Predicting Objects of Interest with Deep Learning

Stefan Hoermann* and Klaus Dietmayer*

Abstract: As a continuation of learned object detection in dynamic occupancy grid maps, this work aims on long-term prediction of individual objects of interest using deep learning. The algorithms focus on a complex shared space scenario including cars, bikes and pedestrians. A main advantage of the approach is that no manual labeling is required for both, object detection and prediction, due to acausal object track and shape refinement. Predicting future occupancy of single objects given the whole perceived scene, special attention is paid on the interactive influence between dynamic objects and also the peripherals.

Keywords: Automatic Labeling, Deep Learning, Dynamic Occupancy Grid Map, Long-Term Prediction, Object Detection

1 Introduction

Predictive driving behavior makes human drivers superior to today's automated vehicles. Thus, situation prediction in terms of motion or trajectory prediction is a key component to advance automated vehicles to smart systems. From the uncertain nature of prediction, it derives the objective to reduce the prediction uncertainty as much as possible, but without gaining false negative predictions. Besides kinematic limitations, interactions and context are the main sources of information to reduce the uncertainty of the evolution of a scene. Convolutional neural networks (CNNs) are known for their capability to exploit context, and as shown in the experiments of this work they are also capable to model human interactions. A dynamic occupancy grid map (DOGMa) [1] includes the entire perceivable scene in an image-like structure, naturally suggesting the usage of CNNs for situation analysis. However, the representation is model-free, what means that grid cells are considered to be independent and no hypotheses of objects, e.g. a list of vectors containing shape and pose, are provided. Although, long-term prediction can be learned model-free on cell basis [2], modeled objects are favorable for further processing like decision making. Object detection, nonetheless, is a critical task in automated driving. A misinterpretation between static background, clutter and dynamic objects can result in fatal accidents [3, 4]. A complementing use of grid maps and modeled objects seems reasonable to avoid a trade-off between false positive and false negative object hypotheses in classical early fusion stages. We argue that predicting single objects of interest in addition to the model-free prediction [2] of the entire scene can increase robustness and safety of an overall system. In this work, therefore, detected objects are predicted individually employing a CNN as motion predictor given the entire perceived DOGMa and a selection mask of single detected objects from [5]. An often claimed disadvantage

*The authors are with the Institute of Measurement, Control, and Microtechnology at Ulm University, Albert-Einstein-Allee 41, 89081 Ulm (firstname.lastname@uni-ulm.de).

of artificial neuronal networks is the high demand for training examples. Therefore, we use automatic generated labels [6]. The concept is based on offline sequence processing, in which an object trajectory is extracted forward and backward in time.

Deep learning on grid maps gains great popularity. Piewak et al. [7] employed a CNN to reduce false positive velocity estimates in a DOGMa with the goal to improve classical clustering techniques to unite cells of the same object. Learned object detection is studied with great interest in the computer vision community. In particular, the concept of anchors [8], where default boxes are classified and the offset to the true object is estimated by regression, gained great success. We adapted these techniques to object detection in grid maps [5]. The group around Posner published the Deep Tracking series [9, 10] where recurrent network structures are employed to resolve occlusions in a grid map via temporal grid cell prediction and intrinsic dynamics estimation. To clarify, the word tracking in this work does not refer to state estimation of moving objects. Deep Tracking can be seen as a learned alternative to the sequential filter employed in the DOGMa from Nuss [1]. The presented work is complementing our previous work [2], where a DOGMa is feed to a CNN used as long-term motion predictor. Instead of predicting the future occupancy of the entire scene, in the present work only selected objects are predicted.

In the remaining document, an overview of the framework is given in Section 2, features of dataset and training are given in Section 3. Experiments are carried out in Section 4 followed by conclusions.

