The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XLVIII-2/W3-2023
ISPRS Intl. Workshop “Photogrammetric and computer vision techniques for environmental and infraStructure monitoring, Biometrics and Biomedicine”
PSBB23, 24–26 April 2023, Moscow, Russia

MONITORING OF DYNAMIC OBJECTS ON A 2D OCCUPANCY MAP USING NEURAL NETWORKS AND MULTIMODAL DATA

A. S. Krishtopik[1,2], D. A. Yudin[1]

[1] Moscow Institute of Physics and Technology (National Research University), Moscow, 141701 Russia
[2] Federal Research Center Computer Science and Control, Russian Academy of Sciences, Moscow, 119333 Russia

Commission II, WG II/8

KEY WORDS: Mobile robot, Dynamic environment, Dynamic occupancy map, Multimodal approach, Instance segmentation, Image, LiDAR Point Cloud

ABSTRACT:

The paper deals with the construction of dynamic occupancy maps, where the grid cell can contain not only information about the presence or absence of an obstacle, but also information about its velocity. We propose a multimodal approach to constructing 2D dynamic occupancy maps from LiDAR point clouds and camera images. The approach involves building a static occupancy map from LiDAR data and then adding information about cell velocities based on neural network instance segmentation and object tracking in monocular onboard camera images. Pedestrians and vehicles were considered as dynamic objects. One of the important stages is the projection of the object masks found in the images onto a 2D occupancy map. We compared the proposed approach with the classical method of constructing dynamic occupancy maps from LiDAR data based on the Monte Carlo method. An experimental quality evaluation of the approaches was carried out using the popular Mapillary Vistas and Waymo Open Datasets containing street scenes and a large number of dynamic objects. We demonstrate that the considered approaches can work in real time, which indicates the possibility of their application as part of the on-board control systems of autonomous cars or ground mobile robots. The software implementation of the proposed dynamic occupancy reconstruction approach is publicly available at the link: https://github.com/andrey1908/nn-dynamic-occupancy.

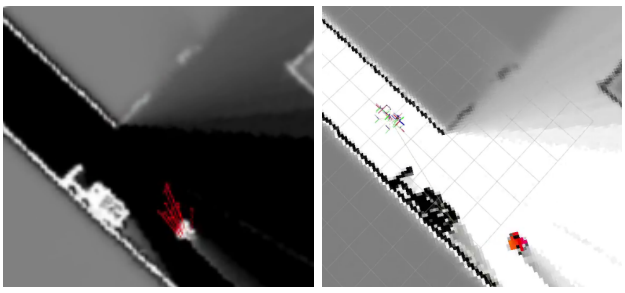
1. INTRODUCTION

The construction of occupancy map as 2D grid (Moravec and Elfes, 1985) is simple and useful way of representing surrounding environment. A low probability of occupancy means that there is no obstacle in this cell and it is possible to pass through it, and a high probability of occupancy means that there is an obstacle in the cell and it is impossible to pass through it. This representation of the environment is convenient for use in planning algorithms. Planning algorithms use occupancy maps to avoid obstacles when moving from one point of the map to another.

Occupancy maps are well suited for representing static environments. However, a presence of a dynamic obstacles can make it more difficult to navigate in the environment. Scenes with dynamic objects need additional information about dynamic state of the map to represent moving objects. For this purpose, dynamic occupancy maps are generated with information about states of occupied cells (dynamic or static) and optionally velocity of dynamic cells.

The Figure 1 shows examples of visualization of a dynamic occupancy map built from mobile robot LiDAR data. The figure shows two visualization options: visualization of the velocity vectors of the map cells and visualization of the velocity vector directions using colored pixels. Cells in which particles have a large spread in velocities are considered static and their velocities are not displayed.

In our paper we propose an multimodal approach for reconstruction dynamic occupancy maps using neural network-based image instance segmentation and LiDAR data. This approach is described in Section 3.1.



(a) (b)

Figure 1. Visualization of the dynamic occupancy maps: (a) - visualization with velocity vectors, (b) - visualization of velocity using colored pixels.

We also compare its performance on a fragment of the Waymo Open Dataset (Sun et al., 2020) with the classical approach based on the Monte Carlo method (Section 3.2), and evaluate their performance.

