2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

Volumetric Propagation Network: Stereo-LiDAR Fusion for Long-Range Depth Estimation

Jaesung Choe[1], Kyungdon Joo[2], Tooba Imtiaz[3] and In So Kweon[3]

***Abstract*— Stereo-LiDAR fusion is a promising task in that we can utilize two different types of 3D perceptions for practical usage – dense 3D information (stereo cameras) and highly-accurate sparse point clouds (LiDAR). However, due to their different modalities and structures, the method of aligning sensor data is the key for successful sensor fusion. To this end, we propose a geometry-aware stereo-LiDAR fusion network for long-range depth estimation, called *volumetric propagation network*. The key idea of our network is to exploit sparse and accurate point clouds as a cue for guiding correspondences of stereo images in a unified 3D volume space. Unlike existing fusion strategies, we directly embed point clouds into the volume, which enables us to propagate valid information into nearby voxels in the volume, and to reduce the uncertainty of correspondences. Thus, it allows us to fuse two different input modalities seamlessly and regress a long-range depth map. Our fusion is further enhanced by a newly proposed feature extraction layer for point clouds guided by images: *FusionConv*. FusionConv extracts point cloud features that consider both semantic (2D image domain) and geometric (3D domain) relations and aid fusion at the volume. Our network achieves state-of-the-art performance on the KITTI and the Virtual-KITTI datasets among recent stereo-LiDAR fusion methods.**

I. Introduction

Sensor fusion is the process of merging data from multiple sensors, which makes it possible enrich the understanding of 3D environments (*i.e.*, 3D perception) for autonomous driving or robot perception. Each sensor has its unique properties and can complement other sensors' limitations by fusion. In particular, sensor fusion – such as two cameras (stereo matching) or LiDAR and a single camera (depth completion) – allows us to estimate accurate depth information. Several studies have explored fusion-based depth estimation algorithms [1], [2], [3]. The quality of depth estimated by these traditional methods has been further improved with the advent of deep learning [4], [5], [6], [7], [8], [9].

Recently, stereo-LiDAR fusion has been getting more attention for practical usage in autonomous driving [10], [11], [12]. Compared to traditional fusion-based depth estimation, stereo-LiDAR fusion is a novel task where we can utilize two different types of 3D perception: dense 3D information from stereo cameras and sparse 3D point clouds from LiDAR, which can further improve the depth quality. Within this fusion framework, aligning different sensor data into a unified space is an essential step to fully operate depth estimation task. Previous works [10], [11], [12] perform fusion on the 2D image domain using the geometric relationship between sensors (*i.e.*, intrinsic and extrinsic parameters). For example, Wang *et al.* [10] project point clouds into the 2D image domain to align images with sparse depth maps of the projected point clouds. However, because neighboring pixels in a 2D image space are not necessarily adjacent in the 3D space, this 2D fusion in the image domain may lose depth-wise spatial connectivity, thus have difficulty in estimating accurate depth at distant regions. For geometry-aware fusion of stereo images and point clouds, it is necessary to maintain the spatial connectivity in a unified 3D space.

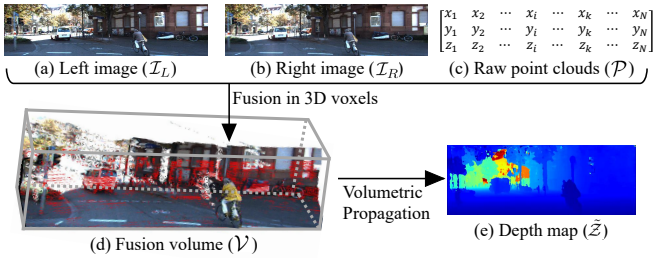

Fig. 1. **Pipeline of the proposed stereo-LiDAR fusion.** Given (a,b) stereo images and (c) point clouds, we fuse two different input modalities at 3D voxels of (d) fusion volume, which is a unified 3D volumetric space, to infer (e) depth map by volumetric propagation.

In this paper, we propose a geometry-aware stereo-LiDAR fusion network for long-range depth estimation, called *volumetric propagation network*. To this end, we define 3D volumetric features, called *fusion volume*, as a unified 3D volumetric space for fusion, where the proposed network computes the correspondences from both stereo images and point clouds, as shown in Fig. 1. Specifically, sparse points become seeds of correct correspondence to reduce the uncertainty of matching between the stereo images. Then, our network propagates this valid matching to the overall volume and computes the matching cost of the rest of the volume through stereo images. Furthermore, we facilitate the volumetric propagation by embedding point features into the fusion volume. The point features are extracted by our image-guided feature extraction layer from raw point clouds, called *FusionConv*. Our FusionConv considers both semantic (2D image domain) and geometric (3D domain) relation for the tight fusion of image features and point features. Finally,

This work (K. Joo) was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2020-0-01336, Artificial Intelligence Graduate School Program (UNIST)). *(Corresponding author: I. S. Kweon.)*

[1]J. Choe is with the Division of the Future Vehicle, KAIST, Daejeon 34141, Republic of Korea. jaesung.choe@kaist.ac.kr

[2]K. Joo is with the Artificial Intelligence Graduate School and the Department of Computer Science, UNIST, Ulsan 44919, Republic of Korea. kdjoo369@gmail.com, kyungdon@unist.ac.kr

[3]T. Imtiaz, and I. S. Kweon are with the School of Electrical Engineering, KAIST, Daejeon 34141, Republic of Korea. {timtiaz, iskweon77}@kaist.ac.kr

This paper is for the presentation in ICRA 2021.

our approach achieves state-of-the-art performance among stereo-LiDAR fusion methods for the KITTI dataset [13] and the Virtual-KITTI dataset [14].

II. Related work

We review sensor fusion-based depth estimation methods according to types of sensor systems: stereo cameras, mono-LiDAR fusion and stereo-LiDAR fusion.

Stereo cameras. Stereo matching is a task that reconstructs the 3D environment captured from a pair of cameras [1]. By computing the dense pixel correspondence between a rectified image pair, stereo matching infers the inverse depth *i.e.*, disparity map. Recently, deep-learning techniques have paved the way for more accurate and robust matching by using volume-based deep architectures [5], [15], [16]. Cost volume is one of the popular volumetric representations that encompass the 3D space in the referential camera view* along the disparity axis. This property is beneficial for computing matching cost between stereo images. Despite the large improvement, stereo matching still lacks accurate depth estimation at distant regions. To address this issue, LiDAR is a prominent sensor to estimate long-range depth.

