IEEE Robotics and Automation Letters (RAL) paper presented at the
2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Macau, China, November 4-8, 2019

Real-Time Dense Depth Estimation using Semantically-Guided LIDAR Data Propagation and Motion Stereo

Atsuki Hirata[1], Ryoichi Ishikawa[1], Menandro Roxas[1], Takeshi Oishi[1]

***Abstract*— In this paper, we present a method for estimating a dense depth map from a sparse LIDAR point cloud and an image sequence. Our proposed method relies on a directionally biased propagation of known depth to missing areas based on semantic segmentation. Additionally, we classify different object boundaries as either occluded or connected to limit the extent of the data propagation. At the regions with large missing point cloud data, we depend on estimated depth using motion stereo. We embed our method on a bounded interpolation strategy which also considers pixel distance, depth difference and color gradient. We then perform an optimization step based on tensor-based TGV-L2 denoising. Our results show that directional propagation and semantic boundary classification can improve the accuracy of interpolation along the edges for different types of objects. Moreover, our motion stereo scheme increases the reliability of extrapolated depth at the regions with large missing point cloud data. Finally, we show that our implementation strategy can achieve reliable results in real time.**

I. INTRODUCTION

Depth estimation is an integral part of many applications including robot navigation, autonomous driving and 3D modeling. Most of these applications require the depth map to be dense, accurate, and solved in real time. For example, in outdoor robot navigation and autonomous driving, detecting distant objects such as humans or traffic signs is difficult when using low resolution depth. In 3D modeling, dense depth maps are used for reconstructing an accurate and detailed 3D map of the environment.

Generally, depth maps can be estimated in two ways depending on the type of sensor used - passive (image-based) and active (3D sensors). Image-based methods (stereo, motion stereo, structure-from-motion) can generate a dense depth map from a pair of images, or image sequences. However, these methods are largely dependent on accurate image correspondences, pose estimation (for motion stereo) and sufficient baseline (for stereo, in general) and are insufficient when mapping distant objects.

In contrast to passive sensing, active sensors such as RGBD cameras and LIDAR can measure the depth in a more straightforward manner. However, for outdoor applications, RGBD cameras are insufficient because of their short range and inaccuracy due to sunlight. LIDAR sensors, on the other hand, have long range capabilities and are mostly unaffected by the ambient lighting. However, the inherent sparsity of the measurement points, which is limited by the number of simultaneous firing lasers, makes LIDARs undesirable when dense depth maps are required.

This work was partially supported by the social corporate program (Base Technologies for Future Robots) sponsored by NIDEC corporation. This work was also supported by JSPS KAKENHI Grant Number JP16747698, JP17923471.

[1]The authors are with the The University of Tokyo, Tokyo 113-8654, Japan hirata@cvl.iis.u-tokyo.ac.jp

Digital Object Identifier (DOI): see top of this page.

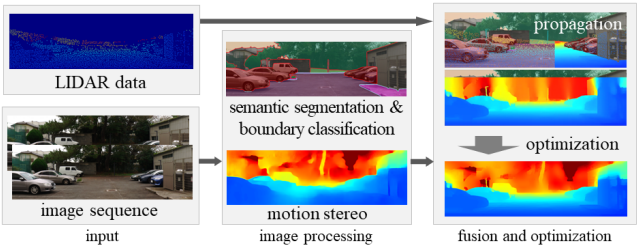

Fig. 1. Overview of our proposed dense depth estimation method using LIDAR, image sequence, semantic information, and motion stereo.

In this work, we utilize the advantages of both active and passive sensing by estimating a dense depth map using an image sequence and LIDAR point clouds as input. Like in methods such as [1] and [2], we also embed our method in a data upsampling framework with improvement on handling boundaries and extrapolation. We utilize both the information that can be extracted from an image sequence (color, gradient, visual object recognition, structure-from-motion) and LIDAR data (accurate depth values), and combine them into one framework that generates a dense depth map in real-time.

In our method, we use the LIDAR data as anchor points upon which the depth values of unknown pixels are based. We design this basis to be dependent on several properties such as geometry, color, motion and semantic segmentation. We also improve the handling of object boundaries by using our proposed boundary class labeling which adjusts the effects of the neighboring depth value based on the relationship between semantic classes.