2. RELATED WORK

Dynamic occupancy map reconstruction. To estimate velocities on occupancy map different approaches are used. In approach described in (Coué et al., 2006) authors use 4D grid to represent dynamic map. Two additional dimensions are used to represent velocities of the cells. Significant disadvantage of this approach is that it requires too much time to process 4D grid, as there are a lot of cells.

Another approach (Pietzsch et al., 2009) uses LiDAR to build occupancy map. Changes of cells states on occupancy grid are

This contribution has been peer-reviewed.
https://doi.org/10.5194/isprs-archives-XLVIII-2-W3-2023-137-2023 | © Author(s) 2023. CC BY 4.0 License.

monitored to detect dynamic objects. Velocity of dynamic objects are determined using radar measurements.

There are approaches that don't use velocities from sensors, but determine velocities by some method. For example approach described in (Danescu et al., 2011) uses particle-based method. First stereo image is used to build occupancy grid map. Then particles representing assumptions about occupied cells velocities are initialized. Each next occupancy grid map is used to verify particles that successfully predicted displacement of occupied cells and particles reevaluation takes place. Another particle filter-based approaches are (Tanzmeister et al., 2014), (Rummelhard et al., 2015), (Nuss et al., 2018). Using particles allows to parallelize computation efficiently, that allows to widely use these approaches on different mobile platforms.

Another way to determine dynamic objects is represented in (Li and Ruichek, 2014). This method uses stereo image sensor to estimate odometry of the ego-vehicle. Outliers from odometry estimation considered to lie on dynamic objects. To segment dynamic objects on the image, depth images generated from stereo images are used.

Dynamic objects can be segmented not only by using depth images, but also using semantic segmentation of camera images. This approach is used in (Shepel et al., 2021). Semantic labels of potentially dynamic objects are mapped onto occupancy map. The map itself is built using stereo images.

Object segmentation methods. The main modern methods for segmenting a scene surrounding a vehicle and including dynamic objects (people, cars, etc.) are deep neural networks of various architectures.

Object segmentation on a three-dimensional scene using LiDAR data can be carried out:

- directly in LiDAR point clouds, such as PointNet (Qi et al., 2017) or KPConv (Thomas et al., 2019);
- by constructing an intermediate spherical projection like SqueezeSegV3 (Xu et al., 2020a), SalsaNeXt (Cortinhal et al., 2020) and its modifications (Stokolesov and Yudin, 2021);
- using a voxel representation of the scene, such as Cylinder3D (Zhou et al., 2020a) and AF2S3Net (Cheng et al., 2021b);
- using a complex representation of scenes. For example, SPVNAS (Xu et al., 2021) uses a simultaneous point-voxel representation of the scene, RPVNet (Xu et al., 2021) uses the representation of the scene in the form of projections, points and voxels;
- with multimodal approaches, such as 2DPASS(Yan et al., 2022), which simultaneously use LiDAR and camera data and demonstrate the best results on popular benchmarks.

It should be noted that the existing open implementations of the highest quality LiDAR data segmentation methods (multimodal or using a complex representation of the scene) are far from the possibility of their operation in real time.

In this regard, in our paper, the main attention is paid to the methods of dynamic object segmentation in the images of onboard cameras. One of the basic methods is the Mask R-CNN approach (He et al., 2017),(Yudin et al., 2019), but it loses significantly in quality to the methods of DeterctoRS(Qiao et al., 2021), HTC++(Liu et al., 2021b), and recently appeared transformer models, for example Mask2former(Cheng et al., 2021a), Oneformer(Jain et al., 2022), etc.

The possibility of real-time performance for a neural network model is the most important criterion for application in approaches related to the online mapping in robotics. One of the fastest instance segmentation models is YOLACT Edge (Liu et al., 2021a). Another model that provides similar performance but significantly higher quality is the recently introduced YOLOv8 (Jocher et al., 2023) model. In our paper we focus on usage of such real-time models.

Object tracking methods. We consider the problem of multi-object tracking, where until now one of the basic algorithms is the Hungarian algorithm for association of found object identifiers (Basharov and Yudin, 2021).

The recognition of identifiers of different objects (instances) and their tracking on a sequence of data, in particular images, can be carried out based on simple kinematic motion models and using the Kalman filter, for example, using the SORT (Bewley et al., 2016) or ByteTrack (Zhang et al., 2022) approach.

Another approach is to predict unique vector representations (embeddings) for each of the objects, this is dominated by neural network models, for example, PointTrack++(Xu et al., 2020b), CenterTrack(Zhou et al., 2020b). There are transformer models for tracking objects, for example, TrackFormer(Meinhardt et al., 2022).

