2019 IEEE Intelligent Transportation Systems Conference (ITSC)
Auckland, NZ, October 27-30, 2019

6DoF Vehicle Pose Estimation Using Segmentation-Based Part Correspondences

Thomas Barowski[1], Magdalena Szczot[1] and Sebastian Houben[2]

***Abstract*— Over the last years, pixel-wise analysis of semantic segmentation was established as a powerful method in scene understanding for autonomous driving, providing classification and 2D shape estimation even with monocular camera systems. Despite this positive resonance, a way to take advantage of this representation for the extraction of 3D information solely from a single-shot RGB image has never been presented.**
In this paper we present a full-fledged six degree-of-freedom vehicle pose estimation algorithm, demonstrating that a segmentation representation can be utilized to extract precise 3D information for non-ego vehicles. We train a neural network to predict a multi-class mask from segmentation, defining classes based on mechanical parts and relative part positions, treating different entities of a part as separate classes. The multi-class mask is transformed to a variable set of key points, serving as a set of 2D-3D correspondences for a Point-n-Perspective-solver. Our paper shows not only promising results for 3D vehicle pose estimation on a publicly available dataset but also exemplifies the high potential of the representation for vehicle state analysis. We present detailed insight on network configuration as well as correspondence calculation and their effect on the quality of the estimated vehicle pose.

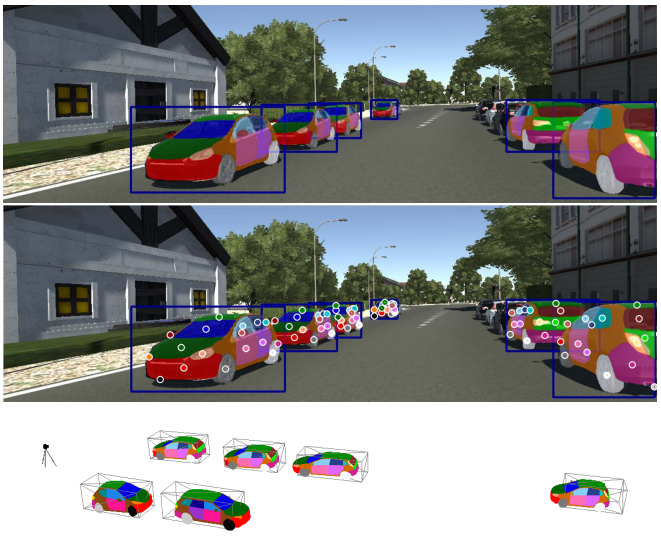

Fig. 1: Exemplary result from the presented 6DoF pose estimation pipeline. *Top*: Prediction of the multi-class Mask-RCNN on the Virtual KITTI dataset [1]. Our vehicle detector outputs part-level segmentation masks. *Middle*: For each detection, part correspondences (colored points) are generated from the segmentation mask and passed to a PnP-solver. *Bottom*: Resulting 3D Visualization of the 6DoF vehicle poses, black boxes represent the ground truth.

I. Introduction

To act in a permanently changing environment, an autonomous vehicle requires an extensive understanding of the surrounding scene and a precise perception of other traffic participants, mainly non-ego vehicles. Over the last years semantic segmentation was established as a powerful method to extract large amounts of information in a holistic manner by pixel-wise image analysis, applicable even on monocular camera systems. Despite this positive resonance, the obtained representation was rarely used for 3D object detection: Applications either remained in the two-dimensional domain or were related to point cloud algorithms requiring stereo or LIDAR setups. In contrast to this, we want to show that semantic segmentation is capable of more than 2D analysis, even in the challenging circumstances of a monocular setup: By describing a vehicle in a dense, spatially precise manner, it provides an information-rich baseline that can be utilized by pose estimation algorithms. Therefore our research focuses on a proof-of-concept of how 3D object information can be extracted from the semantic segmentation of a single image, i.e., the estimation of the size and six degree-of-freedom (6DoF) pose of non-ego vehicles. This is motivated by the following facts: Firstly, future use cases in autonomous driving will require a deeper understanding of vehicles, their dynamic structure and state, e.g., the analysis of dynamic parts, wheels and indicators. Semantic segmentation, as a spatially precise analysis will help providing the required information. Secondly, the structure of semantic segmentation implies a strict limitation to visible features, providing an interpretable input for subsequent algorithms. Lastly, establishing such a pipeline creates a path redundant to current bounding box algorithm designs, which is required for certain automotive security verification standards.

Relying on monocular single-shot object detection, simultaneously estimating the size and pose of an object solely based on geometrical considerations is not possible. To overcome this limitation, model-based approaches provide an efficient solution by limiting the problem to the pose estimation by adding knowledge about the size of the object. Research showed that this assumption is valid for rigid objects like cars, providing good results using Perspective-n-Point (PnP) fitting [2], in which the vehicle's pose is estimated using 2D-3D correspondences to solve a projection-based optimization problem.

Unfortunately, the current semantic segmentation output does not directly provide usable correspondences, delineating

[1]Thomas Barowski and Magdalena Szczot are with BMW AG, Munich, Germany firstname.lastname@bmw.de
[2]Sebastian Houben is with the Institute for Neural Computation, University of Bochum, Germany sebastian.houben@ini.rub.de

vehicles as filled silhouettes which lack detailed information: Not only do silhouettes underlie a high variance w.r.t. occlusions and truncations, even under good conditions the axial symmetry of cars remains a problem. Therefore, to apply model-based pose estimation approaches, better representations need to be found.