Mono-LiDAR fusion. Depth completion is a task that estimates a depth map using a monocular camera and a LiDAR sensor. By propagating highly accurate but sparse points from a LiDAR sensor, this task aims to densely complete the depth map with the help of image information. Recent deep-learning-based methods [7], [6], [17], [18] largely increase the quality of depth. These methods [7], [6], [18] employ a pre-processing stage that projects point clouds into image domain, and feed this sparse depth map as input to a network (*i.e.*, early fusion). On the other hand, Chen *et al*. [17] introduce an intermediate fusion scheme in the feature space. They initially extract features from each modality and implement the fusion in the image feature space by projecting point features. Despite the improvement thus achieved, depth completion task has difficulties in estimating depth at unknown areas not encompassed by point clouds.

Stereo-LiDAR fusion. Stereo camera-LiDAR fusion (simply denoted as stereo-LiDAR fusion) has been recently demonstrated to further increase the accuracy of depth estimation by utilizing additional sensory information. Relying on the highly accurate but sparse depth map from point clouds, Park *et al*. [11] refine the disparity map from stereo cameras using a sparse depth map. Despite the increased quality of estimated depth, the fusion within this method lies in 2D image domain and is therefore insufficient to maintain metric accuracy of point clouds for long-range depth estimation. Another recent work, Wang *et al*. [10], introduce the idea of input fusion (*i.e.*, stereo RGB-D input) and volumetric normalization conditioned by sparse disparity (*i.e.*, projected point clouds). Typically, this conditional cost volume normalization [10] mainly affects the 3D volumetric aggregation and looks closer to the idea of our volumetric propagation. However, this method has difficulty in estimating the accurate depth in remote area.

To address this issue, we introduce the volumetric propagation network that aims to fuse the two input modalities: stereo images and point clouds in a unified 3D volume space, fusion volume. Within the fusion volume, we regard the sparse points as the seed of valid matching between stereo images and propagate the valid matching to the overall volume space. To do so, our network (1) maintains metric accuracy of point clouds during fusion and (2) reduces the uncertainty of stereo matching where point clouds do not exist. We further facilitate fusion at the fusion volume by our feature extraction layer for point clouds, FusionConv. Among recent stereo-LiDAR fusion works [12], [11], [10], with these two contributions, we achieve remarkable state-of-the-art depth estimation performance.

III. Overview

We aim to estimate a long-range, highly-accurate, and dense depth map $\widetilde{\mathcal{Z}}$ in the referential camera viewpoint from rectified stereo images $\mathcal{I}_L$ and $\mathcal{I}_R$, LiDAR point clouds $\mathcal{P}$ and their calibration parameters (*e.g.*, camera intrinsic matrix $\mathbf{K}$). The point clouds are presented as a set of N number of 3D points $\mathcal{P}=\{\mathbf{p}_i\}_{i=1}^{N}$, where each point $\mathbf{p}=[x,y,z]^\top$ lies in the referential camera coordinates.† Under the known $\mathbf{K}$, we can project each point into the image domain, $\mathbf{x} \simeq \mathbf{K}\mathbf{p}$, where the projected image point $\mathbf{x}$ can be located at the pixel location of (u,v). Using this geometric relation in conjunction with interpolation, we fuse the two modalities at the fusion volume (Sec. IV) and FusionConv (Sec. V). The overview of the proposed approach is presented in Fig. 2.

IV. Volumetric Propagation Network

In this section, we detail how we construct our fusion volume $\mathcal{V}$ to fuse two different input modalities, stereo images and raw point clouds, in a manner of volumetric propagation. The proposed fusion volume $\mathcal{V}$ is a unified 3D volumetric space for stereo-LiDAR fusion to estimate long-range depth map $\widetilde{\mathcal{Z}}$ in the referential view. Unlike the traditional voxel representation – cost volume [5], [15], [19] which quantizes the 3D data along the disparity axis, our fusion volume $\mathcal{V}$ describes the 3D environment as an evenly distributed grid voxel space along the depth range (*i.e.*, metric scale). This is to reduce the quantization loss when we embed sparse points $\mathcal{P}$ into the fusion volume. While embedding points into the cost volume dramatically increase the quantization loss in distant regions, our metric-scale fusion volume does not. Specifically, we define the fusion volume $\mathcal{V}$ as a 3D volume representation and each voxel $\mathbf{v} \in \mathcal{V}$ contains a feature vector of a certain dimension. That is, $\mathcal{V} \in \mathbb{R}^{W\times H\times D\times(2C+1)}$, where W, H and D denote the number of voxels along the width, height, and depth axes, respectively, and C represents the dimension of the image feature vector. Note that the feature dimension of the volume $\mathcal{V}$ is set as $(2C+1)$ to

*We set the left camera as the referential [1].

†For simplicity, we transform the point clouds in the LiDAR coordinates into the referential camera coordinates using extrinsic parameters.

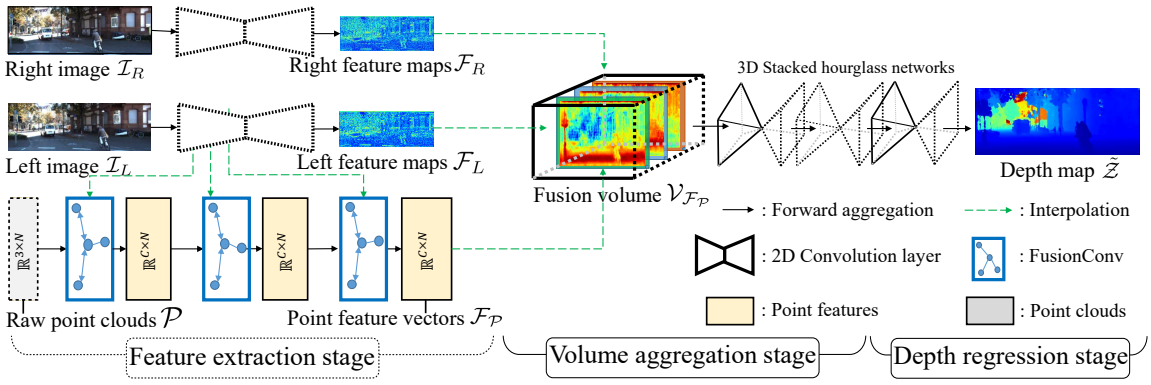

Fig. 2. **Overall architecture.** Our network consists of three stages: feature extraction stage, volume aggregation stage, and depth regression stage. We initially extract features from input modalities. In particular, we use FusionConv layers to infer point feature vectors $\mathcal{F}_\mathcal{P}$. These extracted features are embedded into the fusion volume $\mathcal{V}_{\mathcal{F}_\mathcal{P}}$ to compute the correspondence with stereo feature maps $\mathcal{F}_L$, $\mathcal{F}_R$. After volume aggregation through 3D stacked hourglass networks, we finally obtain the depth map $\tilde{\mathcal{Z}}$ in the referential camera view.

embed stereo image features and point cloud features along the different channels but in a unified volume $\mathcal{V}$.