Popular modern approaches are real-time neural network modifications of the SORT approach, for example, DeepSORT (Wojke et al., 2017), StrongSORT (Du et al., 2023), etc.

A limitation of using neural network approaches for tracking objects is their possible overfitting for a specific data domain.

3. METHODOLOGY

3.1 Multimodal approach for 2D occupancy reconstruction using neural networks

Approach for 2D dynamic occupancy reconstruction. Our approach uses image instance segmentation and tracking algorithms to recognize potentially dynamic objects. Fig. 2 gives an overview of our approach. Camera images are used to segment and track dynamic objects (people, vehicles, etc.). Tracking information is added to LiDAR point cloud using extrinsic camera-LiDAR calibration and then static occupancy map with tracking information is built. To estimate velocity of tracked objects on the map we determine displacement of each tracked object between two sequential maps and using time between this maps estimate velocity. Detailed description of each step is given in the following paragraphs.

Image instance segmentation neural network This section describes approach used to segment potentially dynamic objects on images. We used YOLOv8 (Jocher et al., 2023) segmentation network and fine-tuned it on Mapillary Vistas dataset (Neuhold et al., 2017). Dataset images were split into training and validation sets containing 18,000 and 2,000 images respectively. For training and validation were used two categories: vehicle and person. Mapillary Vistas dataset contains

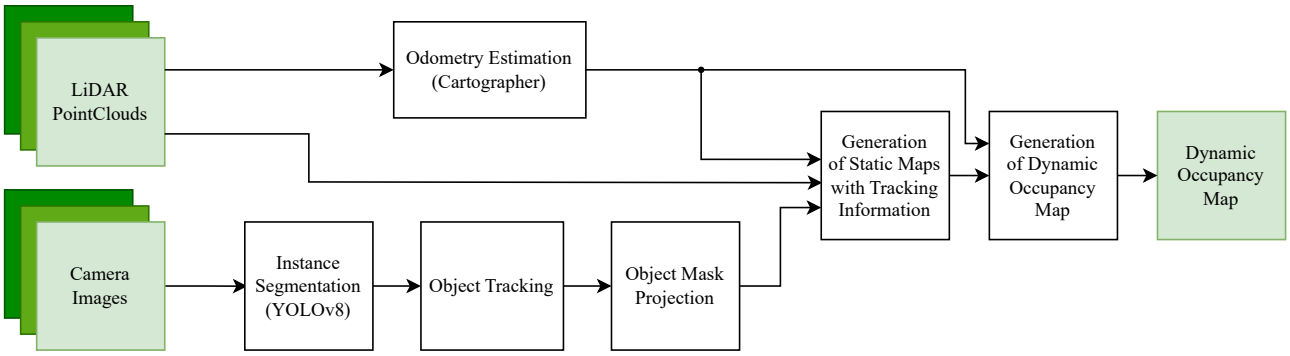


Figure 2. Structure of the proposed approach for 2D dynamic occupancy reconstruction based on LiDAR and monocular camera data.

more detailed category labels, so we combined multiple categories into one. Vehicle category contains Bus, Car, Caravan, Trailer, Truck and Other Vehicle categories from original dataset categories. Person contains Person, Bicyclist, Motorcyclist and Other Rider. Number of person instances is about 60,000, vehicle instances - about 140,000. For training and validation we used input resolution 640x640 to ensure low latency of semantic segmentation.

Dynamic object tracking approach. To perform tracking of detected objects we use multi-object tracking algorithm ByteTrack (Zhang et al., 2022) which has effective software implementations (Broström, 2023). This implementation is specialized to work with YOLOv8 networks. It was modified to output tracking images where each tracked object is shown with a unique color.

Static occupancy map generation using LiDAR data. To build static occupancy grid map we use LiDAR data. Ego vehicle motion is estimated using real-time SLAM algorithm Cartographer (Hess et al., 2016). First, points from LiDAR point cloud are filtered by their height. Only points that are considered obstacles for the ego vehicle are left, that is low points on the ground and high points above the ego vehicle are filtered out. Remaining obstacle points are projected onto the horizontal plane, simply by removing Z coordinate. Then each 2D projected point is assigned to a cell on the grid map making that cell occupied. After that ray tracing algorithm fills gaps between ego vehicle and occupied cells creating empty areas with free cells. We call occupancy grid built from one LiDAR point cloud a local map.