2 System Overview

Inspired by Zhan et al. [11], considering all objects, static and dynamic, with object state $x^{(\alpha)}$ where objects are enumerated by α, the entire scene is described by $\mathcal{X} = \{x^{(1)}, x^{(2)}, ...\}$. From a planning perspective, it is favorable to get future object states $x_+^{(\alpha)}$ as predictions $p(x_+^{(\alpha)}|\mathcal{X})$. Unfortunately, the classic approach of object detection, tracking and prediction

$$\mathcal{X} \rightarrow \{p(x^{(1)}), p(x^{(2)}), ...\} \rightarrow \{p(x_+^{(1)}), p(x_+^{(2)}), ...\} \tag{1}$$

suffers already in the early stage from error prone object detections. Forward-looking driving with long-term prediction is more sensitive to false positive object detection than pure reactive driving. Also, describing the entire static and dynamic scene as object vectors can be complex without hard simplifications. An alternative to the object domain is the grid domain, where the environment is divided in a grid of cells c. For each cell the probability that the location is occupied $p(\mathrm{O}^{(c)}(\mathcal{X}))$ is estimated. Dynamic grid maps, in addition, estimate dynamic states $\mathrm{d}^{(c)}(\mathcal{X})$ without individual object hypotheses. Following this concept and feeding a DOGMa into neural networks, our system provides three outputs, as illustrated in Figure 1. The first branch transforms the grid cell representation into object domain and provides

$$\left(x^{(\alpha)}, r^{(\alpha)} = P(x^{(\alpha)}|\mathrm{O}(\mathcal{X}), \mathrm{d}(\mathcal{X}))\right) \ , \tag{2}$$

an object hypothesis with existence probability r. It was introduced in [5] and results can be seen under https://youtu.be/Rr9LOrQMgKA. The second network predicts future scene occupancy in form of a grid map

$$p(\mathrm{O}(\mathcal{X}_+)|\mathrm{O}(\mathcal{X}), \mathrm{d}(\mathcal{X})) \ . \tag{3}$$

