ADVANCES IN CYBER-PHYSICAL SYSTEMS

Vol. 10, No. 1, 2025

IMPROVING THE LOCALIZATION OF MOBILE ROBOT BY FILTERING DYNAMIC OBJECTS USING CAMERA IMAGE SEGMENTATION

Maksym Shavarskyi, Yurii Kryvenchuk

Lviv Polytechnic National University, 12, S. Bandery str., Lviv, 79013, Ukraine
Authors' e-mails: *maksym.a.shavarskyi@lpnu.ua, yurii.p.kryvenchuk@lpnu.ua*

https://doi.org/10.23939/acps2025.01.117

Submitted on 10.03.2025

***Abstract*: This paper presents an approach to improving mobile robot localization by filtering dynamic objects using camera image segmentation. The proposed algorithm integrates a Particle Filter with state-of-the-art computer vision techniques, specifically employing the YOLO model for segmentation, which effectively differentiates static elements of the environment from moving objects. This approach reduces the impact of noisy data and enhances localization accuracy in dynamic conditions, which is crucial for the reliable autonomous operation of mobile robots.**

***Index terms*: localization, particle Filter, neural network, YOLOv8, LiDAR, camera, segmentation.**

I. INTRODUCTION

Modern robots use localization methods for orientation in complex environments. One popular approach is the Particle Filter algorithm, which allows estimating the spatial location of the robot using statistical methods. However, dynamic obstacles significantly reduce the localization accuracy, because they can cause the algorithm to mistakenly assume static features of the room. To improve the reliability and accuracy of localization, it is necessary to develop a strategy that separates dynamic elements from static ones, based on additional information from visual sensors. Modern robots navigating complex indoor environments rely heavily on probabilistic localization methods. Traditional filters, such as the Extended Kalman Filter (EKF) and Particle Filter, assume a static world and fixed noise characteristics, which can lead to divergence or degraded performance when process or measurement uncertainties are mis-specified. Nguyen et al. [1] mitigate this by embedding a fuzzy logic system within the EKF to adaptively tune its noise covariance matrices via a neural network, yielding more accurate pose estimates under varying noise conditions. Likewise, floor plan based methods employ particle filters to match sensor features against a priori maps. Boniardi et al. [2] use a convolutional network to extract room layout edges from monocular images and feed these into a particle filter, achieving robust localization against map inaccuracies without exhaustive sensor map data collection. More recently, deep learning approaches have further advanced map representation and global localization. Zang et al. [3] introduce Local_INN, an invertible neural network that learns an implicit map representation in the forward pass and recovers pose (with uncertainty estimates) in the inverse pass. They demonstrate on par performance with existing methods at much lower latency, and show both local and global (kidnapping scenario) localization using their INN framework [3]. Cabrera et al. (2024) explore a Siamese CNN architecture on omnidirectional imagery, framing global localization as an image retrieval problem: descriptors from a paired network indicate room identity, and a downstream particle filter refines the pose (particularly improving performance under challenging lighting conditions) [4]. However, all these approaches, whether filter based or learning based on implicitly treat the environment as static. Dynamic obstacles (e. g., moving people or carts) introduce spurious measurements that can mislead both statistical filters and learned models into treating transient features as part of the static map. Akai's Slader [5] begins to address this by jointly modeling pose uncertainty and sensor based environment recognition, fusing map priors with online LiDAR based semantic detections to recognize "no entry" zones and open doors. Yet Slamer still assumes that dynamic changes can be recognized as semantic classes, rather than truly isolating moving elements from static structure.

II. LITERATURE REVIEW AND PROBLEM STATEMENT

Several methods have been proposed to localize robots or, more generally, devices, in 2D maps using RGB and range/depth measurements. For example, the approaches proposed by Wolf et al. [7] and Bennewitz et al. [8] use MCL and employ a database of images recorded in an indoor environment. Mendez et al. [9] proposed a sensor model for MCL that leverages the semantics of the environment, namely doors, walls and windows, obtained by processing RGB images with a CNN. They enhance the standard likelihood fields for the occupied space on the map with suitable likelihood fields for doors and windows. Although such a sensor model can be also adapted to handle range-less measurements, it shows increased accuracy with respect to standard MCL only when depth measurements are used.

[12] was proposed recently to localize a camera in a 2D map. The authors employ a CNN for floor segmen-

tation from which they identify which lines in an edge image belong to the floor plan. The detected edges are reprojected into the 3D world using the current estimate of the floor plane. They are then used as virtual measurement in an extended Kalman filter.