In our earlier work [3] we investigated this problem and found a solution by defining a fine-grained vehicle representation, which defines sub-classes for mechanical parts of a vehicle, embedding strong pose information into a segmentation by treating multiple entities of the same mechanical part as different classes, e.g., different doors or wheels.

As a continuation of our work, in this paper we want to analyze how well this representation can be utilized to estimate a full 6DoF pose of a vehicle using a model-based PnP-fitting approach. To this end, we perform the following three steps: Firstly, we define and train a convolutional neural network[1] to predict our multi-class vehicle segmentation (Fig. 1, *top*). The annotation of said multi-class vehicle segmentation is extensive, leading us to a semi-automated labeling pipeline to easily generate segmentation masks for training. Secondly, we present two schemes to generate 2D-3D correspondences from the multi-class segmentation (Fig. 1, *middle*) respectively from annotated CAD models. Thirdly, a PnP-fitting pipeline is established which takes the correspondences as an input and calculates a 6DoF pose by optimizing the correspondence reprojection error (Fig. 1, *bottom*). The extent of the vehicle is obtained by choosing a suited model from the model database.

The evaluation of our newly approach must overcome several issues: There is no public object detection benchmark that provides both 3D object bounding boxes and semantic segmentation ground truth. Additionally, the multi-class segmentation has high requirements w.r.t. labeling quality, which is also not satisfied by object detection benchmarks. To still provide reasonable and reproducible results we perform experiments on the synthetic dataset Virtual KITTI [1]. The synthetic character enables us to perform this very early proof-of-concept experiments in a controlled environment. Our future work will focus on evaluating the approach on real-world data.

In summary, our contributions are:

- A new segmentation-based 6DoF vehicle pose estimation approach, which generates 2D-3D correspondences for a model-based PnP-solver
- Utilizing mask object detectors for detailed multi-class segmentation
- Evaluation of the approach on a publicly available dataset

[1]Although our goal is a pipeline for full image segmentation, we utilize a Mask-RCNN approach in this paper, which does not bring full redundancy from bounding box based methods. However, full image instance segmentation remains a challenging task which led us to too many problems, mostly caused by fractured object predictions. We see this paper as an intermediate step to focus on the pose estimation component of the pipeline by decoupling the pose estimation from the quality of an instance segmentation.

II. Related Work

In this paper we follow a highly experimental approach, estimating a 6DoF vehicle pose with correspondences solely from segmentation inputs. To the best of our knowledge, there is no published research in this direction. Thus, we provide a review of segmentation background and other vehicle pose estimation approaches, highlighting and discussing the underlying data representations. For pose estimation approaches we focus on monocular camera systems, which constitute only a small part among the methods for other sensor types in autonomous driving, mainly LIDAR and RADAR.

Pixel-wise segmentation as a bottom-up approach has a long history in computer vision. The core idea is to build an understanding of the whole scene from the smallest unit, a pixel. By classifying every pixel into a set of classes, different types of information like the surrounding, infrastructure or dynamic objects can be extracted from the image plane. In automotive context, semantic segmentation had a breakthrough by a combination of the function approximation power of deep learning [4] and the availability of large scale datasets [5], [6], [7]. Research interest focused either on a better generalization of feature extraction [8], [9], network design [10], [11] or exploitation of temporal data [12]. A major downside of the segmentation representation is that the silhouettes of different objects of the same class merge when spatially close to each other. A separation of objects needs to be done in an additional means, commonly done by the utilization of point clouds [13]. To predict instances of a class by network design is a challenging task, because the number of objects within the image is arbitrary and the successful neural network structures are designed with static in- and outputs. Solutions to this problem are recurrent networks [14], [15], [16] or network architectures which transform the input into a representation that can be clustered [17], [18]. These techniques provide promising visual results, but the structural downsides of the received segmentation representation remain: Firstly, the received amount of data is high and hard to compress hindering further processing. Secondly, the spatial arrangement is not robust against adding or removing data points. Finally, the current interpretation solely allows us to extract silhouettes, which conflict with the lateral and longitudinal symmetries of vehicles. These are the main causes for the current stagnation toward the extraction of 3D information. In this paper we provide solutions to the first and third problem: The amount of data cannot be reduced, but we increase the amount of information that is captured by adding classes and do not only identify components but also describe the structure of the vehicle.

Our approach is contrary to commonly applied object pose estimation algorithms that utilize bounding boxes as a core representation of objects. Popular concepts for object

detectors are one-stage [19], [20] or two-stage detectors [21], [22] that are deployed in various applications due to their strong deep learning baselines. We refer the reader to [23] for an extensive overview of the different architectures and their performances. The work of Mask-RCNN [24] builds a bridge between bounding box detectors and instance segmentation but does not bring this into relations with pose estimation. Instead a manifold of different techniques evolved to estimate object poses in a downstream fashion, ranging from coarse viewpoint sub-classification [25], [26] over geometric priors [27] to model-based approaches [2]. The work of [28] is one attempt to include segmentation outputs in an energy function but only on silhouette level.

Although deep learning affected all of these methods, a pure learning-based approach never outperformed all of the specifically tailored methods. The combination of a model-based approach built on a powerful intermediate representation (e.g., key points), which can be learned by a neural network, established the current state-of-the-art for monocular camera setups [2]. However, we want to discuss the fact that the static structure of the neural network leads to a prediction of all key points in every detection, even if simultaneous visibility is not possible. To circumvent this, additional visibility analysis is done by the authors of [2]. Our own experiments with these approaches showed error cases, in which the arrangement of key points was learned relatively to each other instead of relying on visual cues. By design, a segmentation approach bypasses this problem since it is not bound to a static number of correspondences.