To filter out noise in LiDAR measurements we combine two sequential local maps into one occupancy grid map. To do this we shift local maps according to the ego vehicle motion and mark a cell in output occupancy grid map occupied only if this cell is occupied in both local maps. Resulting map shows only surrounding environment around the ego vehicle and do not remember passed locations.

Dynamic occupancy map reconstruction using camera image. We use camera to retrieve semantic information and add it to occupancy map. First, we perform image instance segmentation on the input image to produce semantic image (mask). Semantic image shows detected objects with colors corresponding to the object categories. Then LiDAR points are projected onto semantic image using extrinsic LiDAR-camera calibration and intrinsic camera parameters. Each LiDAR point projected onto semantic image takes color of the pixel where this point was projected to. Thus we can retrieve colored point cloud, where color of the point corresponds to the semantic label of the detected object. Building occupancy map using colored point clouds allows us to add semantic information to a 2D grid.

If tracking image is used instead of semantic image, then color on occupancy grid map shows not object category, but unique tracked object itself.

To formalize the process of assigning color from tracked image to LiDAR points, we use following expressions:

$$\widetilde{p_i} = T_{C \to L} * p_i,$$

$$s * \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = K * \widetilde{p_i},$$

$$K = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix},$$

where $T_{C \to L}$ - 4x4 transformation matrix, that transforms camera frame to LiDAR frame (extrinsic camera-LiDAR calibration), p_i - LiDAR point, K - intrinsic camera parameters, s - normalization factor, (u, v) - image pixel where LiDAR point p_i is projected.

3.2 Dynamic occupancy map reconstruction based on Monte Carlo method

In our paper we investigate the Dynamic Occupancy Grid Map (DOGM) algorithm (Nuss et al., 2018). It is a particle filter based approach for building dynamic occupancy grid maps. As input, the algorithm takes a sequence of static occupancy maps and position of the robot to perform motion compensation. Motion compensation is needed to subtract the movement of the robot from the movement of obstacles.

The main idea of the algorithm is to represent the dynamic occupancy map as a set of a large number of particles (10^6) on a plane. Each particle has coordinates on the plane, velocity projections and weight. The weight determines the contribution of the particle to the probability of occupancy of the cell in which it is located. That is, the sum of the weights of particles in a cell is equal to the probability of its occupancy.

Initialization of the particles occurs during the processing of the first static occupancy map. Particles are generated in occupied cells, and the higher the probability of occupancy, the

more particles are generated in that cell. Particle velocities are distributed randomly with the mathematical expectation equal to zero. The weights of all particles are chosen to be the same and the sum of particle weights in a cell is equal to occupancy probability of that cell.

Each time before updating the particles with a new static map, the position of the particles is predicted based on their velocities. Particles are shifted from their current position, taking into account their velocities and the time elapsed from the moment the previous static map was processed until the moment when the new map was received.

Part of the static map around the robot is used to update the particles. Since the robot is moving, the area of the map that is used to update the particles is also moving, and relative to this area, all static obstacles are moving at a speed opposite to the speed of the robot. In order for static obstacles to remain static when the robot moves, it is necessary to perform motion compensation. To do this, all particles are shifted opposite to the displacement of the robot, that is, in the same way as static obstacles are displaced when the robot moves relative to the latter.

After predicting the position of particles and compensating for motion, the particle weights are updated and new particles are generated. Particle weights are updated according to which areas of the static map they fall into after predicting their position. If a particle hits a free area, then its weight decreases, if it enters an occupied area, then it increases. New particles are generated in those cells where there are few particles, but which have a high probability of being occupied on a static map. The velocities of the new particles are initialized by a random distribution with the mathematical expectation equal to zero.

After the generation of new particles, a selection is made from old and new particles so that the number of particles remains unchanged. After selection, the probability of occupancy of each cell of the dynamic map is calculated as the sum of the weights of the particles in this cell, and the obstacle velocity in the cell is calculated as the average of the particle velocities in this cell. Thus, a dynamic occupancy map is formed that contains information not only about the probabilities of cell occupancy, but also about their velocities.

4. EXPERIMENTS

4.1 Hardware and Software Setup

All the experiments are performed on desktop computer with Intel(R) Core(TM) i5-8400 CPU @ 2.80GHz, NVidia GeForce RTX 2080 GPU and on embedded platform NVidia Jetson Xavier.

