Semantic Grid Estimation with a Hybrid Bayesian and Deep Neural Network Approach

Özgür Erkent[1], Christian Wolf[1,2,3], Christian Laugier[1], David Sierra Gonzalez[1], Victor Romero Cano[4],

***Abstract*— In an autonomous vehicle setting, we propose a method for the estimation of a semantic grid, i.e. a bird's eye grid centered on the car's position and aligned with its driving direction, which contains high-level semantic information about the environment and its actors. Each grid cell contains a semantic label with divers classes, as for instance {*Road, Vegetation, Building, Pedestrian, Car* ...}. We propose a hybrid approach, which combines the advantages of two different methodologies: we use Deep Learning to perform semantic segmentation on monocular RGB images with supervised learning from labeled groundtruth data. We combine these segmentations with occupancy grids calculated from LIDAR data using a generative Bayesian particle filter. The fusion itself is carried out with a deep neural network, which learns to integrate geometric information from the LIDAR with semantic information from the RGB data. We tested our method on two datasets, namely the KITTI dataset, which is publicly available and widely used, and our own dataset obtained with our own platform, equipped with a LIDAR and various sensors. We largely outperform baselines which calculate the semantic grid either from the RGB image alone or from LIDAR output alone, showing the interest of this hybrid approach.**

I. Introduction

Autonomous vehicles require to perceive and comprehend their surroundings accurately in order to navigate safely and successfully. The limited capacity of the sensors, occlusions, complexities and uncertainties in the environment make it a challenging task. Deep Learning is arguably the dominant methodology for autonomous systems and Advanced Driver Assistance Systems (ADAS) in intelligent vehicles. Adopted by most of the major industrial players in the field, the high capacity of deep networks allows them to create high level semantic predictions from low-level sensor data (RGB, stereo, LIDAR etc.). On the downside, handling uncertainty in a principled way is still a difficult task with deep models alone. Bayesian techniques, on the other hand, have a built-in capability for managing uncertainty with a long history of applications in sensor fusion.

A widely used method is to construct occupancy grids without having to recognize the objects in the surroundings. Occupancy grids are spatial 2D maps of the environment which also model regions containing moving objects [1], [2], [3]. The cells of these grids contain the probability of the state of a cell. One of the advantages of the occupancy maps is that they are dense and provide information about free space which is important for vehicles and mobile robots. Another advantage is that they do not depend on the type of the sensor (e.g. stereo cameras are used in [4], laser range sensors are used in [5]). Since the model is generative, different types of sensors can be integrated easily by adapting the observation model. Compared to discriminative methods, no re-training is necessary. However, although these models provide an accurate map of the scene with possible obstacles, they do not provide the semantics of the cells, which are important for decision making.

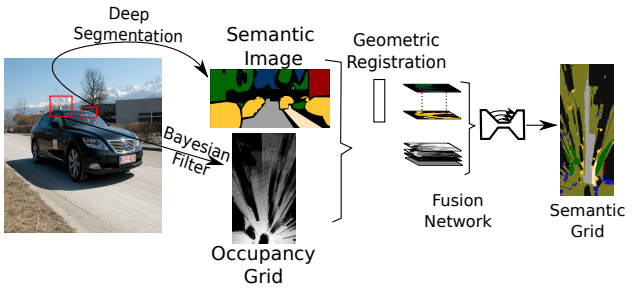


Fig. 1: An overview with a sample semantic grid.

In this work, we are interested in 2D egocentric representations. We propose a method, which estimates an occupancy grid containing detailed semantic information. The semantic characteristics include classes like *road, car, pedestrian, sidewalk, building, vegetation, etc.*. To this end, we leverage and fuse information from multiple sensors including LIDAR, odometry and monocular RGB video. To benefit from the respective advantages of the two different methodologies, we propose a hybrid approach leveraging i) the high-capacity of deep neural networks as well as ii) Bayesian filtering, which is able to model uncertainty in a unique way.

In our approach, Bayesian particle filtering processes the LIDAR data as well as odometry information from the vehicle's motion in order to robustly estimate an egocentric bird's eye view in the form of an occupancy grid. This grid contains a 360° view of the environment around the car and integrates information from the observation history through temporal filtering; however, it does not include fine-grained semantic classes.

Deep Learning is used for two different tasks in our work. Firstly, a deep network performs semantic segmentation of monocular RGB images. This network has been pre-trained on large scale datasets for image classification [6] and finetuned on the vehicle datasets. Secondly, a deep network fuses the occupancy grid with the segmented image

This work was supported by Toyota Motor Europe.

[1] INRIA, Chroma Team, Rhône-Alpes, France [2] Université de Lyon, INSA-Lyon, CNRS, LIRIS, F-69621, France [3] CITI, INSA-Lyon, F-69621, France [4] Departamento de Automática y Electrónica Universidad Autónoma de Occidente - Cali, Colombia. Correspondence: ozgur.erkent@inria.fr

of the projective view in order to estimate the semantic grid. Since the occupancy grid is dense, the semantic grid is also expected to be dense. We pay particular attention to correctly model the transformation from the egocentric projective view of the RGB image to the bird's eye view of the occupancy grid as input to the neural network.

We evaluate our approach on the KITTI dataset and on our own dataset(Fig. 1). Our contributions and the advantages of our method can be summarized as follows:

- We propose a hybrid approach, which combines the advantages of Bayesian filtering and deep neural networks.
- Bayesian filtering provides robust temporal/geometrical filtering and integration and allows for modelling of uncertainty.
- RGB information and deep neural networks provide knowledge about the semantic class labels like *sideway* vs *road*.
- The fusion process is fully learned.
- Due to dense structure of occupancy grid, we can construct a dense semantic grid even if we have a sparse point cloud.