We also propose a directional propagation scheme that relates the direction of the sequential data interpolation and extrapolation based on the semantic classes. Additionally, in parts where there are very few LIDAR points, we use the motion stereo depth to make the extrapolation more reliable. Finally, we perform a global optimization scheme to further smooth the resulting depth and refine the object boundaries using the visual edge information.

II. RELATED WORK

Several work have been presented that address the sparsity of LIDAR data. In [3], an interpolation method based on partial differential equations applied an energy minimization on sparse point clouds to achieve a smooth upsampled depth

map. To improve on boundary conditions, bilateral filter-based approach [4] was proposed which can estimate depth while preserving edges. In [5], a multilateral filter was used in addition to semantic segmentation of point clouds that further improves handling of object boundaries. Recently, deep neural network based interpolation methods have been proposed such as in [6]. In general, point-cloud-only methods are superior to methods that require calibration and synchronization such as RGBD image pairs. Nevertheless, these methods conduct interpolation with smoothness assumption and lacks strict boundary handling.

Since a camera image often has higher resolution than a LIDAR depth map, image-guided point cloud interpolation have been also explored. High resolution images are used to refine rough boundary information from low resolution depth maps. This interpolation is usually performed using various approach such as filtering [7], [8], [9], geodesic distance [10], anisotropic total generalized variation [11], autoregressive model [12], and semantic information [2]. Some methods also take temporal information into consideration [13], [14] while others use neural network for further depth map enhancement [15], [16].

Aside from having a higher resolution, a camera image can also cover a wider area and can be used as a guide for extrapolation or inpainting. In [17], a color image was used to extract structural information and used for depth map inpainting. On the other hand, edge information and semantic segmentation was used to extrapolate the LIDAR data points to missing regions in [2], which results in reliable depth along object boundaries. However, the use of semantic segmentation results in over-dominant extrapolation in areas with no LIDAR points clouds.

III. DENSE DEPTH MAP ESTIMATION

Our method requires a calibrated sparse depth map and an image sequence. The sparse depth map can come from different sources such as LIDAR point clouds. In this paper, we assume that a pair of images from a single-camera system is given in real-time, but our method is easily extendable on multiple images and/or multiple camera system (binocular or multi-view stereo). Our goal is to estimate a complete and dense depth map by interpolating the known depth map in areas with a little gap between known points, and extrapolate at the areas where depth is completely unknown. Using the information that can be extracted from the image pair, such as color gradient, semantic classes, and motion stereo, our proposed scheme can propagate the known depth to missing areas. After propagation, we perform a global optimization step to further improve the appearance of the resulting depth map. We show the overview of our method in Figure 1.

A. Propagation

Our propagation method is dependent on a geodesic distance-based data interpolation scheme, which solves the depth of an unknown pixel based on the nearby known values. Given an image $I : \Omega \rightarrow \mathbb{R}^+$ of the image sequence S, with corresponding sparse depth map D_p, our aim is to find $D = D_p \cup D_e$ where $D_p \cap D_e \equiv \emptyset$ and D_e is the combined interpolated and extrapolated depth. In order to solve for D_e, we define the depth $d_{\mathbf{x}} \in D_e$ for every pixel $\mathbf{x}$ in the image domain $\Omega \in \mathbb{R}^2$ of I as:

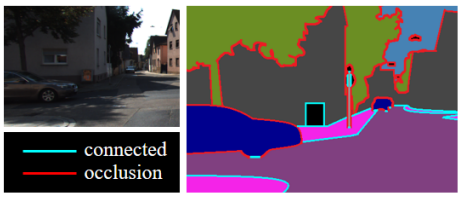

Fig. 2. Classifying boundaries (*Connected*, *Occlusion*) based on semantic labels.

$$d_{\mathbf{x}} = (1 - w_m)\frac{\sum_{\mathbf{y} \in N} w_l w_d w_c w_s d_{\mathbf{y}}}{\sum_{\mathbf{y} \in N} w_l w_d w_c w_s} + w_m d_{m\mathbf{x}} \quad (1)$$

where $d_{\mathbf{y}} \in D_p$ corresponds to known-depth pixels in the $N \in \mathbb{R}^2$ nearest neighborhood of $\mathbf{x}$ and $d_{m\mathbf{x}}$ is the depth at $\mathbf{x}$ solved using motion stereo between frames I and $\bar{I} \in S$ and $I \neq \bar{I}$. The weights w_l, w_d, w_c, w_s, and w_m are calculated form five properties – pixel proximity, depth difference, image gradient, semantic labels and motion stereo, respectively.