III. Method

This section is divided as follows: First, we give an introduction to our multi-class vehicle segmentation and how we apply it to CAD models. Then, our neural network structure is presented along the corresponding data generation pipeline, which is required to automatically annonate ground truth in a large scale manner. In the final subsection we describe our new fitting algorithm as our core contribution. An overview of the two-stage pipeline is given in Fig. 3.

Fine-grained vehicle representation. In this paper we utilize a fine-grained vehicle representation which divides a car into several sub-classes C (e.g., *apron*, *door*) based on material, function and relative position. By this means, we achieve multiple goals: Foremost, pose information is embedded by defining single occurrence parts with fixed positions and treating different entities of the same part as different classes (e.g., *front left door* vs. *front right door*). At the same time, we separate movable parts from static ones and define function-specific parts, e.g., indicators. Full background on the vehicle representations is given in our earlier work [3]. In this work we use the fragmentation-level *full*, which consists of 27 classes, extended by two classes for the side mirrors.

A vehicle model is defined as a mesh quadruplet $M = (V, F, E, Z)$ with the model's vertices V, the model's faces F, the physical extent of the model $E = (E_W, E_H, E_L)$ and an additional class assignment vector $Z(f)$, which holds a class label from C for each face $f \in F$. Using this representation, the vehicle models and their respective part segmentations can easily be projected into camera images using a given calibration. An exemplary visualization of the models is given in Fig. 2.

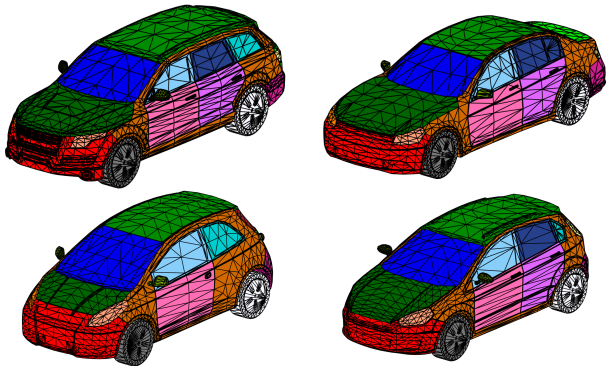

Fig. 2: Visualization of the multi-class vehicle representation annotated on four CAD models.

Multi-class mask prediction. The prediction of the multi-class vehicle representation on image level is done by a mask prediction network based on the framework Mask R-CNN introduced in [24].

The architecture of Mask R-CNN consists of the two-stage bounding box detector Faster R-CNN [22], which itself is divided into a Region Proposal Network (RPN) as first stage and the classification network Fast R-CNN [29] as second stage. The RPN generates promising object candidates, which are evaluated by the Fast R-CNN stage with classification and bounding box regression. In Mask R-CNN, this second stage is extended with an additional output head for pixel-wise instance mask prediction using the fully convolutional network architecture (FCN) [11].

To predict vehicles with our multi-class representation we restructure the outputs of the classification and mask branch while keeping the structure for bounding box regression: Firstly, we extend the mask class prediction from a singular foreground value to the prediction of a probability distribution between all sub-classes from the part segmentation. Secondly, we reduce the architecture the classification task to a binary foreground-background problem to only detect vehicles. Therefore the convolutional output kernel changes from Km^2 (with K number of object classes) as in [24] to $|C|Km^2 = |C|m^2$, with m representing the size of the mask kernel. Instead of treating all part masks individually with a sigmoid loss, we apply a softmax cross-entropy loss L_{mask} over all sub-class outputs for each position within the kernel. With added weighting and L_{cls} and L_{box} as defined in [21] the final loss of the second stage is defined as:

$$L = \alpha_{\text{cls}} L_{\text{cls}} + \alpha_{\text{box}} L_{\text{box}} + \alpha_{\text{mask}} L_{\text{mask}}$$