In the constructed fusion volume, which is empty at the beginning, we first fill in the stereo information. Inspired by the differential cost volume [5], [15], filling the volume with features from sensory data enables the end-to-end learnable voxel representation to estimate depth. From stereo images $\mathcal{I}_L$, $\mathcal{I}_R \in \mathbb{R}^{4W\times 4H\times 3}$, we extract stereo feature maps $\mathcal{F}_L$, $\mathcal{F}_R \in \mathbb{R}^{W\times H\times C}$ using feature extraction layers from our baseline method [15]. By projecting the pre-defined location of each voxel $\mathbf{v}$ into the image domain, we fill the fusion volume $\mathcal{V}$ with stereo features (see Fig. 2). A precise description of the fusion volume composition is included in the supplementary material.

Our fusion volume encapsulates the 3D environments linearly to the metric scale as depth volumes do [20], [21]. However, while traditional depth volumes [20], [21] are the results of transformation from the initially-built cost volume (*i.e.*, from disparity to depth), our fusion volume directly built from sensory features. Since the quantization loss happens when points are embedded into the initially-built volume, direct construction of a metric-scale volume can reduce the quantization loss of points, especially at a farther area. For this purpose, with the known camera matrix $\mathbf{K}$ and the pre-defined location of each voxel in the volume $\mathcal{V}$, we are able to embed point clouds $\mathcal{P}$ into the volume $\mathcal{V}$ as binary representation with less quantization loss at farther area. Voxels embedded by points are filled with 1 (occupied) and the others are filled with 0 (non-occupied or empty). To do so, our fusion volume maintains the geometric and spatial relation of stereo images and point clouds within a unified volumetric space.

So far, we have incorporated stereo feature maps $\mathcal{F}_L$, $\mathcal{F}_R$ and raw point clouds $\mathcal{P}$ into the fusion volume $\mathcal{V}$. With this volume $\mathcal{V}$, we can propagate the embedded points to the overall volume for computing the matching cost between stereo features ($\mathcal{F}_L$, $\mathcal{F}_R$) or among stereo features and point clouds ($\mathcal{F}_L$, $\mathcal{F}_R$, $\mathcal{P}$) through the following 3D convolution layers in stacked hourglass networks (see Fig. 2). Meanwhile, the 3D convolution layers compute both the spatial correspondence and the channel-wise features within the volume $\mathcal{V}$, so that channel-wise information is also an one important factor in our metric-aware fusion. To further facilitate the fusion, we discuss feature extraction from raw point clouds $\mathcal{P}$ in the next section.

V. Fused Convolution Layer

Recently, many research works [22], [23], [24], [25], [26] have proposed feature extraction layers for point clouds and have demonstrated the potential of leveraging local neighbors in aggregating point feature vectors. Despite the progress in deep architectures for point clouds, previous methods [22], [23], [24], [25], [26] focus purely on utilizing raw point clouds (for classification [27] or segmentation tasks [28], [29]) instead of fusing them with different sets of sensor information.

In this section, we introduce an image-guided feature extraction layer for point clouds, called *FusionConv*, which is specialized in sensor fusion-based depth estimation task. Under the known geometric mapping relation between image and point clouds, we design a FusionConv layer that exploits image guidance in several ways. (1) We adaptively cluster neighbors of each point while considering geometric relation (3D metric domain) as well as the semantic relation (2D image domain), which enables us to determine relevant neighbors. (2) We directly fuse the input point feature with the corresponding image feature via interpolation, which implicitly helps to extract distinctive point features.

The proposed FusionConv takes the input feature map of the left image $\mathcal{F}_\mathcal{L}$ and the input point feature vectors $\mathcal{F}_\mathcal{P}^{in} \in \mathbb{R}^{C\times N}$ (extracted from raw point clouds $\mathcal{P} \in \mathbb{R}^{3\times N}$) and estimates the output point feature vectors $\mathcal{F}_\mathcal{P} \in \mathbb{R}^{C\times N}$ by fusing the two different features while taking account of relevant neighbors (see Fig. 3). For simplicity, let $\mathbf{p} \in \mathcal{P}$ and $\mathbf{x}$ be a 3D point and its projected image point by $\mathbf{K}$, respectively. We then denote the corresponding 3D feature vector and 2D image feature vector as $\mathcal{F}_\mathcal{P}(\mathbf{p}) \in \mathbb{R}^{C\times 1}$ and $\mathcal{F}_\mathcal{L}(\mathbf{x}) \in \mathbb{R}^{C\times 1}$, respectively.

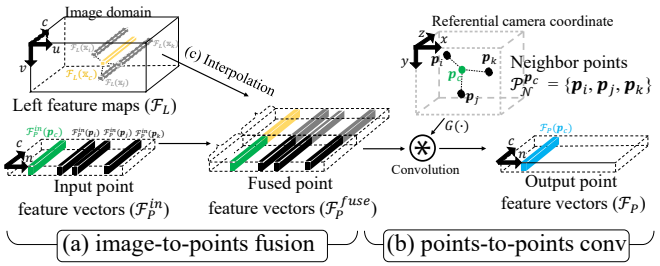

Fig. 3. **FusionConv.** We illustrate the process of FusionConv layer that extracts output point feature vector $\mathcal{F}_{\mathcal{P}}(\mathbf{p}_c)$ from neighboring point clouds $\mathcal{P}_{\mathcal{N}}^{\mathbf{p}_c}$ of a point $\mathbf{p}_c$. In (a) image-to-point fusion, interpolation is used to fuse $\mathcal{F}_L$ and $\mathcal{F}_{\mathcal{P}}^{in}$ and generate the fused point feature vector $\mathcal{F}_{\mathcal{P}}^{fuse}$. Then, we operate (b) points-to-points convolution of fused features $\mathcal{F}_{\mathcal{P}}^{fuse}$ and geometric distance $G(\cdot)$ between adjacent points $\mathcal{P}_{\mathcal{N}}^{\mathbf{p}_c}$ to infer output point feature vector at the point $\mathbf{p}_c$, denoted as $\mathcal{F}_{\mathcal{P}}(\mathbf{p}_c)$. Note that the overall flow is processed after the clustering stage ($\mathcal{P}_{\mathcal{N}}^{\mathbf{p}_c} = \{\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}_k\}$).