Adaptive variants of the Extended Kalman Filter (EKF) and Particle Filter have been proposed to improve pose estimation under uncertain noise models. Nguyen et al. integrate a fuzzy neural system into the EKF to tune its process and measurement covariances online, yielding more robust localization when sensor noise characteristics vary unexpectedly [1]. Zang et al. further explore an invertible neural network within a Particle Filter framework, enabling efficient implicit map representation and rapid pose recovery in both local and kidnapping scenarios [3].

Classical SLAM algorithms assume a static world and thus suffer when dynamic obstacles introduce spurious observations. Akai's Slamer system augments a LiDAR based SLAM with semantic recognition of dynamic classes (e. g., doors, people), filtering out measurements that contradict the static map prior [5]. However, SLAMER relies on predefined semantic labels and still treats all nonsemantic outliers equivalently.

Extracting persistent structural features – especially planar surfaces such as walls has proven effective for rejecting dynamic points before localization. Boniardi et al. employ a CNN to extract room layout edges from monocular images and guide a Particle Filter, improving robustness to map inaccuracies without extensive retraining [2]. Cabrera et al. apply a Siamese CNN on omnidirectional imagery to identify room identity and support downstream filtering of nonstatic elements in a particle based framework [4].

While prior work either adapts filter parameters [1], learns implicit representations [3], or semantically labels dynamics [5], fewer approaches explicitly segment walls in LiDAR scans to differentiate static from dynamic returns. In this paper, we extend the Kalman Filter by first performing LiDAR based wall segmentation: each point is classified as "on wall" or "off wall" via planar fitting and learned appearance cues. We then exclude off wall points as potential dynamic outliers, feeding only the static subset into the filter. This hybrid strategy preserves the probabilistic rigor of the Kalman framework while leveraging structural priors to reject moving obstacles.

III. SCOPE OF WORK AND OBJECTIVES

This research presents a hybrid localization framework that enhances the classical Particle Filter by rejecting dynamic LiDAR measurements through wall based segmentation. To realize this, we first acquire synchronized RGB imagery and LiDAR point clouds in diverse indoor settings, ensuring precise temporal alignment and rigorous sensor calibration so that 3D points can be accurately mapped onto image pixels. Using state-of-the-art neural segmentation techniques, we then delineate wall surfaces within each RGB frame, leveraging high-capacity classification models to distinguish static structural elements from transient objects. Once segmentation masks are obtained, each LiDAR point is projected into the corresponding image plane via known camera intrinsics and extrinsics. Projection accuracy is verified through reprojection error analysis to guarantee that only well-aligned points proceed to the next stage. Points whose projections fall within the segmented wall regions are retained as reliable measurements, while those that land outside are treated as potential dynamic outliers and discarded. This selective filtering effectively isolates static elements of the environment, preventing moving objects, such as pedestrians or carts, from corrupting the localization process. Filtered wall points are incorporated into the observation model of the Particle Filter, where each particle's weight is updated exclusively based on these static cues. By excluding dynamic returns, the filter maintains a sharper, more focused likelihood distribution, resulting in faster convergence and reduced pose ambiguity. We evaluate the proposed approach against a baseline Particle Filter that uses all raw LiDAR measurements, measuring improvements in positional error, convergence speed, and robustness under varying levels of environmental dynamics. The primary objective of this work is to demonstrate that explicit wall-based dynamic-point rejection significantly enhances localization accuracy in highly dynamic indoor environments. In addition, we aim to establish best practices for segmentation-filter design and real-time sensor fusion parameters. To facilitate reproducibility and encourage further research, we will release our implementation alongside a curated evaluation dataset featuring annotated ground truth across multiple indoor scenarios.

IV. PARTICLE FILTER AND LIDAR USAGE OBJECTIVES

Localization of mobile robots is a fundamental problem in robotics, which consists in determining the exact position and orientation of the robot in the environment. This task is complicated by uncertainties associated with sensor measurements and dynamic changes in the environment. One of the most effective approaches to solving this problem is the use of Particle Filter (PF) – Monte Carlo methods that are used to estimate the state of nonlinear dynamical systems. PF allows you to represent the probability distribution of the robot's position using a set of random samples (particles), which provides flexibility and accuracy in modeling complex systems. Particle Filter works by representing the probability distribution of the state of the system as a set of particles, where each particle describes a possible state of the robot. The algorithm usually starts with an initialization of many particles, after which, in the prediction phase, each particle is moved according to the dynamic model. In the next phase, the particle weights are updated based on new sensor data, which allows you to assess how well each hypothesis is true. The final step is resampling, where particles with higher weights are duplicated and less likely ones are discarded, which helps to focus the probability distribution on the most promising

hypotheses. However, despite its effectiveness, Particle Filter has some limitations. Increasing the number of particles can lead to high computational costs, which makes it difficult to apply the algorithm in real time. In addition, the accuracy of the method depends on the quality of the initial initialization and motion model, as well as on the reliability of the sensor data, which may contain noise.

