Quadtree Segmentation Network for Obstacle Avoidance in Monocular Navigation

Daniel Braun, Olivier Morel, Pascal Vasseur, Cédric Demonceaux

Quadtree Segmentation Network for Obstacle Avoidance in Monocular Navigation

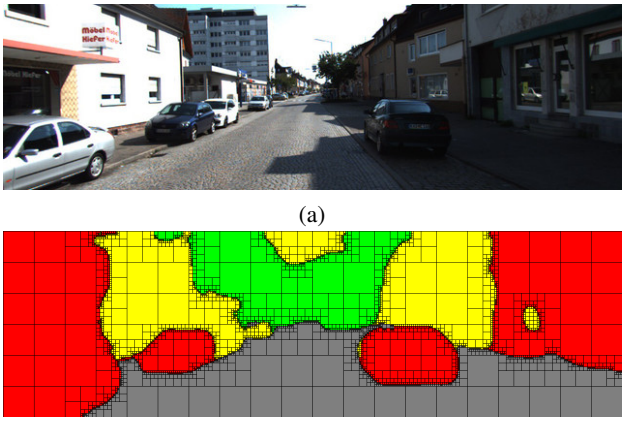
Daniel Braun[1], Olivier Morel[1], Pascal Vasseur[2] and Cédric Demonceaux[1]

***Abstract*— Monocular depth map prediction has become in recent years a major research topic in computer vision. Especially with the emergence of self-supervised methods that have demonstrated that based on geometric properties, it is possible to obtain good results without human generated labels. But these methods are, for the moment, mainly oriented towards high resolution dense reconstruction and therefore neglect applications for robotic navigation. In this paper, we propose addressing this problem by developing a solution for navigation and obstacle avoidance. The method takes advantage of dense depth prediction to segment the view into a limited number of classes of interest for navigation. Four classes have been thus retained: one to segment the road and three to cluster obstacles by their distance (close, middle and far). As a result, the system is able to directly identify threats and areas where it can navigate. Furthermore, efficient information compression can be considered using a quadtree data structure derived from Quadtree Generating Network. Experiments conducted on the Kitti dataset have shown our proposed method can efficiently predict a quadtree segmentation directly from monocular input images. In addition, the approach tends to significantly reduce the amount of information to be predicted without any loss of accuracy.**

I. Introduction

Autonomous navigation is one of the key subjects in computer vision and robotics. In recent years, deep learning approaches have worked on developing new methods to outperform the existing geometric based algorithm such as SLAM. Among them, we find the approaches of depth prediction from a single image. These methods have the particularity to give solutions to an ill-posed problem. Indeed, at least two images are normally required to be able to extract depth information using geometric methods. Monocular depth is, therefore, a prospect to lighten robotic systems. If the depth can be efficiently extracted from one image, robotics system would benefit from the latter and become RGB-D sensitive while equiped with a single sensor. However, most of these methods are engaged in a race for accuracy [13] at the expense of computation time. Only a few of them addressed the question of real-time embedded solutions [14], [10].

However, monocular depth prediction methods have limited applications. The predicted depth is only reliable over a few meters. It is therefore unusable on the entire data range, which can be up to 80 or 100 meters for outdoor applications. In obstacle avoidance applications, this degree of accuracy is unnecessary. It is often more appropriate to have a binary or segmented output providing a simplified representation of the scene. Semantic segmentation methods aim at enriching the understanding of the scene and thus improving the navigation performances. This semantics is sometimes even integrated to depth prediction methods to improve its quality [18]. However, segmentation approaches rely on annotated data, which makes it difficult to generalize to every situation.



(a)

(b)

Fig. 1. Quadtree scene segmentation for obstacle avoidance. (a) is the input monocular RGB image and (b) is the output segmented view for obstacle avoidance, in which we highlighted the quadtree subdivision.

Upon those observations, we propose here to develop a hybrid solution between depth prediction and scene segmentation, presented in Figure 1. This method predicts from a single image a segmentation of the scene based on depth to perform autonomous navigation with obstacle avoidance. This segmentation is initially constructed geometrically from a self-supervised depth prediction method, which will be used as ground truth during the training phase. It is segmented into four classes: one is segmenting the ground and the other three are splitting the depth according to the distance of the obstacles. The *ground* class represents the safe area in which the autonomous system is allowed to navigate. The following three classes are based on the depth and indicate the level of priority with which the system must take into account the obstacles. This limited number of classes allows a fast training, with a light network, which can be further refined by generating a prediction in the form of a quadtree data structure using a Quadtree Generating Network [2]. The quadtree decoder brings the advantage of considerably reducing the computational complexity of the

[1]Daniel Braun, Olivier Morel and Cédric Demonceaux are with ImViA Laboratory, University of Bourgogne Franche-Comté, 71200 Le Creusot, France.

