



# LIQUID DETECTION AND INSTANCE SEGMENTATION BASED ON MASK R-CNN IN INDUSTRIAL ENVIRONMENT

Grzegorz Gawdzik <sup>1,2</sup> and Arkadiusz Orłowski <sup>2</sup>

<sup>1</sup>*Security and Defense Systems Division, Łukasiewicz Research Network –*

*Industrial Research Institute for Automation and Measurements PIAP, Warsaw, Poland*

<sup>2</sup>*Institute of Information Technology, Warsaw University of Life Sciences – SGGW, Warsaw, Poland*

**Abstract** The goal of the paper is to present an efficient approach to detect and instantiate liquid spilled in the industrial and industrial-like environments. Motivation behind it is to enable mobile robots to automatically detect and collect samples of spilled liquids. Due to the lack of useful training data of spilled substances, a new dataset with RGB images and masks was gathered. A new application of the Mask-RCNN-based algorithm is proposed which has the functionalities of detecting the spilled liquid and segmenting the image.

**Keywords:** AI, Mask-RCNN, liquid detection, dataset.

## 1. Introduction

Robotics is a domain in which machine vision is frequently used. It has multiple applications, including e.g. automotive domain for lines and signs detection, or manufacturing industry for pick & place, assembly and quality control operations. This paper is focused on a specific usage of machine vision in mobile robotics for liquid spill detection. Liquid leakages or liquid spills are something typical in industrial environment, especially in chemical production including oil, detergents, adhesives, cosmetic chemicals, and other specialty chemicals. It is common to see chemical substances on the floor, close to the open tanks covers or leaking tank flanges. Some of those chemicals are hazardous or even toxic for humans and shall be treated and removed in a specific way. To understand the type of a substance, it shall be sampled and analysed. This operation can be performed manually by human or with the use of a dedicated tool or machine. The goal for this paper is to propose a solution that would enable automatic sampling of spilled liquids. The solution is solely based on RGB data analysis and enables the detection of different types of substances that were spilled on the factory-like floor. The solution provides both bounding box of detected substances as well as masks that can be used for further processing and utilized for substance localization in 3D environment. The proposed algorithm is based on convolutional neural networks and is trained in a supervised manner. Dataset used for training is a custom one and was prepared exclusively for this purpose. Multiple databases were searched to gather required data, however due to niche application no available dataset was found.

## 2. Dataset

Dataset consisting spilled liquids in a factory-like environment was not available in any open access database. The searched terms were related to stains (incl. stains on the clothes), oil leakage, oil slick, spill of oil, liquid leak, fluid leak, or their variations. Among verified databases there were for instance: COCO dataset [9,10] (over 80 classes), Kaggle datasets [20] (over 280 000 datasets), OpenML database [16] (over 5300 datasets), Penn Machine Learning Benchmarks [15,17,18,19] (over 400 datasets). Therefore, a new dataset was required to partially fill the gap and to provide basic images related to spilled liquids. The dataset consists of objects of one class only, which is a liquid. Liquid class encloses two different types of liquids which are tea and coffee. In the future the dataset will be expanded by other types of liquids, especially chemical or chemical-like substances typical for chemical industry. Database was gathered using a camera with Full HD resolution and 30 fps in different lighting conditions, both natural and artificial. The above-mentioned substances were spilled on two different surface types, which are industrial monocolour epoxy floor (epoxy) and light terracotta tiles (terracotta). Gathering of data for a database was divided into five steps:

1. recording of short videos (approx. 10s) for each example of spilled liquid showing a spill from different angles;
2. reduction of the recorded frames by keeping only every tenth frame;
3. division of the video into separate images;
4. annotation of the images by drawing polygons and adding contextual information like type of substance and type of surface;
5. transfer of the frames with an annotation file into one dataset.

For dataset preparation 29 short videos were recorded. Dataset consists of images and a file with annotations. Examples of selected frames together with extracted from annotations masks can be seen in Fig. 1. The dataset in its current form (without extracted masks) is available in the repository [3].

Videos were annotated with the use of CVAT programme [12] both in local and server-based versions. Extracted video frames with annotations were saved in a COCO-dataset-like format including following information: bounding box of the object, polygon describing the edges of the object (segmentation), area of the polygon (in image pixels), category (always one – liquid) and attributes showing a type of substance and type of surface.