1) Pixel proximity (PP): This weight depends on the Eucledian distance between the estimated pixel $\mathbf{x}$ and the known-depth pixel $\mathbf{y}$. As the distance between the two pixels increases, the contribution of $\mathbf{y}$ to the depth value of $\mathbf{x}$ decreases. We define the weight as:

$$w_l = \frac{1}{\beta_l + \|\mathbf{x} - \mathbf{y}\|_2} \quad (2)$$

The parameter β_l is used for normalization.

2) Depth (DE): We define the depth weight as:

$$w_d = \frac{1}{\beta_d + |d_{\mathbf{g}} - d_{\mathbf{y}}|} \quad (3)$$

where w_d is dependent on the difference between depth values of $\mathbf{y}$ and the nearest known depth to pixel $\mathbf{x}$. We find the nearest depth, $d_{\mathbf{g}}$, as the value at pixel $\mathbf{g}$ with smallest local geodesic distance from $\mathbf{x}$. The local geodesic distance is dependent on the difference in proximity and color similarity of the estimated pixel and the candidate known-depth pixel. We define this distance as:

$$G = \lambda_l \|\mathbf{x} - \mathbf{y}\|_2 + \lambda_c \|I(\mathbf{x}) - I(\mathbf{y})\|_2 \quad (4)$$

where λ_l and λ_c are normalization weights.

3) Image gradient (IG): Natural object boundaries are often indicated by the difference in color or intensity of adjacent pixels. We utilize this assumption to further weight a pixel based on the similarity of its appearance to the estimated pixel. To do this, we find the maximum normalized image gradient, $\dot{I}_{max}$ along the path between $\mathbf{x}$ and $\mathbf{y}$ and define the color weight as:

$$w_c = \frac{1}{\beta_c + \dot{I}_{max}} \quad (5)$$

The image gradient can be solved using edge detection techniques such as SED [18].

4) Semantic boundary labels (SB): We use semantic segmentation to further identify object boundaries. We classify different object boundaries as either *Connected* or *Occlusion* as shown in Figure 2. *Connected* boundaries usually exist along the edges between the ground and objects on it such as buildings, cars, or trees, and usually located at the bottom-most part of these objects. On the other hand, *Occlusion* boundaries happen along the edges of two vertical objects such as buildings and cars.

The difference in semantic labels and the identified boundary determine whether the weight is increased or decreased. For example, when the boundary is labeled as *Connected*, the neighboring pixel should be counted during estimation and therefore the weight is increased. On the other hand, when the boundary is classified as *Occlusion*, there should be an obvious disconnection between the depth of the two neighboring pixels and the weight should be decreased.

We define the weight for the semantic boundary as:

$$w_s = \begin{cases} 1 & (L_{\mathbf{x}} = L_{\mathbf{y}}) \\ \alpha_s & (otherwise) \end{cases} \tag{6}$$

This formula reduces the weight when the semantic labels between estimated point $L_{\mathbf{x}}$ and the known-depth point $L_{\mathbf{y}}$ are different and the boundary is classified as *Occlusion*.

5) Motion stereo (MS): Using sequential images gives more information other than just color gradient and object boundaries. In particular, assuming a non-zero translational motion, dense depth map of static objects can be estimated from successive frames. Using this information, we can further improve the estimation of unknown depths in the image, particularly outside the boundary of the known point clouds (typically upper part of images when using LIDAR-image pairs).

We define the weighting of the depth from motion stereo using concatenated inverse oriented distance function:

$$w_m = \begin{cases} 0 & (\mathbf{x} \in B) \\ h(\mathbf{x}, \delta B) & (\mathbf{x} \notin B) \end{cases} \tag{7}$$

where $h(\cdot)$ defines the Euclidean distance of $\mathbf{x}$ to the boundary δB of the known point cloud area B. The weight w_m increases linearly as the distance to the boundary increases, which decreases the contribution of the known-depth points along the boundary to the estimated pixel.

B. Semantically Dependent Propagation

In contrast to other methods that also handles interpolation of sparse data ([1], [2], [19]), our method relies on directionally biased propagation. This means that we give different importance to the direction the data is propagated with respect to the semantic classes.

Our idea is based on observable visual properties of objects in a projective imaging system, where in object shapes are generally preserved with obvious limits to the inherent effects of 3D-2D projection. By relying on the semantic classes, it is possible to guess the geometric structure of the object and as a result allows us to constrain the relationship between neighboring depth pixels.