II. Related Work

Occupancy grids are 2D spatial maps whose grids represent a probabilistic estimation of the occupancy. These are helpful to plan the motion of a mobile robot as they provide the obstacles in the environment. The Bayesian Occupancy Filter (BOF) is one of the earlier methods used to update the occupancy and dynamic properties of the cells in parallel [7]. The high computational complexity of this accurate approach motivated other methods based on Bayesian Filters. For example BOFUM [8] used a prior map, which was not always feasible. The Hybrid-Sampling Bayesian Occupancy Filter [9] introduced the separation of static and dynamic cells which reduced the required computation time significantly. Inspired by this separation, Conditional Monte Carlo Dense Occupancy Tracker (CMCDOT) [1] introduced further states which increased the speed of the Bayesian Filter approach further. Another advantage of this method is that it can be adopted to be used with different sensor modalities [4], [5] easily. We will use CMCDOT to produce our occupancy grid inputs due to its speed and accuracy.

Semantic information — Although occupancy grids provide accurate information about the state of the cells, this may not be sufficient to make decisions/plans for various tasks including navigation. Few studies deal with integration of maps with semantic characteristics. A 3D volumetric semantic map via CRFs is constructed by Kundu et. al. [10]. They use a monocular camera and integrate the images by using the pose information of the camera only by visual SLAM. Hand-crafted features like color, histogram of oriented gradients (HOG), pixel location features and several filter banks are used. Similarly, Floros et. al. [11] use CRFs to obtain the semantic segmentation in 3D. Stereo images are used to estimate the depth and the static structures are integrated via Kalman filtering onto the map. 3D reconstruction is used to couple individual image segmentations. A model-free method is used to compute 3D semantic segmentations by Tung et. al. [12]. They combine the region proposal network output [13], which outputs the features for a region and depth-based descriptors of that region obtained from stereo image pairs. Babahajiani et. al. [14] use a rule-based strategy followed by a boosted decision tree detector. First, the road surfaces and building facades are detected and excluded from the point cloud, then the voxels are classified with a classifier. A strong *a priori* assumption about the environment is made which is not always feasible. The common point among these approaches is that they use 3D reconstruction which is computationally expensive. Instead, some methods construct a 2D semantic map. Dequaire et. al. [15] discover the semantic properties of the grids by using recurrent neural networks. RGB information is not used; therefore, the approach cannot discriminate between classes like road and sidewalks where the laser data is similar. Giovani et. al. [16] builds an evidential perception grid by using segmentation maps obtained via texton maps and dispton maps. The performance of the semantic segmentation is not superior to recent methods.

Semantic segmentation of RGB images is an important part of our method. Various studies have been made in this field. Early work is based on flat classifiers like Support Vector Machines [17], random forests [18] and boosting [19] processing hand-crafted features. However, deep learning methods improved the performance significantly. An early work by Long et. al. [20] showed that a fully convolutional network can give good results in segmentation. However, the feature maps have a low resolution. To overcome this low resolution problem, encoder-decoder networks were proposed. SegNet [21] or U-Net [22] are two successful examples of such segmentation networks. Networks which are combined with Conditional Random Fields (CRFs) are developed to improve the the errors at the boundaries of the class regions ([23]) at the cost of higher computation time. Grid Networks [24] model segmentation as a flow through a Grid, where each path corresponds to a specific choice of neural network with its own change of resolution during the forward pass.

Multiple Object Tracking (MOT) is an alternative approach to detect objects in motion (but not the static environment) [25], [26]. Although these approaches may provide accurate results for certain classes of objects, they make assumptions regarding the shapes or semantic characteristics of the objects which may not always be correct and may require expert knowledge. Some MOT approaches that do not use such assumptions need segmentation and classification steps [27], [28] to detect the objects which would require additional complexity.

III. Semantic Grid Estimation

The objective of our work is to fuse 3D point cloud input $\boldsymbol{l}=\{\boldsymbol{l}_t\}$ from a LIDAR with 2D image input $\boldsymbol{x}=\{\boldsymbol{x}_t\}$ from an RGB camera in order to obtain a semantic grid $\boldsymbol{y}=\{\boldsymbol{y}_t\}$ from a bird's eye view centered on the car. All representations

are indexed by time t, but this index will be dropped for convenience if time is not required.

If required, additional indices x and y index 2D positions in the representations: $\boldsymbol{x}_{t,x,y}$ (or $\boldsymbol{x}_{x,y}$) gives the value of position (x, y) at time t in the RGB input. Each value $\boldsymbol{s}_{t,x,y}$ of a segmented image can take values in the alphabet Λ (C different semantic classes), whereas each grid value $\boldsymbol{o}_{t,x,y}$ of the occupancy grid can take values in the alphabet Ω (see section III-B).

An overview of the hybrid Bayesian/Deep method can be seen in Fig. 2. The LIDAR input $\boldsymbol{l}$ is geometrically and temporally integrated with a Bayesian particle filter to produce an occupancy grid $\boldsymbol{o}=\{\boldsymbol{o}_t\}$. This grid has the advantage of holding a 360° view of the environment and of encoding the full LIDAR observation history through temporal integration. However, its semantic information is low, as the states of each cell encode possible values $\{empty, occupied, in{-}motion\}$ but do not contain fine-grained information on the semantic classes of occupied cells. This information is available from the output $\boldsymbol{s}$ of a deep network taking as input the RGB image $\boldsymbol{x}$. However, the two representations are of different types (3D projective geometry vs. bird's eye view) and need to be fused.

In the following sub sections, we will describe the different parts of this process: i) semantic segmentation of the RGB input image, ii) occupancy grid estimation from LIDAR input, and iii) learned fusion of these two representations with a deep neural network.

