DSGN: Deep Stereo Geometry Network for 3D Object Detection

Yilun Chen[1] Shu Liu[2] Xiaoyong Shen[2] Jiaya Jia[1,2]
[1]The Chinese University of Hong Kong [2]SmartMore
{ylchen, leojia}@cse.cuhk.edu.hk {sliu, xiaoyong}@smartmore.com

Abstract

Most state-of-the-art 3D object detectors heavily rely on LiDAR sensors because there is a large performance gap between image-based and LiDAR-based methods. It is caused by the way to form representation for the prediction in 3D scenarios. Our method, called Deep Stereo Geometry Network (DSGN), significantly reduces this gap by detecting 3D objects on a differentiable volumetric representation – 3D geometric volume, which effectively encodes 3D geometric structure for 3D regular space. With this representation, we learn depth information and semantic cues simultaneously. For the first time, we provide a simple and effective one-stage stereo-based 3D detection pipeline that jointly estimates the depth and detects 3D objects in an end-to-end learning manner. Our approach outperforms previous stereo-based 3D detectors (about 10 higher in terms of AP) and even achieves comparable performance with several LiDAR-based methods on the KITTI 3D object detection leaderboard. Our code is publicly available at https://github.com/chenyilun95/DSGN.

1. Introduction

3D scene understanding is a challenging task in 3D perception, which serves as a basic component for autonomous driving and robotics. Due to the great capability of LiDAR sensors to accurately retrieve 3D information, we witness fast progress on 3D object detection. Various 3D object detectors were proposed [9, 25, 60, 28, 29, 35, 41, 55, 10] to exploit LiDAR point cloud representation. The limitation of LiDAR is on the relatively sparse resolution of data with several laser beams and on the high price of the devices.

In comparison, video cameras are cheaper and are with much denser resolutions. The way to compute scene depth on stereo images is to consider disparity via stereo correspondence estimation. Albeit recently several 3D detectors based on either monocular [38, 7, 6, 32, 50] or stereo [27, 47, 39, 58] setting push the limit of image-based 3D object detection, the accuracy is still left far behind compared with the LiDAR-based approaches.

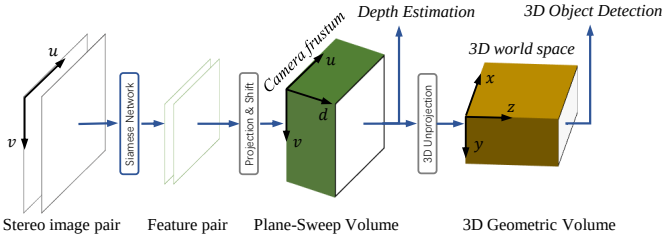

Figure 1. DSGN jointly estimates depth and detects 3D objects from a stereo image pair. It intermediately generates a plane-sweep volume and 3D geometric volume to represent 3D structure in two different 3D space.

Challenges One of the greatest challenges for image-based approaches is to give appropriate and effective representation for predicting 3D objects. Most recent work [27, 38, 50, 39, 42, 2] divides this task into two sub ones, *i.e.*, depth prediction and object detection. Camera projection is a process that maps 3D world into a 2D image. One 3D feature in different object poses causes local appearance changes, making it hard for a 2D network to extract stable 3D information.

Another line of solutions [47, 58, 49, 32] generate intermediate point cloud followed by a LiDAR-based 3D object detector. This 3D representation is less effective since the transformation is non-differentiable and incorporates several independent networks. Besides, the point cloud faces the challenge of object artifacts [19, 49, 58] that limits the detection accuracy of the following 3D object detector.

Our Solution In this paper, we propose a stereo-based end-to-end 3D object detection pipeline (Figure 1) – Deep Stereo Geometry Network (DSGN), which relies on space transformation from 2D features to an effective 3D structure, called 3D geometric volume (3DGV).

The insight behind 3DGV lies in the approach to construct the 3D volume that encodes 3D geometry. 3D geometric volume is defined in 3D world space, transformed from a plane-sweep volume (PSV) [11, 12] constructed in the camera frustum. The pixel-correspondence constraint can be well learned in PSV, while 3D features for real-world

objects can be learned in 3DGV. The volume construction is fully differentiable and thus can be jointly optimized for learning of both stereo matching and object detection.

This volumetric representation has two key advantages. First, it is easy to impose the pixel-correspondence constraint and encode full depth information into 3D real-world volume. Second, it provides 3D representation with geometry information that makes it possible to learn 3D geometric features for real-world objects. As far as we know, there was no study yet to explicitly investigate the way of encoding 3D geometry into an image-based detection network. Our contribution is summarized as follows.

- To bridge the gap between 2D image and 3D space, we establish stereo correspondence in a plane-sweep volume and then transform it to 3D geometric volume for capability to encode both 3D geometry and semantic cues for prediction in 3D regular space.
- We design an end-to-end pipeline for extracting *pixel-level* features for stereo matching and *high-level* features for object recognition. The proposed network jointly estimates scene depth and detects 3D objects in 3D world, enabling many practical applications.
- Without bells and whistles, our simple and fully-differentiable network outperforms all other stereo-based 3D object detectors (10 points higher in terms of AP) on the official KITTI leaderboard [14].

2. Related Work

We briefly review recent work on stereo matching and multi-view stereo. Then we survey 3D object detection based on LiDAR, monocular images, and stereo images.

Stereo Matching In the field of stereo matching on binocular images, methods of [22, 4, 59, 15, 45, 48] process the left and right images by a Siamese network and construct a 3D cost volume to compute the matching cost. Correlation-based cost volume is applied in recent work [33, 57, 54, 15, 30, 44]. GC-Net [22] forms a concatenation-based cost volume and applies 3D convolution to regress disparity estimates. Recent PSMNet [4] further improves the accuracy by introducing pyramid pooling module and stacks hourglass modules [34]. State-of-the-art methods already achieved less than 2% 3-pixel error on KITTI 2015 stereo benchmark.

Multi-View Stereo Methods of [5, 56, 20, 21, 18, 17] reconstruct 3D objects in a multi-view stereo setting [1, 3].

MVSNet [56] constructs plane-sweep volumes upon a *camera frustum* to generate the depth map for each view. Point-MVSNet [5] instead intermediately transforms the plane-sweep volume to point cloud representation to save computation. Kar *et al*. [21] proposed the differentiable projection and unprojection operation on multi-view images.

LiDAR-based 3D Detection LiDAR sensors are very powerful, proven by several leading 3D detectors. Generally two types of architectures, i.e., voxel-based approaches [60, 9, 26, 10] and point-based approaches [36, 37, 41, 55, 51], were proposed to process point cloud.

Image-based 3D Detection Another line of detection is based on images. Regardless of monocular- or stereo-based setting, methods can be classified into two types according to intermediate representation existence.

3D detector with depth predictor: the solution relies on 2D image detectors and depth information extraction from monocular or stereo images. Stereo R-CNN$_{\text{Stereo}}$ [27] formulates 3D detection into multiple branches/stages to explicitly resolve several constraints. We note that the keypoint constraint may be hard to generalize to other categories like *Pedestrian*, and the dense alignment for stereo matching directly operating raw RGB images may be vulnerable to occlusion.

MonoGRNet$_{Mono}$ [38] consists of four subnetworks for progressive 3D localization and directly learning 3D information based solely on semantic cues. MonoDIS$_{\text{Mono}}$ [42] disentangles the loss for 2D and 3D detection. It achieves both tasks in an end-to-end manner. M3D-RPN$_{\text{Mono}}$ [2] applies multiple 2D convolutions of non-shared weights to learn location-specific features for joint prediction of 2D and 3D boxes. Triangulation$_{\text{Stereo}}$ [39] directly learns offset from predefined 3D anchors on bird's eye view and establishes object correspondence on RoI-level features. Due to low resolutions, pixel correspondence is not fully exploited.

3D representation based 3D Detector: 3DOP$_{\text{Stereo}}$ [7, 8] generates point cloud by stereo and encodes the prior knowledge and depth in an energy function. Several methods [47, 58, 49, 32] transform the depth map to Pseudo-LiDAR (point cloud) intermediately followed by another independent network. This pipeline yields large improvement over previous methods. OFT-Net$_{\text{Mono}}$ [40] maps image feature into an orthographic bird's eye view representation and detects 3D objects on bird's eye view.