[2]Pascal Vasseur is with MIS Laboratory, University of Picardie Jules Verne, 80000 Amiens, France.

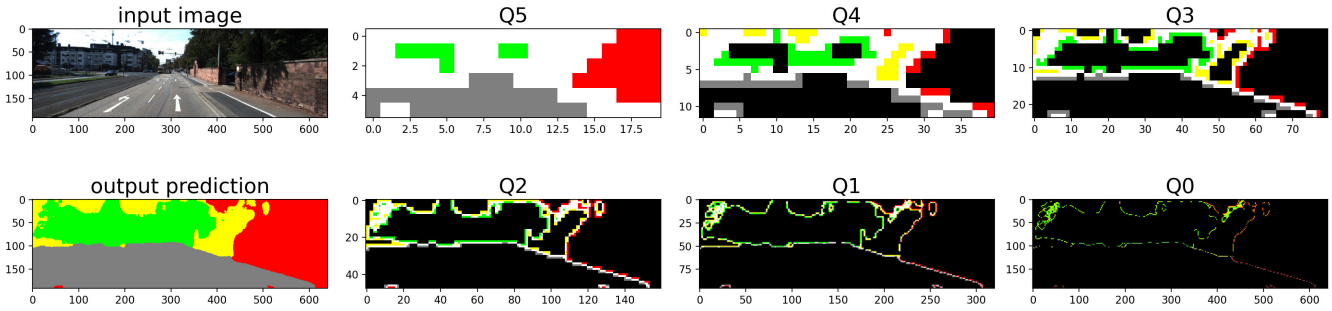


Fig. 2. Quadtree prediction decomposition. On the top left the input image and the bottom left the recomposed output prediction. Q_5 to Q_0 are the outputs of each prediction layers. We have the *mixed* class in white, the *Ground* class in gray, the *Close* class in red, the *Middle* class in yellow and the *Far* class in green. The pixels in black stores no value: it's the area which have been predicted by previous layers of the decoder.

network. Indeed, some zones in the image can be classified at low resolution. Thus, the pixels that have already been classified can be ignored in the rest of the decoder, thanks to the usage of submanifold sparse convolutions [8]. It permits to focus the attention on areas requiring a higher degree of accuracy to be segmented.

II. Related Works

A. Self-supervised Monocular Depth

Self-supervised learning consists of training a network to predict information without interference of human made labels. The network's output has to be learnt through an optimization process only based on available input information. For monocular depth prediction, the learning process consists of injecting an adjacent view with small changes in the camera motion. If the relative pose between the views is known, the depth prediction can be projected into this adjacent view in order to minimize the photometric reconstruction error as introduced in [4]. The method has been improved over the years with the introduction of the left-right consistency with stereo supervision [6] and the simultaneous pose and depth prediction for ego-motion training [19], [7]. However, some problems persist, essentially caused by occlusions and moving objects between views, leading to bias as the photometric minimization error tends towards an incorrect result. This issue is addressed by Chen et al. [1] which proposes an occlusion-aware module which allows to focus the training on non-occluded areas.

The results obtained for monocular depth prediction are encouraging and raise the question of the use of these methods for navigation applications as expressed in Dong et al. survey [3]. For the moment, only a minority of papers address the issue of lightweight architecture, usable on embedded systems [14], [10]. These methods seek to optimize their network structure to find the best trade-off between lightness and precision.

B. Quadtree Data Structure

Quadtree is a hierarchical tree data structure that is deeply tight to the image processing field since it was first introduced nearly 40 years ago by Samet et al. [16]. Therefore, this method has often been used for image compression especially for navigation applications [17]. But it has only recently been introduced in deep learning [2] probably because its inconsistent data structure does not fit the standard tensor representation. The introduction of sparse convolutions [11] permitted to overcome this limitation.