Network training and data generation. To train the de-

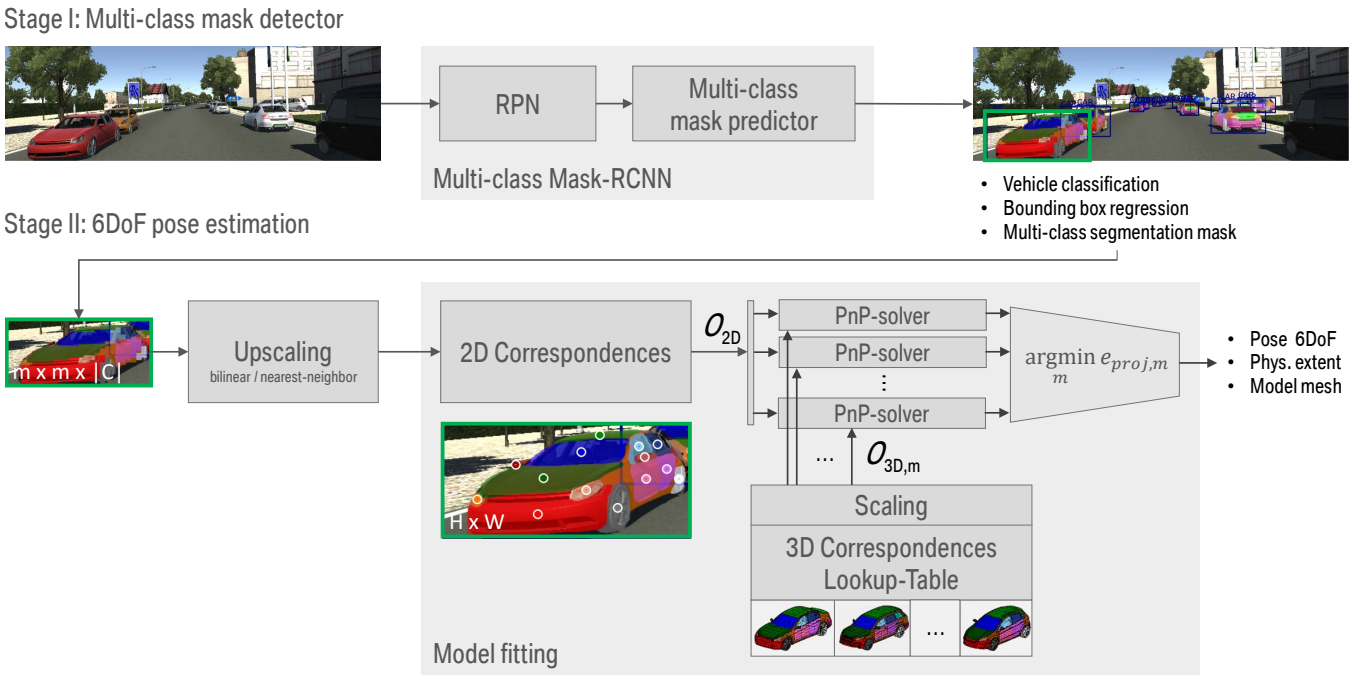

Fig. 3: Overview of the presented 6DoF pose estimation pipeline. In the first stage, an image is passed through the introduced multi-class mask detector to predict a list of bounding boxes with corresponding part-level vehicle masks. After that, a 2D center point is calculated for each part class of an upscaled mask. A filtered set of 2D correspondences O_{2D} is passed to a PnP-solver which performs PnP-fitting with the 3D correspondences $O_{3D,\,m}$ for each model m of a given model database. The 3D correspondences are scaled to fit the detection. After fitting, the algorithms returns the estimated 6DoF pose and the physical extent for the model with the lowest reprojection error.

fined neural network in a supervised fashion, a large amount of annotated data is required as training input. Commonly segmentation labeling is done manually, which is time consuming, especially for the high number of classes included in our representation. Additionally, the manual annotation of fine-grained vehicle parts is not viable for vehicles in far distances due to limited resolution. To overcome both problems, a semi-automated labeling pipeline [2] is set up.

An existing dataset which provides three dimensional bounding box ground truth is augmented with new data using CAD models using projection with calibrated camera: For each bounding box a model from our model zoo is assigned w.r.t. vehicle category and size (L_1 distance). The selected model is scaled to fit the dimensions of the labeled bounding box and placed at the bounding box position in 3D space. With a calibrated camera the dense multi-class vehicle segmentation can be rendered into the image. To handle occlusions correctly, we propagate only pixels that have been labeled as class car in an additionally given image segmentation. As a result, the dataset that is fed to the prediction network consists of bounding box coordinates, a foreground-background classification and the multi-class mask.

Vehicle pose estimation from part-level segmentation. The core of our algorithm is the downstream estimation of the vehicle pose, i.e., the joint search for the translation vector t and the rotation vector r – either three degrees of freedom – for each vehicle detection provided by the multi-class mask detector. Input to the algorithm is the triplet (B, P, M) with the detected bounding box B, the upscaled multi-class part segmentation mask P and the annotated model zoo M containing the model quadruplets described above. The multi-class detection mask P is a set of pixels with positions (p_x, p_y), which can be divided into several sub-masks P^c for each class c, holding only pixels predicted as the respective class. From this set of masks we generate 2D-3D correspondences for the PnP-solver.

In 2D image space, we use two different methods to generate a correspondence O_{2D} for a part class. The first one is a center-of-mass calculation as the average over all pixel coordinates within the mask predicted as the respective class:

$$O^c_{2D,CM} = \left(\frac{1}{|P^c|} \sum_{p \in P^c} p_x, \frac{1}{|P^c|} \sum_{p \in P^c} p_y \right).$$

The second method uses a rectangular approximation of all pixels belonging to a certain vehicle part class s:

$$tl^c_{2D} = \left(\min_{p \in P^c} p_x, \min_{p \in P^c} p_y \right)$$

$$br^c_{2D} = \left(\max_{p \in P^c} p_x, \max_{p \in P^c} p_y \right)$$

$$O^c_{2D,RECT} = \frac{tl^c_{2D} + br^c_{2D}}{2}.$$

In 3D model space, corresponding coordinates on each model can be precomputed for each vehicle instance in the model

database. There is no depedency w.r.t. the viewpoint in 3D. As in 2D space, we define the key point O as either calculated center of mass $O_{\text{3D,CM}}$ or a cuboid approximation $O_{\text{3D,CUB}}$. Let $V^c := \left\{v \in V^f, f \in F \mid Z(f) = c\right\}$ be the vertices incident to all faces of a given vehicle part:

$$O^c_{\text{3D,CM}} = \left(\frac{1}{|V^c|}\sum_{v\in V^c} v_x, \frac{1}{|V^c|}\sum_{v\in V^c} v_y, \frac{1}{|V^c|}\sum_{v\in V^c} v_z,\right)$$

$$tl^c_{\text{3D}} = \left(\min_{v\in V^c} v_x, \min_{v\in V^c} v_y, \min_{v\in V^c} v_z\right)$$

$$br^c_{\text{3D}} = \left(\max_{v\in V^c} v_x, \max_{v\in V^c} v_y, \max_{v\in V^c} v_z\right)$$

$$O^c_{\text{3D,CUB}} = \frac{br^c_{\text{3D}} + tl^c_{\text{3D}}}{2}$$

During inference, the 2D center points O^c_{2D} are generated for all classes that occur in the mask and then passed as tuples with their 3D correspondences O^c_{3D} to a standard PnP-solver, in our case the fast ePnP-solver from [30]. To avoid outliers, we integrate an option to use solely parts with a percentage of pixels larger than a threshold $\frac{|P^c|}{A^B} \leq d$ in the PnP-solver, in which A^B denotes the area of the detected bounding box B. A lower bound of 4 correspondences is required to estimate a pose with the PnP-solver.