## 3. Solution

The proposed solution is based on transfer learning, with a use of Mask R-CNN algorithm as a foundation. The algorithm was selected due to its well known performance in detection and segmentation tasks [5]. The inputs for the algorithm are images and

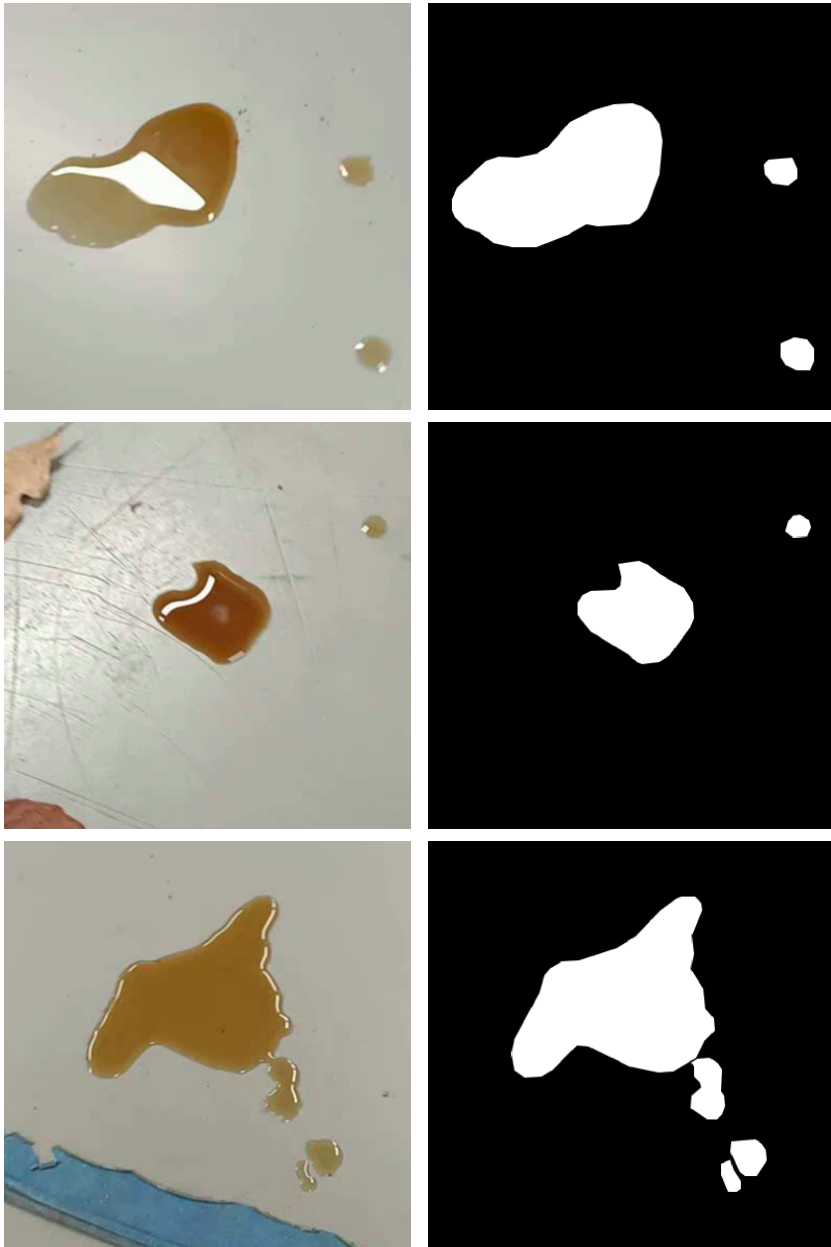


Fig. 1. Samples of the dataset. Images presenting RGB images and created masks.

masks. To comply with requirements the masks had to be extracted from the dataset (based on annotations) and saved as a set of images associated with original RGB ones. The created masks are presented in Fig. 1.

### 3.1. Environment and Tools

The algorithm was trained, tested and verified on two machines.

The first one is Lenovo ThinkPad T16p with 12th Gen Intel® Core™ i7-12700H×20, 32 GiB RAM, NVIDIA GeForce RTX 3050 Laptop GPU (4GiB VRAM) and installed Ubuntu 20.04.6 LTS.