4.2 Image Instance Segmentation Results

We trained two versions of YOLOv8 (Jocher et al., 2023): YOLOv8 nano and YOLOv8 medium. These networks differ in number on parameters, so YOLOv8 medium is more accurate but YOLOv8 nano is faster.

For training we used Mapillary Vistas dataset. Training was performed on two classes: person and car. Input image resolution was 640x640. Table 1 shows metrics on validation set for selected classes. These metrics are good enough for our approach. Fig. 3 shows inference examples of YOLOv8 nano network on validation images of Mapillary Vistas dataset.

Table 1. The quality of instance segmentation on the Mapillary Vistas dataset

	person AP	vehicle AP	mAP
YOLOv8 nano	30.8	54.2	42.5
YOLOv8 medium	41.5	60.9	51.2

Table 2. The performance of instance segmentation using NVidia GeForce RTX2080

	Inference time, ms
YOLOv8 nano	6.9
YOLOv8 medium	9.6

Performance of both YOLOv8 nano and medium are presented in Table 2. We could achieve high performance due to using small network architectures and small input image size (640x640). High performance is important for our approach since it is intended to be used in dynamic environments, where low response time is required.

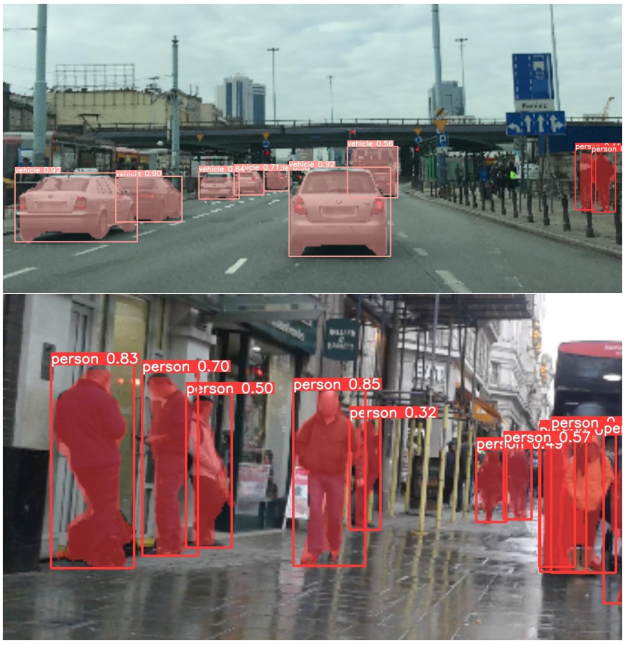


Figure 3. Example of YOLOv8 nano inference on validation images of Mapillary Vistas dataset.

4.3 Dynamic Object Tracking Results

Table 3 shows tracking metrics on a fragment of the Waymo Open dataset (3D Tracking task). When computing metrics ground truth odomtery was used instead of Cartographer odometry. We used ground truth odometry to exclude influence of odometry drift on tracking metrics.

From the Table 3 we can see that tracking metrics are quite low. There are multiple reasons for that. First of all, we used low input resolution for image segmentation (640x640) to ensure high processing speed, because information about dynamic objects should be available as soon as possible to make decisions about collision avoidance. So objects located far away are hard to segment and track due to low input image resolution. Another reason for low metrics is that we can track only objects in front of camera, while Waymo Open dataset provides tracking

information about all objects that are visible in LiDAR point cloud. While LiDAR can retrieve information from all directions, camera can see only small area in front of ego vehicle. One more reason is that instance segmentation and projection of semantic information on LiDAR data might be inaccurate, which causes false detections on the background structures (e.g. walls).

Later we will show two examples of tracking dynamic objects with both our approach and Monte Carlo method. We will be able to see, that our method successfully tracks dynamic objects on the scene.

Table 3. Tracking metrics on a fragment of the Waymo Open dataset using ground truth odometry

	Recall	Precision AP	MOTP
YOLOv8 nano	0.0351	0.2428	0.3486
YOLOv8 medium	0.0392	0.2427	0.3502

4.4 Dynamic Occupancy Map Reconstruction Results

Performance of neural network-based approach. Table 4 presents performance estimation results for our neural-network-based approach on Nvidia GeForce RTX 2080. The map was chosen to be 100x100 meters in size. All pipeline takes 22 ms, which is acceptable in dynamic environments.