3. Our Approach

In this section, we first explore the proper representation for 3D space and motivate our network design. Based on the discussion, we present our complete 3D detection pipeline under a binocular image pair setting.

3.1. Motivation

Due to perspective, objects appear smaller with the increase of distance, which makes it possible to roughly estimate the depth according to the relative scale of objects sizes and the context. However, 3D objects of the same category may still have various sizes and orientations. It

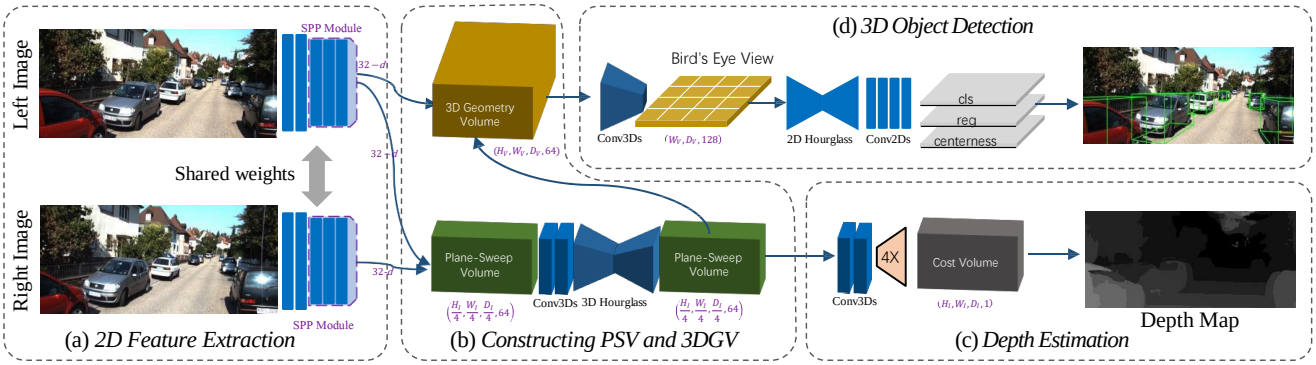

Figure 2. Overview of Deep Stereo Geometry Network (DSGN). The whole neural network consists of four components. (a) A 2D image feature extractor for capture of both *pixel-* and *high-level* feature. (b) Constructing the plane-sweep volume and 3D geometric volume. (c) Depth Estimation on the plane-sweep volume. (d) 3D object detection on 3D geometric volume.

greatly increases the difficulty to make accurate prediction.

Besides, the visual effect of *foreshortening* causes that nearby 3D objects are not scaled evenly in images. A regular cuboid car appears like an irregular frustum. These two problems impose major challenges for 2D neural networks to model the relationship between 2D imaging and real 3D objects [27]. Thus, instead of relying on 2D representation, by reversing the process of projection, an intermediate 3D representation provides a more promising way for 3D object understanding. The following two representations can be typically used in 3D world.

Point-based Representation Current state-of-the-art pipelines [47, 58, 32] generate intermediate 3D structure of point cloud by depth prediction approaches [13, 4, 22] and apply LiDAR-based 3D object detectors. The main possible weakness is that it involves several independent networks and potentially loses information during intermediate transformation, making the 3D structure (such as cost volume) boiled down to point cloud.

This representation often encounters streaking artifacts near object edges [19, 49, 58]. Besides, the network is hard to be differentiated for multi-object scenes [5, 10].

Voxel-based Representation Volumetric representation, as another way of 3D representation, is investigated less intensively. OFT-Net$_{\text{mono}}$ [40] directly maps the image feature to the 3D voxel grid and then collapses it to the feature on bird's eye view. However, this transformation keeps the 2D representation for this view and does not explicitly encode the 3D geometry of data.

Our Advantage The key to establishment of an effective 3D representation relies on the ability to encode accurate 3D geometric information of the 3D space. A stereo camera provides an explicit pixel-correspondence constraint for computing depth. Aiming to design a unified network to exploit this constraint, we explore deep architectures capable of extracting both *pixel-level* features for stereo correspondence and *high-level* features for semantic cues.

On the other hand, the pixel-correspondence constraint is supposedly imposed along the projection ray through each pixel where the depth is considered to be *definite*. To this end, we create an intermediate plane-sweep volume from a binocular image pair to learn stereo correspondence constraint in camera frustum and then transform it to a 3D volume in 3D space. In this 3D volume with 3D geometric information lifted from the plane-sweep volume, we are able to well learn 3D features for real-world objects.

3.2. Deep Stereo Geometry Network

In this subsection, we describe our overall pipeline – Deep Stereo Geometry Network (DSGN) as shown in Figure 2. Taking the input of a binocular image pair (I_L, I_R), we extract features by a Siamese network and construct a plane-sweep volume (PSV). The pixel-correspondence is learned on this volume. By differentiable warping, we transform PSV to a 3D geometric volume (3DGV) to establish 3D geometry in *3D world space*. Then the following 3D neural network on the 3D volume learns necessary structure for 3D object detection.

3.2.1 Image Feature Extraction

Networks for stereo matching [22, 4, 15] and object recognition [16, 43] have different architecture designs for their respective tasks. To ensure reasonable accuracy of stereo matching, we adopt the main design of PSMNet [4].

Because the detection network requires a discriminative feature based on high-level semantic features and large context information, we modify the network for grasping more high-level information. Besides, the following 3D CNN for cost volume aggregation takes much more computation, which gives us room to modify the 2D feature extractor without introducing extra heavy computation overhead in the overall network.

Network Architecture Details Here we use the notations `conv_1`, `conv_2`, ..., `conv_5` following [16]. The key modification for 2D feature extractor is as follows.

- Shift more computation from `conv_3` to `conv_4` and `conv_5`, *i.e.*, changing the numbers of basic blocks of `conv_2` to `conv_5` from $\{3, 16, 3, 3\}$ to $\{3, 6, 12, 4\}$.
- The SPP module used in PSMNet concatenates the output layers of `conv_4` and `conv_5`.
- The output channel number of convolutions in `conv_1` is 64 instead of 32 and the output channel number of a basic residual block is 192 instead of 128.

Full details of our 2D feature extraction network are included in the supplementary material.

3.2.2 Constructing 3D Geometric Volume

To learn 3D convolutional features in 3D regular space, we first create a 3D geometric volume (3DGV) by warping a plane-sweep volume to 3D regular space. Without loss of generality, we discretize the region of interest in *3D world space* to a 3D voxel occupancy grid of size (W_V, H_V, D_V) along the right, down and front directions in camera view. W_V, H_V, D_V denote the width, height and length of the grid, respectively. Each voxel is of size (v_w, v_h, v_d).

Plane-Sweep Volume In binocular vision, an image pair (I_L, I_R) is used to construct a disparity-based cost volume for computing matching cost, which matches a pixel i in the left image I_L to the correspondence in the right image I_R horizontally shifted by an integral disparity value d. The depth is inversely proportional to disparity.

It is thus hard to distinguish among distant objects due to the similar disparity values [27, 47, 58]. For example, objects 40-meter and 39-meter away have almost no difference (< 0.25pix) on disparity on KITTI benchmark [14].

In a different way to construct the cost volume, we follow the classic plane sweeping approach [11, 12, 56] to construct a plane-sweep volume by concatenating the left image feature F_L and the reprojected right image feature $F_{R->L}$ at equally spaced depth interval, which avoids imbalanced mapping of features to 3D space.

The coordinate of PSV is represented by (u, v, d), where (u, v) represents (u, v)-pixel in the image and it adds another axis orthogonal to the image plane for depth. We call the space of (u, v, d) grid *camera frustum space*. The depth candidates d_i are uniformly sampled along the depth dimension with interval v_d following the pre-defined 3D grid. Concatenation-based volume enables the network to learn semantic features for object recognition.

We apply 3D convolution to this volume and finally get a matching cost volume for all depth. To ease computation, we apply only one 3D hourglass module, contrary to the three used in PSMNet [4]. We note that the resulting performance degradation can be compensated in the following detection network since the overall network is differentiable.