The second one is Clevo x370 SNW-G with 13th Gen Intel® Core™ i9-13900HX×32, 64 GiB RAM, NVIDIA GeForce RTX 4090 Laptop GPU (12GiB VRAM) and installed Ubuntu 20.04.6 LTS.

To guarantee same software version on both machines the conda environment [11] was prepared and installed. As the main framework PyTorch [13] in version 1.13 with CUDA support was used. The PyTorch-based algorithm was trained and tested both with a use of CUDA and CPU. CUDA was used with NVIDIA GeForce RTX 4090 Laptop GPU and CPU with Intel® Core™ i7-12700, due to limitations of NVIDIA GeForce RTX 3050 Laptop GPU.

The usage of two calculation means emphasised a dramatic difference in training speed on both machines, even though the values of parameters and number of epochs were the same. Training with GPU for 10 epochs lasted around 4 minutes, while training on CPU lasted around 5 hours. Due to this difference, further calculations were performed on the GPU and therefore all outcomes presented in the paper are based on the outputs from the algorithm run on the GPU.

### 3.2. Algorithm

Mask R-CNN-based algorithms are widely used in multiple domains especially were detectable object presence and its exact shape is required. The algorithm has application in the fields of geospatial surveying [2, 22], object detection in technical processes [21], atmospheric analysis [23], fruits detection [6], biomedical studies [7] and many others.

The algorithm was created with the use of PyTorch framework. The implementation is based on a finetuned Mask R-CNN model, pre-trained on COCO train2017 dataset [1, 10] with default weights. Used Mask R-CNN model is founded on top of ResNet-50-FPN [5] architecture. Model's head for bounding box prediction and image segmentation was changed and trained on the dataset to output both bounding box and segmentation mask of a detected object. The algorithm classifies two classes, which are *liquid spill* – class 1, and *background* – class 0. In the algorithm a few hyperparameters was used including number of batches, learning rate, number of epochs and split ratio. As the optimiser a stochastic gradient descent with momentum was selected.

### 3.3. Training and Testing Datasets

To create a training and testing sets, the dataset had to be split, however not to bias the results it was decided that there will be no permanent division of the sets. On the contrary, at each run of the algorithm, the dataset was randomly divided into two sets of 70% to 30%, respectively. To reduce randomness of results each run (with unchanged parameters' values) was repeated 10 times. The approach was selected to keep the results objective and not to skew them at the dataset division level. The inputs are twofold as the algorithm accepts two tuples, which are: 1) RGB images, 2) annotations, including bounding boxes, labels, masks and image ID to reference with RGB images.

### 3.4. Hyperparameters

Based on the initial algorithm performance, it was observed that the hyperparameters with the biggest impact were: 1) batch size and 2) learning rate. The number of epochs did not change the results significantly and after looping for 6-7 times the loss value was stabilised. According to initial results, 1000 additional runs of the algorithm was performed, with hyperparameters that were set as follows: batch size from 1 to 10, learning rate from 0.001 to 0.01 and number of epochs to 10. The others were set as follows: momentum factor to 0.9 and weight decay to 0.0005.

During training the images were shuffled in each run to generalise the training set and limit impact of the consecutive images from one video. While testing the images were not shuffled.

### 3.5. Predictions

Predictions are composed of a set of values. Each prediction consists of information about detected objects, including:

1. coordinates of bounding boxes in reference to analysed RGB image,
2. class number (in this case always liquid, as another class is a background),
3. prediction score,
4. coordinates of vertices creating a polygon of the segmentation mask.

Based on the predictions, a mask is generated to visualise a successful detection. Examples of the predicted masks are presented in Fig. 2.

## 4. Evaluation

The algorithm was run multiple times with consecutively changed parameters. During each run the algorithm calculated the loss value after each epoch and adjusted the weights. The sample of calculated train loss curves can be seen in Fig. 3. The differences are based on the different hyperparameters chosen at the beginning of each run.



Fig. 2. Examples of visualised predictions that led to liquid detection and masks creation.