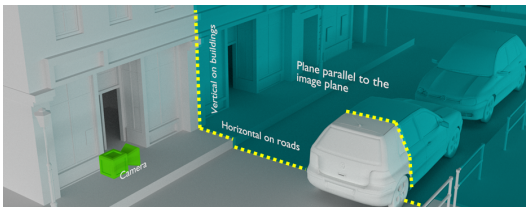

Fig. 3. The intersection between the plane parallel to the image plane and the 3D structures defines lines of which direction is dependent on the semantic class.

For example, in a perspective projection where the camera is perpendicular to the ground, we can define a plane that is parallel to the camera frame (see Figure 3). Points in this plane are equidistant to the camera frame, hence have equal depth values. If we intersect this plane with the 3D objects, the direction of the lines that are formed appears to be dependent on the semantic class.

The value of $d_{\mathbf{x}}$ is solved sequentially along the direction indicated by the semantic label. After $d_{\mathbf{x}}$ is calculated for $\mathbf{x}$, it is removed from D_e and becomes a subset of D_p. Since the depth map is propagated sequentially, this means that succeeding pixels will also use the newly added value to the interpolation. Using this scheme, we were able to improve the accuracy of estimated depth especially along the object boundaries. Figure 4 shows the effect of this approach.

C. Optimization

After solving the dense depth map using our proposed propagation method, we implemented an optimization step to solve for a smooth depth map $\bar{D}$. We borrow the tensor-based TGV-L2 (total generalized variation) denoising method described in [11]. To do this, we solve for the depth u at pixel $\mathbf{x}$, where $u \in \bar{D}$, as:

$$\begin{aligned} \arg\min_{u,v} \int_{\Omega} \{ & \alpha_1 |T^{\frac{1}{2}} (\nabla u - v)| \\ & + \alpha_0 |\nabla v| + \lambda \|u - d_{\mathbf{x}}\|^2 \} d\mathbf{x} \end{aligned} \tag{8}$$

where T is an anisotropic diffusion tensor as described in [11]. The above optimization function allows u to be smooth by imposing a small TGV (∇v) through the relaxation variable v, while constraining the value around $d_{\mathbf{x}}$. It improves the values along the natural object boundaries described by the edge images and guided by the diffusion tensor. We assign the parameters α_0, α_1 and λ in the same manner as in [11] and solve (8) using primal-dual decomposition and the second-order approximation of TGV.

IV. IMPLEMENTATION

Our implementation requires a calibrated [20] image sequence and a sparse 3D point cloud. We first assume that the 3D points are projected onto the image plane for each frame

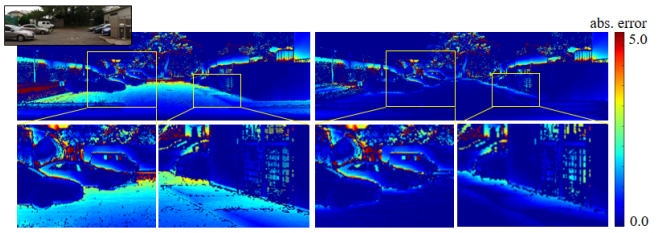

Fig. 4. Comparison of depth accuracy along boundaries without directional propagation (left) and with our proposed directionally biased propagation (right).

in the sequence as a sparse depth map. This RGBD pairing is common in publicly available datasets such as [21].

A. Semantic Segmentation

We process each image in the sequence for the semantic segmentation. We use a publicly available implementation of the ICNET method [22][23] trained on the CITYSCAPES [24] dataset. The semantic classes are: road, sidewalk, building, wall, fence, pole, traffic light, traffic sign, vegetation, terrain, sky, person, rider, car, truck, bus, train, motorcycle, bicycle and void. We run ICNET on a GTX1080Ti GPU computer and achieved a 30fps frame rate on 1242x375 image size.

B. Boundary Labeling

To implement our boundary labeling scheme, we use the semantic classes generated by ICNET. We devise a simple boundary traversal in the semantic segmentation image to determine the type of boundary. We first re-categorized the objects such as road, sidewalk and terrain as *ground*, of which self-boundaries are labeled as *Connected*. Except for *vehicles* (car, truck, bus, train, motorcycle, bicycle), crossing a boundary to the *ground* during vertical traversal, indicates a *Connected* boundary. For *vehicles*, the bottom most section of the segment bounded by the ground (wheels) are also labeled as *Connected*. All other boundaries are then considered as *Occlusion*.