Table 4. Performance of the neural network-based approach

Neural network-based stage	Nvidia GeForce RTX 2080 (100x100m map)
Semantic segmentation (for YOLOv8 medium)	9,6
Tracking	1,3
Occupancy grid building	9,8
Velocity estimation	1,3

Performance of the approach based on the Monte Carlo method. Approach testing was performed on static occupancy maps built from LiDAR data. The Table 5 presents the results of measuring the performance of the Monte Carlo method-based approach on various platforms. All pipeline takes 28.5 ms on the same platform that our approach was tested on. This time does not include static occupancy grid building and map resolution is much smaller. But this approach also shows good performance to be used in dynamic environments.

Table 5. Performance of the approach based on the Monte Carlo method for various platforms

Monte-Carlo approach stage	Nvidia GeForce RTX 2080 (20x20m map)	Nvidia Jetson Xavier (10x10m map)
Converting a static occupancy map from ROS format to a format for updating a dynamic map, ms	2	8
Motion compensation, ms	0,5	1
Dynamic map update, ms	23	39
Converting a dynamic occupancy map to ROS format, ms	3	1

Example of Map Reconstruction Fig. 4 shows an example of dynamic occupancy map reconstruction using both approaches: based on a neural network and the Monte Carlo method. The figure shows two different scenes in two columns. The first row shows dynamic map built by Monte Carlo method. The map highlights moving cells with a certain color depending on the direction where the cell is moving. The second row provides additional information about velocities of dynamic cells on the map built by Monte Carlo method. Velocities of dynamic cells are shown with arrows. The third row shows dynamic map built by neural network-based approach. Segmented objects are highlighted on the map with the same color as on the segmented image (the fourth row). Arrows show the velocity of tracked objects. The fourth row presents segmented images after tracking, that is color of segmented object represents unique tracking identifier of that object.

From these two examples we can see that neural network-based approach successfully detects and tracks dynamic objects on the scene. Estimated velocity is not much different from the velocity estimated by Monte Carlo method, as can be seen from the pictures.

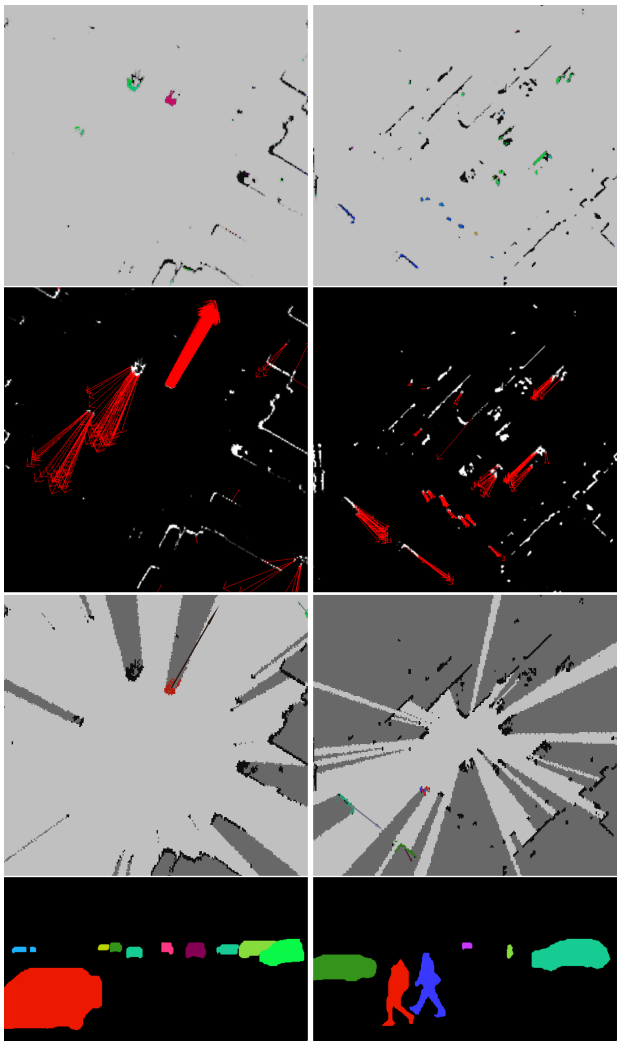


Figure 4. Example of dynamic map reconstruction. Each column is a separate example. The first row is Monte Carlo method result visualization (color indicates velocity direction). The second row is Monte Carlo method result with each dynamic cell velocity shown with an arrow. The third row is result of neural network-based approach. The fourth row contains image segmentation results with color representing the tracking id.