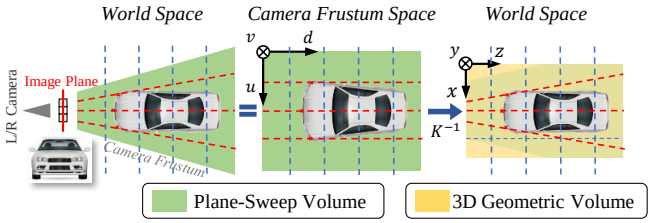

Figure 3. Illustration of volume transformation. The image is captured at the image plane (red solid line). PSV is constructed by projecting images at equally spaced depth (blue dotted lines) in left camera frustum, which is shown in the *3D world space* (left) and *camera frustum space* (middle). Car is shown to be distorted in the middle. Mapping by the camera intrinsic matrix K, PSV is warped to 3DGV, which restores the car.

3D Geometric Volume With known camera internal parameters, we transform the last feature map of PSV before computing matching cost from *camera frustum space* (u, v, d) to *3D world space* (x, y, z) by reversing 3D projection with

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 1/f_x & 0 & -c_u/f_x \\ 0 & 1/f_y & -c_v/f_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} ud \\ vd \\ d \end{pmatrix} \quad (1)$$

where f_x, f_y are the horizontal and vertical focal lengths. This transformation is fully-differentiable and saves computation by eliminating background outside the pre-defined grid, such as the sky. It can be implemented by warp operation with *trilinear* interpolation.

Figure 3 illustrates the transformation process. The common pixel-correspondence constraint (red dotted lines) is imposed in *camera frustum* while object recognition is learned in regular *3D world space* (*Euclidean space*). There obviously is difference in these two representations.

In the last feature map of plane-sweep volume, a low-cost voxel (u, v, d) means the high probability of object existing at depth d along the ray through the focal point and image point (u, v). With the transformation to regular *3D world space*, the feature of low cost suggests that this voxel is occupied in the front surface of the scene, which can serve as a feature for 3D geometric structure. Thus it is possible for the following 3D network to learn 3D object features on this volume.

This operation is fundamentally different from differentiable unprojection [21], which directly lifts the image feature from 2D image frame to 3D world by *bilinear* interpolation. Our goal is to lift geometric information from cost volume to 3D world grid. We make pixel-correspondence constraint easy to be imposed along the projection ray.

The contemporary work [58] applies a similar idea to

construct depth-cost-volume like plane-sweep volume. Differently, we aim to avoid imbalanced warping from plane-sweep volume to 3D geometric volume, and deal with the streaking artifact problem. Besides, our transformation keeps the distribution of depth instead of deducting it to a depth map. Our strategy intriguingly avoids object artifacts.

3.2.3 Depth Regression on Plane-Sweep Cost Volume

To compute the matching cost on the plane-sweep volume, we reduce the final feature map of plane-sweep volume by two 3D convolutions to get 1D cost volume (called plane-sweep cost volume). Soft arg-min operation [22, 4, 59] is applied to compute the expectation for all depth candidates with probability $\sigma(-c_d)$ as

$$\hat{d} = \sum_{d \in \{z_{\min}, z_{\min}+v_d, \ldots, z_{\max}\}} d \times \sigma(-c_d) \tag{2}$$

where the depth candidates are uniformly sampled within pre-defined grid $[z_{\min}, z_{\max}]$ with interval v_d. The softmax function encourages the model to pick a single depth plane per pixel.

3.2.4 3D Object Detector on 3D Geometric Volume

Motivated by recent one-stage 2D detector FCOS [46], we extend the idea of *centerness* branch in our pipeline and design a distance-based strategy to assign targets for the real world. Because objects of the same category are of similar size in 3D scene, we still keep the design of anchors.

Let $\mathcal{V} \in \mathbb{R}^{W \times H \times D \times C}$ be the feature map for 3DGV of size (W, H, D) and denote the channels as C. Considering the scenario of autonomous driving, we gradually downsample along the height dimension and finally get the feature map $\mathcal{F}$ of size (W, H) for bird's eye view. The network architecture is included in the supplementary material.

For each location (x, z) in $\mathcal{F}$, several anchors of different orientations and sizes are placed. Anchors $\mathbf{A}$ and ground-truth boxes $\mathbf{G}$ are represented by the location, prior size and orientation, *i.e.*, $(x_\mathbf{A}, y_\mathbf{A}, z_\mathbf{A}, h_\mathbf{A}, w_\mathbf{A}, l_\mathbf{A}, \theta_A)$ and $(x_\mathbf{G}, y_\mathbf{G}, z_\mathbf{G}, h_\mathbf{G}, w_\mathbf{G}, l_\mathbf{G}, \theta_\mathbf{G})$. Our network regresses from anchor and gets the final prediction $(h_\mathbf{A} e^{\delta h}, w_\mathbf{A} e^{\delta w}, l_\mathbf{A} e^{\delta l}, x_\mathbf{A} + \delta x, y_\mathbf{A} + \delta y, z_\mathbf{A} + \delta z, \theta_\mathbf{A} + \pi / N_\theta \tanh(\delta\theta))$, where N_θ denotes the number of anchor orientations and $\delta\cdot$ is the learned offset for each parameter.

Distance-based Target Assignment Taking object orientation into consideration, we propose distance-based target assignment. The distance is defined as the distance of 8 corners between anchor and ground-truth boxes as

$$distance(\mathbf{A}, \mathbf{G}) = \frac{1}{8} \sum_{i=1}^{8} \sqrt{(x_{\mathbf{A}_i} - x_{\mathbf{G}_i})^2 + (z_{\mathbf{A}_i} - z_{\mathbf{G}_i})^2)}$$

In order to balance the ratio of positive and negative samples, we let the anchors with top N nearest distance to ground-truth as positive samples, where $N = \gamma \times k$ and k is the number of voxels inside ground-truth box on bird's eye view. γ adjusts the number of positive samples. Our centerness is defined as the exponent of the negative normalized distance of eight corners as

$$centerness(\mathbf{A}, \mathbf{G}) = e^{-\text{norm}(distance(\mathbf{A}, \mathbf{G}))}, \tag{3}$$

where norm denotes min-max normalization.

3.3. Multi-task Training

Our network with stereo matching network and 3D object detector is trained in an end-to-end fashion. We train the overall 3D object detector with a multi-task loss as

$$Loss = \mathcal{L}_{depth} + \mathcal{L}_{cls} + \mathcal{L}_{reg} + \mathcal{L}_{centerness}. \tag{4}$$

For the loss of depth regression, we adopt smooth L_1 loss [22] in this branch as

$$\mathcal{L}_{depth} = \frac{1}{N_D} \sum_{i=1}^{N_D} smooth_{L_1} \left(d_i - \hat{d}_i \right), \tag{5}$$

where N_D is the number of pixels with ground-truth depth (obtained from the sparse LiDAR sensor).

For the loss of classification, focal loss [31] is adopted in our network to deal with the class imbalance problem in 3D world as

$$\mathcal{L}_{cls} = \frac{1}{N_{pos}} \sum_{(x,z) \in \mathcal{F}} Focal\_Loss(p_{\mathbf{A}_{(x,z)}}, p_{\mathbf{G}_{(x,z)}}), \tag{6}$$

where N_{pos} denotes the number of positive samples. Binary cross-entropy (BCE) loss is used for centerness.

For the loss of 3D bounding box regression, smooth L_1 Loss is used for the regression of bounding boxes as

$$\mathcal{L}_{reg} = \frac{1}{N_{pos}} \sum_{(x,z) \in F_{pos}} \text{centerness}(\mathbf{A}, \mathbf{G}) \times \text{smooth}_{L_1}(l1\_\text{distance}(\mathbf{A}, \mathbf{G})) \tag{7}$$

where F_{pos} denotes all positive samples on bird's eye view.

We try two different regression targets with and without jointly learning all parameters.

- *Separably optimizing box parameters.* The regression loss is directly applied to the offset of $(x, y, z, h, w, l, \theta)$.
- *Jointly optimizing box corners.* For jointly optimizing box parameters, the loss is made on the average *L1* distance of eight box corners between predicted boxes from 3D anchors and ground-truth boxes following that of [35].

In our experiments, we use the second regression target for *Car* and the first regression target for *Pedestrian* and *Cyclist*. Because it is hard for even human to accurately predict or annotate the orientation of objects like *Pedestrian* from an image, other parameter estimation under joint optimization can be affected.