C. Motion Stereo

For motion stereo, we implemented a depth estimation method described in [25]. This method solves the motion stereo problem for two views in a variational framework and runs in real time. We estimate the correct scale relative pose between two frames using the LIDAR point cloud data and the dense correspondence from optical flow [26], and perform a perspective-n-point [27] estimation. After solving for the poses, we then estimate the dense depth using [25]. On a GTX1080Ti GPU, we were able to achieve a 10fps frame rate which is suitable for our method.

D. Propagation and Optimization

We set the parameters of each term in the propagation step to normalize the values and scale the range between different terms (i.e. pixel distance, depth, intensity values). In our implementation, we used the values: $\beta_l = 0.4$, $\beta_d = 1.0$, $\lambda_l = 0.1$, $\lambda_c = 1.0$, $\beta_c = 0.1$, and $\alpha_s = 10.0$.

We use the sparse depth map from projected LIDAR points, RGB, semantic segmentation and depth from motion stereo images as inputs to our proposed method. Both propagation and optimization steps are parallelized using the same GPU as above with C++/CUDA to achieve real-time results. The propagation step requires 19ms. In our experiments, there is a trade-off between optimization iteration steps and accuracy and smoothness along object boundaries. Higher iterations result in a more defined object boundaries at the cost of processing time. For a 1242x375 image, we determine a range of 29ms to 105ms for iteration values between 50 and 200. In our results, we use an iteration value of 100 and achieve a processing time of 50ms using parameter values $\alpha_0 = 17.0$, $\alpha_1 = 1.2$ and $\lambda = 5.0$.

V. RESULTS AND COMPARISON

A. Effective Contribution of Each Term

We evaluated the contribution of each term on the accuracy of the resulting depth map and summarize the results in Table I and Figure 5 showing the error map as used in [21]. We added and accumulated the term one by one starting from the pixel proximity. From the results, each additional term gradually reduces the MAE and RMSE.

Our proposed semantic boundary labeling scheme improves the RMSE by 27.1mm and the MAE by 210.4mm for Frame 12 of our dataset. From the images, it is apparent that the errors along the object boundaries are reduced. Moreover, by adding the motion stereo term, the accuracy of estimated depth outside of measured LIDAR regions is greatly improved, with total RMSE improvement of 1458.5mm and MAE of 476.7mm.

However, while applying the optimization step improves the RMSE slightly by 7.9mm, the MAE was worse at +95.1mm. The degradation after optimization is due to the naive edge smoothing which often excludes semantic information especially in geometrically smooth areas (e.g. ground) with visually varying textures (e.g. paints and markings).

B. Comparison with Existing Methods

We compare the results of our propagation technique with our implementation of two existing methods [1] and [2] using our outdoor dataset which consists of image pairs with dense ground truth depth map, ground truth semantic segmentation

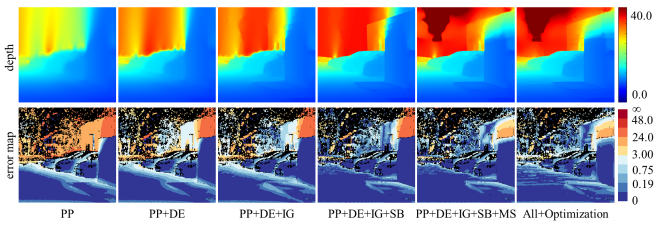

Fig. 5. Contribution of each term on the accuracy of depth estimation.