5. CONCLUSION

State-of-the-art neural networks such as YOLOv8 allow to perform fast yet accurate image instance segmentation. Using tracking results and accurate LiDAR range data we can

build precise occupancy grid maps with dynamic object location and velocity estimation. We provided examples of successful dynamic objects detection and velocity estimation by two approaches: based on neural network and on Monte Carlo method. Promising directions for improving the proposed neural network-based approach are the usage of images from a larger number of cameras installed around the car and the ability to work with an increased image resolution. It should be noted that the real-time performance of the approaches discussed in this paper allows them to be used in on-board systems of autonomous cars and ground mobile robots.

ACKNOWLEDGMENTS

This work was supported by Russian Science Foundation, project №22-21-00716.

REFERENCES

Basharov, I., Yudin, D., 2021. Real-time deep neural networks for multiple object tracking and segmentation on monocular video. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 44, 15–20.

Bewley, A., Ge, Z., Ott, L., Ramos, F., Upcroft, B., 2016. Simple online and realtime tracking. *2016 IEEE international conference on image processing (ICIP)*, IEEE, 3464–3468.

Broström, M., 2023. Real-time multi-object tracking and segmentation using Yolov8 with StrongSORT and OSNet.

Cheng, B., Choudhuri, A., Misra, I., Kirillov, A., Girdhar, R., Schwing, A. G., 2021a. Mask2former for video instance segmentation. *arXiv preprint arXiv:2112.10764*.

Cheng, R., Razani, R., Taghavi, E., Li, E., Liu, B., 2021b. 2-s3net: Attentive feature fusion with adaptive feature selection for sparse semantic segmentation network. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 12547–12556.

Cortinhal, T., Tzelepis, G., Erdal Aksoy, E., 2020. Salsanext: Fast, uncertainty-aware semantic segmentation of lidar point clouds. *Advances in Visual Computing: 15th International Symposium, ISVC 2020, San Diego, CA, USA, October 5–7, 2020, Proceedings, Part II 15*, Springer, 207–222.

Coué, C., Pradalier, C., Laugier, C., Fraichard, T., Bessière, P., 2006. Bayesian occupancy filtering for multitarget tracking: an automotive application. *The International Journal of Robotics Research*, 25(1), 19–30.

Danescu, R., Oniga, F., Nedevschi, S., 2011. Modeling and tracking the driving environment with a particle-based occupancy grid. *IEEE Transactions on Intelligent Transportation Systems*, 12(4), 1331–1342.

Du, Y., Zhao, Z., Song, Y., Zhao, Y., Su, F., Gong, T., Meng, H., 2023. Strongsort: Make deepsort great again. *IEEE Transactions on Multimedia*.

He, K., Gkioxari, G., Dollár, P., Girshick, R., 2017. Mask r-cnn. *Proceedings of the IEEE international conference on computer vision*, 2961–2969.

Hess, W., Kohler, D., Rapp, H., Andor, D., 2016. Real-time loop closure in 2d lidar slam. *2016 IEEE international conference on robotics and automation (ICRA)*, IEEE, 1271–1278.

Jain, J., Li, J., Chiu, M., Hassani, A., Orlov, N., Shi, H., 2022. OneFormer: One Transformer to Rule Universal Image Segmentation. *arXiv preprint arXiv:2211.06220*.

Jocher, G., Chaurasia, A., Qiu, J., 2023. YOLO by Ultralytics.

Li, Y., Ruichek, Y., 2014. Occupancy grid mapping in urban environments from a moving on-board stereo-vision system. *Sensors*, 14(6), 10454–10478.

Liu, H., Soto, R. A. R., Xiao, F., Lee, Y. J., 2021a. Yolactedge: Real-time instance segmentation on the edge. *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 9579–9585.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B., 2021b. Swin transformer: Hierarchical vision transformer using shifted windows. *Proceedings of the IEEE/CVF international conference on computer vision*, 10012–10022.

Meinhardt, T., Kirillov, A., Leal-Taixe, L., Feichtenhofer, C., 2022. Trackformer: Multi-object tracking with transformers. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 8844–8854.