The PnP-problem is solved with 3D correspondences $O_{\text{3D},m}$ from each model m and the final detection is returned with the size and pose estimation from the model with the lowest reprojection error $e_{\text{proj},m}$.

IV. Experiments

The experiments section starts with a description of the dataset followed by details of the code framework and the parameters used in training. The first evaluation is a short 2D analysis of the bounding box detector performance, yet we want to focus on the multi-class mask and 6DoF pose evaluation. To that end, we first analyze the mask quality using pixel accuracy for different kernel sizes and scaling methods before we evaluate the predicted 6DoF poses using 3D IoU and average distances.

Dataset. As we are projecting a dense representation, the demand for accurate labeling increases significantly compared to algorithms that use sparse correspondences: Aberrations within a few centimeters already result in erroneous projections of the multi-class segmentation masks. Therefore, our method is not applicable to established pose estimation benchmarks, for example the KITTI benchmark [6]. However, to obtain first results with this new approach, we utilize the synthetic dataset of Virtual KITTI [1] for a proof-of-concept. Earlier work [31], [32] already provided strong examples of how deep learning algorithms can rely on synthetic data for training.

Since the dataset does not provide a fixed benchmark definition, we define our own: The dataset was split into 13743 images for training, 1527 for validation and 4540 for testing, in which each dataset contains scenes with various weather and lighting conditions. Consecutive frames of sequences are prevented from appearing in the training and the test set. An extensive description can be found in [3]. Within the labeling pipeline we utilized the original CAD models that were used in the rendering of the Virtual KITTI dataset. Therefore, the projected part-segmentation matches the vehicles exactly.

Implementation. Our framework is based on the code of the tensorflow object detection framework [23]. The parameters of the two-stage detectors are configured as described in [22], [24]. In short, we use resized images with 600 pixels height, 5 scales and 3 aspect ratios for the RPN anchors and 300 object proposals. Feature extraction is done with the ResNet-101 backbone [8]. We set the loss weights of the detection heads to $\alpha_{\text{cls}} = 1$, $\alpha_{\text{box}} = 2$, $\alpha_{\text{mask}} = 4$ to focus on multi-class mask predictions since we use pretrained weights from a network trained solely on classification and bounding box regression. Our training is limited to 400,000 steps, starting from a pretrained bounding box detector. We evaluate different mask kernel sizes $m = 14, 28, 42, 56$ to analyze if higher mask resolutions affect the algorithm.

Detector evaluation. To evaluate the performance of the object detector we take a look at precision, recall and mean IoU using the Pascal criterion with a threshold th_{iou} = 0.5. In later evaluations we refer to detections complying with this criterion as assigned.

Two different training sets are considered at this first stage: The dataset holds a lot of highly truncated and occluded objects, marked with the label *dontcare* in the dataset. For evaluation, we ignore overlapping detections with these *dontcare* vehicles, but want to identify their influence during training. Therefore we evaluate separate trainings with and without considering the *dontcare* vehicles. Table I shows that the precision, recall and mean IoU of all trained detectors are very high and therefore provide a lot of well-overlapping 2D detections as input to our pose estimation algorithm. Training with the *dontcare* vehicles results in lower precision but higher recall, which is reasonable since the detector learns from harder examples. Due to the fact that the mean IoU is higher for training without *dontcare* vehicles, we solely use this training in the following experiments.

Kernel size	Dontcare in training	Precision	Recall	mIoU
14x14	–	**99.89**	96.11	**92.46**
14x14	x	93.59	**99.79**	91.43
28x28	–	**99.97**	96.74	**92.27**
28x28	x	94.22	**99.8**	91.43
42x42	–	**99.93**	96.33	**91.91**
42x42	x	93.54	**99.79**	91.7
56x56	–	**99.94**	95.88	**92.4**
56x56	x	93.3	**99.79**	91.74

TABLE I: Results for the bounding box detector in **Precision**, **Recall** and mean intersection of union (**mIoU**) for different mask kernel sizes and different training sets.

Mask evaluation. The prediction of the multi-class ve-

hicle masks is evaluated by pixel-wise accuracy P_A for each assigned detection. The final metric mP_A is the mean over the accuracies P_A for all assigned detections of the testing dataset. We evaluate the varying mask kernel sizes and scaling methods, namely bilinear and nearest-neighbor scaling, exemplary visualized in Fig. 4. To neglect the influence of erroneous bounding box sizes, pixels of the ground truth bounding box that are not in the respective detection bounding box are not considered false positive.

Fig. 4: Visual comparison of the resulting multi-class masks for the same detection using the nearest-neighbor or bilinear scaling method. The different kernel sizes imply different underlying network weights.