Through the mapping relation, the FusionConv layer clusters the neighboring point feature vectors to aggregate the local response in point feature vectors. While many approaches [23], [24], [25], [26] cluster point clouds $\mathcal{P}$ only in a metric space, we need to cluster the neighboring point clouds following the voxel alignment in the volume $\mathcal{V}$, which is also aligned in the referential camera view. Specifically, for each point $\mathbf{p}$ in $\mathcal{P}$, we dynamically cluster neighboring point clouds $\mathcal{P}_{\mathcal{N}}$ that reside within the pre-defined size of the voxel window. For instance in Fig. 3, there are three neighboring points $\mathbf{p}_i$, $\mathbf{p}_j$ and $\mathbf{p}_k$ within the voxel window whose center indicates the point $\mathbf{p}_c$. From these three neighbor points and the center point $\mathbf{p}_c$ itself, FusionConv layer is able to calculate the local response of the point feature vectors at the point $\mathbf{p}_c$ in alignment with the fusion volume $\mathcal{V}$.

The following process of the FusionConv layer consists of image-to-points fusion and points-to-points aggregation. For the fusion process, we first interpolate $\mathcal{F}_L$ into $\mathcal{F}_{\mathcal{P}}$ following the projection mapping relation to obtain the fused point feature vectors $\mathcal{F}_{\mathcal{P}}^{fuse}$. These fused features $\mathcal{F}_{\mathcal{P}}^{fuse}$ have the same number of points as N but they have an extended length of channels containing both $\mathcal{F}_L$ and $\mathcal{F}_{\mathcal{P}}^{in}$. After the image-to-points fusion, we convolve the fused point feature vectors $\mathcal{F}_{\mathcal{P}}^{fuse}$ and the geometric distance between the neighboring points $\mathcal{P}_{\mathcal{N}}$ to aggregate the local response, called points ($\mathcal{P}^{\mathcal{N}}$)-to-points ($\mathcal{F}_{\mathcal{P}}^{fuse}$) convolution. For instance in Fig. 3, the weighted geometric distance $G(\cdot)$ from the center point $\mathbf{p}_c{=}[x_c, y_c, z_c]^\top$ to its neighboring point $\mathbf{p}_i{=}[x_i, y_i, z_i]^\top$ is calculated as:

$$G(\mathbf{p}_c{-}\mathbf{p}_i){=}G(\Delta\mathbf{p}){=}A_0 + A_1{\cdot}\Delta x + A_2{\cdot}\Delta y + A_3{\cdot}\Delta z, \quad (1)$$

where A_0, A_1, A_2 and A_3 are learnable weights and we define $\Delta\mathbf{p}{=}\,[\Delta x, \Delta y, \Delta z]^\top$ as $\Delta x{=}x_c{-}x_i$, $\Delta y{=}y_c{-}y_i$, and $\Delta z{=}z_c{-}z_i$. This weighted geometric distance $G(\cdot)$ becomes the weight of the convolution with the fused point feature vector $\mathcal{F}_{\mathcal{P}}^{fuse}$ to compute the local response at point $\mathbf{p}_c$ as:

$$\mathcal{F}_{\mathcal{P}}(\mathbf{p}_c) = \frac{1}{|\mathcal{P}_{\mathcal{N}}^{\mathbf{p}_c}|} \sum_{\mathbf{p}_i \in \mathcal{P}_{\mathcal{N}}^{\mathbf{p}_c}} \mathcal{F}_{\mathcal{P}}^{fuse}(\mathbf{p}_i) \cdot G(\mathbf{p}_c - \mathbf{p}_i), \quad (2)$$

where $|\mathcal{P}_{\mathcal{N}}^{\mathbf{p}_c}|$ is the number of neighboring points near the point $\mathbf{p}_c$. Thus, this is the way of imposing different weights $G(\cdot)$ on the neighboring point features $\mathcal{F}_{\mathcal{P}}^{in}$ and left feature maps $\mathcal{F}_L$ when extracting the center point feature vectors $\mathcal{F}_{\mathcal{P}}(\mathbf{p}_c)$. To fully compute the output feature vectors from all point clouds, the FusionConv layer iteratively calculates the output point feature vectors $\mathcal{F}_{\mathcal{P}}$ as:

$$\mathcal{F}_{\mathcal{P}} = \left[\mathcal{F}_{\mathcal{P}}(\mathbf{p}_1), \quad \mathcal{F}_{\mathcal{P}}(\mathbf{p}_2), \quad \cdots, \quad \mathcal{F}_{\mathcal{P}}(\mathbf{p}_N)\right]. \quad (3)$$

Finally, we extract the output point feature vector $\mathcal{F}_{\mathcal{P}} \in \mathbb{R}^{C\times N}$ from raw point clouds $\mathcal{P}$ and left feature maps $\mathcal{F}_L$. This point feature vector becomes embedded into the modified fusion volume $\mathcal{V}_{\mathcal{F}_{\mathcal{P}}} \in \mathbb{R}^{3C\times D\times H\times W}$ as in Fig. 2. The embedded location of $\mathcal{F}_{\mathcal{P}}$ is identical to the corresponding spatial locations of raw point clouds $\mathcal{P}$ within the extended channel-wise voxels $2C+1$ $(\mathcal{V}) \rightarrow 3C$ $(\mathcal{V}_{\mathcal{F}_{\mathcal{P}}})$ to maintain the metric accuracy from raw point clouds $\mathcal{P}$. With this fusion volume $\mathcal{V}_{\mathcal{F}_{\mathcal{P}}}$, we can fuse the two different modalities to compute the subpixel matching cost by the following 3D convolution layers as in Fig. 2.

VI. Depth Map Regression

After we fuse features from the two modalities at the fusion volume $\mathcal{V}_{\mathcal{F}_{\mathcal{P}}}$, we propagate the point features to the overall volume and compute the matching cost through the stacked hourglass networks [31], [15] to regress the depth map. Following [15], we perform a cost aggregation process by aggregating the fusion volume along the depth dimension as well as the spatial dimension. In our stacked hourglass networks, the networks consist of three encoder-decoder networks that sequentially refine the cost aggregation via intermediate loss [15], [31], [19]. After aggregation, the cost aggregation reduces the channels of $\mathcal{V}_{\mathcal{F}_{\mathcal{P}}}$ into a 3D structure $\mathcal{A} \in \mathbb{R}^{D\times H\times W}$. From $\mathcal{A}$, we can estimate the depth value $\tilde{z}_{u,v}$ at pixel (u, v) as:

$$\tilde{z}_{u,v} = \sum_{d=0}^{D-1} \frac{d}{D-1} \cdot z_{max} \cdot \sigma(\mathbf{a}_{u,v}^d), \quad (4)$$