Dynamic interference causes false matches between LiDAR observations and the static map. This reduces the efficiency of the Particle Filter, as the algorithm receives information about moving objects that are not reflected in the static map, and therefore incorrectly estimates the robot's position.

V. LOCALIZATION ALGORITHM

The localization algorithm is the following:

1. Data collection: The robot receives images from RGB cameras and spatial data from the LiDAR sensor, which contain information about the 3D coordinates of the points.
2. Image segmentation: Using modern computer vision methods, each RGB image is segmented to identify walls. The use of neural networks or classification algorithms allows you to determine with high accuracy the image areas corresponding to static elements of the room.
3. Projection of LiDAR points: The obtained 3D coordinates of the points are projected onto the RGB image plane taking into account the camera parameters. This allows you to compare the spatial data with the obtained segmentation.
4. Selection of relevant points: If the projected point falls into the area marked as a "wall", it is stored for further use in the Particle Filter algorithm. Points that do not belong to the segmented wall are considered potentially dynamic and are excluded from the calculations.
5. Integration with Particle Filter: Filtered data is used to update particle weights, allowing for a more accurate estimate of the robot's position in a static map.

VI. DATA PREPARATION

The data collection process is as follows: the robot's wheelbase is equipped with sensors that allow simultaneous recording of images, 2D points from the lidar, and information about the current position relative to the origin of the map. The map is represented as a set of links and joints that describe the main elements of the environment (walls, doors, toilets, urinals). The map format is similar to URDF, but modified to meet the specific needs of the study.

The recorded data is downloaded from the robot and transformed to create a single structured dataset for testing. The first step is manual data filtering, which helps to avoid errors and exclude incorrect records. For comparative analysis, both the original dataset and the data that have been denoised and augmented are used, which helps to increase the generalizability of the developed localization system. The data for segmentation are given below:

The recorded data is downloaded from the robot and transformed to create a single structured dataset for testing. The first step is manual data filtering, which helps to avoid errors and exclude incorrect records. For comparative analysis, both the original dataset and the data that have been denoised and augmented are used, which helps to increase the generalizability of the developed localization system. The data for segmentation are shown in Fig. 1.

Fig. 1. Training data examples

VII. YOLO FOR IMAGE SEGMENTATION

The YOLO (You Only Look Once) algorithm from Ultralytics is a modern tool for real-time object segmentation in images. It uses a single convolutional neural network to simultaneously detect and segment objects, which provides high processing speed and accuracy. YOLO (You Only Look Once) works according to the following algorithm:

1. *Image processing:* The input image is passed through a single CNN, which extracts features using convolutional layers.
2. *Grid partitioning:* The image is divided into a grid of cells, where each cell is responsible for detecting objects within its boundaries.
3. *Prediction:* For each cell, the network simultaneously predicts multiple bounding boxes, their confidence scores, and classification probabilities for different object classes.
4. *Summary:* non-maximum suppression algorithm removes redundant frames, leaving only the most relevant predicted objects.

Thus, YOLO detects objects in a single pass through the network, which provides high speed and efficiency in real-time. YOLO training results are shown in Fig. 2. "X" axis shows train epochs, "Y" axis in '*_loss' charts shows how loss was reduced during training, and "Y" axis shows how good was model on validation data on different metrics.