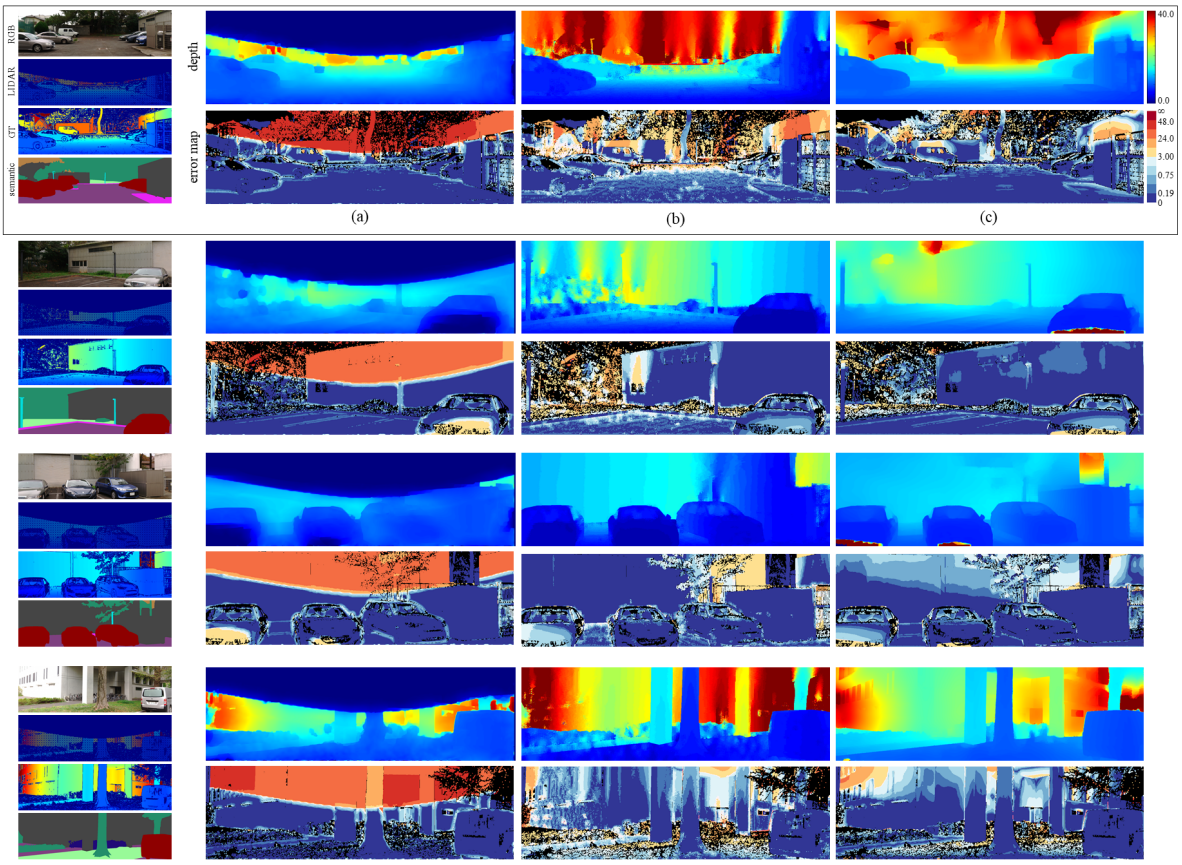

Fig. 6. Comparison of (a) [1], (b) [2] and (c) our method with error map [21] on samples of our dataset with ground truth dense depth map and semantic labels. (Top to bottom: frames 12, 18, 29, and 35.)

TABLE I

CONTRIBUTION OF EACH TERM ON THE ACCURACY OF DEPTH ESTIMATION ON FRAME 12 OF OUR DATASET USING MAE AND RMSE (IN MM) AND ERROR REDUCTION (DIFF.) WITH THE ADDED TERM.

Term	MAE	Diff.	RMSE	Diff.
PP	2092.3	-0.0	4184.2	-0.0
PP+DE	1614.0	-478.3	3685.8	-498.4
PP+DE+IG	1409.0	-205.1	3420.6	-265.3
PP+DE+IG+SB	1198.6	-210.4	3393.5	-27.1
PP+DE+IG+SB+MS	**721.8**	-476.7	1935.0	-1458.5
All+Optimization	817.0	+95.1	**1927.1**	-7.9

and known pose. Figure 6 shows the depth results from the three methods as well as the error map. We also compare the three methods using maximum absolute error (MAE in mm) and root mean square error (RMSE in mm) measures and summarize the results in Table II. In all but one image, our method outperforms the two other methods in terms of accuracy. Compared with [1], our obvious advantage is the availability of estimated depth even without the LIDAR inputs (top part of the image). Using interpolation-based completion suffers from the limitation of estimation window size which is not enough to cover the whole image especially when large portions are missing. Increasing the window size, however, significantly increases computation time.