where z_{max} is a hyper-parameter defining the maximum range of depth estimation, $\sigma(\cdot)$ represents the softmax operation, and $\mathbf{a}_{u,v}^d$ is the d-th value of the cost aggregation vector $\mathbf{a}_{u,v} \in \mathbb{R}^{D\times 1}$ at (u, v). For our experiments, we set the hyper-parameters as $D = 48$ and $z_{max} = 100$. Specifically, we compute the depth loss $\mathcal{L}_{depth}$ from the estimated depth map $\tilde{\mathcal{Z}}$ and the true depth map $\mathcal{Z}$ as follows:

$$\mathcal{L}_{depth} = \frac{1}{M} \sum_u \sum_v smooth_{L_1}(z_{u,v} - \tilde{z}_{u,v}), \quad (5)$$

where M is the number of valid pixels in $\mathcal{Z}$ for the normalizing factor, $\tilde{z}_{u,v}$ is the value of the predicted depth map $\tilde{\mathcal{Z}}$ at pixel location (u, v), and $smooth_{L_1}(\cdot)$ is the smooth L1 loss function used to compute the loss [5], [15]. Finally, the total loss $\mathcal{L}_{total}$ of our network is computed from the three different intermediate results of depth maps from the three stacked hourglass networks [15], [31] as below:

$$\mathcal{L}_{total} = \sum_{i=1}^{3} w_i \cdot \mathcal{L}_{depth}^i, \quad (6)$$

TABLE I

QUANTITATIVE RESULTS OF DEPTH ESTIMATION NETWORKS IN KITTI COMPLETION VALIDATION BENCHMARK.

* REPRESENTS THE REPRODUCED RESULTS.

Method	Modality	Depth Evaluation (Lower the better)			
		RMSE (mm)	MAE (mm)	iRMSE (1/km)	iMAE (1/km)
PSMnet* [15]	Stereo	884	332	1.649	0.999
Sparse2Dense* [6]	Mono + LiDAR	840.0	-	-	-
Guidenet [7]	Mono + LiDAR	777.78	221.59	2.39	1.00
NLSPN [18]	Mono + LiDAR	771.8	197.3	2.0	0.8
CSPN++ [30]	Mono + LiDAR	725.43	207.88	-	-
Park *et al.* [11]	Stereo + LiDAR	2021.2	500.5	3.39	1.38
LiStereo [12]	Stereo + LiDAR	832.16	283.91	2.19	1.10
CCVN [10]	Stereo + LiDAR	749.3	252.5	**1.3968**	**0.8069**
Ours	Stereo + LiDAR	**636.2**	**205.1**	1.8721	0.9870

TABLE II

QUANTITATIVE RESULTS OF DEPTH ESTIMATION NETWORKS IN THE VIRTUAL-KITTI 2.0 DATASET.

* REPRESENTS THE REPRODUCED RESULTS. S, M+L AND S+L REPRESENT STEREO CAMERAS, MONOCULAR CAMERA WITH A LIDAR AND STEREO CAMERAS WITH A LIDAR RESPECTIVELY.

Method	Modality	Depth Evaluation (Lower the better)			
		RMSE (mm)	MAE (mm)	iRMSE (1/km)	iMAE (1/km)
PSMnet [15]*	S	5728	2235	9.805	4.380
Sparse2Dense [6]*	M+L	3357±8	1336±2.0	12.136±0.045	6.243±0.013
CCVN [10]*	S+L	3726.83±10.83	915.6±0.4	8.814±0.019	**2.456±0.004**
Ours	S+L	**3217.16±1.84**	**712±2.0**	**7.168±0.048**	2.694±0.011

where w_i is the weight of the i-th depth loss $\mathcal{L}^i_{depth}$ (we set the weight to w_1=0.5, w_2=0.7 and w_3=1.0). During the evaluation, we only consider the predicted depth map at the last network, as illustrated in Fig. 2.

VII. EXPERIMENTAL EVALUATION

In this section, we describe the implementation details of our network. Our network is trained on two independent datasets, the KITTI dataset [13] and the Virtual-KITTI dataset [14]. We separately evaluate the accuracy of the depth for each dataset against existing techniques. Additionally, we conduct an ablation study to validate each dominant component of our method and to contrast the early fusion [10] with our intermediate fusion at the fusion volume $\mathcal{V}_{\mathcal{F}_\mathcal{P}}$.

A. Architecture

Our network consists of three stages: the feature extraction stage, volume aggregation stage, and depth regression stage, as depicted in Fig. 2. In the feature extraction stage, we follow the architecture of image feature extraction layers as in Chang and Chen [15], but extract the intermediate left feature map to operate image-to-points fusion in FusionConv layers. We use three FusionConv layers to infer the point feature vectors. Then, the extracted features from input modalities are embedded in the fusion volume as explained in Secs. IV and V. We aggregate the volume through the three stacked hourglass networks and finally regress the depth map as described in Sec. VI.

B. Datasets and training schemes.

KITTI dataset. The KITTI Raw benchmark [32] provides sequential stereo images and LiDAR point clouds under different road environments. Within the benchmark, KITTI completion benchmark [13] provides the true depth maps and the corresponding sensor data, consisting of 42,949 training samples and 1,000 validation samples. Given the input sensory data, we train our network by setting the learning rate as 0.001 for 5 epochs and as 0.0001 for 30 epochs. We use three NVIDIA 1080-Ti GPUs for training, and the batch size is 9. The entire training scheme takes three days and the inference speed of our network during the test is 0.71 FPS (1.40 sec per frame). We use random-crop augmentation during the training phase. For training, the size of the cropped images is 256×512. We also crop point clouds $\mathcal{P}$ that reside within the cropped left images. Usually, there are ∼5K point clouds in the training phase, but it can vary depending on the location of the cropped area within images. For testing, we fully utilize the original shape of images and raw point clouds (∼25K points per image) without any augmentation or filtering.

Virtual KITTI 2.0 dataset. Virtual KITTI 2.0 [14] is a recently published dataset that provides much more realistic images than the previous version of the Virtual-KITTI 1.3.1 [33]. The merits of the synthetic environment include access to dense ground truth at the farther areas, while the KITTI completion benchmark [13] provides relatively sparse ground truth. There are five scenes in this dataset and each scene contains ten different scenarios, such as rain, sunset, *etc*. We set the two scenes (Scene01, Scene02) for training the network and the other scenes are exploited for evaluating the accuracy of depth maps. For each scene, we take the only scenario (15-deg-left) for training and evaluation. In total, there are 680 images in the training set and 1,446 images in the test set. Though the raw point cloud data is not provided in this dataset, we randomly sample the ground truth depth pixels and regard the pixels as the point clouds. We select the same number of selected point clouds as for the KITTI dataset (*i.e.*, 5K points for training and 25K points for testing). Given pre-trained weights from the KITTI dataset, we fine-tune the network for 5K iterations under the identical augmentation methods as we adopt for the KITTI dataset.