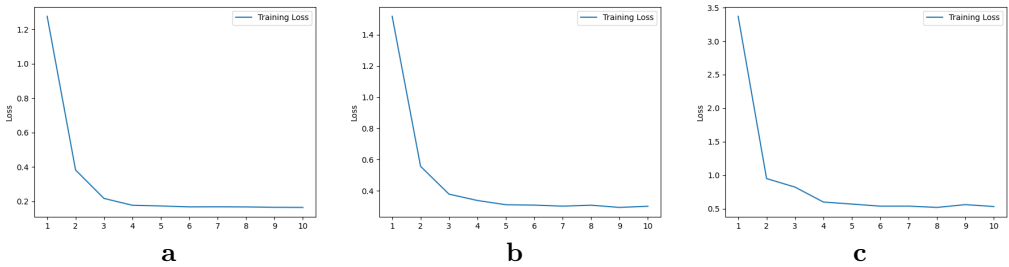


Fig. 3. The training loss curves for runs with different parameters: (a) Batch Size = 2, RL = 0.001, loss value: 0.165; (b) Batch Size = 5, RL = 0.001, loss value: 0.301; (c) Batch Size = 10, RL = 0.005, loss value: 0.532.

To measure the prediction algorithm results, an evaluation method based on MS COCO [8] and available in PyTorch Computer Vision library [14] was used. The method provides 10 classification quality measures of mean average precision and recall for both bounding box and segmentation masks detections.

To decide whether the prediction should be considered a True Positive case usually a threshold is set for the Intersection over Union (IoU) of the predicted and ground truth regions. In our study we have used a set of thresholds, starting from 0.5 up to 0.95. So, a prediction with the value of IoU higher than the selected threshold is considered as a True Positive one. Based on the True Positive, False Positive and False Negative

results, the average precision (AP) and average recall (AR) with different thresholds is calculated.

For evaluation purposes, ten IoU thresholds between 0.5 and 0.95 with step of 0.05 were used to calculate the mean values for average precision (mAP) and average recall (mAR):

$$\text{mAP}_{\text{COCO}} = \frac{\text{AP}_{0.50} + \text{AP}_{0.55} + \dots + \text{AP}_{0.95}}{10} \quad (1)$$

The evaluation method takes under consideration the size of the objects and divide them into 3 groups: small with size lower than  $32 \times 32$  pixels; medium with size between  $32 \times 32$  pixels and  $96 \times 96$  pixels; large with size higher than  $96 \times 96$  pixels.

The measures mAP and mAR are presented below and were calculated both for bounding boxes and masks.

### Mean average precision

- mean average precision (mAP) over 10 consecutive IoU thresholds, from 0.5 to 0.95, step 0.05 (0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95) for all scales;
- mAP at IoU = .50 for all scales;
- mAP at IoU = .75 for all scales;
- mAP over 10 consecutive IoU thresholds for small scale;
- mAP over 10 consecutive IoU thresholds for medium scale;
- mAP over 10 consecutive IoU thresholds for large scale.

### Mean average recall

- mean average recall (mAR) over 10 consecutive IoU thresholds, from 0.5 to 0.95, step 0.05 (0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95) for all scales;
- mAR over 10 consecutive IoU thresholds for small scale;
- mAR over 10 consecutive IoU thresholds for medium scale;
- mAR over 10 consecutive IoU thresholds for large scale.

As it was mentioned before the algorithm was run 100 times with different sets of parameters. Each run was repeated 10 times, which ends up with 1000 runs. To calculate and select optimal parameters for the algorithm, the results of 5 measures of classification quality were taken under consideration, which are: Bounding Box Average Precision, Bounding Box Average Recall, Segmentation Average Precision, Segmentation Average Recall and training loss. First four measures were calculated over 10 IoU consecutive thresholds and provides general precision and recall of the model. Based on the results there is no set of parameters that provides outstanding results in comparison to other sets, however there is a clear trend. Tables 1 and 2 present sets of five best and five worst results, respectively, based on loss value.

Tab. 1. Five best results based on the loss value. No.: batch no.; BS: batch size; LR: learning rate; bb mAP: bounding box mean average precision, bb mAR: bounding box mean average recall, s mAP: segmentation mean average precision, s mAR: segmentation mean average recall; TL: training loss.

No.	BS	LR	bb mAP	bb mAR	s mAP	s mAR	TL
4	1	0.004	0.839	0.878	0.873	0.901	0.117
5	1	0.005	0.850	0.886	0.888	0.914	0.117
6	1	0.006	0.840	0.871	0.886	0.906	0.120
7	1	0.007	0.844	0.875	0.887	0.908	0.117
8	1	0.008	0.859	0.886	0.890	0.910	0.119

Tab. 2. Five worst results based on the loss value. For denotations see Tab. 1.