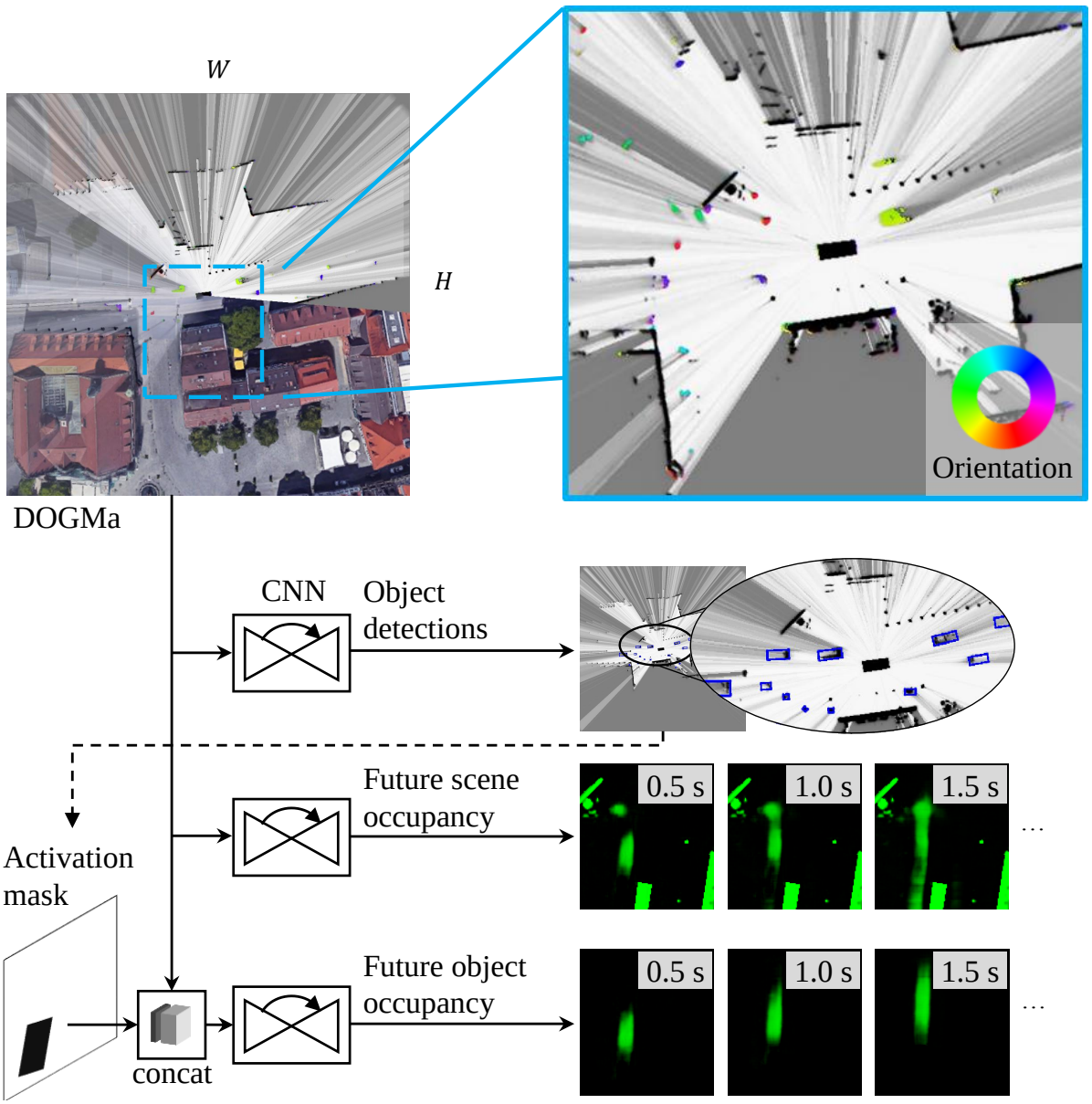

Figure 1: System overview: Dynamic occupancy grid map (DOGMa) is the input to three neural networks. The first branch detects objects in the grid fusion. The second branch provides future occupancy of the entire scene. The bottom network predicts future occupancy related to an object of interest which is selected by an additional input channel.

The network was introduced in [2] and a video showing the results can be seen under https://youtu.be/C1hhEYGs49Q. The third network is novel in this work complementing the other branches. For a selected object α, it provides predictions of future object occupancy

$$p(\mathrm{O}(x_{+}^{(\alpha)})|\mathrm{O}(\mathcal{X}), \mathrm{d}(\mathcal{X})) \ . \tag{4}$$

We claim that using these tools, robustness of planning with respect to false positive and false negative object detection can be improved compared to pure object based methods due to the following reasons:

- No gating/thresholding of sensor data is done since object detection is performed after sensor fusion and dynamics estimation.
- False negatives in object detection are still included in the DOGMa.
- False velocity estimates in the DOGMa are trained to be correctly predicted in scene occupancy prediction, e.g. walls with high velocity are not predicted with spatial movement.
- The influence of false positive object detections can be reduced when limiting object assessment to objects of interest.

The last bullet point might be seen controversial. However, we consider the planning task dividable in two sub tasks: reactive and strategic planning. Reactive planning, on the one hand, reacts on the environment without the intention to influence other objects and thus doesn't need object hypothesis generation, but can react on future scene occupancy from (3) to yield forward-looking driving behavior. Strategic planning, on the other hand, exploits object specific features, e.g. kinematic constraints or the capability to cooperate. To reduce the risk of considering false positive objects for strategic planning, objects of interest can be defined according to the scenario, e.g. the leading vehicle in a following scenario or oncoming vehicles in a turning scenario.

2.1 Network Structure

Except for input and output, the three networks have the same core architecture which relies on a simple convolution/deconvolution structure with skip connections similar to [12]. Input and output of all networks can be seen as a top view representation, i.e. grid map, while the spatial input dimensions are always equal to the spatial output dimensions. Pooling is used for down sampling and deconvolution for upscaling. Skip connections include a convolutional layer to reduce the channel dimension. The skip connection result is concatenated to the main tensor before upscaling via deconvolution.

2.1.1 Input and Output Data

The dynamic occupancy grid map (DOGMa) from Nuss [1] serves as input to the three CNNs. Multiple sensors are fused in a grid representation where the environment is divided in cells c. Employing a particle filter, dynamic cell states can be estimated even if velocity can not be measured directly, i.e. using lidar sensors. An example for the resulting bird's-eye image is illustrated in Figure 1. Each cell stores occupancy and

dynamics information in channels $\Omega = \left\{M_{\mathrm{O}}, M_{\mathrm{F}}, v_{\mathrm{E}}, v_{\mathrm{N}}, \sigma^2_{v_{\mathrm{E}}}, \sigma^2_{v_{\mathrm{N}}}, \sigma^2_{v_{\mathrm{E}},v_{\mathrm{N}}}\right\}$. $M_{\mathrm{F}} \in [0,1]$ and $M_{\mathrm{O}} \in [0,1]$ are the Dempster-Shafer masses for free space and occupancy, respectively. Velocity east and north components are stored in v_{E} and v_{N}, respectively. σ^2 describes the velocity variance or covariance. The occupancy probability is calculated by $p(\mathrm{O}) := 0.5 \cdot M_{\mathrm{O}} + 0.5 \cdot (1 - M_{\mathrm{F}})$. The DOGMa data is provided in $\mathbb{R}^{W \times H \times |\Omega|}$ with the spatial width W and height H. The grid is constant and the ego vehicle moves within the grid.