A. Semantic Segmentation from Monocular RGB Input

A deep neural network takes monocular RGB input $\boldsymbol{x}$ and predicts an estimate of the segmentation $\boldsymbol{s}$. High capacity neural networks are capable of estimating high-level semantic information from low-level input data provided that they are trained on sufficiently large databases. We follow the standard practice of pre-training the model first for classification on a large-scale dataset like ImageNet/ILSVRC [6] followed by fine-tuning on our target datasets.

The main challenge in image segmentation using deep networks is to preserve resolution, i.e. to provide segmentation at the same spatial resolution as the input. The difficulty lies in the fact that, internally, convolutional neural networks perform downsampling and pooling between successive layers in order to increase the spatial support of convolutional filters in later layers. Preserving resolution can be solved using the *à trous* algorithm [29], or through upsampling, which was first introduced in [20] and then refined with switch variables in [30] and with skip connections in [22].

We use the SegNet variant of these techniques [21] for obtaining semantic information $\boldsymbol{s}$. The accuracy of SegNet is not the highest among other reported results [31], but its runtime/accuracy trade-off is very favorable. As [20], [30], [22], SegNet is an encoder-decoder network.

We use the parameters from a previously trained version with a VGG16 [32] architecture trained for object recognition. The pixels are classified by using a soft-max layer. The labels are *road, car, sideways, vegetation, pedestrian, bicycle, building, signage, fence* and *unknown*.

B. Bayesian Occupancy Grids

Bayesian Filter methods are used to predict the occupancy and dynamic state of the cells in an occupancy grid map. This is achieved via deducting the occupancy probability given the sensor measurements for each cell and the previous states of the cell. A detailed mathematical formalization of the problem can be found in [1], here we will mainly summarize the steps to obtain the probability by using CMCDOT, which is one of the recent Bayesian Filter methods to deduct the probabilities of the grid occupancies and dynamics.

First of all, instead of defining each cell as being occupied or not, CMCDOT [1] introduces free, statically occupied, dynamically occupied and unknown states. Free state represents the probability of a cell being free of any type of object. However, it should be noted that it does not necessarily imply the area where the vehicle can navigate to. For example, a sideways has a high probability of being free, but a vehicle should not be allowed to drive into this area. Statically occupied cell refers to the probability of a cell being occupied by an obstacle. Although it may be part of a permanently static structure like a building, it can also be part of a temporary static object, like a parked car. The dynamically occupied cells show the probability of the cell being occupied by a dynamic object. This kind of state also includes information about the velocity of the cell which are contained in the particles in that cell. Only the particles in this dynamic cell regions have velocity related information, which reduces the computation complexity significantly.

The first step of a Bayesian filter is to predict the current state probabilities from the previous states. To be able to achieve this, we define transition probabilities. They represent the probability of the transition of a state in the previous time into another state in the current time. For example a statically occupied cell will remain as a statically occupied cell with a transition probability of 0.99, or it will become a dynamically occupied cell with a transition probability of 0.01. The details of transition for prediction can be found in [1].

In the next step, we evaluate the updated probabilities. A probabilistic sensor model [33] is used for the observation model. It should be noted that due to the flexibility of the observation model, the Bayesian Filter can be used with a wide variety of sensors which is one of the main advantages. In this study, we use LIDAR sensor data $\boldsymbol{l}$. After the evaluation step, the state distributions are estimated. A particle re-sampling is applied to assign new particles to new dynamically occupied regions and to focus on significant regions. After particle re-sampling, the iteration continues with prediction step if it is necessary.

Although the occupancy and dynamic information about the state of the cells can be obtained accurately via a Bayesian Filter method, this cannot always be used to plan the next action of the vehicle. As aforementioned, the same state may refer to different semantic classes. For example a

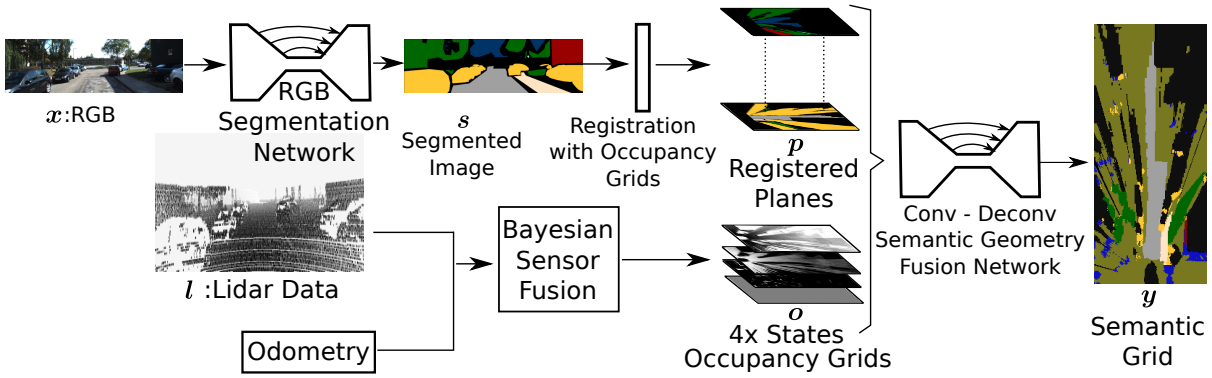


Fig. 2: Functional overview of the method. The four states of the occupancy grids represent the free cells, statically occupied cells, dynamically occupied cells and unknown cell. $\boldsymbol{x}$: RGB Image, $\boldsymbol{l}$: LIDAR data, $\boldsymbol{o}$: occupancy grids, $\boldsymbol{s}$: segmented image, $\boldsymbol{p}$: registered planes as inner representations, $\boldsymbol{y}$: semantic grid

free cell can be a road or a sideways which would result in different behavior. Therefore, we enrich the occupancy grids with the semantic information.