No.	BS	LR	bb mAP	bb mAR	s mAP	s mAR	TL
71	8	0.001	0.413	0.622	0.571	0.813	0.500
81	9	0.001	0.433	0.646	0.588	0.827	0.478
82	9	0.002	0.436	0.630	0.596	0.834	0.463
91	10	0.001	0.432	0.611	0.580	0.790	0.489
92	10	0.002	0.443	0.641	0.586	0.817	0.465

It seems that batch size has a highest impact on the results achieved during algorithm training and testing and the lower the value is, the better results are achieved.

Learning rate has lower impact but it seems that the best results can be achieved with learning rate different from minimal or maximal values. The full table with all runs can be found in the repository [4].

The results present averaged outputs of the model from 10 runs of testing dataset with selected set of parameters. The detailed measures showing results for different IoU and area values was presented in Table 3 and 4. The tables show the outputs from a sample run with following parameters: batch size = 1, learning rate = 0.005, epochs number = 10, dataset split = 70%.

## 5. Conclusion

The achieved results seem to be promising. They present high values of the classification quality measures. Moreover, the results show a clear relation between batch size, learning rate and achieved outcomes. Used dataset is recorded with a high resolution camera which is expected to be used in real case scenario. The liquids in that case will be located in the close range to the camera, therefore the evaluation shall be focused on



Tab. 3. Mean Average Precision: mAP, and Mean Average Recall: mAR, for bounding boxes.

Measure	IoU	Area	Value
mAP	0.50:0.95	all	0.855
mAP	0.50:0.95	small	0.790
mAP	0.50:0.95	medium	0.756
mAP	0.50:0.95	large	0.947
mAR	0.50:0.95	all	0.877
mAR	0.50:0.95	small	0.820
mAR	0.50:0.95	medium	0.782
mAR	0.50:0.95	large	0.964

Tab. 4. Mean Average Precision: mAP, and Mean Average Recall: mAR, for segmentation masks.

Measure	IoU	Area	Value
mAP	0.50:0.95	all	0.899
mAP	0.50:0.95	small	0.819
mAP	0.50:0.95	medium	0.847
mAP	0.50:0.95	large	0.978
mAR	0.50:0.95	all	0.908
mAR	0.50:0.95	small	0.833
mAR	0.50:0.95	medium	0.853
mAR	0.50:0.95	large	0.982

metrics considering detections over large scale objects (96×96 pixels and more). The mean average recall and precision values for segmentation received in this study were very high and close to 90%. The mean average precision and recall values for bounding boxes were a bit lower and reached 86% and 88%, respectively.

In the next stages of the research it is planned to use the current results as a benchmark and to test other algorithms with modified architecture to identify those with best performance, and to increase the dataset with images comprising new liquids and surfaces on which the substances are spilled.

## References

- [1] awsaf49. COCO 2017 Dataset, 2017. <https://www.kaggle.com/datasets/awsaf49/coco-2017-dataset>, [Accessed: May 2023].
- [2] A.-A. Dalal, Y. Shao, A. Alalimi, and A. Abdu. Mask R-CNN for geospatial object detection. *International Journal of Information Technology and Computer Science (IJITCS)*, 12(5):63–72, 2020. doi:10.5815/ijitcs.2020.05.05.
- [3] G. Gawdzik. Liquid dataset. PIAP Cloud Resources, 2023. <https://cloud.piap.pl/index.php/s/ApiXNzt4ZUUSRks>, [Accessed: 10 Dec 2023].