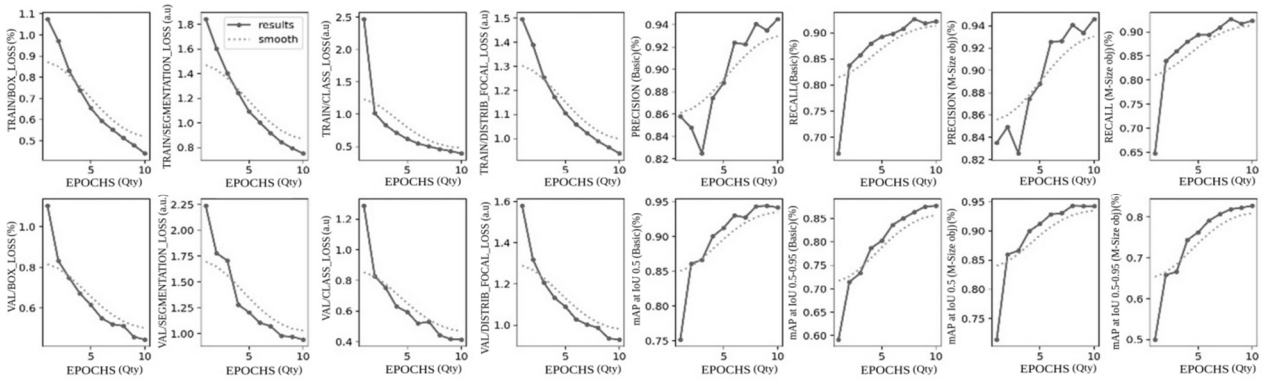


Fig. 2. Segmentation training results

VIII. PROJECTION 3D LIDAR POINTS ONTO 2D CAMERA IMAGE

The first step is to find the transformation matrix between the LiDAR coordinate system and the camera coordinate system. Since we have the LiDAR and cameras on the robot, and the robot is described using the URDF format, we can easily find the transition from one coordinate system to the other. The size of the matrix will be 4×4, consisting of a rotation matrix (3×3) and a translation vector (3×1)

$$T = \begin{bmatrix} R & T \\ 0_{1x3} & 1 \end{bmatrix}, \quad (1)$$

where R is rotation matrix, T is translation vector 0_{1x3} is vector [0, 0, 0].

To map LiDAR points to the camera coordinate system, you need to multiply the transformation matrix T by each LiDAR point. A LiDAR point has the form [X, Y, Z, 1] (4×1):

$$P_{LiDAR} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \quad (2)$$

The result of matrix multiplication will be a 4×1 matrix that represents the same LiDAR point only in the camera coordinate system. P_{camera} is every lidar point in camera coordinate system. P_{camera} is calculated as:

$$P_{camera} = T \cdot P_{LiDAR}. \quad (3)$$

All that remains is to project the 3D point onto the 2D camera image. To do this, we need to know the camera's intrinsic parameters, also called the intrinsic matrix:

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \quad (4)$$

where f_x and f_y are the horizontal and vertical focal lengths (usually in pixels); cx and cy are the coordinates of the principal point (image center), i. e. the point through which the optical axis of the camera passes.

Now we perform the initial transformation of the 3D point represented in the camera coordinate system P_{camera}, using the internal camera matrix K.

$$\begin{bmatrix} u' \\ v' \\ w \end{bmatrix} = K \cdot \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}, \quad (5)$$

where u and v are pixels on image.

This vector is not yet the final pixel coordinates, as it contains the scaling factor w'. To obtain the real pixel coordinates, it is necessary to normalize the resulting vector, i. e. divide the first two coordinates by the third. This gives the final coordinates (u,v) in the image.

$$u = u'/w' \quad v = v'/w'. \quad (6)$$

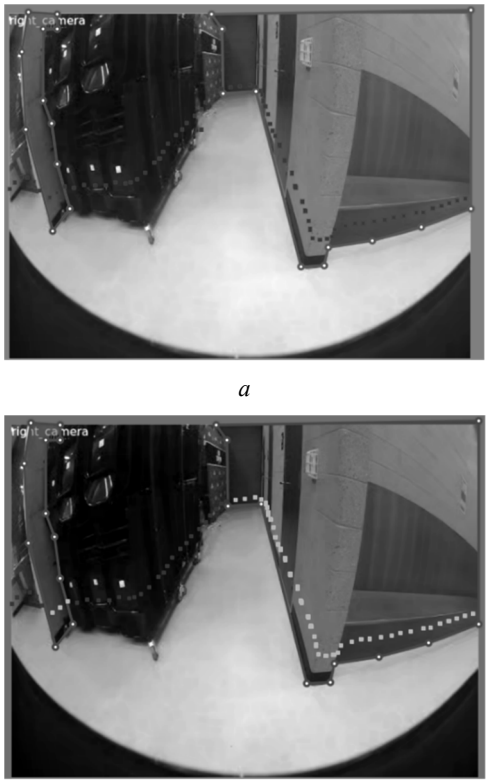
This gives the final coordinates (u,v) in the image. Now our LiDAR point is in the image. We pass the image from the camera to the segmentation model so that it marks the walls in the picture. In order to check whether a 2D LiDAR point belongs to a segmentation (polygon), there are three algorithms: Ray-Casting (Even-Odd Rule), Winding Number Algorithm. They all have the same linear complexity $O(n)$, since each of them checks all n edges of the polygon. I chose the Crossing Number Algorithm, because it is considered the fastest, since it does not use trigonometric functions, and is also simpler to implement. The disadvantage of this algorithm is that the algorithm cannot work with points that lie on a vertex or an edge. But this is not a problem for my algorithm, so this drawback can be ignored. After all the points from the LiDAR have been filtered, they can be passed to the Particle Filter algorithm.