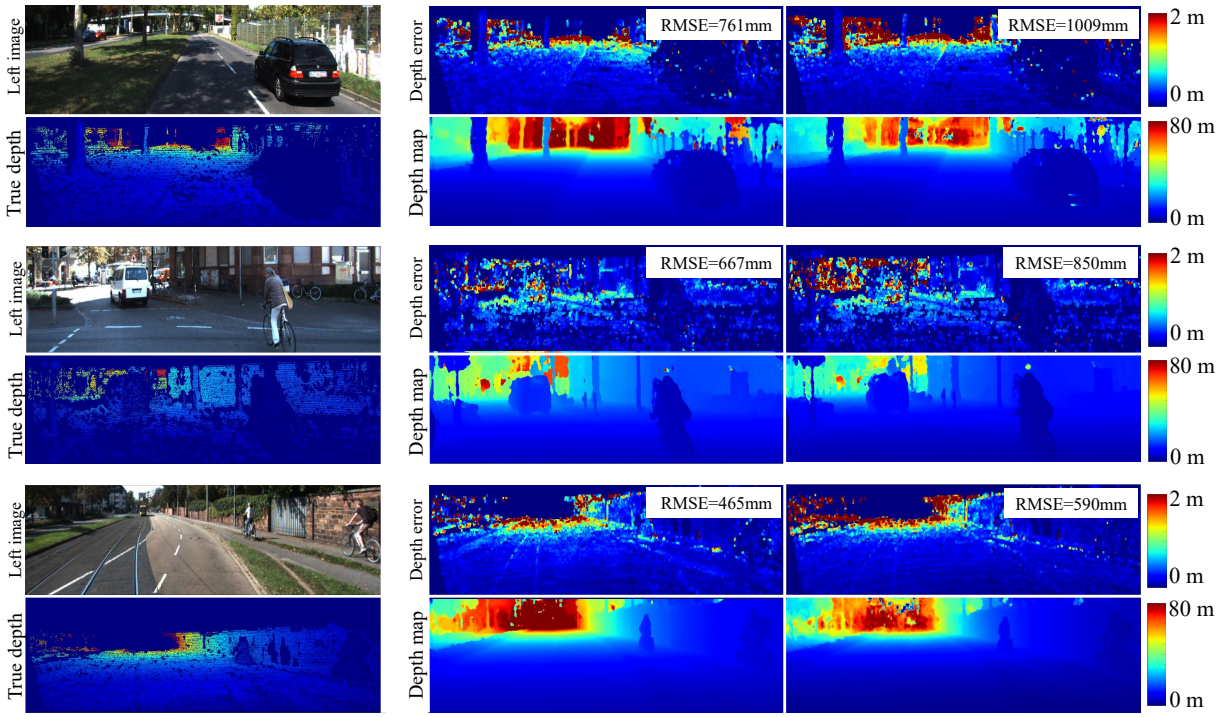

Fig. 4. **Qualitative results on the KITTI dataset.** We visualize depth maps and depth errors from ours and the recent stereo-Lidar method (CCVN [10]) in three different cases. We also include the depth metric RMSE (lower the better).

Metrics. We evaluate the quality of the estimated depth by our network. We follow the metric scheme proposed by Eigen *et al*. [34] which is identical to the official KITTI depth completion benchmark [13], *i.e*., RMSE, MAE, iRMSE, iMAE, where RMSE and MAE are the target metrics or evaluating the metric distance. These metric formulations are equivalently applied to measure the performance on the Virtual-KITTI dataset [14]. Since the Virtual-KITTI dataset does not provide raw point clouds data, we evaluate the depth metric by 5 times of repetitive sampling of the ground truth depth pixels. The resulting metric in Table II is the average over the samples. Note that we re-train the network by accessing the open-source code of the target method and denote the re-evaluated results by a '*' (*e.g*., PSMnet*) as in Tables I and II.

Comparison. We evaluate our network against existing methods [12], [11], [10] for the KITTI completion validation benchmark (Table I) and the Virtual-KITTI 2.0 (Table II). We also include the performance of other depth estimation networks, such as stereo matching methods [15], [5] and depth completion studies [6], [7], [18], [30]. Among the existing methods, our method achieves state-of-the-art depth performance in RMSE and MAE as in Tables I and II. These results suggest that our method shows higher accuracy in the estimation of the depth at a distant area. This is consistent with our intention to create a metric-linear volumetric design. To further verify the strength of our method, we provide the qualitative results for the KITTI dataset and the Virtual-KITTI dataset in Figs. 3 and 4. We attribute this improvement to the fusion at the fusion volume that simultaneously computes matching cost between features of the two input modalities. More quantitative results and information about the inference time are available in the supplementary material.

TABLE III

ABLATION STUDY OF THE TYPE OF VOLUME FOR DEPTH ESTIMATION. WE DIFFERENTIATE THE TYPE OF VOLUME AS COST VOLUME [5] (*i.e*., COSTV), DEPTH VOLUME (*i.e*., DEPTHV) BY YOU *et al*. [35], AND OUR FUSION VOLUME $\mathcal{V}$ (*i.e*., FUSIONV). NOTE THAT DEPTH VOLUME (*i.e*., DEPTHV) BY YOU *et al*. [35] IS A TRANSFORMED VOLUME FROM THE COST VOLUME, BUT OUR METHOD DIRECTLY INTERPOLATES SENSOR DATA INTO OUR VOLUME $\mathcal{V}$. WE EVALUATE EACH METHOD ON THE KITTI COMPLETION VALIDATION BENCHMARK [13].

	Preserve (✓) Type of Volume			Depth Evaluation (Lower the better)			
	CostV [5]	DepthV [35]	FusionV (ours)	RMSE (mm)	MAE (mm)	iRMSE (1/km)	iMAE (1/km)
1	✓			884	332	**1.649**	0.999
2		✓		797	286	2.046	1.070
3			✓	**636.2**	**205.1**	1.8721	**0.9870**

VIII. ABLATION STUDY

In this section, we extensively investigate our neural modules: the fusion volume and the FusionConv layer. The following experiments are conducted on the KITTI dataset [13].