The results from table II show three characteristics: While the first two characteristics – larger mask kernel size results in a better mP_A score and bilinear scaling is better than the nearest-neighbor method – are quite intuitive, the latter is of greater interest: Nearest-neighbor scaling mP_A shows a significant performance difference (-6.48%) for the smallest kernel size but for larger kernels, this difference is rather small (-4.28% to -2.74%). We conclude that a lower bound for nearest-neighbor scaling should be estimated in relation to the object size distribution of the dataset. Since the results for larger kernel sizes using the significantly faster nearest-neighbor scaling are in the same range as for bilinear scaling, we analyze both methods in the pose evaluation section.

Kernel size	Scaling	mP_A
14x14	bilinear	81.27
28x28	bilinear	84.7
42x42	bilinear	85.37
56x56	bilinear	**85.71**
14x14	nearest-neighbor	74.69
28x28	nearest-neighbor	80.42
42x42	nearest-neighbor	82.09
56x56	nearest-neighbor	**82.97**

TABLE II: Results for the prediction of the multi-class masks in mean pixel accuracy mP_A w.r.t. different kernel sizes and scaling methods.

Pose estimation evaluation. The 6DoF fitting algorithm is evaluated by using 3D IoU as well as average Euclidean distance and discrepancy of the Euler angles. As stated earlier, only assigned 2D detections are considered in this analysis. Since each model appears in the dataset with varying extents, we scale the 3D correspondences O_{3D} with the factor extracted from ground truth. Table III and IV compare the 3D performance and model precision with respect to different scaling methods, mask kernel sizes and correspondence calculation methods O_{2D}, O_{3D}. 3D IoU is calculated from cuboids using the predicted box extent.

The overall performance settles at about 40 percent 3D IoU, which we consider good for a monocular camera system. Qualitative results (Fig. 5) show many well estimated examples, even under challenging weather or lighting conditions. We want to remind the reader that the 3D IoU metric is very punishing and even a small error in distance estimation already results in a low score for a normally sized car. The small average errors for y, α_x and α_z prove that the algorithm is well-suited for full 6DoF pose estimation. In all configurations there were no detections with too few correspondences to perform PnP-fitting. Regarding model precision, over half of the detections are predicted with the correct model. We consider a prediction of the model with the neural network in future research.

Comparing the different configurations we conclude with the following statements: The calculation of the correspondences is in favor of the rectangular approximations $O_{2D,RECT}$, $O_{3D,CUB}$, resulting in significantly higher model precision (up to +9.2%) and slightly higher 3D IoU (up to +3.97%). Another implication is that larger kernel sizes do not improve quality of the pose estimation. Neither 3D IoU nor average distance errors follow any trend – as can be seen by the various configurations that perform best in the different metrics. Instead, we consider this algorithmic noise. This statement appears also to be valid for the two scaling methods: Solely the smallest kernel for bilinear scaling is an outlier, supporting our theory from mask evaluation that the estimation of a minimum kernel size is recommended. For kernel sizes above 28x28 the 3D performance does not increase.

An abnormality in the presented results is the lower average yaw error δ_y for the smallest mask kernel size. Our analysis showed that a significant number of vehicles closer than 5 meters are better for this kernel size, which is caused by truncated parts that are solely detected at larger kernel sizes and affect the fitting in a negative way.

Parameter tuning. The presented algorithm permits a lot of tuning possibilities that are open for further research. In our experiments we already found the following tweaks to improve performance: As stated earlier, removing parts with low pixel counts may be considered (-SP) but we do not apply this for the *mirror* and *lights* classes by whitelisting (+WL) due to their relatively small size. Another way to increase performance is the blacklisting of certain parts (+BL), i.e., the classes for the sides, aprons and the roof due to their high variance w.r.t. the camera viewpoint. Lastly, we found a solution to the earlier mentioned truncation problems by removing correspondences that are directly at the image borders (+CFIX), resulting in significantly lower δ_y error (-0.10rad). Another solution for this problem could be outlier analysis with methods like RANSAC or a

Kernel	Scaling	O_{2D}	O_{3D}	Model	3D IoU	ΔT/m	ΔR/rad	Δx/m	Δy/m	Δz/m	$\Delta\alpha_x$/rad	$\Delta\alpha_y$/rad	$\Delta\alpha_z$/rad
14x14	bilinear	CM	CM	44.51	40.84	3.44	**0.35**	**0.96**	0.14	2.34	0.08	**0.20**	**0.07**
14x14	bilinear	RECT	CUB	43.78	40.33	**3.43**	0.38	0.97	0.13	**2.32**	0.09	0.21	0.08
28x28	bilinear	CM	CM	50.36	**43.52**	3.56	0.41	0.99	0.13	2.44	**0.07**	0.25	0.08
28x28	bilinear	RECT	CUB	55.01	42.31	3.98	0.45	1.13	0.13	2.71	0.09	0.28	0.08
42x42	bilinear	CM	CM	47.67	42.14	4.18	0.41	1.14	0.15	2.90	0.08	0.25	0.08
42x42	bilinear	RECT	CUB	**56.87**	42.27	4.31	0.44	1.20	0.14	2.97	0.09	0.26	0.08
56x56	bilinear	CM	CM	49.47	42.98	3.51	0.42	0.98	0.13	2.4	**0.07**	0.26	0.08
56x56	bilinear	RECT	CUB	55.03	42.03	3.64	0.45	1.03	**0.12**	2.47	0.09	0.29	0.09

TABLE III: Results for the pose evaluation based on bilinearly upscaled masks. We compare different kernel sizes and the two different correspondence calculation methods using Euclidean distances (**Δx**, **Δy**, **Δz**), discrepancy of the Euler angles ($\Delta\alpha_x$, $\Delta\alpha_y$, $\Delta\alpha_z$), model precision (**Model**) and 3D intersection over union (**3D IoU**). Euclidean distances are summed up to **ΔT** while **ΔR** represents the accumulated angular errors. The reported distances are defined in the camera frame with x pointing to the right, y pointing downwards and z pointing to the front.