C. Fusing Geometry and Semantic Information

Fusing the geometric information from the occupancy grid $\boldsymbol{o}$ obtained from LIDAR input $\boldsymbol{l}$ and the semantic information $\boldsymbol{s}$ obtained from the projective RGB input $\boldsymbol{x}$ is a non trivial problem, which cannot be solved with direct geometric computations, unless depth information is available together with the RGB input: one point in projective images $\boldsymbol{x}$ or $\boldsymbol{s}$ can potentially correspond to several 3D points in the scene and therefore to several points in the grid $\boldsymbol{o}$. Depth from stereo could provide sufficient constraints. We propose an alternative solution, where this fusion process is learned, avoiding the calculation of disparity maps from stereo.

Directly learning this fusion in a black box strategy by providing both representations as an input to a neural network would be sub-optimal. The network would need to learn multiple mappings, namely (i) the geometric transformation between the projective view and the bird's eye view, (ii) solving the point-to-line correspondence due to missing depth information, and (iii) fusing occupancy grid states with semantic classes from the segmentation. Whereas (ii) and (iii) are inevitable in our approach, (i) can be solved (up to the point-to-line correspondence) directly.

For this reason, we introduce an intermediate representation $\boldsymbol{p}$, which transforms the projective input $\boldsymbol{s}$ to a bird's eye view representation compatible with the coordinate system of the occupancy grid $\boldsymbol{o}$ (Fig. 3). Each cell in $\boldsymbol{p}$ takes values from the alphabet Λ of semantic classes, i.e. the same alphabet used for segmented images $\boldsymbol{s}$. The objective is to create a representation, where a cell with given spatial coordinates (x, y) corresponds to a grid cell in $\boldsymbol{o}$ with the same coordinates. To model the ambiguity resulting from the missing depth information, $\boldsymbol{p}$=$\{\boldsymbol{p}_i\}$ is a layered 3D map organized as a collection of D planes $\boldsymbol{p}_i$ which are parallel to the ground. Here, i indexes different heights. In what follows, the time index t has been dropped, as fusion is calculated separately for each time instant t.

The distance between the ground and each plane p_i is d_i. We assume D planes with distance δd to each other, therefore the distance of i^{th} plane to ground is $i\delta d$. The relation between an image and a plane is straightforward if the camera is calibrated and can be derived from projective geometry. For each of the planes p_i, for any point $\{x_i^j, y_i^j, z_i^j\}$ on this plane, first we find the coordinates of the point in the image plane by using a transformation ${}^p_I tf$ from plane to image coordinates $\left(\hat{x}_i^j, \hat{y}_i^j, \hat{z}_i^j, 1\right)^\intercal = {}^p_I tf \left(x_i^j, y_i^j, z_i^j, 1\right)^\intercal$, the corresponding pixel in the image plane can be found as

$$\begin{pmatrix} px_i^j \\ py_i^j \\ 1 \end{pmatrix} = K \begin{pmatrix} \hat{x}_i^j / \hat{z}_i^j \\ \hat{y}_i^j / \hat{z}_i^j \\ 1 \end{pmatrix} \tag{1}$$

where K is the camera calibration matrix. px_i^j and py_i^j are the pixel coordinates of the projection of the j^{th} point in the i^{th} plane. For a given pixel in the segmented image $\boldsymbol{s}$, we assign its semantic label to a set of points in the representation $\boldsymbol{p}$, each of which corresponds to a different depth value (and therefore to a different height value).

As stated above, the spatial coordinates of representations $\boldsymbol{p}$ and $\boldsymbol{o}$ are compatible. The objective is to train a learned mapping which integrates the two representations into a segmented occupancy grid $\boldsymbol{o}$. The underlying assumption is that objects with height $h < D$ are visible in the RGB image and are in the limits of the occupancy grid. Then, if δd is small enough, at least one of the points in the projected planes $\boldsymbol{p}$ will have the correct label of the object, and the learned mapping can pick up the integration with the occupancy grid.

D. Joint Dimensionality Reduction and Fusion

The occupancy grid $\boldsymbol{o}$ and the intermediate representation $\boldsymbol{p}$ are both fed into a deep neural network, which outputs the semantic grid $\boldsymbol{y}$. The semantic grid is a 3D tensor of the same spatial dimensions as the input tensors $\boldsymbol{o}$ and $\boldsymbol{p}$. The model therefore needs to be resolution preserving, which is a non-trivial task if the intermediate layers of the network use any sort resolution reducing pooling. As also done for

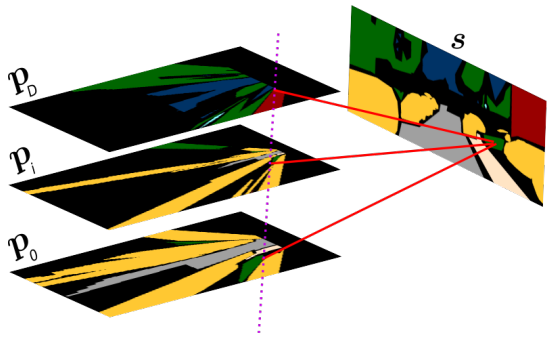


Fig. 3: Alignment of projective view to bird's eye view (only three planes are shown). A single point in the projective view corresponds to a set of points in the bird's eye view lying on a straight line.