4. Experiments

Datasets Our approach is evaluated on the popular KITTI 3D object detection dataset [14], which is union of $7,481$ stereo image-pairs and point clouds for training and $7,518$ for testing. The ground-truth depth maps are generated from point clouds following [47, 58]. The training data has annotation for *Car*, *Pedestrian* and *Cyclist*. The KITTI leaderboard limits the access to submission to the server for evaluating test set. Thus, following the protocol in [9, 27, 47], the training data is divided into a training set (3,712 images) and a validation set (3,769 images). All ablation studies are conducted on the split. For the submission of our approach, our model is trained from scratch on the 7K training data only.

Evaluation Metric KITTI has three levels of difficulty setting of easy, moderate (main index) and hard, according to the occlusion/truncation and the size of an object in the 2D image. All methods are evaluated for three levels of difficulty under different IoU criteria per class , *i.e.*, IoU ≥ 0.7 for *Car* and IoU ≥ 0.5 for *Pedestrian* and *Cyclist* for 2D, bird's eye view and 3D detection.

Following most image-based 3D object detection setting [47, 27, 39, 38, 2, 42], the ablation experiments are conducted on *Car*. We also report the results of *Pedestrian* and *Cyclist* for reference in the supplementary file. KITTI benchmark recently changes evaluation where AP calculation uses 40 recall positions instead of the 11 recall positions proposed in the original Pascal VOC benchmark. Thus, we show the main test results following the official KITTI leaderboard. We generate the validation results using the original evaluation code for fair comparison with other approaches in ablation studies.

4.1. Implementation

Training Details By default, models are trained on 4 *NVIDIA Tesla V100* (32G) GPUs with batch-size 4 – that is, each GPU holds one pair of stereo images of size 384×1248. We apply *ADAM* [23] optimizer with initial learning rate 0.001. We train our network for 50 epochs and the learning rate is decreased by 10 at 50-th epoch. The overall training time is about 17 hours. The data augmentation used is horizontal flipping only.

Following other approaches [58, 47, 60, 52, 41, 10], another network is trained for *Pedestrian* and *Cyclist*, we first pre-train the network with all training images for the stereo network and then apply fine-tune with 3D box annotation for both branches because only about $1/3$ images have annotations of these two objects.

Implementation Details For constructing plane-sweep volume, the image feature map is shrunk to 32D and downsampled by 4 for both left and right images. Then by reprojection and concatenation, we construct the volume of shape $(W_I/4, H_I/4, D_I/4, 64)$, where the image size is $(W_I = 1248, H_I = 384)$ and the number of depth is $D_I = 192$. It is followed by one 3D hourglass module [4, 34] and extra 3D convolutions to get the matching cost volume of shape $(W_I/4, H_I/4, D_I/4, 1)$. Then interpolation is used to upsample this volume to fit the image size.

To construct 3D geometric volume, We discretize the region in range $[-30.4, 30.4] \times [-1, 3] \times [2, 40.4]$ (meters) to a 3D voxel occupancy grid of size $(W_V = 300, H_V = 20, D_V = 192)$ along the right (X), down (Y) and front (Z) directions in camera's view. 3D geometric volume is formed by warping the last feature map of PSV. Each voxel is a cube of size $(0.2, 0.2, 0.2)$ (meter).

Other implementation details and the network architecture are included in the supplementary file.

4.2. Main Results

We give comparison with state-of-the-art 3D detectors in Tables 1 and 2. Without bells and whistles, our approach outperforms all other image-based methods on 3D and BEV object detection. We note that Pseudo-LiDARs [47, 58] is with pre-trained PSMNet [4] on a large-scale synthetic scene flow dataset [33] (with 30,000+ pairs of stereo images and dense disparity maps) for stereo matching. Stereo R-CNN [27] uses ImageNet pre-trained ResNet-101 as backbone and has input images of resolution 600×2000.

Differently, our model is trained from scratch only on these 7K training data with input of resolution 384×1248. Also, Pseudo-LiDARs [47, 58] approaches apply two independent networks including several LiDAR-based detectors, while ours is just one unified network.

DSGN without explicitly learning 2D boxes surpasses those applying strong 2D detectors based on ResNet-101 [27] or DenseNet-121 [2]. It naturally achieves duplicate removal by non-maximum suppression (NMS) in 3D space, which coincides with the common belief that there is no collision between regular objects.

More intriguingly, as shown in Table 1, DSGN even achieves comparable performance on BEV detection and better performance on 3D detection on KITTI easy regime with MV3D [9] (with LiDAR input only) – a classic LiDAR-based 3D object detector, *for the first time*. This result demonstrates a promising future application at least in the scenario of low-speed autonomous driving.

The above comparison manifests the effectiveness of 3D geometric volume, which serves as a link between 2D im-

Modality	Method	**3D Detection AP (%)**			BEV Detection AP (%)			2D Detection AP (%)		
		Easy	**Moderate**	Hard	Easy	**Moderate**	Hard	Easy	**Moderate**	Hard
LiDAR	MV3D (LiDAR) [9]	68.35	54.54	49.16	86.49	78.98	72.23	–	–	–
Mono	OFT-Net [40]	1.61	1.32	1.00	7.16	5.69	4.61	–	–	–
	MonoGRNet [38]	9.61	5.74	4.25	18.19	11.17	8.73	88.65	77.94	63.31
	M3D-RPN [2]	14.76	9.71	7.42	21.02	13.67	10.23	89.04	85.08	69.26
	AM3D [32]	16.50	10.74	9.52	25.03	17.32	14.91	92.55	**88.71**	77.78
Stereo	3DOP [7]	–	–	–	–	–	–	93.04	88.64	**79.10**
	Stereo R-CNN* [27]	47.58	30.23	23.72	61.92	41.31	33.42	93.98	85.98	71.25
	PL: AVOD* [47]	54.53	34.05	28.25	67.30	45.00	38.40	85.40	67.79	58.50
	PL++: P-RCNN* [58]	61.11	42.43	36.99	78.31	58.01	51.25	94.46	82.90	75.45
	DSGN (Ours)	**73.50**	**52.18**	**45.14**	**82.90**	**65.05**	**56.60**	**95.53**	86.43	78.75

Table 1. Comparison of main results on KITTI test set (official KITTI leaderboard). The results are evaluated using new evaluation metric on the KITTI leaderboard. Several methods undergoing old evaluation are not available on the leaderboard. PL/PL++* uses extra Scene Flow dataset to pre-train the stereo matching network and Stereo R-CNN* uses ImageNet pre-trained model.

Modality	Method	**3D Detection AP (%)**			BEV Detection AP (%)			2D Detection AP (%)		
		Easy	**Moderate**	Hard	Easy	**Moderate**	Hard	Easy	**Moderate**	Hard
LiDAR	MV3D (LiDAR) [9]	71.29	56.60	55.30	86.18	77.32	76.33	88.41	87.76	79.90
Mono	OFT-Net [40]	4.07	3.27	3.29	11.06	8.79	8.91	–	–	–
	MonoGRNet [38]	13.88	10.19	7.62	43.75	28.39	23.87	–	78.14	–
	M3D-RPN [2]	20.27	17.06	15.21	25.94	21.18	17.90	90.24	83.67	67.69
	AM3D [32]	32.23	21.09	17.26	43.75	28.39	23.87	–	–	–
Stereo	MLF [50]	–	9.80	–	–	19.54	–	–	–	–
	3DOP [7]	6.55	5.07	4.10	12.63	9.49	7.59	–	–	–
	Triangulation [39]	18.15	14.26	13.72	29.22	21.88	18.83	–	–	–
	Stereo R-CNN* [27]	54.1	36.7	31.1	68.5	48.3	41.5	**98.73**	**88.48**	71.26
	PL: F-PointNet* [47]	59.4	39.8	33.5	72.8	51.8	33.5	–	–	–
	PL: AVOD* [47]	61.9	45.3	39.0	74.9	56.8	49.0	–	–	–
	PL++: AVOD* [58]	63.2	46.8	39.8	77.0	63.7	56.0	–	–	–
	PL++: PIXOR* [58]	–	–	–	79.7	61.1	54.5	–	–	–
	PL++: P-RCNN* [58]	67.9	50.1	45.3	82.0	**64.0**	57.3	–	–	–
	DSGN (Ours)	**72.31**	**54.27**	**47.71**	**83.24**	63.91	**57.83**	89.25	83.59	**78.45**

Table 2. Comparison of main results on KITTI *val* set. As described in Section 4, we use original KITTI evaluation metric here. PL/PL++* uses extra Scene Flow dataset to pre-train the stereo matching network and Stereo R-CNN* uses ImageNet pre-trained model.

ages and 3D space by combining the depth information and semantic feature.