Kernel	Scaling	O_{2D}	O_{3D}	Model	3D IoU	ΔT/m	ΔR/rad	Δx/m	Δy/m	Δz/m	$\Delta\alpha_x$/rad	$\Delta\alpha_y$/rad	$\Delta\alpha_z$/rad
14x14	nearest-n.	CM	CM	42.26	26.63	4.41	**0.36**	1.16	0.18	3.07	0.08	**0.21**	**0.07**
14x14	nearest-n.	RECT	CUB	42.05	36.78	3.72	0.38	1.02	0.16	2.54	0.09	0.22	0.08
28x28	nearest-n.	CM	CM	49.17	38.24	3.85	0.40	1.04	0.15	2.66	**0.07**	0.25	0.08
28x28	nearest-n.	RECT	CUB	53.94	42.21	4.49	0.43	1.12	0.15	2.75	0.09	0.27	0.08
42x42	nearest-n.	CM	CM	47.55	40.62	4.41	0.4	1.19	0.16	3.05	0.08	0.24	0.08
42x42	nearest-n.	RECT	CUB	**55.93**	42.25	4.49	0.43	1.22	0.16	3.10	0.09	0.26	0.08
56x56	nearest-n.	CM	CM	49.6	**42.30**	**3.50**	0.41	**0.96**	**0.14**	**2.40**	**0.07**	0.26	0.08
56x56	nearest-n.	RECT	CUB	55.03	42.03	3.64	0.45	1.03	**0.14**	2.48	0.08	0.28	0.08

TABLE IV: Results for the pose evaluation based for masks upscaled with the nearest-neighbor scheme. We compare different kernel sizes and the two different correspondence calculation methods using Euclidean distances (**Δx**, **Δy**, **Δz**), discrepancy of the Euler angles ($\Delta\alpha_x$, $\Delta\alpha_y$, $\Delta\alpha_z$), model precision (**Model**) and 3D intersection over union (**3D IoU**). Euclidean distances are summed up to **ΔT** while **ΔR** represents the accumulated angular errors. The reported distances are defined in the camera frame with x pointing to the right, y pointing downwards and z pointing to the front.

visibility analysis during fitting. Table V gives an overview of the performance increases: In the best configuration, model precision is boosted by +9% and for 3D IoU we were able to achieve more than +6%. Applying these techniques, we had a ratio of 0.09% unfittable detections.

Configuration	Model	3D IoU	ΔT/m	ΔR/rad
56x56 bilinear RECT	55.03	42.03	3.64	0.45
-SP +BL +WL	63.83	48.87	2.48	0.29
-SP +BL +WL +CFIX	**64.17**	**49.34**	**2.42**	**0.28**

TABLE V: Results in model precision (**Model**), 3D intersection over union (**3D IoU**) and the summed errors (**ΔT**, **ΔR**) for improved algorithmic configurations.

Execution performance. The inference time of Mask-RCNN is not affected by the multi-class extension, yet the training time is higher and scales with the configured mask kernel size. Calculating the correspondences and PnP-fitting is within a few milliseconds and therefore can be considered real-time capable. The choice of the scaling method makes a difference since bilinear scaling takes about two orders of magnitude longer than nearest-neighbor scaling, which is usually within a double-digit microsecond range.

V. Conclusion

We presented a 6DoF pose estimation algorithm utilizing the representation of a multi-class segmentation, which is a first achievement toward the goal of extracting 3D information from pixel-wise semantics with a monocular camera setup. The evaluation shows promising 3D results under various image conditions and provides extensive insights on configuration and parameters. The high accuracy of the underlying multi-class mask representation brings great benefits for detailed vehicle analysis, extracting dynamic parts and indicators. The significantly increased performance from parameter tuning highlights the fact that the used representation holds a lot of unexplored potential, which can be exploited by a better fitting procedure considering all pixel information and the underlying predicted probability distributions. A detailed analysis regarding a judicious trade-off between computational demand for fitting and performance is necessary. Our future work will investigate under which conditions the used representation can be applied to real-world data.

References

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

[2] F. Chabot, M. Chaouch, J. Rabarisoa, and T. Chateau, "Deep manta: A coarse-to-fine many-task network for joint 2d and 3d vehicle analysis from monocular image," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[3] T. Barowski, M. Szczot, and S. Houben, "Fine-grained vehicle representations for autonomous driving," in *Proc. of the IEEE Conference on Intelligent Transportation Systems*, 2018.

[4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Computing Research Repository (CoRR)*, vol. abs/1409.1556, 2014.