Moravec, H., Elfes, A., 1985. High resolution maps from wide angle sonar. *Proceedings. 1985 IEEE international conference on robotics and automation*, 2, IEEE, 116–121.

Neuhold, G., Ollmann, T., Rota Bulo, S., Kontschieder, P., 2017. The mapillary vistas dataset for semantic understanding of street scenes. *Proceedings of the IEEE international conference on computer vision*, 4990–4999.

Nuss, D., Reuter, S., Thom, M., Yuan, T., Krehl, G., Maile, M., Gern, A., Dietmayer, K., 2018. A random finite set approach for dynamic occupancy grid maps with real-time application. *The International Journal of Robotics Research*, 37(8), 841–866.

Pietzsch, S., Vu, T. D., Burlet, J., Aycard, O., Hackbarth, T., Appenrodt, N., Dickmann, J., Radig, B., 2009. Results of a precrash application based on laser scanner and short-range radars. *IEEE Transactions on Intelligent Transportation Systems*, 10(4), 584–593.

Qi, C. R., Su, H., Mo, K., Guibas, L. J., 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 652–660.

Qiao, S., Chen, L.-C., Yuille, A., 2021. Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10213–10224.

Rummelhard, L., Negre, A., Laugier, C., 2015. Conditional monte carlo dense occupancy tracker. *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, IEEE, 2485–2490.

Shepel, I., Adeshkin, V., Belkin, I., Yudin, D. A., 2021. Occupancy grid generation with dynamic obstacle segmentation in stereo images. *IEEE Transactions on Intelligent Transportation Systems*, 23(9), 14779–14789.

Stokolesov, M., Yudin, D., 2021. Improvement of projection-based lidar data segmentation algorithms using object-contextual representations. *Journal of Physics: Conference Series*, 1925number 1, IOP Publishing, 012035.

Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B. et al., 2020. Scalability in perception for autonomous driving: Waymo open dataset. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2446–2454.

Tanzmeister, G., Thomas, J., Wollherr, D., Buss, M., 2014. Grid-based mapping and tracking in dynamic environments using a uniform evidential environment representation. *2014 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 6090–6095.

Thomas, H., Qi, C. R., Deschaud, J.-E., Marcotegui, B., Goulette, F., Guibas, L. J., 2019. Kpconv: Flexible and deformable convolution for point clouds. *Proceedings of the IEEE/CVF international conference on computer vision*, 6411–6420.

Wojke, N., Bewley, A., Paulus, D., 2017. Simple online and realtime tracking with a deep association metric. *2017 IEEE international conference on image processing (ICIP)*, IEEE, 3645–3649.

Xu, C., Wu, B., Wang, Z., Zhan, W., Vajda, P., Keutzer, K., Tomizuka, M., 2020a. Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation. *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII 16*, Springer, 1–19.

Xu, J., Zhang, R., Dou, J., Zhu, Y., Sun, J., Pu, S., 2021. Rpvnet: A deep and efficient range-point-voxel fusion network for lidar point cloud segmentation. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 16024–16033.

Xu, Z., Zhang, W., Tan, X., Yang, W., Su, X., Yuan, Y., Zhang, H., Wen, S., Ding, E., Huang, L., 2020b. Pointtrack++ for effective online multi-object tracking and segmentation. *arXiv preprint arXiv:2007.01549*.

Yan, X., Gao, J., Zheng, C., Zheng, C., Zhang, R., Cui, S., Li, Z., 2022. 2dpass: 2d priors assisted semantic segmentation on lidar point clouds. *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVIII*, Springer, 677–695.

Yudin, D., Ivanov, A., Shchendrygin, M., 2019. Detection of a human head on a low-quality image and its software implementation. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 42, 237–241.

Zhang, Y., Sun, P., Jiang, Y., Yu, D., Weng, F., Yuan, Z., Luo, P., Liu, W., Wang, X., 2022. Bytetrack: Multi-object tracking by associating every detection box. *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXII*, Springer, 1–21.

Zhou, H., Zhu, X., Song, X., Ma, Y., Wang, Z., Li, H., Lin, D., 2020a. Cylinder3d: An effective 3d framework for driving-scene lidar semantic segmentation. *arXiv preprint arXiv:2008.01550*.

Zhou, X., Koltun, V., Krähenbühl, P., 2020b. Tracking objects as points. *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV*, Springer, 474–490.