Fusion volume. We compare our fusion volume with other volume representations, such as cost volume [5] and depth volume [35] as shown in Table III. Note that the depth volume [35] is built via cost volume, but our fusion volume directly encodes point feature vectors $\mathcal{F}_{\mathcal{P}}$ into the metric-aware voxels. For a fair comparison, we evaluate each type of

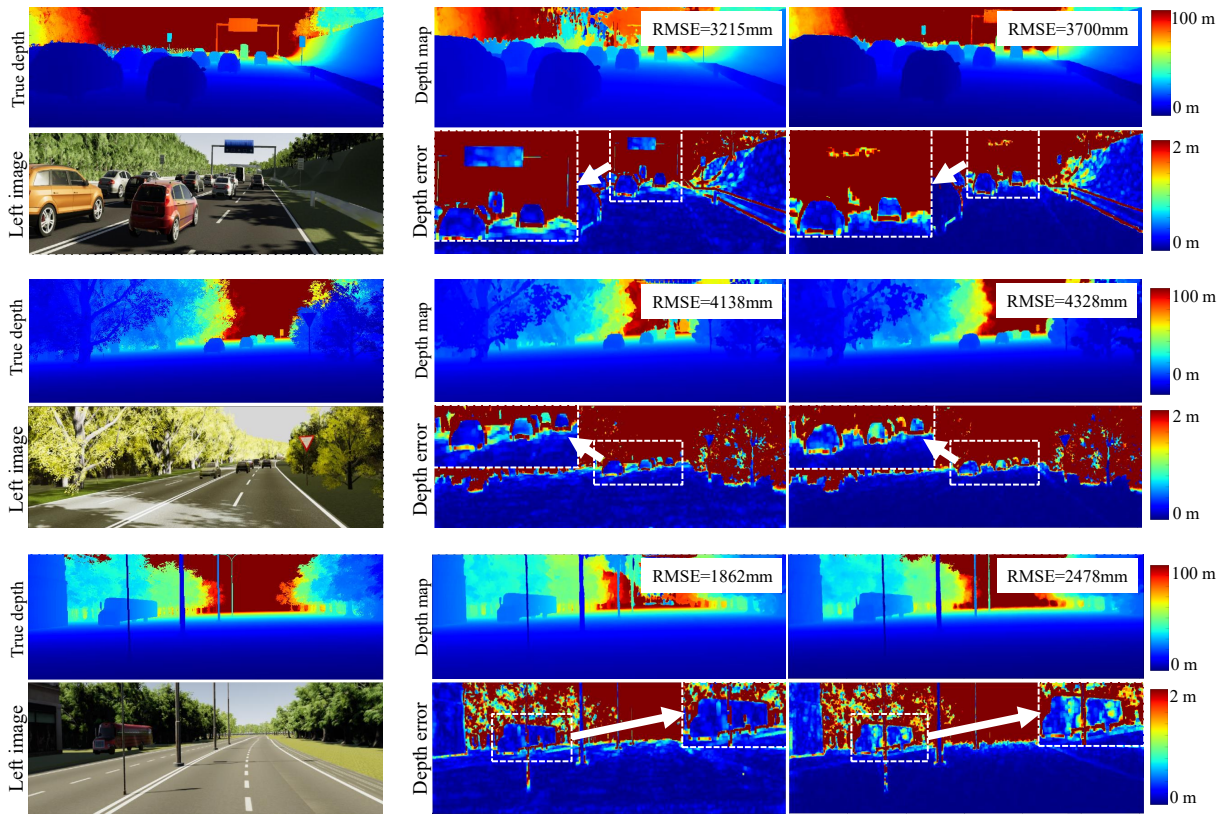

Fig. 5. **Qualitative results on the Virtual-KITTI 2.0 dataset.** We evaluate the estimated depth from both our network and the recent Stereo-LiDAR fusion network by Wang *et al*. [10]. This synthetic data covers the wide range of depth upto $655m$, but we clamp true depth maps and estimated depth maps upto $100m$ during the evaluation, as in Table II. For the detailed visualization, we crop and enlarge the part of depth error maps in each frame. Mainly, the cropped images correspond to the farther area to validate our long-range depth estimation.

TABLE IV

ABLATION STUDY OF TYPE OF FUSION FOR DEPTH ESTIMATION. EACH METHOD EMBEDS DIFFERENT INFORMATION INTO THE FUSION VOLUME, SUCH AS RAW POINT CLOUD $\mathcal{P}$, POINT FEATURE VECTORS FROM MULTILAYER PERCEPTRON (MLP) BY QI *et al*. [22] AND POINT FEATURE VECTORS FROM OUR FUSIONCONV LAYER.

	Preserve (✓) Type of point network			Depth Evaluation (Lower the better)			
	Raw points	MLP [22]	FusionConv (ours)	RMSE (mm)	MAE (mm)	iRMSE (1/km)	iMAE (1/km)
4	✓			669	226	2.169	1.120
5		✓		652	212	2.099	1.066
6			✓	**636.2**	**205.1**	**1.8721**	**0.9870**

TABLE V

ABLATION STUDY OF DIFFERENT LEVELS OF FUSION. EARLY FUSION TAKES PRE-PROCESSING STEPS TO PROJECT POINT CLOUDS INTO IMAGE DOMAIN FOR FUSION, WHILE INTERMEDIATE FUSION PROPOSES FUSION AT FEATURE SPACE, *e.g.*, FUSION VOLUME.

	Preserve (✓) Level of fusion			Depth Evaluation (Lower the better)			
	Early fusion	Intermediate fusion (ours)	Volume	RMSE (mm)	MAE (mm)	iRMSE (1/km)	iMAE (1/km)
7	✓		CostV	744	249	2.026	1.022
8	✓		FusionV	650	215	1.912	**0.964**
9		✓	FusionV	**636.2**	**205.1**	**1.8721**	0.9870

volume under identical conditions, such as deep architectures (*e.g.*, FusionConv) and input sensory data (*e.g.*, stereo-lidar fusion). In Table III, our fusion volume shows the highest accuracy among different voxel representations. We deduce the reason that our volume can directly encode the 3D metric point into the volume, while the other voxel representation [35] is constructed via cost volume, which can lose metric information during the transformation.

FusionConv. In Table IV, we decompose FusionConv into three factors: convolution, cluster, and fusion, to analyze each component in the proposed layers. As a baseline (Method 4 in Table IV), we embed the raw point clouds $\mathcal{P}$ into the volume $\mathcal{V}_{\mathcal{P}}$ as described in Sec. IV. Moreover, we set another baseline network as a multilayer perceptron layer (*i.e.*, fully-connected layer) for point feature extraction [22] (Method 5), which does not infer point feature vectors $\mathcal{F}_{\mathcal{P}}$ via clustering. Though the quality of depth from the two baseline methods outperforms the previous method [10], the convolution operation among the neighboring points $\mathcal{P}_{\mathcal{N}}$, as in FusionConv, further increases the accuracy of the depth estimation (Method 6). This confirms that our clustering strategy and image-to-point fusion are effective for the sensor fusion-based depth estimation task. Finally, our FusionConv layer (Method 6) shows the highest metric performance.