In Quadtree Generating Networks [2], the author demonstrated the usage of quadtree permits to significantly reduce the computation cost for segmentation inference with no loss of accuracy. It was made possible using sparse convolutions by extracting in the early stage of the decoder the parts of the image in which all the pixels already belonged to a unique class. The extracted parts are then not being processed in the rest of the decoder, to avoid unnecessary computation.

C. Segmentation for Navigation

Depth prediction and semantic segmentation are tight since they are both methods to enhance scene understanding. For this reason, some methods predict both information together and fuse them to enhance results [18]. Some pre-deep learning methods like Liu et al. [12] proposed to predict depth information from a single image using segmentation labels and Markov random fields. Wang et al. [18] proposes a Semantic Divide-and-Conquer method predicting the segmentation and using it to infer the depth per class. Such decomposition simplifies the learning and permits a greater generalization.

The combination of the semantic and depth have demonstrated they capability reciprocally improve each other results. Yet, all the semantic classes are not relevant to improve the depth understanding of the scene which can even aggregate confusion in the prediction. Additionally, those methods have in common the need of semantic segmentation labels and, from the best of our knowledge, there is no method generating them without human interference. Therefore, those methods are at most semi-supervised making them harder to be generalized.

III. Method

In this section, we present our method to predict the quadtree segmentation of the scene. We will explain the quadtree representation and its implementation into a convolutional deep neural network. We will also describe the process to build our self-supervised reference map and the way it is employed to train the quadtree prediction.

A. Quadtree representation

1) Definition: A quadtree is a herarchical tree data structure composed of nodes connected by branches. Each node is defined by its position in the tree, i.e. its depth level and location at that level, and by the value it contains. A node can be subdivided into four sub-nodes if more information is needed to describe that location in the tree. It can be defined as follows in equation (1), by a set of N nodes, of which the i^{th} node is characterized by its depth level l_i, its centroid coordinates (x_i, y_i) and its value v_i.

$$Q = \{l_i, x_i, y_i, v_i \mid i = \{1, .., N\}\} \tag{1}$$

In the rest of the paper, the quadtree will often be separated by its nodes depth level and noted $Q_l = \{Q \mid l_i = l\}$. Indeed, the depth level is directly related to the squares dimension that represent the nodes. Therefore, all these nodes form an image of the scene at a given resolution, see Figure 2. In general, the quadtree can be seen as a multi-resolution sparse representation of the image.

2) Quadtree Segmentation: For the segmentation approach, each node in the quadtree has a value v_i which is a vector of $k+1$ elements, representing the k segmentation classes plus one *mixed* class. The values in v_i represent the probability of the node to belong to the corresponding classes. If the *mixed* class obtains the highest score, then it means the area is composed of several classes. Therefore, the node has to be subdivided in order to better describe the corresponding location in the image. The same process is applied recursively on each sub-node until either the *mixed* does not obtains the highest probability or the maximum depth level in the quadtree is reach.

B. The network

1) Architecture: The network architecture is based on the quadtree generating network (QGN) proposed by [2]. It is a U-Net [15] architecture composed of a dense ResNet [9] encoder and sparse ResNet decoder. The sparse decoder consists of submanifolds sparse convolutions blocks [8], which exclusively performs operations on active sites, i.e. where there is data to be processed. Initially designed for sparse input data processing, they are suited for quadtree construction, where the information is voluntarily reduced.

The strategy behind QGN is to directly generate a quadtree through the decoder without having to compute the full resolution dense prediction. Indeed, some information can be correctly predicted at low resolution from the first layers of the decoder. Further processing of this information will not change the output. Subsequently, they can be stored into the quadtree and removed from the set of active sites to not be processed by the following layers of the decoder.

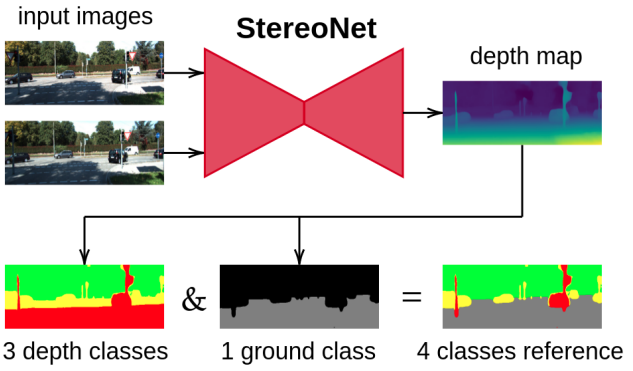


Fig. 3. The reference segmentation map is constructed based on a depth map generated by stereoNet. The image is segmented into four classes: three classes separating the depth and one class for the ground.