Inference Time On a NVIDIA Tesla V100 GPU, the inference time of DSGN for one image pair is 0.682s on average, where 2D feature extraction for left and right images takes 0.113s, constructing the plane-sweep volume and 3D geometric volume takes 0.285s, and 3D object detection on 3D geometric volume takes 0.284s. The computation bottleneck of DSGN lies on 3D convolution layers.

4.3. Ablation Study

4.3.1 Ablation study of 3D Volume Construction

One of the main obstacles to construct an effective 3D geometric representation is the appropriate way of learning 3D geometry. We therefore investigate the effect of following three key components to construct a 3D volume.

Input Data Monocular-based 3D volume only has the potential to learn the correspondence between 2D and 3D feature, while stereo-based 3D volume can learn extra 2D feature correspondence for pixel-correspondence constraint.

Constructing 3D Volume One straightforward solution to construct 3D volume is by directly projecting the image feature to 3D voxel grid [21, 40] (denoted as IMG→3DV). Another solution in Figure 3 transforms plane-sweep volume or disparity-based cost volume to 3D volume, which provides a natural way to impose pixel-correspondence constraint along the projection ray in camera frustum (denoted as IMG→(PS)CV→3DV).

Supervising Depth Supervised with or without the point cloud data, the network learns the depth explicitly or implicitly. One way is to supervise the voxel occupancy of 3D grid by ground-truth point cloud using *binary cross-entropy loss*. The second is to supervise depth on the plane-sweep cost volume as explained in Section 3.3.

For fair comparison, the models IMG→3DV and IMG→(PS)CV→3DV have the same parameters by adding the same 3D hourglass module for the model IMG→3DV. In addition, several important facts can be revealed from

Input	Transformation	Supervision	AP_{3D} / AP_{BEV} / AP_{2D}
Mono	IMG→3DV	×	6.22 / 11.98 / 58.23
		3DV	13.66 / 19.92 / 65.89
Stereo	IMG→3DV	×	11.03 / 15.17 / 57.30
		3DV	42.57 / 54.55 / 81.86
	IMG→CV→3DV	CV	45.89 / 58.40 / 81.71
	IMG→PSCV→3DV	×	38.48 / 52.85 / 77.83
		PSCV	**54.27 / 63.91 / 83.59**

Table 3. Ablation study of depth encoded approaches. "PSCV" and "3DV" with "Supervision" header represent that the constraint is imposed in (plane-sweep) cost volume and 3D volume, respectively. The results are evaluated in moderate level.

Networks	Targets	Depth Error (meters)		AP_{3D} / AP_{BEV} / AP_{2D}
		Mean	Median	
PSMNet-PSV*	Depth	0.5337	0.1093	—-
DSGN		**0.5279**	**0.1055**	—-
PSMNet-PSV*	Both	0.5606	0.1157	46.41 / 57.57 / 80.67
DSGN		0.5586	0.1104	**54.27 / 63.91 / 83.59**

Table 4. Influence on depth estimation, evaluated on KITTI val images. PSMNet-PSV* is a variant of PSMNet [4], which uses one 3D hourglass module instead of three of them for refinement considering limited memory space and takes the plane-sweep approach to construct cost volume.

Table 3 and are explained in the following.

Supervision of point cloud is important. The approaches under the supervision of the LiDAR point cloud consistently perform better than those without supervision, which demonstrates the importance of 3D geometry for image-based approaches.

Stereo-based approaches work much better than monocular ones under supervision. The discrepancy between stereo and monocular approaches indicates that direct learning of 3D geometry from semantic cues is a quite difficult problem. In contrast, image-based approaches without supervision make these two lines yield similar performance, which indicates that supervision only by 3D bounding boxes is insufficient for learning of 3D geometry.

Plane-sweep volume is a more suitable representation for 3D structure. Plane-sweep cost volume (54.27 AP) performs better than disparity-based cost volume (45.89 AP). It shows that balanced feature mapping is important during the transformation to 3D volume.

Plane-sweep volume, as an intermediate encoder, more effectively contains depth information. The inconsistency between IMG→PSCV→3DV and IMG→3DV manifests that plane-sweep volume as the intermediate representation can effectively help learning of depth information. The observation explains that the *soft arg-min* operation encourages the model to pick a single depth plane per pixel along the projection ray, which shares the same spirit as the assumption that only one depth-value is true for each pixel. Another reason can be that PSCV and 3DV have different matching densities – PSCV intermediately imposes the dense pixel correspondence over all image pixels. In contrast, only the left-right pixel pairs through the voxel centers are matched on 3DV.

From above comparison of volume construction, we observe that the three key facts affect the performance of computation pipelines. The understanding and recognition of how to construct a suitable 3D volume is still at the very early stage. More study is expected to reach comprehensive understanding of the volume construction from the multi-view images.

4.3.2 Influence on Stereo Matching

We conduct experiments for investigating the influence of depth estimation, which is evaluated on KITTI *val* set following [47]. The average and median value of absolute depth estimation errors within the pre-defined range of $[z_{\text{min}}, z_{\text{max}}]$ is shown in Table 4. A natural baseline for our approach is PSMNet-PSV* modified from PSMNet [4] whose 2D feature extractor takes 0.041s while ours takes 0.113s.

Trained with depth estimation branch only, DSGN performs slightly better than PSMNet-PSV* with the same training pipeline in depth estimation. For joint training of both tasks, both approaches suffer from larger and similar depth error (0.5586 meter for DSGN vs. 0.5606 meter for PSMNet-PSV*). Differently, DSGN outperforms the alternatives by 7.86 AP on 3D object detection and 6.34 AP on BEV detection. The comparison indicates that our 2D network extracts better high-level semantic features for object detection.

5. Conclusion

We have presented a new 3D object detector on binocular images. It shows that an end-to-end stereo-based 3D object detection is feasible and effective. Our unified network encodes 3D geometry via transforming the plane-sweep volume to a 3D geometric one. Thus, it is able to learn high-quality geometric structure features for 3D objects on the 3D volume. The joint training lets the network learn both *pixel-* and *high-level* features for the important tasks of stereo correspondence and 3D object detection.

Without bells and whistles, our one-stage approach outperforms other image-based approaches and even achieves comparable performance with a few LiDAR-based approaches on 3D object detection. The ablation study investigates several key components for training 3D volume in Table 3. Although the improvement is clear and explained, our understanding of how the 3D volume transformation works will be further explored in our future work.

A. Appendix

A.1. More Experiments

Correlation between stereo (depth) and 3D object detection accuracy. Following KITTI stereo metric, we consider a pixel as being correctly estimated if its depth error is less than an *outlier_thresh*. The percentage of non-outlier pixels inside the object box is deemed as depth estimation precision.

With several *outlier_threshes* (0.1, 0.3, 0.5, 1.0, 2.0 meters), scatter plots of stereo and detection precision are drawn. We observed that when *outlier_thresh* is 0.3 meter, the strongest linear correlation is yielded.

Outlier_thresh	> 2m	> 1m	> 0.5m	> 0.3m	> 0.1m
PCC	0.249	0.353	0.438	0.450	0.417

Table 5. Pearson's correlation coefficients (PCC) for a set of *outlier_threshes* between depth estimation precision and detection accuracy. *Outlier_thresh*$= 0.3m$ yields the strongest linear correlation.

Figure 4 shows the scatter plots with the *outlier_thresh* 0.3m and 0.1m. Quite a few predictions get over 0.7 detected precision within a certain range of depth estimation error. This reveals that the end-to-end network gives rise to its ability to detect 3D objects even with a larger depth estimation error. The following 3D detector enables compensation for the stereo depth estimation error by 3D location regression with back-propagation.