To select the object of interest, an additional binary channel is fed to the network with dimension $W \times H \times 1$. We call this channel the activation mask or selection mask, since grid cells covered by the object of interest are set to 1 and other cells are filled with 0.

Future scene occupancy prediction and the novel future object occupancy prediction have similar output structure. The main difference is, that object prediction contains only occupancy of a single selected object and no static/dynamic segmentation is performed. The network result $\hat{P}_{\mathrm{O}} \in \mathbb{R}^{W \times H \times T}$ provides occupancy maps at discrete future time steps $k \in (1, T)$. In our experiments we chose a prediction step size of 0.5 s with $t_k = t_0 + k \cdot 0.5\,\mathrm{s}$. The output interpretation of a tensor element $\hat{P}_{\mathrm{O}}(\mathrm{E}, \mathrm{N}, k) := \hat{P}_{\mathrm{O}}(c, k) := \hat{p}(\mathrm{O}_k^{(c)})$ is the occupancy probability at time step k and cell c with coordinates east E and north N. The hat symbol $\hat{}$ indicates the network output while for labels the hat is omitted.

In contrast to the result from scene occupancy prediction, where occupancy couldn't be associated to an object, the result here is completely related to the object chosen at the activation mask in the input. Normalizing the output grid maps $P_{\mathrm{O}}(k)$ at a single time step to the object size, i.e. number of cells, allows the probabilistic interpretation. Figure 3 illustrates an example.

3 Dataset and Training

All three tasks were trained without any manual labeling. Instead of humans annotating data, algorithms were employed to annotate over 2 h recordings. This proceeding, of course, is only successful since the annotation algorithm can use more information than the network, i.e. a whole sequence instead of only data from the past. We call this concept acausal long-term label extraction. The main benefits are as follows: Firstly, we refine estimates in the sequence. For example, velocity estimates are corrected by comparing it with actual recorded movement. Also, the estimated object shape is corrected when the object was seen from all sides during the sequence. Secondly, recorded and annotated data can be smoothed. Sequential filtering, i.e. a Kalman filter is not necessary and can be substituted by simple low pass smoothing and outlier reduction. Thirdly, we correct false positives, e.g., when an object trajectory is unreasonable, the object label is deleted. We also search for false negatives. For example, when an object is detected, it was certainly missed earlier in the sequence when it just entered the field of view. By tracking the object backwards in time, labels can be generated at these time steps. Lastly, the concept benefits from the fact that runtime is no issue, as long as the algorithm is faster than human annotations, enabling thorough calculation instead of fast approximations. A detailed explanation of the object trajectory extraction algorithm is given in [6].

Compared to human annotations, the acquired data can be considered messy. It contains approximately 5 % false labels. This has to be considered during training. Common training methods rely on perfect labels, like hard example mining where only the worst training samples of an overrepresented class are used to update the network. In our case,

however, a hard example might be a false label. Thus, we perform balancing by weighting underrepresented samples more than overrepresented examples but use a massive dataset to gain an appropriate number of underrepresented samples while keeping the number of training epochs low. Training the network to not predict corrupted labels is a matter of generalization, i.e. finding a local minimum instead of the global optimum with respect to the corrupted training data set. To gain good generalization we benefit from the possibility to produce high amounts of training data and thus, we use every example at most three times to update the network.

3.1 Spatial Balancing Loss Function

Due to the nature of temporal prediction, the result is always uncertain. Choosing an occupancy grid map representation as the network output helps to represent spatial uncertainty about the future object position. False positives, i.e. cells with $\hat{P}_{\mathrm{O}}(c) > 0$ outside the vehicle silhouette, can be interpreted as prediction uncertainty. False negatives, i.e. cells occupied by the label but with $\hat{P}_{\mathrm{O}}(c) = 0$, must be considered as prediction failure. Due to a high imbalance between occupied and free labels, we balance the loss by weighting cells with label $P_{\mathrm{O}}(c) > 0$ more with

$$L = \frac{1}{2} \sum_{k=1}^{T} \sum_{c=1}^{W \times H} \left(1 + \lambda_k P_{\mathrm{O}}(c,k)\right) \left(\hat{P}_{\mathrm{O}}(c,k) - P_{\mathrm{O}}(c,k)\right)^2 \tag{5}$$

where λ_k increases with the prediction horizon. In our experiments we chose $(\lambda_k) = (40, 50, 100, 200)$ for $(0.5\,\mathrm{s}, 1.0\,\mathrm{s}, 1.5\,\mathrm{s}, 2.0\,\mathrm{s})$, respectively.