IX. TESTING

Below is an example of using this algorithm in real conditions. The images on the right show segmented objects that are important for the robot's localization. The darker gray dots represent LiDAR points that are projected onto the camera image (Fig. 3, *a*). In the Fig. 3, *b* after using the Crossing Number algorithm, the points that affect localization have turned light gray (Fig. 3, *b*).

Results illustrates the benefit-cost trade-off introduced by our wall-segmentation enhancement. While the average per-iteration computation time increases modestly from 0.0255 s to 0.0294 s (≈ 15 % overhead) and CPU load remains essentially unchanged (≈ 50 % vs. 51 %), the localization error drops dramatically – from 5.08 cm down to 3.45 cm – an improvement of roughly 32 %. At the

same time, our approach converges in fewer iterations (42 vs. 55), indicating a 24 % reduction in convergence time, and exhibits lower run-to-run variability (σ = 0.0092 m vs. 0.0135 m), demonstrating both faster and more consistent performance. The memory footprint does increase (145 MB vs. 120 MB) due to the segmentation module, but remains well within the limits of typical robotic platforms. Together, these results confirm that a modest increase in processing and memory demands can yield substantial gains in accuracy, convergence speed, and robustness through dynamic-point rejection via wall segmentation.



a

b

Fig. 3. An example of a real environment without using(a) and using the Crossing Number algorithm (b)

Comparison of the results of particle filter (basic) and PF + segmentation (author approach)

Method	Avg. position calc. time, sec	CPU load, %	Avg. localization error, m	Convergence Speed, iterations	Std. Dev. of error, m	Memory Footprint, MB
Basic	0.02549	~50	0.05077	55	0.0135	~120
Our approach	0.02938	~51	0.03451	42	0.0092	~170

X. CONCLUSION

In this work, we proposed a hybrid localization approach for mobile robots operating in dynamic indoor environments, which leveraged wall segmentation in RGB images to filter out moving objects from LiDAR point clouds before feeding measurements into a Particle Filter. Experimental results demonstrated that by excluding dynamic points, our method substantially reduced the noise introduced by moving obstacles and achieved significantly higher pose estimation accuracy. Importantly, this enhancement could be integrated into existing Particle Filter-based systems without overhauling their core architecture – simply by inserting segmentation and projection modules upstream of the filter. The synergistic fusion of camera and LiDAR data enabled a more precise understanding of static structures: segmented wall regions guide the selection of reliable LiDAR returns, and the corresponding 3D geometry confirmed their spatial consistency. As a result, our framework converged more rapidly to the correct pose and exhibits greater resilience in cluttered or partially altered scenes. Despite these advantages, two main limitations remained. Firstly, the additional computational steps – image segmentation and point projection – might increase processing time, necessitating further optimization for deployment on resource constrained platforms. Secondly, the overall performance hinged on segmentation quality: misclassified wall areas could lead either to the loss of valuable static measurements or to the inadvertent inclusion of dynamic objects. Moreover, achieving precise 3D–2D alignment required careful calibration between camera and LiDAR, which could be a nontrivial technical challenge. Overall, our results confirmed that wall oriented filtering of LiDAR points enhanced both the reliability and accuracy of localization in dynamic environments. Future work will focus on accelerating the segmentation and projection pipeline, improving the robustness of the segmentation model, and streamlining calibration procedures to facilitate real time applications. The localization accuracy is improved by ~32 % compared to the baseline system.