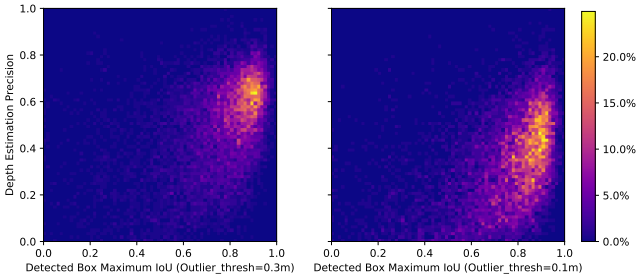

Figure 4. BEV Detection precision versus depth precision for all predicted boxes for *Car* on KITTI *val* set. Only TPs with IoU$>$ 0.01 and score > 0.1 are shown.

Relationship between distance and detection accuracy. Figure 5 illustrates, as distance increases, all detection accuracy indicators have a shrinking tendency. The average accuracy maintains 80%+ within 25 meters. In all indicators, 3D AP decreases the fastest, followed by BEV AP and last 2D AP. This observation suggests that the 3D detection accuracy is determined by the BEV location precision beyond 20 meters.

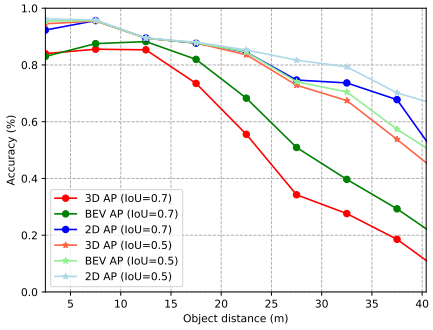

Figure 5. Detection accuracy versus distance. We separate the range [0, 40] (meters) into 8 intervals, each with 5 meters. All evaluations are conducted within each interval.

Influence of different features for 3D geometry. We discuss the efficiency of different geometric representations for volumetric structure. Most depth prediction approaches [13, 22, 4, 15] apply the strategy of "depth classification" (such as cost volume) instead of "depth regression". Thus, we have several choices for encoding the depth information of cost volume into a 3D volume. The intuitive one is to use a 3D voxel occupancy (denoted by "Occupancy"). An advanced version is by keeping the probability of voxel occupancy (denoted by "Probability"). They both have explicit meaning for 3D geometry and can be easily visualized. Another one is by using the last feature map for cost volume as geometric embedding for 3D volume (denoted by "Last Features").

Voxel Features	AP_{3D} / AP_{BEV} / AP_{2D}
Occupancy	37.86 / 50.64 / 70.79
Probability	43.24 / 54.87 / 74.93
Last Features	**54.27 / 63.91 / 83.59**

Table 6. Ablation study on 3D geometric representation. "Occupancy" indicates only using binary feature for 3D volume. It is 1 for voxel of minimum cost along the projection ray and is 0 otherwise. "Probability" denotes keeping the probability of voxel occupancy instead of quantizing it to 0 or 1. "Last Features" represents transforming the last features of cost volume to 3D volume.

Table 6 reveals the performance gap between "Occupancy" / "Probability". "Last Features" indicates the latent feature embedding (64D) that enables the network to extract more 3D latent geometric information and even semantic cues than the explicit voxel occupancy. It aids learning of 3D structure.

Technical details We explore several technical details used in DSGN and discuss their importance in the pipeline in Table 7. Joint optimization of bounding box regression improves accuracy ($+4.80$ AP) than the separable optimiza-

JOINT	IMG	ATT	Depth	HG	Flip	AP_{3D}/AP_{BEV}/AP_{2D}
						40.71 / 53.71 / 76.11
✓						45.51 / 56.65 / 78.31
✓	✓					44.79 / 56.24 / 81.58
✓	✓	✓				46.52 / 57.44 / 82.41
✓			✓			45.79 / 56.89 / 78.49
✓	✓	✓		✓		51.73 / 61.74 / 83.6
✓	✓	✓		✓	✓	54.27 / 63.91 / 83.59

Table 7. Ablation study evaluated in moderate level. "JOINT" indicates using joint optimization instead of separable optimization for bounding boxes regression. "IMG" denotes concatenating the mapped left image feature to 3DGV for retrieving more 2D semantics. "ATT (Attention)" represents concatenating the mapped image feature weighted by the corresponding depth probability. "Depth" indicates warping the final matching cost volume to 3DGV. "HG (Hourglass)" represents applying an hourglass module in 3DGV. "Flip" means using random horizontal flipping augmentation.

tion. The intermediate 3D volume representation enables the network to naturally retrieve image feature for more 2D semantic cues. However, 3DGV cannot directly benefit from the concatenation of mapped 32D image features and warped predicted cost volume. Instead, the image feature weighted by the depth probability achieves $+1.01$ AP gain. Further, Involving more computation by an extra hourglass module on 3D object detector and flip augmentation, DSGN finally achieves 54.27 AP on 3D object detection.

***Pedestrian* and *Cyclist* detection.** The main challenges for detecting *Pedestrian* and *Cyclist* are the limited data (about only 1/3 of images are annotated) and the difficulty to estimate their poses in an image even for human. As a result, most image-based approaches yield poor performance or are not validated on *Pedestrian* and *Cyclist*. Since the evaluation metric is changed on the official KITTI leaderboard, We only report the available results from original papers and the KITTI leaderboard.

Experimental results in Table 8 shows that our approach achieves better results on *Pedestrian* but worse ones on *Cyclist* compared with PL: F-PointNet. We note that PL: F-PointNet used Scene Flow dataset [33] to pre-train the stereo matching network. Besides, PL: F-PointNet achieves the best result on *Pedestrian* and the model PL: AVOD works best on *Car* and *Cyclist*. Table 9 shows the submitted results on the official KITTI leaderboard.

A.2. More Implementation Details

Network Architecture. We show the full network architecture in Table 10, including the networks for 2D feature extraction, constructing plane-sweep volume and 3D geometric volume, stereo matching and 3D object detection.

Implementation Details of 3D Object Detector. Given the feature map $\mathcal{F}$ on bird's eye view, we put four anchors of different orientation angles $(0, \pi/2, \pi, 3\pi/2)$ on all locations of $\mathcal{F}$. The box sizes of pre-defined anchors used for respectively *Car*, *Pedestrian*, *Cyclist* are $(h_{\mathbf{A}} = 1.56, w_{\mathbf{A}} = 1.6, l_{\mathbf{A}} = 3.9)$, $(h_{\mathbf{A}} = 1.73, w_{\mathbf{A}} = 0.6, l_{\mathbf{A}} = 0.8)$, and $(h_{\mathbf{A}} = 1.73, w_{\mathbf{A}} = 0.6, l_{\mathbf{A}} = 1.76)$.

The horizontal coordinate $(x_{\mathbf{A}}, z_{\mathbf{A}})$ of each anchor lies on the center of each grid in bird's eye view and its center along the vertical direction locates on $y_{\mathbf{A}} = 0.825$ for *Car* and $y_{\mathbf{A}} = 0.74$ for *Pedestrian* and *Cyclist*. We set $\gamma = 1$ for *Car* and $\gamma = 5$ for *Pedestrian* and *Cyclist* for balancing the positive and negative samples. The classification head of 3D object detector is initialized following RetinaNet [31]. NMS with IoU threshold 0.6 is applied to filter out the predicted boxes on bird's eye view.

Implementation Details of Differentiable Warping from PSV to 3DGV. Let $\mathcal{U} \in \mathbb{R}^{H_I \times W_I \times D_I \times C}$ be the last feature map of PSV, where C is the channel size of features. We first construct a pre-defined 3D volume $\in \mathbb{R}^{H_V \times W_V \times D_V \times 3}$ to store the center coordinate (x, y, z) of each voxel in 3D space (Section 4.1). Then we get the projected pixel coordinate (u, v) by multiplying the projection matrix. z is directly concatenated to pixel coordinate to get (u, v, z) in *camera frustum space*.

As a result, we get a coordinate volume $\in \mathbb{R}^{H_V \times W_V \times D_V \times 3}$, which stores the mapped coordinates in *camera frustum space*. By *trilinear interpolation*, we fetch the corresponding feature of $\mathcal{U}$ at the projected coordinates to construct the 3D volume $\mathcal{V} \in \mathbb{R}^{H_V \times W_V \times D_V \times C}$, *i.e.*, 3D geometric volume. We ignore the projected coordinates outside the image by setting these voxel features to 0. In backward operations, the gradient is passed and computed using the same coordinate volume.

A.3. Future Work

More further studies on stereo-based 3D object detection are recommended here.

