of GPU-based optimizations to achieve low cost computing. In the evaluation, FastFusion achieves impressive generalization power and responsiveness comparing to the state-of-the-art stereo-LiDAR fusion methods.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grant 62072478; in part by the Key-Area Research and Development Programs of Guangdong Province.

References

[1] T.-H. Wang *et al.*, "3d lidar and stereo fusion using stereo matching network with conditional cost volume normalization," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.
[2] X. Cheng *et al.*, "Noise-aware unsupervised deep lidar-stereo fusion," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.
[3] K. Park *et al.*, "High-precision depth estimation using uncalibrated lidar and stereo fusion," *IEEE Transactions on Intelligent Transportation Systems*, 2019.
[4] M. Menze *et al.*, "Object scene flow for autonomous vehicles," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.
[5] K. Batsos *et al.*, "Cbmv: A coalesced bidirectional matching volume for disparity estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.
[6] G. Chen *et al.*, "Gpu-accelerated real-time stereo estimation with binary neural network," *IEEE Transactions on Parallel and Distributed Systems*, 2020.
[7] J.-R. Chang *et al.*, "Pyramid stereo matching network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.
[8] T. Schöps *et al.*, "A multi-view stereo benchmark with high-resolution images and multi-camera videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
[9] D. Hernandez-Juarez *et al.*, "Embedded real-time stereo estimation via semi-global matching on the gpu," *Procedia Computer Science*, 2016.
[10] J. Zbontar, Y. LeCun *et al.*, "Stereo matching by training a convolutional neural network to compare image patches." *Journal of Machine Learning Research*, 2016.
[11] W. Maddern *et al.*, "Real-time probabilistic fusion of sparse 3d lidar and dense stereo," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.
[12] W. Tang *et al.*, "How to train a compact binary neural network with high accuracy?" in *AAAI Conference on Artificial Intelligence*, 2017.
[13] K. Zhang *et al.*, "Cross-based local stereo matching using orthogonal integral images," *IEEE Transactions on Circuits and Systems for Video Technology*, 2009.
[14] M. Courbariaux *et al.*, "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1," *arXiv preprint arXiv:1602.02830*, 2016.
[15] CUDA, "https://docs.nvidia.com/cuda/," Accessed Sept. 20, 2019.
[16] J. Uhrig *et al.*, "Sparsity invariant cnns," in *International Conference on 3D Vision (3DV)*, 2017.