4 Experiments

In this section we show the overall performance, as well as examples to illustrate the network capabilities for multi modal situation prediction and interaction consideration. Including GPU data transfer, the network runs with 73.5 ms per prediction on a Nvidia GTX 1080Ti. The overall prediction performance is illustrated in Figure 2. The plot compares the prediction performance of the neural network to Monte Carlo simulations. A Monte Carlo simulation forward propagates 5000 particles. The particle states were initialized with a labeled object while velocity and orientation distribution was drawn from the covariance in the occupied grid cell states. Particle acceleration a and turn rate ω are assumed to be constant over the prediction time frame but drawn from $\mathcal{N}(0, \sigma_a)$ and $\mathcal{N}(0, \sigma_\omega)$, respectively. 1000 randomly picked objects were predicted with different constellations of $(\sigma_a, \sigma_\omega)$, as well as the trained network, to exemplify the prediction performance of the CNN in terms of the relationship between false negatives and uncertainty. False positive (FP), false negative (FN) and true positive (TP) occupied cells were counted comparing any cell with predicted occupancy > 0 with labeled occupied cells. The false negative rate (FNR=$\frac{\mathrm{FN}}{\mathrm{TP+FN}}$) indicates how often the prediction fails. The uncertain area gain ($\frac{\mathrm{FP+TP}}{\mathrm{TP+FN}}$) indicates how large the predicted occupied area grows compared to the object size. It can be seen in the plot, that the network prediction fails less often and has a lower uncertainty as the first two Monte Carlo simulations. A very high process noise in the Monte Carlo run is necessary to gain similar low FNR but gaining very high uncertainty.

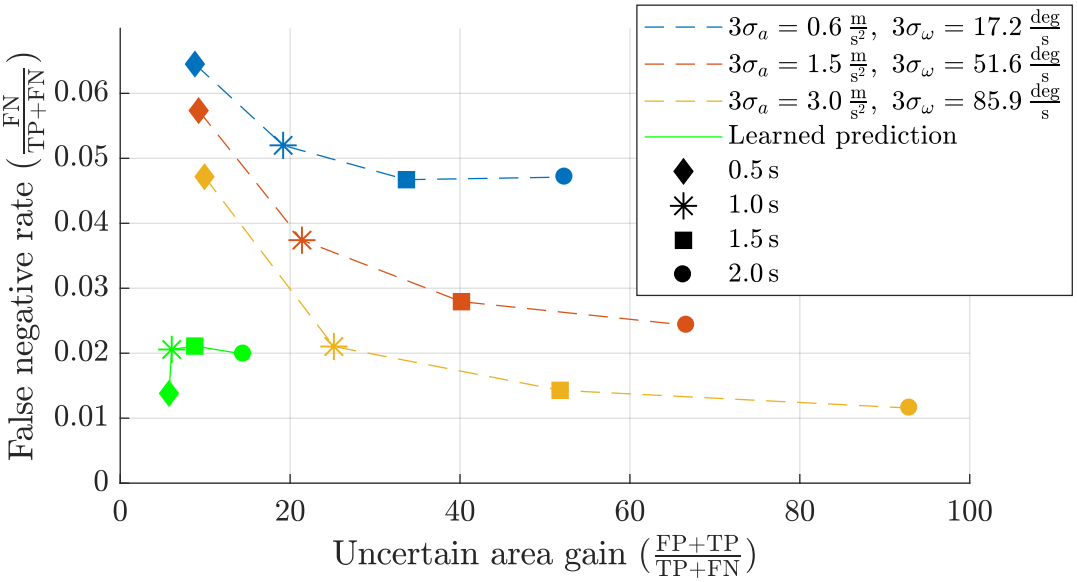

Figure 2: Comparison of learned prediction to Monte Carlo simulation. 1000 objects were predicted by the CNN and with constant velocity constant turn rate model, while 5000 particles draw process noise parameters for acceleration σ_a and turn rate σ_ω. Markers indicate prediction time steps. The plots illustrate the false negative rate, i.e. when prediction misses the true object position, over uncertainty, i.e. the area predicted as possibly occupied in ratio to occupied area. While engineered motion model prediction reaches low miss rate only with very high uncertainty, the network reaches lower miss rate with lower uncertainty in most cases.