The Gap with state-of-the-art LiDAR-based approaches. Although our approach achieves comparable performance with some LiDAR-based approaches on 3D object detection, there remains a large gap with state-of-the-art LiDAR-based approaches [55, 41, 10, 52]. Besides, an obvious problem is the accuracy gap on bird's eye view (BEV) detection. As shown in the table of main results, there is almost 12 AP gap on the moderate and hard level in BEV detection.

One possible solution is high-resolution stereo matching [53], which can help obtain more accurate depth information to increase the robustness for highly occluded, truncated and far objects.

Modality	Method	**3D Detection AP (%)**			BEV Detection AP (%)			2D Detection AP (%)		
		Easy	**Moderate**	Hard	Easy	**Moderate**	Hard	Easy	**Moderate**	Hard
Pedestrian										
Mono	M3D-RPN [2]	–	11.09	–	–	11.53	–	–	–	–
Stereo	PL: F-PointNet* [47]	33.8	27.4	24.0	41.3	34.9	30.1	–	–	–
	DSGN (ours)	**40.16**	**33.85**	**29.43**	**47.92**	**41.15**	**36.08**	**59.06**	**54.00**	**49.65**
Cyclist										
Mono	M3D-RPN [2]	–	2.81	–	–	3.61	–	–	–	–
Stereo	PL: F-PointNet* [47]	**41.3**	**25.2**	**24.9**	**47.6**	**29.9**	**27.0**	–	–	–
	DSGN (ours)	37.87	24.27	23.15	41.86	25.98	24.87	49.38	33.97	32.40

Table 8. Comparison of results for *Pedestrian* and *Cyclist* on KITTI *val* set. PL: F-PointNet* uses extra Scene Flow dataset to pretrain the stereo matching network.

Modality	Method	**3D Detection AP (%)**			BEV Detection AP (%)			2D Detection AP (%)		
		Easy	**Moderate**	Hard	Easy	**Moderate**	Hard	Easy	**Moderate**	Hard
Pedestrian										
Mono	M3D-RPN [2]	4.92	3.48	2.94	5.56	4.05	3.29	56.64	41.46	37.31
Stereo	RT3DStereo [24]	3.28	2.45	2.35	4.72	3.65	3.00	41.12	29.30	25.25
	DSGN (ours)	**20.53**	**15.55**	**14.15**	**26.61**	**20.75**	**18.86**	**49.28**	**39.93**	**38.13**
Cyclist										
Mono	M3D-RPN [2]	0.94	0.65	0.47	1.25	0.81	0.78	61.54	41.54	35.23
Stereo	RT3DStereo [24]	5.29	3.37	2.57	7.03	4.10	3.88	19.58	12.96	11.47
	DSGN (ours)	27.76	18.17	16.21	31.23	21.04	18.93	49.10	35.15	31.41

Table 9. Comparison of results for *Pedestrian* and *Cyclist* on KITTI test set (official KITTI Leaderboard).

3D Volume Construction. Table 3 shows basic comparison of volume construction in DSGN. We expect a more in-depth analysis of the volume construction from multi-view or binocular images, which serves as an essential component design for 3D object understanding. Besides, the effectiveness of 3D volume construction methods still requires more investigation since it needs to balance and provide both depth information and semantic information.

Computation Bottleneck. The computation bottleneck of DSGN locates on the computation of 3D convolutions for computing cost volume. Recent stereo matching work [57, 48] focused on accelerating the computation of cost volume. Another significant aspect of constructing cost volume is that current cost volume [22, 4] is designed for regressing disparity but not depth. Further research might explore more efficient feature encoding for the plane-sweep cost volume.

Network Architecture Design. There is a trade-off between stereo matching network and 3D detection network for balancing the feature extraction of pixel- and high-level features, which can be conducted by recent popular Network Architecture Search (NAS).

Application on Low-speed Scenario. Our approach shows comparable performance with the LiDAR-based approach on 3D and BEV detection in the close range in the KITTI easy set. Most importantly, it is affordable even with one strong GPU Tesla V100 ($11,458 (USD)) compared with the price of a 64-beam LiDAR $75,000 [58]. It is a promising application of image-based autonomous driving system for low-speed scenarios.

A.4. Qualitative Results

We provide a video demo [1] for visualization of our approach, which shows both the detected 3D boxes on front view and bird's eye view. The ground-truth LiDAR point cloud is shown on bird's eye view. The detection results are obtained by DSGN trained on KITTI training split only. The unit of the depth map is meter.

Some noise observed in the predicted depth map is mainly caused by the implementation details. (1) Noise in the near and far part: 3D volumes are constructed in [2, 40.4] (meters). (2) Noise and large white zone in the higher region (>3m): The stereo branch is trained with a sparse GT depth map (64 lines around [-1,3 (meters) along the gravitational z-axis, captured by a 64-ray LiDAR).

References

[1] Henrik Aanæs, Rasmus Ramsbøl Jensen, George Vogiatzis, Engin Tola, and Anders Bjorholm Dahl. Large-scale data for multiple-view stereopsis. volume 120, pages 153–168. Springer, 2016.

[2] Garrick Brazil and Xiaoming Liu. M3d-rpn: Monocular 3d region proposal network for object detection. 2019.

[3] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model

[1] https://www.youtube.com/watch?v=u6mQW89wBbo

Layers	Kernel Size	Chl	Output Size
Image Feature Extractor			
Input Image		3	$H_I \times W_I$
conv1_x	$(3 \times 3) \times 3$	64	$H_I/2 \times W_I/2$
conv2_x	$(3 \times 3) \times 3$	32	$H_I/2 \times W_I/2$
conv3_x	$(3 \times 3) \times 6$	64	$H_I/4 \times W_I/4$
conv4_x	$(3 \times 3) \times 12$	128	$H_I/4 \times W_I/4$
conv5_x	$(3 \times 3) \times 4$ (dila=2)	192	$H_I/4 \times W_I/4$
conv5_4 → SPP Module			
branch_1	64×64 *avgpool* 3×3 bilinear interpolation	32	$H_I/4 \times W_I/4$
branch_2	32×32 *avgpool* 3×3 bilinear interpolation	32	$H_I/4 \times W_I/4$
branch_3	16×16 *avgpool* 3×3 bilinear interpolation	32	$H_I/4 \times W_I/4$
branch_4	8×8 *avgpool* 3×3 bilinear interpolation	32	$H_I/4 \times W_I/4$
Fusion of shadow and deep layers			
concat [conv3_6, conv4_12, conv5_4, branch1~4]		512	$H_I/4 \times W_I/4$
fusion_1	3×3	256	$H_I/4 \times W_I/4$
fusion_2	3×3	32	$H_I/4 \times W_I/4$
(fusion_2 (left), fusion_2 (right)) → Constructing Plane-Sweep Volume			
Plane-Sweep Volume		64	$H_I/4 \times W_I/4 \times D_I/4$
PS_conv1_x	$(3 \times 3 \times 3) \times 2$	64	$H_I/4 \times W_I/4 \times D_I/4$
PS_conv2_x	$(3 \times 3 \times 3) \times 2$ *add* PS conv1_2	64	$H_I/4 \times W_I/4 \times D_I/4$
PS_stack_1x	$(3 \times 3 \times 3) \times 2$	128	$H_I/8 \times W_I/8 \times D_I/8$
PS_stack_2x	$(3 \times 3 \times 3) \times 2$	128	$H_I/16 \times W_I/16 \times D_I/16$
PS_stack_3	deconv $3 \times 3 \times 3$ *add* PS_stack_12	128	$H_I/8 \times W_I/8 \times D_I/8$
PS_stack_4	deconv $3 \times 3 \times 3$ *add* PS_conv2_2	64	$H_I/4 \times W_I/4 \times D_I/4$
PS_stack_4 → Stereo Matching			
Depth conv_1	$3 \times 3 \times 3$	64	$H_I/4 \times W_I/4 \times D_I/4$
Depth conv_2	$3 \times 3 \times 3$	1	$H_I/4 \times W_I/4 \times D_I/4$
Upsample	*trilinear interpolation*	1	$H_I \times W_I \times D_I$
***soft argmin* function**		1	$H_I \times W_I$
PS_stack_4 → 3D Geometric Volume			
3D Geometric Volume		64	$H_V \times W_V \times D_V$
3DG_conv	$(3 \times 3 \times 3)$	64	$H_V \times W_V \times D_V$
3DG_ stack_1x	$(3 \times 3 \times 3) \times 2$	128	$H_V/2 \times W_V/2 \times D_V/2$
3DG_ stack_2x	$(3 \times 3 \times 3) \times 2$	128	$H_V/4 \times W_V/4 \times D_V/4$
3DG_ stack_3	deconv $3 \times 3 \times 3$ *add* 3DG_stack_12	128	$H_V/2 \times W_V/2 \times D_V/2$
3DG_ stack_4	deconv $3 \times 3 \times 3$ *add* 3DG_conv	64	$H_V \times W_V \times D_V$
3DG_stack_4→3D Geometric Volume on BEV			
$4 \times 1 \times 1$ *avgpool* and reshape to bev		$64 \times H_V/4$	$W_V \times D_V$
3DGVbev_ conv_x	$(3 \times 3) \times 2$	128	$W_V \times D_V$
3DGVbev_conv_2 → Classification			
cls_conv_x	$(3 \times 3) \times 4$	128	$W_V \times D_V$
bbox_cls	3×3	4×3	$W_V \times D_V$
3DGVbev_conv_2 → Regression			
reg_conv_x	$(3 \times 3) \times 4$	128	$W_V \times D_V$
bbox_cls	3×3	10×3	$W_V \times D_V$
reg_conv_4 → Centerness			
bbox_centerness	3×3	4	$W_V \times D_V$