Compared to [2], our method achieves better results in terms of accuracy especially in regions with large missing LIDAR data and mostly uniform semantic segmentation because of our motion stereo scheme. Generally speaking, when an object is perfectly segmented even without sparse depth prior (such as the tree trunk in frame 35 of Figure 6), the method described in [2] works very well. However, in recent cases, semantic segmentation methods can only identify general object classes and leaves out natural and specific object boundaries, hence the advantage of motion stereo-based depth estimation. For example, the wrong depth of the trees in frame 12, windows in frame 18, and the fence and building boundary in frame 29, were propagated to the top of the image when using method [2] due to the flat semantic label. On the other hand, these regions were more accurately estimated with our proposed method.

We also evaluated our method on a the KITTI [21] dataset and show our propagation and optimization result in Figure 7. Even though the semantic segmentation is not perfect, our proposed approach was able to estimate the depth of even thin objects outside the boundary of the measured LIDAR points, such as traffic signs and poles. Moreover, the optimization step refines the boundary conditions where the semantic labels fail.

TABLE II

COMPARISON OF OUR METHOD WITH [1] AND [2] USING MAE AND RMSE (IN MM).

frame	[1]		[2]		Ours	
	MAE	RMSE	MAE	RMSE	MAE	RMSE
8	4328.3	8563.3	1022.2	3008.7	**529.3**	**1765.6**
12	5733.8	12528.6	1823.5	4031.0	**721.8**	**1935.0**
18	5817.6	9498.4	502.8	1494.2	**242.2**	**792.3**
29	4970.0	8261.9	415.4	1159.1	**387.0**	**807.4**
35	8845.8	14098.1	770.6	**1939.1**	**737.7**	1948.0

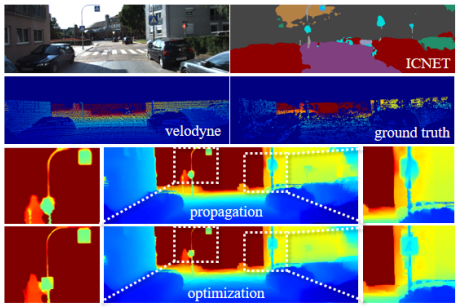

Fig. 7. Sample result of our method with the KITTI dataset [21] using the computed semantic segmentation from ICNET [22]. Thin objects outside of the measured LIDAR region and inaccurate semantic segmentation were estimated correctly

VI. CONCLUSIONS

In this paper, we proposed a dense depth estimation method by using a sparse LIDAR data and an image sequence. Our results show that using our proposed directionally-biased propagation, we were able to improve the accuracy of the result along object boundaries. Furthermore, by utilizing the semantic labels to classify different type of boundaries, we were able to make the depth estimation more reliable. We were able to accurately estimate the depth at the regions with large missing LIDAR points using our motion stereo scheme. In our implementation, we were able to achieve real-time processing using modern GPUs.

However, our boundary labeling method is dependent on the accuracy of the semantic segmentation. For future work, a segmentation method that can classify between individual objects can be used which will allow for detecting occlusion boundaries between similar class objects. Additionally, a wider range of propagation strategy, i.e. non-strict direction, can be extracted from the semantic classes and can increase the flexibility of the proposed approach. Moreover, the naive optimization scheme can be improved to include semantic information such that the edge refinement is limited to geometric object boundaries and ignores the visual texture of smooth surfaces.

REFERENCES

[1] L. Chen, Y. He, J. Chen, Q. Li, and Q. Zou, "Transforming a 3d lidar point cloud into a 2d dense depth map through a parameter self-adaptive framework," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 1, pp. 165–176, Jan 2017.

[2] N. Schneider, L. Schneider, P. Pinggera, U. Franke, M. Pollefeys, and C. Stiller, "Semantically guided depth upsampling," in *Proc. IEEE German Conf. Pattern Recognit.*, 2016, pp. 37–48.

[3] P. Gurram, S. Hu, and A. Chan, "Uniform grid upsampling of 3d lidar point cloud data," in *Proc. 3D Image Process. App.*, vol. 8650, 2013.

[4] C. Premebida, L. Garrote, A. Asvadi, A. P. Ribeiro, and U. Nunes, "High-resolution lidar-based depth mapping using bilateral filter," *arXiv preprint arXiv:1606.05614*, pp. 2469–2474, 2016.

[5] M. Dimitrievski, P. Veelaert, and W. Philips, "Semantically aware multilateral filter for depth upsampling in automotive lidar point clouds," in *Proc. IEEE Intel. Vehic.*, 2017, pp. 1058–1063.