[5] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for

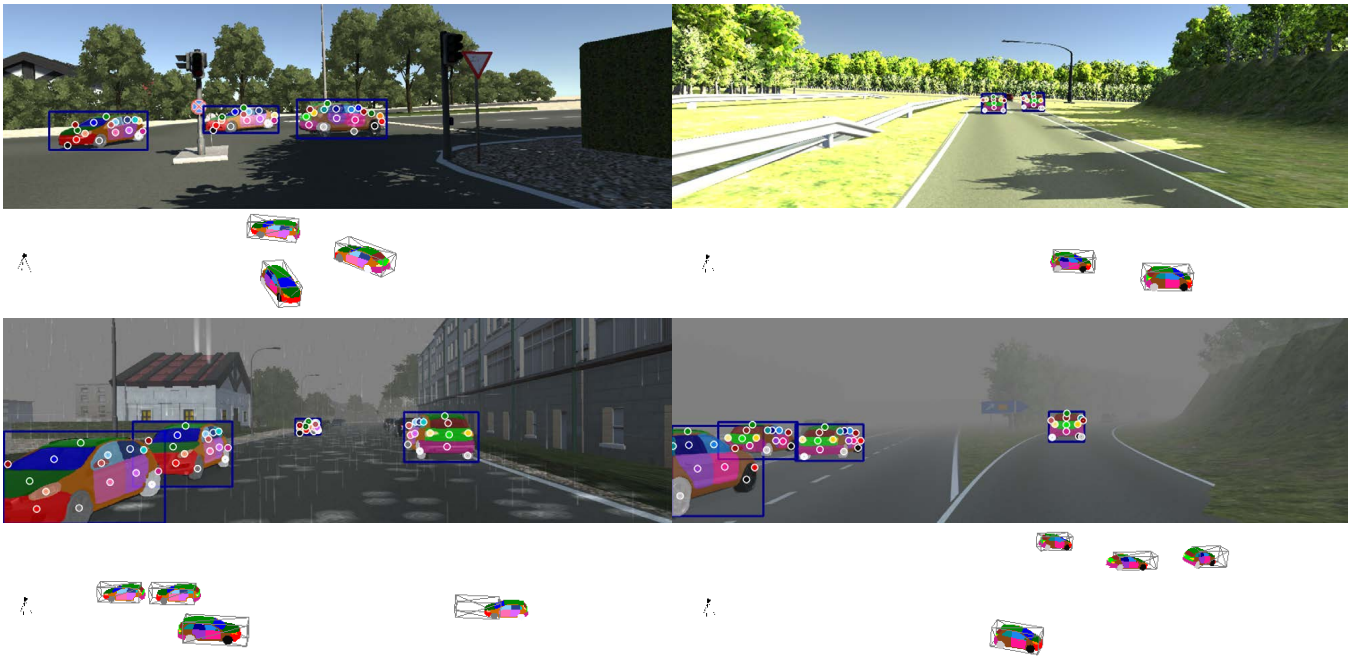

Fig. 5: Several example frames from the evaluation on the Virtual KITTI dataset [1]. The upper half of the example shows the processed frame with bounding box vehicle detections, mask predictions and calculated center points (colored dots). The lower half shows the 3D visualization of the scene with the predicted models for detections and wireframe bounding boxes for ground truth.

semantic urban scene understanding," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[6] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[7] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," *Computing Research Repository (CoRR)*, vol. abs/1405.0312, 2014.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[9] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2016.

[10] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, 2017.

[11] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[12] 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," vol. 40, no. 4, 2018.

[13] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, "Learning rich features from rgb-d images for object detection and segmentation," in *Proc. of the European Conference on Computer Vision*, 2014.

[14] B. Romera-Paredes and P. H. S. Torr, "Recurrent instance segmentation," in *European conference on computer vision*, 2016.

[15] M. Ren and R. S. Zemel, "End-to-end instance segmentation with recurrent attention," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[16] E. Park and A. C. Berg, "Learning to decompose for object detection and instance segmentation," *arXiv preprint arXiv:1511.06449*, 2015.

[17] B. De Brabandere, D. Neven, and L. Van Gool, "Semantic instance segmentation with a discriminative loss function," *arXiv preprint arXiv:1708.02551*, 2017.

[18] J. Uhrig, M. Cordts, U. Franke, and T. Brox, "Pixel-level encoding and depth layering for instance-level semantic labeling," in *German Conference on Pattern Recognition*, 2016.

[19] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[20] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Proc. of the European Conference on Computer Vision*, 2016.

[21] R. Girshick, "Fast r-cnn," *arXiv preprint arXiv:1504.08083*, 2015.

[22] 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.

[23] V. R. J Huang, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy, "Speed/accuracy trade-offs for modern convolutional object detectors," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[24] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proc. of the IEEE International Conference on Computer Vision*, 2017.

[25] Y. Xiang, W. Choi, Y. Lin, and S. Savarese, "Data-driven 3d voxel patterns for object category recognition," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[26] B. Pepik, M. Stark, P. Gehler, and B. Schiele, "Multi-view and 3d deformable part models," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 11, 2015.

[27] M. Arsalan, D. Anguelov, J. Flynn, and J. Kosecka, "3d bounding box estimation using deep learning and geometry," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[28] X. Chen, K. Kundu, Y. Zhu, A. G. Berneshawi, H. Ma, S. Fidler, and R. Urtasun, "3d object proposals for accurate object class detection," in *Advances in Neural Information Processing Systems*, 2015.

[29] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[30] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o (n) solution to the pnp problem," *International journal of computer vision*, vol. 81, no. 2, 2009.

[31] S. R. Richter, V. Vineet, S. Roth, and V. Koltun, "Playing for data: Ground truth from computer games," in *Proc. of the IEEE European Conference on Computer Vision*, 2016.

[32] H. Abu Alhaija, S. K. Mustikovela, L. Mescheder, A. Geiger, and C. Rother, "Augmented reality meets computer vision: Efficient data generation for urban driving scenes," *International Journal of Computer Vision*, 2018.