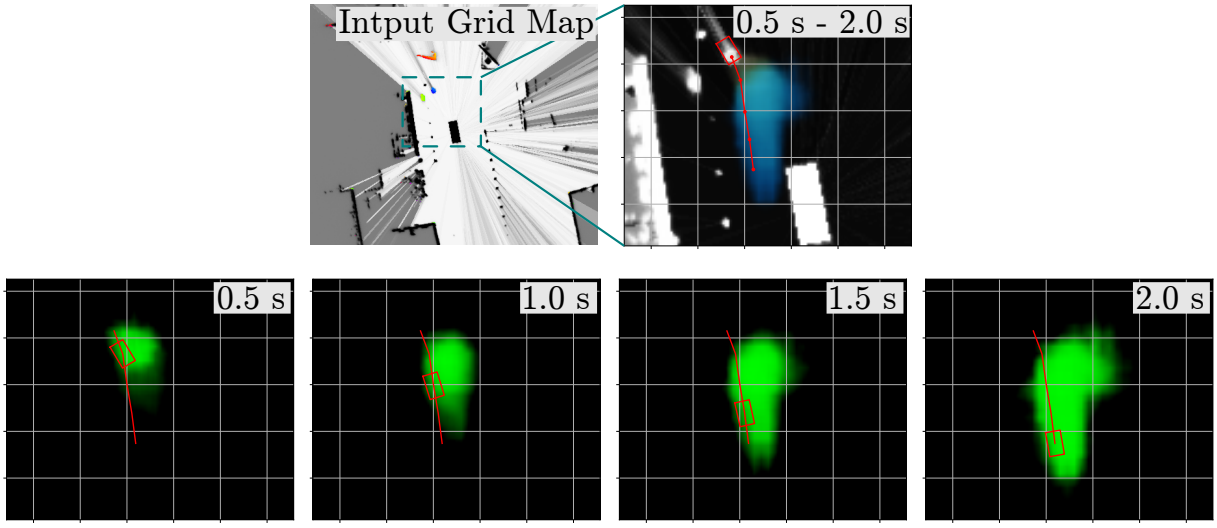

Figure 3: Example scene for object prediction: A cyclist is predicted multi modal. The complete scene is illustrated in the top left. An overview of the complete prediction result of the selected bicycle is given in the top left. Prediction results for the single time steps are shown in the bottom row. Red rectangles illustrate the true object bounding box, the red line indicates the 2 s trajectory.

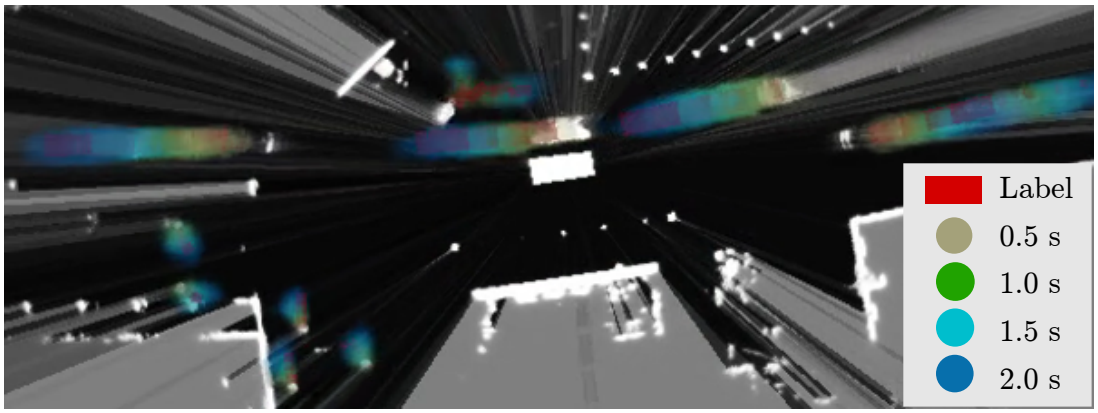

Figure 4: Example scene for object prediction: Multiple objects were predicted including pedestrians in the bottom and cars driving horizontally in center. In this image, four different prediction horizons and labels are drawn in stacked layers with transparency with the occupancy grid map of the current scene as background.