2) Quadtree Subdivision: Each prediction layer in the decoder outputs a sparse segmented map, noted Q_l with l representing the quadtree level. From this map, one can build the activation map, which will define the set of active sites for the subsequent layer of the decoder. It is obtained by applying a filter on the v_i values of Q_l. Indeed, as presented in section III-A.2, only the v_i with their highest probability belonging to the *mixed* class are subdivided, i.e. conserved as active sites to be computed by in the upper layers of the decoder. In the other cases, the value has already been assigned to a class and is therefore removed from the active sites. This activation map $\mathcal{A}$ is defined for each v_i as follow:

$$\mathcal{A}(v_i) = \begin{cases} 1, & \textit{if } \arg\max(v_i) = m, \\ 0, & \textit{else,} \end{cases} \tag{2}$$

where m is the channel number of the *mixed* class.

C. Training

1) Self-Supervised Reference Map: Depth map prediction networks have demonstrated their ability to accurately infer scene geometry without ground truth supervision. The reference map, used to train our model, is derived from depth prediction network to build our segmentation classes.

For urban applications as proposed by the Kitti dataset [5], the position of the cameras is known and similar from one image to another. It is therefore possible to extract useful information with simple image processing tools. The aim of the method is to be able to navigate and avoid obstacles. Subsequently, the image has been segmented into four classes as illustrated in Figure 3:

- three classes to cluster obstacles by distance (close, middle and far),
- one class to detect the ground, where the driving system is allowed to move.

The objective of this segmentation is to allow the system to clearly identify the presence of nearby hazards as well as

the ground on which it is permitted to move. In theory, only a third class would have been necessary to classify distant obstacles. However, as monocular depth prediction can be unreliable, we chose to integrate a buffer class between near and far obstacles. This *middle* class consists, as it will be presented later in the experiments, in producing the *close* and *far* classes disjoint. Hence, we increase the reliability of the whole prediction.

The depth map used to compute the segmentation reference is obtained from a network, which infers depth from a pair of stereo images. This network has been trained upstream using state-of-the-art monocular depth learning methods and will be called stereoNet in the rest of the paper. To counteract scale inconsistency and to have the most reliable reference possible, the network generates the depth from calibrated stereo images as the information is available. Still, it would have worked similarly if the depth was acquired from a single image or with any other approach to capture the depth.

2) Loss function: As in [2], the optimization process is based on the minimization of the cross-entropy loss function, noted $\mathcal{H}$, between the prediction Q and the reference map decomposed into quadtree, noted Q^*, for every classes. The loss is computed for each depth level of the quadtree Q_l as follows:

$$\mathcal{L}_l = \frac{1}{N_l} \sum_{i=1}^{N_l} \mathcal{H}(v_i, v_i^*) \tag{3}$$

where N_l is the number of elements in Q_l. The cross-entropy function $\mathcal{H}$ compares the predicted value v_i to the reference value v_i^* at the same location in the image (x_i, y_i).

Each Q_l is sparse and does not represent the same portion of the image in the prediction (see Table III). Therefore, the global loss function is a weighted sum of the $\mathcal{L}_l$ terms:

$$\mathcal{L} = \frac{1}{N} \sum_{l=0}^{N-1} \lambda_l \mathcal{L}_l \tag{4}$$

with λ_l some constants that weight respectively the $\mathcal{L}_l$ and N the number of depth levels in the quadtree, here N has been set to 6 for application purpose.

3) Active sites: The advantage of using a sparse quadtree decoder is its allowance to only compute necessary operations. But it also implies the quality of the prediction depends on the ability of the network to perform the appropriate operations. They are determined by the set of active sites which indicates the sparse convolutional system the features to consider. During the inference, these active sites are defined by the nodes of the quadtree to be subdivided, i.e. where the *mixed* class gets the highest score. This approach works effectively on a trained model, but is uncertain during training phase. Indeed, the network will have difficulties to converge if the quadtree structure is incorrect. To address this issue, the choice of active sites is supervised during training and is modeled on the quadtree subdivision of the reference map. This guidance allows stabilization and reduction of the training time of the network.