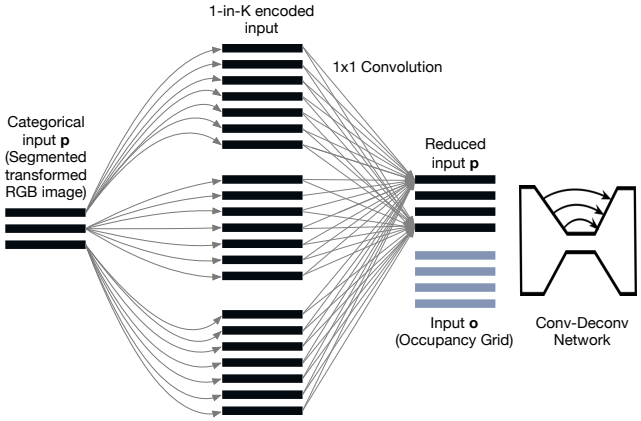


Fig. 4: The categorical semantic input representation is encoded into a 1-to-K space (per height plane), which is then reduced back to a lower dimensional tensor before being concatenated with the occupancy grid.

semantic segmentation (Section III-A), we resort to Conv-Deconv networks [30] in the SegNet variant [22] including skip connections originally introduced in [21].

The input data consist of $D+4$ planes: 4 planes corresponding to the occupancy grid $\boldsymbol{o}$ and D planes corresponding to the intermediate representation $\boldsymbol{p}$ calculated from the segmented RGB image. The latter is categorical data, as each value from the alphabet Λ corresponds to a semantic class label. The data is by definition unordered, which makes learning from it directly inefficient, as the learned mapping needs to be highly (and unnecessarily) complex. We therefore increase the number of data planes by encoding each of the D planes from $\boldsymbol{p}$ in a 1-in-K encoding ("*hot-one*"). This creates well-behaved data with the drawback of significantly increasing the size of the input representation: the input tensor has now $D\times C+4$ planes, where C is the number of semantic classes.

Directly learning this representation is computationally inefficient and results in a network of an unreasonable large capacity. We add a dimensional reduction layer at the input of the encoder, which, as a first operation, reduces the input planes to a lower number. The number of planes is a hyper-parameter, which we set equal to C. This is implemented as 1×1 convolutions, which have the same effect as a pointwise non-linearity with spatially shared parameters. Dimensionality reduction and fusion are trained jointly, end-to-end. This is shown in Fig. 4.

The rest of the encoder part has 13 convolution layers similar to the well-known VGG16 network [32]. Each encoder layer applies a convolution and obtains a set of features. These are then batch-normalized and an element-wise rectified-linear non-linearity (ReLU) is performed. Max-pooling is used to reduce the the size of the images. The indices of the maximum values are kept and delivered to the decoder process which accelerates the decoding process [30]. The decoder has 13 layers. Again a convolution is performed with a batch-normalization and ReLU. To upsample the images, the indices stored in the encoder layers are used. A multi-class softmax classifier is applied in the final layer to estimate the class probabilities of the pixels. Cross-entropy loss [20] is used and the loss is taken as the sum of all the pixels in a batch.

IV. Experiments

To our knowledge, no dataset exists which directly labels the bird's eye view semantically from moving vehicles. We therefore use two different datasets and transform the data to our desired representation:

The KITTI Dataset was introduced in [34]. We use a variant which was semantically annotated by Zhang et. al. [35]. It has 10 class labels, 252 labeled images, corresponding laser scan data, vehicle motion information and camera calibration parameters. We use 142 images for training, or, to be more precise, for fine-tuning after pre-training on ILSVRC/Imagenet [6].

The INRIA-Chroma Semantic Grid Dataset is our own dataset, which was collected with our experimental vehicle equipped with LIDAR, IMU, odometry and stereo cameras. It consists of 657 images and laser range sensor data. For this dataset, we annotated 276 images, of which we use 146 images for training.

Groundtruth Preprocessing — To obtain the groundtruth of the KITTI dataset, we use the images acquired from the frontal view camera labeled by a human user. First, we transform the points obtained by the LIDAR into the RGB camera coordinate frame. After this transformation, some of the pixels are assigned with at least one depth value. For the pixels where there exists more than one depth value, the closer depth value is selected. It should be noted that some of the pixels will not have a depth value due to non-overlapping regions or the sparseness of the point cloud. After assigning depth values to some of the pixel values, these points are transformed into the occupancy grid coordinate frame. However, due to sparseness of laser data and imperfections in the labeling, this transformation is not perfect and contains some limited noise. We apply

some morphological techniques to increase the quality of the labels. Details are beyond the scope of this study.

For the remaining unlabeled grid points, we check the corresponding grid state probabilities in KITTI dataset by computing the occupancy grid states with the CMCDOT approach. If one of the state values at that cell is higher than the remaining states and a threshold, that grid is labeled with the corresponding state, i.e. free, statically occupied or dynamically occupied labels are added to the bird's eye view where appropriate. In this way, we preserve information from the occupancy grids for unlabeled grid cells which have a high certainty of one of the formal grid states.

To obtain the groundtruth of the INRIA-Chroma dataset, we have converted the RGB images into bird's eye view images and then labeled them in the bird's eye view. Since the labeling is made by a human user in the bird's eye view, the additional labels which correspond to the state of the occupancy grid are not added to this dataset.

Bayesian Occupancy Grid Estimation — To obtain the occupancy grid formal states, we use CMCDOT [1]. We set its width to 31 m, length to 71 m and the grid size to 0.2×0.2 m.

Training the Segmentation Network — We started from a pre-trained network ILSVRC/Imagenet [6] for classification and finetune it on our own data for segmentation. We use a learning rate of 1×10^{-5} and momentum of 0.9.