References

[1] Wei, Z., Yang, X. (2021). Mobile Robot Localization Using Fuzzy Neural Network Based Extended Kalman Filter. *World Congress on Intelligent Control and Automation*, 416–421. DOI: https//doi.org/10.1109/WCICA.2011.5970654

[2] Boniardi F., Valada A., Rohit Mohan, Caselitz T., Burgard W. (2019). Robot Localization in Floor Plans Using a Room Layout Edge Extraction Network. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),* 5291–5297. DOI: https//doi.org/10.1109/IROS40897.2019.8967847

[3] Zang, Z., Zheng, H., Betz, J., & Mangharam, R. (2023, May). Local_inn: implicit map representation and localization with invertible neural networks. In *2023 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 11742–11748). IEEE. DOI: https//doi.org/10.1109/ICRA48891.2023.10161015

[4] Cabrera J., Román V., Gil A., Reinoso O., Payá L. (2024). An experimental evaluation of Siamese Neural Networks for robot localization using omnidirectional imaging in indoor environments. *Artif. Intell. Rev.,* 57, 198–215. DOI: https://doi.org/10.1007/s10462-024-10840-0

[5] Akai, N. (2023, May). SLAMER: Simultaneous Localization and Map-Assisted Environment Recognition. In *2023 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 6203–6209). IEEE. DOI: https://doi.org/10.1109/ICRA48891.2023.10160639

[6] Zimmerman N., Sodano M., Marks E., Behley J., Stachniss C. (2023). Constructing Metric-Semantic Maps using floor plan priors for long-term indoor localization. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),* 1366–1372. DOI: https://doi.org/10.1109/IROS55552.2023.10341595

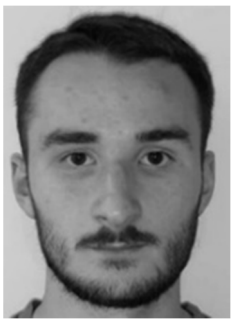
[7] Akai, N. (2023). Reliable Monte Carlo localization for mobile robots. *Journal of Field Robotics*, 40(3), 595–613. DOI: https://doi.org/10.1002/rob.22149

[8] Chen, Z., Li, K., Li, H., Fu, Z., Zhang, H., & Guo, Y. (2024). Metric localization for lunar rovers via cross-view image matching. *Visual Intelligence*, 2(1), 12. DOI: https://doi.org/10.1007/s44267-024-00045-y

[9] Mendez O., Hadfield S., Pugeault N., Bowden R. (2018). SeDARsemantic detection and ranging: Humans can localise without lidar, can robots*? IEEE International Conference on Robotics and Automation (ICRA),* 6053–6060, DOI: https://doi.org/10.1109/ICRA.2018.8461074

[10] Lin C., Li C., Furukawa Y., Wang W. (2018). Floorplan priors for joint camera pose and room layout estimation. *IEEE/CVF International Conference on Computer Vision (ICCV),* 5673–5682. DOI: https://doi.org/10.1109/ICCV.2019.00577

[11] Unicomb J., Ranasinghe R., Dantanarayana L., Dissanayake G. (2018). A monocular indoor localiser based on an extended Kalman filter and edge images from a convolutional neural network. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),* 1–9. DOI: https://doi.org/10.1109/IROS.2018.8594337



Maksym Shavarskyi was born on June 5, 2000. He received the B. S. and M. S. degrees in Computer Science at Lviv Polytechnic National University, Institute of Computer Science, where he is currently pursuing the Ph. D. degree. He is a Software Developer at SOMATIC and also serves as an Assistant at the Institute of Computer Science, Department of Artificial Intelligence Systems, at Lviv Polytechnic National University, where he conducts laboratory courses in robotics. His research interests include robotics and artificial intelligence.

Yurii Kryvenchuk was born on November 11, 1987. Yurii Pavlovych Kryvenchuk is a Ukrainian computer science researcher, PhD in Technical Sciences, and Associate Professor at the Departtment of Artificial Intelligence Systems at Lviv Polytechnic National University. He also serves as Deputy Director for Academic Affairs at the Institute of Computer Science and Information Technologies. In 2017, he defended his PhD thesis titled “Temperature measurement by frequency shift of combinational light scattering” in the specialty of thermal measurement instrumentation. His research interests include Industry 4.0 technologies, system modeling, data analysis, and non-contact methods of measuring physical quantities – especially temperature in nanostructured objects. Dr. Kryvenchuk has authored over 110 scientific publications, including 30 indexed in Scopus and Web of Science. He is also the author of two textbooks, eight patents, and 20 methodological developments.