TABLE I
INTERSECTION OVER UNION PER CLASS AND IN AVERAGE (MIOU).

Method	Close (red)	Middle (yellow)	Far (green)	Ground (gray)	mIoU
Dense ResNet 18	68.48%	54.8%	66.5%	90.7%	70.1%
Sparse ResNet 18	65.3%	50.1%	65.2%	89.9%	67.6%
Dense ResNet 50	**72.1%**	**60.9%**	68.8%	90.2%	73.0%
Sparse ResNet 50	71.9%	59.4%	**70.6%**	**91.4%**	**73.3%**

IV. EXPERIMENTS

Experiments are conducted on the Kitti dataset [5]. It offers urban images acquired from cameras in a car, which allows exploring the scene from a constant point of view on entire video sequences. Our method is evaluated under two architectures: sparse ResNet 18 and 50 and compared with the equivalent dense version, which have been trained using the same loss function but to output a dense prediction. The double objectives of those experiments are to evaluate the reliability of the proposed segmentation solution as well as the capability of the sparse methods to equal or outperform the dense equivalent estimations.

A. Intersection over Union

The table I compares the performance of the methods on the intersection-over-union (IoU) criterion for each class and for the mean value mIoU. The first observation is that the results are similar between sparse and dense, with an advantage for the sparse ResNet 50 approach. The *ground* class is the one that obtains in all cases the most accurate prediction. The method seems to have difficulties to correctly classify the buffer *middle* class, which is in between *close* and *far* obstacles as explained in section 2.

Ultimately, the IoU results, except for the *ground* class are not compelling. However, these three classes are not based on semantic but on depth. Therefore, they might be penalized by border effects induced by class-to-class adjacency.

B. Accuracy

The objective of our method is not to do semantic segmentation, but to use the segmentation to navigate. Besides, our classes are very related to each other because they describe for three of them a depth information. As a result, there are edge effects that minimize the results of intersection over union (IoU) metrics. The aim of this experimentation is to discuss the reliability of the method by analyzing the confusion matrix. The latter, in Table II, highlights the percentage of true positives per class (diagonal values) as well as the false positives, i.e. the areas misclassified from one class to another with regard to the ground truth. Values of the four methods are displayed in the same table. Thus, each cell contains four values and are ordered as indicated in the top left corner of the table.

For navigation purposes, a high accuracy is required for the detection of the nearest obstacles. This task is primarily performed by the two classes *close* and *ground*. One can investigate their respective accuracy (true positives) and the

TABLE II

CONFUSION MATRIX REPRESENTING THE SEGMENTATION DISTRIBUTION PER CLASS OF THE PREDICTION WITH RESPECT TO THE GROUND TRUTH FOR EACH METHOD. **BOLD VALUES** ON THE DIAGONAL REPRESENTS THE ACCURACY AND THE OTHERS ARE THE INCORRECT SEGMENTATION. UNDERLINED VALUES ARE THE HIGHEST ACCURACY PER CLASS. THE VALUES ARE MEANT TO BE READ IN LINE.

Dense 18 / Sparse 18	Dense 50 / Sparse 50	Ground Truth							
		Close		Middle		Far		Ground	
Prediction	Close	**90.18%**	**91.09%**	7.03%	6.47%	0.34%	0.23%	2.45%	2.21%
		88.38%	**91.22%**	8.12%	6.20%	0.47%	0.25%	3.04%	2.33%
	Middle	16.89%	12.69%	**73.51%**	**79.28%**	8.49%	6.97%	1.12%	1.06%
		21.79%	13.84%	**67.40%**	**77.55%**	9.24%	7.35%	1.56%	1.26%
	Far	2.17%	1.01%	22.07%	19.08%	**75.24%**	**79.39%**	0.53%	0.53%
		2.22%	1.02%	21.47%	19.22%	**75.52%**	**79.06%**	0.79%	0.69%
	Ground	3.11%	2.55%	1.26%	1.11%	0.57%	0.54%	**95.06%**	**95.79%**
		2.98%	2.53%	1.17%	1.08%	0.49%	0.51%	**95.36%**	**95.88%**

TABLE III

DISTRIBUTION OF THE DATA PREDICTED BY EACH LAYER OF THE DECODER.