Training the Fusion Network — Since the datasets are not balanced (the numbers of pixels may be highly different over semantic classes), less frequent classes may be ignored by learning (learned bias). We handle this situation with *median frequency balancing* [36]. For each class, the frequency $freq(c)$ is obtained by dividing the number of pixels of that class by all the pixels in the image if the class is available in the image. med_f is the median of these frequencies. Then, each class is weighted by a class coefficient $\alpha_c = med_f / freq(c)$ in the cross-entropy loss. In unbalanced case, all the weights are selected to be identical.

We use a learning rate of 1×10^{-4} and momentum of 0.9, a mini-batch size of 10, which means that approximately every 14 epochs correspond to one full pass over the training set. We train it with 5000 iterations of optimization until the loss converges. We select $D = 20$ planes and $C = 14$ classes.

We compare the performance when the parameters are taken from a pre-trained network for classification ILSVRC/Imagenet [6] and when the parameters have random initial values. We also compare the performance when the semantic input $\boldsymbol{s}$ consists of categorical data (D+4 planes) to the situation where the input $\boldsymbol{s}$ consists of 1-in-K encodings followed by dimensionality reduction (DR), resulting in $D \times C$+4 planes (Table I).

Furthermore, we also compare the cases where we use only semantically segmented images $\boldsymbol{s}$ and occupancy grid data $\boldsymbol{o}$ for training (Table II).

End-to-end training — In addition, we also compare the performance when training is in an end-to-end manner (Table II). However, due to memory constraints, we reduce the encoder and decoder parts to 5 layers in the fusion network.

TABLE I: Results on the KITTI dataset with different input encodings, pre-training and balancing strategies (DR=Dimensionality Reduction).

Input ($\boldsymbol{s}$) Encoding	Pre-Training	Balanced	Pixel Acc.	Class Acc.	FmIoU
Categorical	X	X	72.1	40.3	60.6
Categorical		X	67.9	34.2	56.0
Categorical			67.3	21.9	51.8
Categorical	X		**73.7**	25.8	59.1
1-in-K +DR		X	71.4	34.6	57.2
1-in-K +DR	X	X	72.7	**41.7**	**61.8**

Results and Discussion — We analyze the input types, using parameters from a pretrained network, different losses and finally using only occupancy grids or semantically segmented images as available data.

The measures that we use for evaluating the results consists of three values, which are commonly used in semantic segmentation: pixel accuracy is the proportion of correctly classified pixels; class accuracy is the average accuracy over all the classes, and pixel-frequency weighted average Jaccard index is the mean of intersection over union based on frequency (also called **FmIoU**). We evaluate the results over the semantic grids, which are the semantically segmented bird's eye view grids.

Pre-training the networks on large-scale datasets for image classification increases the performance of the network as it can be seen in Table I. This is a classical effect. Although the parameters are taken from a network trained on a different task, the performance increases, which is probably due to the fact that the early layers of the network learn general filters.

When we include a loss for data balancing, we can observe an increase in accuracy. Class balancing increases the accuracy due to small static and dynamic objects in the environment. In particular, we frequently encounter small regions in the bird's eye view, which tend to be neglected without class balancing. This explains the high global accuracy accompanied by a low mean class accuracy for the unbalanced approach.

The effect of using different input types can be seen in Table I. The best performance belongs to the 1-in-K encoding approach, further confirming the interest of a well-behaved vector-space encoding.

Next, we compared our approach to baselines, which do not fuse the two different sources of information (LIDAR $\boldsymbol{o}$ and segmented RGB $\boldsymbol{s}$). For these tests, we chose the configuration which performed best: pre-training, class balancing, and 1-in-K encoding with dimensionality reduction (DR). We included end-to-end trained model to compare its performance. Surprisingly, the pixel accuracy is even higher in the baseline where only the occupancy grid $\boldsymbol{o}$ is available. In this case, the occupancy grid output is mapped to the semantic grid through the fusion network, which needs to infer semantic classes from the 4 states of the occupancy grid only (*empty, occupied, in-motion, unknown*). This works surprisingly well for the majority of the pixels, which is a

TABLE II: Results on KITTI dataset, comparing the proposed hybrid approach with baselines using one of the two data sources only (LIDAR or RGB).

Data sources Used	Pixel Acc.	Class Acc.	FmIoU
LIDAR only ($\boldsymbol{o}$)	74.4	38.3	62.1
RGB only ($\boldsymbol{s}$)	50.2	19.6	35.6
LIDAR and RGB ($\boldsymbol{o}$, $\boldsymbol{s}$)	72.7	41.7	61.8
LIDAR and RGB ($\boldsymbol{o}$, $\boldsymbol{s}$) (end-to-end)	**81.0**	**49.4**	**69.8**

TABLE III: Results on the INRIA-Chroma-Semantic Grid Dataset

Input ($\boldsymbol{s}$) Encoding	Pre-Training	Balanced	Pixel Acc.	Class Acc.	FmIoU
Categorical	X	X	**79.5**	33.2	66.4
1-in-K + DR	X	X	79.1	34.1	**66.5**
End-to-end	X	X	78.7	**35.2**	65.2

sign of quality of the occupancy information produced by CMCDOT [1]. However, this approach is forced to learn a high amount of "prior" information in a sense that class labels are not available and will be inferred through absolute pixel positions and context. This works less well for smaller objects and less represented classes, as confirmed by the much lower class accuracy and FmIoU. The occupancy grid alone is not sufficient to predict the different semantic classes which can only be detected by a camera. The proposed fusion technique clearly outperforms the LIDAR only approach in this important measure when trained in an end-to-end manner.

Results with the segmented image $\boldsymbol{s}$ alone are clearly inferior. This supports the idea that the semantic grid fusion method depends much more on the occupancy grid structure than on the segmented image. Moreover, the occupancy grid is a result of temporal integration through Bayesian filtering, whereas the segmented image $\boldsymbol{s}$ corresponds to observation of the current time frame only.