4.1 Interactions and Multi Modal Predictions

Multiple predicted objects are illustrated in Figure 4 to give an example. A single cyclist is predicted in Figure 3 showing an example for multi modal prediction capabilities. The ability to cover interactions is examined on manipulated input data, where objects are pasted close to the original trajectory of the object of interest. An example scene is illustrated in Figure 5, showing the original scene of a driving car, the same scenario with a parking car on the lane, and another example with a pedestrian approaching the lane.

5 Conclusion

To complete an existing long-term prediction framework, grid-based prediction was extended to include an individual object predictor. Learned object detection is used to create a spatial binary mask used, to select object cells in a grid map. Both object detection and object prediction was trained with automatic label generation where an engineered acausal algorithm extracts object shapes and trajectories form a sequence. Evaluation shows failure rate over uncertain area gain, a performance measure comparing predicted possible occupied area relative to actual occupied area. Experiments showed that the algorithm has less uncertainty and fails less often, in terms of false negative predictions, compared to straight forward methods. An experiment with artificially inserted traffic participants in a real scene proves intrinsically modeled interactions. In future work, generalization on other traffic scenes and prediction from a moving platform will be investigated.

References

[1] D. Nuss, *A Random Finite Set Approach for Dynamic Occupancy Grid Maps*, ser. Schriftenreihe des Instituts für Mess-, Regel- und Mikrotechnik. Universität