Method	Q5	Q4	Q3	Q2	Q1	Q0
Sparse ResNet 18	42.6%	18.4%	13.6%	9.9%	8.1%	7.5%
Sparse ResNet 50	49.6%	20.4%	13.5%	8.8%	5.2%	2.6%

TABLE IV

NETWORKS COMPLEXITY. COMPARE THE FLOPS, THE NETWORK PARAMETERS AND THE MEMORY CONSUMPTION OF EACH METHOD.

Method	FLOPs	Parameters	Memory
Dense ResNet 18	24G	27.4M	0.78GB
Sparse ResNet 18	10G	26.5M	0.83GB
Dense ResNet 50	56G	72.3M	1.14GB
Sparse ResNet 50	19G	70.7M	1.27GB

prediction errors (false positives). The *ground* class accuracy is over 95% for every methods confirming the good IoU results. The *close* class has an accuracy over 90% for the ResNet 50 methods and presents almost no critical cases since two-thirds of the false positives indicate obstacles closer than they really are. The most problematic is the *middle* class which shows over 20% of false positives, confirming the observations made on the IoU results. In practice, with such results, the *middle* class should be considered as an area to avoid, similarly to *close* class. But it fulfills its initial goal, which is to be a buffer between the *close* and *far* obstacles and ensuring separation.

When comparing the approaches, we observe a superior accuracy when deploying ResNet 50. Besides, for the two classes of interest *close* and *ground*, the best performances are obtained with the sparse method. It is, therefore, the most efficient of the four presented methods.

C. Complexity

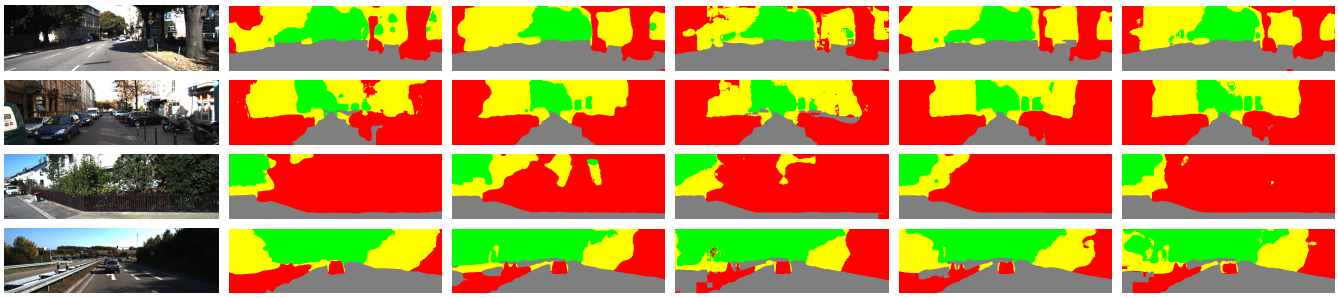
1) Quadtree Data Compression: The quadtree data structure allows compressing significantly the information. Table III presents the data distribution among each depth level of the quadtree. Over 40% of the information can be predicted by the first prediction layer of the decoder (Q_5). It means the rest of the decoder only has to process less than 60% of the information. This percentage decreases after each prediction layer, leaving only a small portion of the data to be processed by the last layers. Since the decoder is composed on submanifold sparse convolutions [8], operations are only computed on active sites, i.e. information that have yet not been extracted by previous prediction layers.

2) FLOPs, parameters and memory: The quadtree subdivision permits to drastically reduce the floating-points operations per seconds (FLOPs). Yet the memory usage remains equivalent due to the necessity to provide a dense prediction at full resolution, reconstructed from the quadtree layers. Note that the FLOPs are related to the quadtree size. Subsequently, it can fluctuate from one view to another. The number of parameters remains mathematically high and is almost equivalent to the dense counterpart due to the similarity in the architecture. Yet, for the same reasons as for the FLOPs, the prediction is highly sparse and all the parameters are not solicited during the inference.

D. Qualitative Evaluation

Qualitative results are presented in Figure 4, allowing a visual comparison of the methods with the reference map. The observations corroborate the results presented in the previous sections. The sparse segmentation networks predict similar outcomes to the dense counterpart and the *ground* and *close* classes are distinctly defined. The notable difference that can be observed is on the *middle* class in yellow which tends to encroach on the *close* and *far* classes. This behavior can be mainly explained by the difficulty of the method to accurately interpret the depth of the scene at medium and long range. As explained previously, these segmentation problems do not jeopardize the method which can be utilized for obstacle avoidance, since the classes for the *ground* and the *close* obstacles are precisely defined.