Finally, we test our network on a new dataset. We select two network approaches with different input types for comparison. The labeled pixels are less in number in this dataset. There are also fewer class types since we don't have free, statically occupied and dynamically occupied labels. The results are not significantly different for both of the approaches (Table III). They are inferior in terms of class accuracy with respect to KITTI dataset, probably due to lack of labeling density. They are superior in global accuracy probably due to fewer number of classes.

The visualization of the results can be seen in Fig.5. **(a)**, **(e)**, **(i)** show the segmentation made by a human user overlayed with the RGB image. **(b)**, **(f)**, **(j)** show the groundtruth for the bird's eye view. **(c)**, **(g)**, **(k)** show the prediction of semantic segmentation in frontal view and **(d)**, **(h)**, **(l)** show the semantic grid estimation of our method with the configuration which performed best in Table I. The semantic grid (SG) estimation generally estimates a road in the middle of the grid, which is usually correct. If the class is estimated correctly in the semantic view, it is also correctly estimated in the SG, i.e. Fig.5-**(d)** and **(l)** have a car in front of our vehicle which is estimated in the semantic view images. On the other hand, if the semantic segmentation fails in detecting a class, then the corresponding SG may also fail to represent that class. For example Fig.5-**(g)** cannot classify the car correctly, and Fig.5-**(h)** cannot represent the car in its view. Another point is that hallucinations in the semantic view can be eliminated in the SG view. For example, Fig.5-**(k)** hallucinates cars on the sideways; however, since the network does not learn to produce cars on the sideways, it does not represent cars on the sideways in Fig.5-**(l)**.

V. Conclusion

We have proposed a hybrid method including Bayesian particle filtering and deep neural networks for the estimation of semantic occupancy grids from LIDAR, odometry and monocular RGB video. Our method uses deep learning for data fusion and directly models registration of the projective RGB images to the bird's eye representation of the occupancy grid. Dimensionality reduction increases efficiency and avoids overfitting. The method has been tested on two datasets, the KITTI Dataset [34] and a dataset we acquired using our own vehicle platform and annotated manually.

Future (and ongoing) work will address direct semantic segmentation in the 3D point cloud with integration of these labels with the labels obtained from the semantic image segmentation. We also work on 3D object detection from the obtained semantic grid through direct regression of 3D bounding boxes using deep neural networks.

Acknowledgement

We thank Nicolas Vignard, Jean-Alix David and Jérôme Lussereau for their assistance with the experimental vehicle during the data collection.

References

[1] L. Rummelhard, A. Nègre, and C. Laugier, "Conditional monte carlo dense occupancy tracker," in *ITSC*. IEEE, 2015, pp. 2485–2490.

[2] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

[3] H. P. Moravec, "Sensor fusion in certainty grids for mobile robots," *AI magazine*, vol. 9, no. 2, p. 61, 1988.

[4] M. Perrollaz, J.-D. Yoder, A. Spalanzani, and C. Laugier, "Using the disparity space to compute occupancy grids from stereo-vision," in *IROS*. IEEE, 2010, pp. 2721–2726.

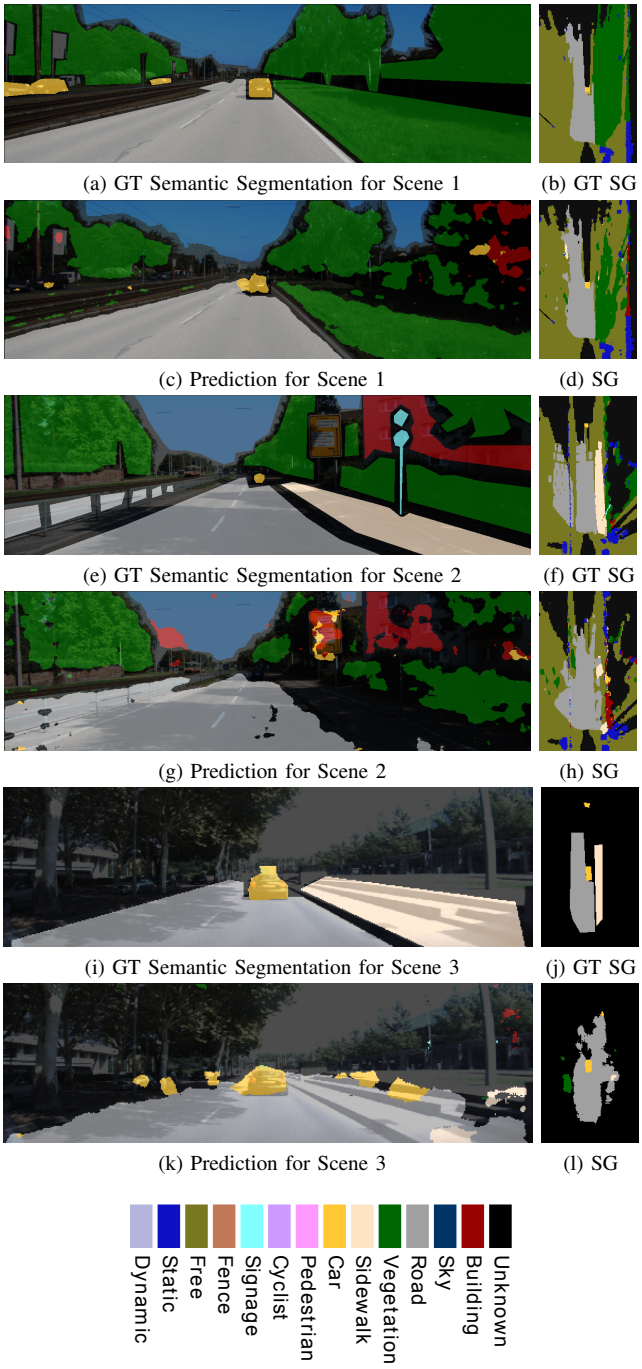
[5] J. D. Adarve, M. Perrollaz, A. Makris, and C. Laugier, "Computing occupancy grids from multiple sensors using linear opinion pools," in *ICRA*. IEEE, 2012, pp. 4074–4079.