Table 10. Full network architecture of DSGN. The color of the table highlights different components.

Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[4] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *CVPR*, pages 5410–5418, 2018.

[5] Rui Chen, Songfang Han, Jing Xu, and Hao Su. Point-based multi-view stereo network. 2019.

[6] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3d object detection for autonomous driving. In *CVPR*, 2016.

[7] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Andrew G Berneshawi, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals for accurate object class detection. In *NIPS*, pages 424–432, 2015.

[8] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals using stereo imagery for accurate object class detection. volume 40, pages 1259–1272. IEEE, 2017.

[9] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *CVPR*, 2017.

[10] Yilun Chen, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Fast point r-cnn. In *ICCV*, pages 9775–9784, 2019.

[11] Robert T Collins. A space-sweep approach to true multi-image matching. In *CVPR*, pages 358–363. IEEE, 1996.

[12] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *CVPR*, pages 5515–5524, 2016.

[13] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *CVPR*, pages 2002–2011, 2018.

[14] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012.

[15] Xiaoyang Guo, Kai Yang, Wukui Yang, Xiaogang Wang, and Hongsheng Li. Group-wise correlation stereo network. In *CVPR*, 2019.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[17] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. Deepmvs: Learning multi-view stereopsis. In *CVPR*, pages 2821–2830, 2018.

[18] Sunghoon Im, Hae-Gon Jeon, Stephen Lin, and In So Kweon. Dpsnet: end-to-end deep plane sweep stereo. 2019.

[19] Saif Imran, Yunfei Long, Xiaoming Liu, and Daniel Morris. Depth coefficients for depth completion. In *CVPR*, June 2019.

[20] Mengqi Ji, Juergen Gall, Haitian Zheng, Yebin Liu, and Lu Fang. Surfacenet: An end-to-end 3d neural network for multiview stereopsis. In *ICCV*, pages 2307–2315, 2017.

[21] Abhishek Kar, Christian Häne, and Jitendra Malik. Learning a multi-view stereo machine. In *Advances in neural information processing systems*, pages 365–376, 2017.

[22] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry.

End-to-end learning of geometry and context for deep stereo regression. In *ICCV*, pages 66–75, 2017.

[23] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 2014.

[24] Hendrik Knigshof, Niels Ole Salscheider, and Christoph Stiller. Realtime 3D Object Detection for Automated Driving Using Stereo Vision and Semantic Information. In *Proc. IEEE Intl. Conf. Intelligent Transportation Systems*, Auckland, NZ, Oct 2019.

[25] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven Waslander. Joint 3d proposal generation and object detection from view aggregation. 2018.

[26] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. 2018.

[27] Peiliang Li, Xiaozhi Chen, and Shaojie Shen. Stereo r-cnn based 3d object detection for autonomous driving. In *CVPR*, pages 7644–7652, 2019.

[28] Ming Liang, Bin Yang, Yun Chen, Rui Hu, and Raquel Urtasun. Multi-task multi-sensor fusion for 3d object detection. In *CVPR*, pages 7345–7353, 2019.

[29] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *ECCV*, 2018.

[30] Zhengfa Liang, Yiliu Feng, Yulan Guo, Hengzhu Liu, Wei Chen, Linbo Qiao, Li Zhou, and Jianfeng Zhang. Learning for disparity estimation through feature constancy. In *CVPR*, pages 2811–2820, 2018.

[31] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. In *ICCV*, 2017.

[32] Xinzhu Ma, Zhihui Wang, Haojie Li, Wanli Ouyang, and Pengbo Zhang. Accurate monocular 3d object detection via color-embedded 3d reconstruction for autonomous driving. 2019.

[33] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, pages 4040–4048, 2016.

[34] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, pages 483–499. Springer, 2016.

[35] Charles R. Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J. Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *CVPR*, 2018.

[36] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. 2017.

[37] Charles R. Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NIPS*, 2017.

[38] Zengyi Qin, Jinglu Wang, and Yan Lu. Monogrnet: A geometric reasoning network for monocular 3d object localization. In *AAAI*, volume 33, pages 8851–8858, 2019.

[39] Zengyi Qin, Jinglu Wang, and Yan Lu. Triangulation learning network: from monocular to stereo 3d object detection. 2019.

[40] Thomas Roddick, Alex Kendall, and Roberto Cipolla. Orthographic feature transform for monocular 3d object detection. 2019.

[41] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. 2018.

[42] Andrea Simonelli, Samuel Rota Bulo, Lorenzo Porzi, Manuel Lopez-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection. In *ICCV*, 2019.

[43] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. 2014.

[44] Xiao Song, Xu Zhao, Hanwen Hu, and Liangji Fang. Edgestereo: A context integrated residual pyramid network for stereo matching. In *Asian Conference on Computer Vision*, 2018.

[45] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *CVPR*, pages 8934–8943, 2018.

[46] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. 2019.

[47] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *CVPR*, pages 8445–8453, 2019.

[48] Yan Wang, Zihang Lai, Gao Huang, Brian H Wang, Laurens van der Maaten, Mark Campbell, and Kilian Q Weinberger. Anytime stereo image depth estimation on mobile devices. In *ICRA*, pages 5893–5900. IEEE, 2019.

[49] Xinshuo Weng and Kris Kitani. Monocular 3d object detection with pseudo-lidar point cloud. 2019.

[50] Bin Xu and Zhenzhong Chen. Multi-level fusion based 3d object detection from monocular images. In *CVPR*, pages 2345–2353, 2018.

[51] Danfei Xu, Dragomir Anguelov, and Ashesh Jain. Pointfusion: Deep sensor fusion for 3d bounding box estimation. In *CVPR*, 2018.

[52] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. 2018.

[53] Gengshan Yang, Joshua Manela, Michael Happold, and Deva Ramanan. Hierarchical deep stereo matching on high-resolution images. In *CVPR*, June 2019.

[54] Guorun Yang, Hengshuang Zhao, Jianping Shi, Zhidong Deng, and Jiaya Jia. Segstereo: Exploiting semantic information for disparity estimation. In *ECCV*, pages 636–651, 2018.

[55] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Std: Sparse-to-dense 3d object detector for point cloud. In *ICCV*, 2019.

[56] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *ECCV*, pages 767–783, 2018.

[57] Zhichao Yin, Trevor Darrell, and Fisher Yu. Hierarchical discrete distribution decomposition for match density estimation. In *CVPR*, pages 6044–6053, 2019.

[58] Yurong You, Yan Wang, Wei-Lun Chao, Divyansh Garg, Geoff Pleiss, Bharath Hariharan, Mark Campbell, and Kilian Q

Weinberger. Pseudo-lidar++: Accurate depth for 3d object detection in autonomous driving. 2019.

[59] Feihu Zhang, Victor Prisacariu, Ruigang Yang, and Philip HS Torr. Ga-net: Guided aggregation net for end-to-end stereo matching. In *CVPR*, pages 185–194, 2019.

[60] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *CVPR*, 2018.