Differences between the ResNet 18 and ResNet 50 methods are visually noticeable but primarily due to some minor defects. The general understanding of the scene is maintained, and the slight qualitative loss does not appear to be redhibitory.



(a) Input (b) Reference Map (c) Dense ResNet 18 (d) Sparse ResNet 18 (e) Dense ResNet 50 (f) Sparse ResNet 50

Fig. 4. Qualitative Results. From left to right, the single input image (a), the reference map (b) and from (c) to (f) are the prediction of each methods.

V. Conclusion

In this paper, we have presented a segmentation framework dedicated to obstacle detection using a monocular navigation system. The proposed network architecture is derived from QGN and provides efficient prediction of a limited set of classes relevant for obstacle avoidance task. The reference segmentation map has been constructed from a stereo depth prediction network and converted into classes.

The experiments conducted on the Kitti dataset highlighted our method capability to achieve similar segmentation precision as the dense counterpart but with a lower computational complexity. Indeed, the quadtree data structure permits to drastically reduce the density of the prediction. It was also demonstrated that most of the information can be extracted within the first few layers of the decoder justifying the usage of quadtrees that avoids unnecessary computations.

The method has been experimented with both ResNet 18 and ResNet 50 constitutional blocks. The quantitative results benefit the ResNet 50 solution, yet it is composed of almost 3 times more parameters. Therefore, it seems the sparse ResNet 18 is more appropriate for lighter applications. The trade-off between the network's complexity and the accuracy of the output would need to be discussed during the deployement of the solution on an autonomous driving system.

Acknowledgment

This work is supported by the French National Research Agency ANR-18-CE33-0004-CLARA. We gratefully acknowledge the support of NVIDIA Corporation with the donation of GPUs used for this research.

References

[1] Z. Chen, X. Ye, W. Yang, Z. Xu, X. Tan, Z. Zou, E. Ding, X. Zhang, and L. Huang, "Revealing the reciprocal relations between self-supervised stereo and monocular depth estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 15 529–15 538.

[2] K. Chitta, J. M. Alvarez, and M. Hebert, "Quadtree generating networks: Efficient hierarchical scene parsing with sparse convolutions," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020.

[3] X. Dong, M. A. Garratt, S. G. Anavatti, and H. A. Abbass, "Towards real-time monocular depth estimation for robotics: A survey," *arXiv preprint arXiv:2111.08600*, 2021.

[4] R. Garg, V. K. Bg, G. Carneiro, and I. Reid, "Unsupervised cnn for single view depth estimation: Geometry to the rescue," in *European conference on computer vision*. Springer, 2016, pp. 740–756.

[5] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3354–3361.

[6] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 270–279.

[7] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow, "Digging into self-supervised monocular depth estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3828–3838.

[8] B. Graham, M. Engelcke, and L. Van Der Maaten, "3d semantic segmentation with submanifold sparse convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9224–9232.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[10] L. Huynh, P. Nguyen, J. Matas, E. Rahtu, and J. Heikkilä, "Lightweight monocular depth with a novel neural architecture search method," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 3643–3653.

[11] B. Liu, M. Wang, H. Foroosh, M. Tappen, and M. Pensky, "Sparse convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 806–814.

[12] B. Liu, S. Gould, and D. Koller, "Single image depth estimation from predicted semantic labels," in *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE, 2010, pp. 1253–1260.

[13] S. M. H. Miangoleh, S. Dille, L. Mai, S. Paris, and Y. Aksoy, "Boosting monocular depth estimation models to high-resolution via content-adaptive multi-resolution merging," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 9685–9694.

[14] M. Poggi, F. Aleotti, F. Tosi, and S. Mattoccia, "Towards real-time unsupervised monocular depth estimation on cpu," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 5848–5854.

[15] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[16] H. Samet, "The quadtree and related hierarchical data structures," *ACM Computing Surveys (CSUR)*, vol. 16, no. 2, pp. 187–260, 1984.

[17] K. Wang, W. Ding, and S. Shen, "Quadtree-accelerated real-time monocular dense mapping," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1–9.

[18] L. Wang, J. Zhang, O. Wang, Z. Lin, and H. Lu, "Sdc-depth: Semantic divide-and-conquer network for monocular depth estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 541–550.

[19] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1851–1858.