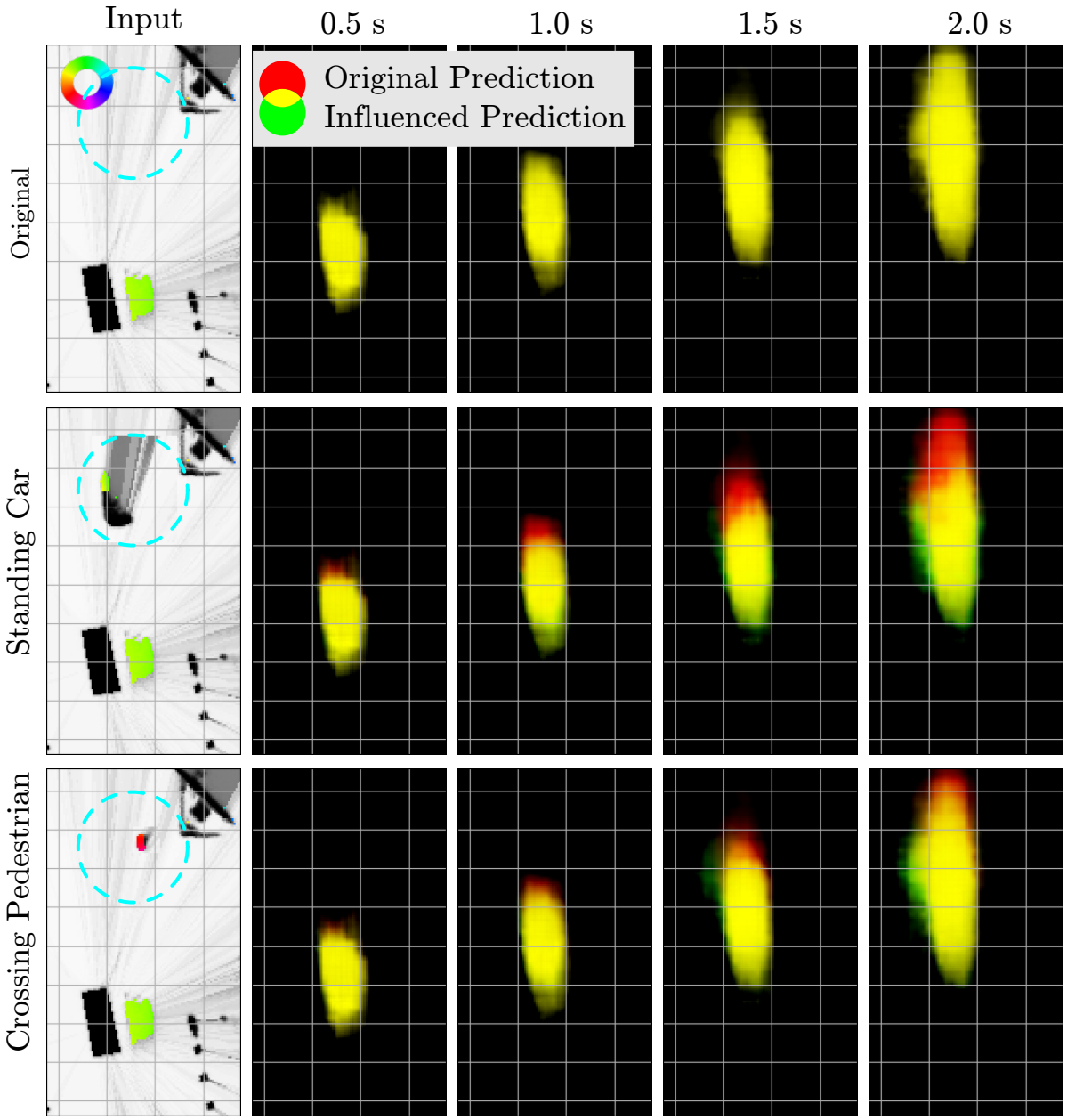

Figure 5: Influence of other vehicles on prediction: In the original scene (top row), a vehicle drives upwards on a free lane. A standing car on the lane (second row) and a pedestrian approaching the lane (third row) was pasted into the input DOGMa. The network predicts the vehicle to stop before the parking car and to slow down for the approaching pedestrian. The prediction results are illustrated in the 4 right columns as RGB images, where the original prediction is drawn in the red channel, the prediction for the manipulated input in the green channel. An overlap appears yellow.

Ulm, Institut für Mess-, Regel- und Mikrotechnik, 2017. [Online]. Available: http://dx.doi.org/10.18725/OPARU-4361

[2] S. Hoermann, M. Bach, and K. Dietmayer, "Dynamic Occupancy Grid Prediction for Urban Autonomous Driving: A Deep Learning Approach with Fully Automatic Labeling," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, to be published.

[3] Tesla driver dies in first fatal crash while using autopilot mode. Visited on 2018-07-06. [Online]. Available: https://www.theguardian.com/technology/2016/jun/30/tesla-autopilot-death-self-driving-car-elon-musk

[4] Technikexperte beantwortet die wichtigsten fragen zum uber-unfall. Visited on 2018-07-06. [Online]. Available: http://www.handelsblatt.com/21103570.html

[5] S. Hoermann, P. Henzler, M. Bach, and K. Dietmayer, "Object Detection on Dynamic Occupancy Grid Maps Using Deep Learning and Automatic Label Generation," in *IEEE Intelligent Vehicles Symposium Proceedings*, June 2018. [Online]. Available: https://arxiv.org/abs/1802.02202

[6] D. Stumper, F. Gies, S. Hoermann, and K. Dietmayer, "Offline Object Extraction from Dynamic Occupancy Grid Map Sequences," in *IEEE Intelligent Vehicles Symposium Proceedings*, June 2018. [Online]. Available: https://arxiv.org/abs/1804.03933

[7] F. Piewak, T. Rehfeld, M. Weber, and J. M. Zoellner, "Fully convolutional neural networks for dynamic object detection in grid maps," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, June 2017, pp. 392–398.

[8] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Computer Vision – ECCV 2016*, 2016, pp. 21–37.

[9] J. Dequaire, D. Rao, P. Ondruska, D. Z. Wang, and I. Posner, "Deep tracking on the move: Learning to track the world from a moving vehicle using recurrent neural networks," *CoRR*, vol. abs/1609.09365, 2016. [Online]. Available: http://arxiv.org/abs/1609.09365

[10] J. Dequaire, P. Ondruska, 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*, 2017. [Online]. Available: https://doi.org/10.1177/0278364917710543

[11] W. Zhan, A. Fortelle, Y. Chen, Y. Chan, and M. Tomizuka, "Probabilistic prediction from planning perspective: Problem formulation, representation simplification and evaluation metric," in *IEEE Intelligent Vehicles Symposium Proceedings*, June 2018.

[12] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec 2015, pp. 1520–1528.