[6] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *IJCV*, vol. 115, no. 3, pp. 211–252, 2015.

[7] C. Coué, C. Pradalier, C. Laugier, T. Fraichard, and P. Bessière, "Bayesian occupancy filtering for multitarget tracking: an automotive application," *The International Journal of Robotics Research*, vol. 25, no. 1, pp. 19–30, 2006.

[8] T. Gindele, S. Brechtel, J. Schroder, and R. Dillmann, "Bayesian occupancy grid filter for dynamic environments using prior map knowledge," in *Intelligent Vehicles Symposium*. IEEE, 2009, pp. 669–676.

[9] A. Nègre, L. Rummelhard, and C. Laugier, "Hybrid sampling bayesian occupancy filter," in *Intelligent Vehicles Symposium*. IEEE, 2014, pp. 1307–1312.



(a) GT Semantic Segmentation for Scene 1 (b) GT SG

(c) Prediction for Scene 1 (d) SG

(e) GT Semantic Segmentation for Scene 2 (f) GT SG

(g) Prediction for Scene 2 (h) SG

(i) GT Semantic Segmentation for Scene 3 (j) GT SG

(k) Prediction for Scene 3 (l) SG

(m) Colors for labels

Fig. 5: Three scenes with ground truth (GT), semantic segmentation predictions and semantic grids (SG). Scene 1 & 2 are from KITTI dataset, whereas Scene 3 is from Inria-Chroma dataset. The legends for the labels are given at the bottom.

[10] A. Kundu, Y. Li, F. Dellaert, F. Li, and J. M. Rehg, "Joint Semantic Segmentation and 3D Reconstruction from Monocular Video," in *ECCV*, 2014, pp. 703–718.

[11] G. Floros and B. Leibe, "Joint 2D-3D temporally consistent semantic segmentation of street scenes," in *CVPR*, 2012, pp. 2823–2830.

[12] F. Tung and J. J. Little, "MF3D: Model-free 3D semantic scene parsing," in *ICRA*, 2017, pp. 4596–4603.

[13] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[14] P. Babahajiani, L. Fan, J. K. Kämäräinen, and M. Gabbouj, "Urban 3D segmentation and modelling from street view images and LiDAR point clouds," *Machine Vision and Applications*, vol. 28, no. 7, pp. 1–16, 2017.

[15] J. Dequaire, P. Ondrúška, D. Rao, D. Wang, and I. Posner, "Deep tracking in the wild: End-to-end tracking using recurrent neural networks," *The International Journal of Robotics Research*, p. 0278364917710543, 2017.

[16] B. V. Giovani, A. C. Victorino, and J. V. Ferreira, "Stereo vision for dynamic urban environment perception using semantic context in evidential grid," in *ITSC*. IEEE, 2015, pp. 2471–2476.

[17] B. Fulkerson, A. Vedaldi, and S. Soatto, "Class segmentation and object localization with superpixel neighborhoods," in *ICCV*, 2009.

[18] M. J. Shotton and R. Cipolla, "Semantic texton forests for image categorization and segmentation," in *CVPR*, 2008.

[19] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context," *IJCV*, 2009.

[20] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *CVPR*, 2015.

[21] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE PAMI*, vol. 39, no. 12, pp. 2481–2495, 2017.

[22] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *MICCAI*, 2015.

[23] L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE PAMI*, vol. PP, no. 99, pp. 1–1, 2017.

[24] D. Fourure, R. Emonet, E. Fromont, D. Muselet, A. Tremeau, and C. Wolf, "Residual Conv-Deconv Grid Network for Semantic Segmentation," in *BMVC*, 2017.

[25] T. E. Fortmann, Y. Bar-Shalom, and M. Scheffe, "Multi-target tracking using joint probabilistic data association," in *Decision and Control including the Symposium on Adaptive Processes, 19th IEEE Conference on*. IEEE, 1980, pp. 807–812.

[26] D. Z. Wang, I. Posner, and P. Newman, "Model-free detection and tracking of dynamic objects with 2d lidar," *The International Journal of Robotics Research*, vol. 34, no. 7, pp. 1039–1063, 2015.

[27] K. O. Arras, O. M. Mozos, and W. Burgard, "Using boosted features for the detection of people in 2d range data," in *ICRA*. IEEE, 2007, pp. 3402–3407.

[28] A. Petrovskaya and S. Thrun, "Model based vehicle detection and tracking for autonomous urban driving," *Autonomous Robots*, vol. 26, no. 2-3, pp. 123–139, 2009.

[29] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *International Conference on Learning Representations (ICLR)*, 2016.

[30] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *ICCV*, 2015, pp. 1520–1528.

[31] G. Lin, C. Shen, A. Van Den Hengel, and I. Reid, "Exploring context with deep structured models for semantic segmentation," *IEEE PAMI*, 2017.

[32] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[33] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.

[34] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *CVPR*, 2012.

[35] R. Zhang, S. A. Candra, K. Vetter, and A. Zakhor, "Sensor Fusion for Semantic Segmentation of Urban Scenes," in *ICRA*, 2015, pp. 1850–1857.

[36] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *ICCV*, 2015, pp. 2650–2658.