Early fusion. Our method proposes fusion in the feature space, called intermediate fusion, through the fusion volume. However, previous methods [12], [11], [10] internally use the pre-processing stage to project raw point clouds $\mathcal{P}$ into the image domain to concatenate them with RGB images,

called early fusion. With this ablation study, we validate the performance gap between early fusion and our intermediate fusion to prove the effectiveness of fusion in a 3D metric space. For a fair comparison with different levels of fusion, we use similar architectures with different types of volume as listed in Table V. For early fusion, we do not embed point clouds into the volume but undergo the pre-processing stage as previous works proposed [12], [11], [10]. Method 9 represents our method. In Table V, it reveals that intermediate fusion in a 3D voxel space (Methods 8, 9) shows better results than the early fusion approach (Method 7). We deduce that our fusion scheme takes into consideration the spatial connectivity of point clouds and stereo images for seamless fusion and obtains long-range depth maps.

IX. Conclusion

In this paper, we propose a volumetric propagation network for stereo-LiDAR fusion and perform the long-range depth estimation. To this end, we design two dominant modules, fusion volume and FusionConv, to facilitate the fusion in a unified volumetric space. Within the fusion volume, we formulate the fusion as volumetric propagation by considering the spatial connectivity of sparse point features and densely-ordered stereo images features. Our method demonstrates state-of-the-art performance on the KITTI and the Virtual-KITTI datasets and delivers a message about the geometric-aware stereo-LiDAR fusion.

References

[1] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[2] H. Hirschmuller, "Accurate and efficient stereo processing by semi-global matching and mutual information," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[3] F. Guney and A. Geiger, "Displets: Resolving stereo ambiguities using object knowledge," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[4] 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 Conference on Computer Vision and Pattern Recognition*, 2016.

[5] A. Kendall, H. Martirosyan, S. Dasgupta, and P. Henry, "End-to-end learning of geometry and context for deep stereo regression," in *IEEE International Conference on Computer Vision*, 2017.

[6] F. Mal and S. Karaman, "Sparse-to-dense: Depth prediction from sparse depth samples and a single image," in *IEEE International Conference on Robotics and Automation*, 2018.

[7] J. Tang, F.-P. Tian, W. Feng, J. Li, and P. Tan, "Learning guided convolutional network for depth completion," *arXiv preprint arXiv:1908.01238*, 2019.

[8] J. Choe, K. Joo, T. Imtiaz, and I. S. Kweon, "Stereo object matching network," in *IEEE International Conference on Robotics and Automation*, 2021.

[9] J. Choe, K. Joo, F. Rameau, G. Shim, and I. S. Kweon, "Segment2regress: Monocular 3d vehicle localization in two stages." in *Robotics: Science and Systems*, 2019.

[10] T.-H. Wang, H.-N. Hu, C. H. Lin, Y.-H. Tsai, W.-C. Chiu, and M. Sun, "3d lidar and stereo fusion using stereo matching network with conditional cost volume normalization," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019.

[11] K. Park, S. Kim, and K. Sohn, "High-precision depth estimation with the 3d lidar and stereo fusion," in *IEEE International Conference on Robotics and Automation*, 2018.

[12] J. Zhang, M. S. Ramanagopalg, R. Vasudevan, and M. Johnson-Roberson, "Listereo: Generate dense depth maps from lidar and stereo imagery," in *IEEE International Conference on Robotics and Automation*, 2020.

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

[14] Y. Cabon, N. Murray, and M. Humenberger, "Virtual kitti 2," *arXiv preprint arXiv:2001.10773*, 2020.

[15] J.-R. Chang and Y.-S. Chen, "Pyramid stereo matching network," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[16] S. Im, H.-G. Jeon, S. Lin, and I. S. Kweon, "Dpsnet: End-to-end deep plane sweep stereo," *International Conference on Learning Representations*, 2019.

[17] Y. Chen, B. Yang, M. Liang, and R. Urtasun, "Learning joint 2d-3d representations for depth completion," in *IEEE International Conference on Computer Vision*, 2019.

[18] J. Park, K. Joo, Z. Hu, C.-K. Liu, and I. S. Kweon, "Non-local spatial propagation network for depth completion," in *European Conference on Computer Vision*, 2020.

[19] F. Zhang, V. Prisacariu, R. Yang, and P. H. Torr, "Ga-net: Guided aggregation net for end-to-end stereo matching," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[20] Y. Wang, W.-L. Chao, D. Garg, B. Hariharan, M. Campbell, and K. Q. Weinberger, "Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[21] Y. Chen, S. Liu, X. Shen, and J. Jia, "Dsgn: Deep stereo geometry network for 3d object detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[22] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[23] S. Wang, S. Suo, W.-C. Ma, A. Pokrovsky, and R. Urtasun, "Deep parametric continuous convolutional neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[24] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "Pointcnn: Convolution on x-transformed points," in *Advances in neural information processing systems*, 2018.

[25] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions on Graphics*, vol. 38, no. 5, pp. 1–12, 2019.

[26] J. Mao, X. Wang, and H. Li, "Interpolated convolutional networks for 3d point cloud understanding," in *IEEE International Conference on Computer Vision*, 2019.

[27] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, *et al.*, "Shapenet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015.

[28] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese, "3d semantic parsing of large-scale indoor spaces," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[29] L. Yi, V. G. Kim, D. Ceylan, I.-C. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, and L. Guibas, "A scalable active framework for region annotation in 3d shape collections," *ACM Transactions on Graphics*, vol. 35, no. 6, pp. 1–12, 2016.

[30] X. Cheng, P. Wang, C. Guan, and R. Yang, "Cspn++: Learning context and resource aware convolutional spatial propagation networks for depth completion," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[31] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *European Conference on Computer Vision*, 2016.

[32] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[33] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig, "Virtual worlds as proxy for multi-object tracking analysis," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[34] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Advances in Neural Information Processing Systems*, 2014.

[35] Y. You, Y. Wang, W.-L. Chao, D. Garg, G. Pleiss, B. Hariharan, M. Campbell, and K. Q. Weinberger, "Pseudo-lidar++: Accurate depth for 3d object detection in autonomous driving," *International Conference on Learning Representations*, 2020.