[6] Y. Tsuji, H. Chishiro, and S. Kato, "Non-guided depth completion with adversarial networks," in *Proc. IEEE Intell. Trans. Sys. Conf.*, 2018, pp. 1109–1114.

[7] Q. Yang, R. Yang, J. Davis, and D. Nistér, "Spatial-depth super resolution for range images," in *Proc. IEEE Int. Conf. Comput. Vis. and Pattern Recognit.*, 2007, pp. 1–8.

[8] F. Garcia, D. Aouada, B. Mirbach, T. Solignac, and B. Ottersten, "A new multi-lateral filter for real-time depth enhancement," in *Proc. IEEE Int. Conf. Adv. Vid. Sig. Surv.*, 2011, pp. 42–47.

[9] J. Park, H. Kim, Y. Tai, M. S. Brown, and I. Kweon, "High quality depth map upsampling for 3d-tof cameras," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2011, pp. 1623–1630.

[10] M.-Y. Liu, O. Tuzel, and Y. Taguchi, "Joint geodesic upsampling of depth images," in *Proc. IEEE Int. Conf. Comput. Vis. and Pattern Recognit.*, 2013, pp. 169–176.

[11] D. Ferstl, C. Reinbacher, R. Ranftl, M. Rüther, and H. Bischof, "Image guided depth upsampling using anisotropic total generalized variation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 993–1000.

[12] J. Yang, X. Ye, K. Li, C. Hou, and Y. Wang, "Color-guided depth recovery from rgb-d data using an adaptive autoregressive model," *IEEE Trans. Image Process.*, vol. 23, no. 8, pp. 3443–3458, 2014.

[13] J. Dolson, J. Baek, C. Plagemann, and S. Thrun, "Upsampling range data in dynamic environments," in *Proc. IEEE Int. Conf. Comput. Vis. and Pattern Recognit.*, 2010, pp. 1141–1148.

[14] L. Pan, Y. Dai, M. Liu, and F. Porikli, "Depth map completion by jointly exploiting blurry color images and sparse depth maps," in *Proc. IEEE Winter Conf. App. Comput. Vis.*, 2018, pp. 1377–1386.

[15] W. Zhou, X. Li, and D. Reynolds, "Guided deep network for depth map super-resolution: How much can color help?" in *Proc. IEEE Int. Conf. Acous. Speech Sig. Process.*, 2017, pp. 1457–1461.

[16] F. Mal and S. Karaman, "Sparse-to-dense: Depth prediction from sparse depth samples and a single image," in *Proc. IEEE Int. Conf. Robot Automat.*, 2018, pp. 1–8.

[17] F. Qi, J. Han, P. Wang, G. Shi, and F. Li, "Structure guided fusion for depth map inpainting," *Pattern Recog. Letters*, vol. 34, no. 1, pp. 70–76, 2013.

[18] P. Dollár and C. L. Zitnick, "Structured forests for fast edge detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 1841–1848.

[19] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "EpicFlow: Edge-Preserving Interpolation of Correspondences for Optical Flow," in *Proc. IEEE Int. Conf. Comput. Vis. and Pattern Recognit.*, 2015, pp. 1164–1172.

[20] R. Ishikawa, T. Oishi, and K. Ikeuchi, "Lidar and camera calibration using motions estimated by sensor fusion," in *Proc. IEEE Int. Work. Robots Sys.*, 2018, pp. 7342–7349.

[21] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *Int. J. Robot. Res.*, vol. 32, pp. 1231–1237, 2013.

[22] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, "Icnet for real-time semantic segmentation on high-resolution images," in *Proc. IEEE Europ. Conf. Comput. Vis.*, 2018, pp. 405–420.

[23] Accessed on 02.24.2019. [Online]. Available: https://github.com/hellochick/ICNet-tensorflow

[24] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Int. Conf. Comput. Vis. and Pattern Recognit.*, 2016, pp. 3213–3223.

[25] M. Roxas and T. Oishi, "Real-time simultaneous 3d reconstruction and optical flow estimation," in *Proc. IEEE Winter Conf. App. Comput. Vis.*, 2018, pp. 885–893.

[26] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Proc. IEEE Int. Conf. Comput. Vis. and Pattern Recognit.*, July 2017, pp. 1647–1655.

[27] V. Lepetit, M. Moreno-Noguer, and P. Fua, "Epnp: An accurate o(n) solution to the pnp problem," *Int. J. Comput. Vis.*, vol. 81, no. 2, pp. 155–166, 2009.