- [4] G. Gawdzik. Table of results for all runs of the liquid detection. PIAP Cloud Resources, 2023. <https://cloud.piap.pl/index.php/s/cE3mJ8CrCYWUkJ9>, [Accessed: 10 Dec 2023].
- [5] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In: *Proc. 2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2980–2988. Venice, Italy, 22–29 Oct 2017. doi:10.1109/ICCV.2017.322.
- [6] W. Jia, Y. Tian, R. Luo, Z. Zhang, J. Lian, et al. Detection and segmentation of overlapped fruits based on optimized Mask R-CNN application in apple harvesting robot. *Computers and Electronics in Agriculture*, 172:105380, 2020. doi:10.1016/j.compag.2020.105380.
- [7] H. Jung, B. Lodhi, and J. Kang. An automatic nuclei segmentation method based on deep convolutional neural networks for histopathology images. *BMC Biomedical Engineering*, 1:24, 2019. doi:10.1186/s42490-019-0026-8.
- [8] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, et al. Detecion Evaluation. In: *COCO. Common Objects in Context* [10]. [Accessed: Dec 2023]. <https://cocodataset.org/#detection-eval>.
- [9] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, et al. Microsoft COCO: Common Objects in Context. *arXiv*, 2015. ArXiv:1405.0312. doi:10.48550/arXiv.1405.0312.
- [10] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, et al., eds. *COCO. Common Objects in Context*, 2020. [Accessed May 2023], <https://cocodataset.org>.
- [11] Anaconda, Inc. Conda Documentation, 2023. <https://docs.conda.io>, [Accessed: 10 Dec 2023].
- [12] CVAT.ai Corporation. CVAT Open Data Annotation Platform, 2023. <https://www.cvat.ai>, [Accessed: 10 Dec 2023].
- [13] PyTorch Foundation, a project of The Linux Foundation. PyTorch Get Started, 2023. <https://pytorch.org/>, [Accessed: 10 Dec 2023].
- [14] TorchVision maintainers and contributors. TorchVision: PyTorch’s Computer Vision library. GitHub repository, 2016. <https://github.com/pytorch/vision>, [Accessed: Jun 2023].
- [15] R. S. Olson, W. La Cava, P. Orzechowski, R. J. Urbanowicz, and J. H. Moore. PMLB: a large benchmark suite for machine learning evaluation and comparison. *BioData Mining*, 10(36):1–13, 2017. doi:10.1186/s13040-017-0154-4.
- [16] J. van Rijn, J. Vanschoren, B. Bischl, M. Feurer, G. Casalicchio, et al. OpenML Datasets. <https://www.openml.org/search?type=data>, [Accessed: Dec 2023].
- [17] J. D. Romano, T. T. Le, W. La Cava, J. T. Gregg, D. J. Goldberg, et al. Penn Machine Learning Benchmarks. <https://epistasislab.github.io/pmlb/>.
- [18] J. D. Romano, T. T. Le, W. La Cava, J. T. Gregg, D. J. Goldberg, et al. PMLB v1.0: an open-source dataset collection for benchmarking machine learning methods. *Bioinformatics*, 38(3):878–880, 2021. doi:10.1093/bioinformatics/btab727.
- [19] J. D. Romano, T. T. Le, W. La Cava, J. T. Gregg, D. J. Goldberg, et al. PMLB v1.0: an open source dataset collection for benchmarking machine learning methods. *arXiv*, 2021. ArXiv:2012.00058v2. doi:10.48550/arXiv.2012.00058.
- [20] D. Sculley, J. Moser, W. Cukierski, J. Rose, M. O’Connell, et al. Kaggle Datasets, 2023. <https://www.kaggle.com/datasets>, [Accessed: Jan 2023].
- [21] S. Sibirtsev, S. Zhai, M. Neufang, J. Seiler, and A. Jupke. Mask R-CNN based droplet detection in liquid–liquid systems, Part 2: Methodology for determining training and image processing parameter values improving droplet detection accuracy. *Chemical Engineering Journal*, 473:144826, 2023. doi:10.1016/j.cej.2023.144826.

- [22] H. Su, S. Wei, M. Yan, C. Wang, J. Shi, et al. Object detection and instance segmentation in remote sensing imagery based on precise Mask R-CNN. In: *Proc. 2019 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pp. 1454–1457. Yokohama, Japan, 28 Jul – 2 Aug 2019. doi:10.1109/IGARSS.2019.8898573.
- [23] P. Su, J. Joutsensaari, L. Dada, M. A. Zaidan, T. Nieminen, et al. New particle formation event detection with Mask R-CNN. *Atmospheric Chemistry and Physics*, 22(2):1293–1309, 2022. doi:10.5194/acp-22-1293-2022.

**Grzegorz Gawdzik**, M.Sc., Eng., graduated from Warsaw University of Life Sciences – SGGW, Faculty of Forestry, with specialization in GIS and 3D scanning. Currently works as a researcher in the Security and Defence Systems Division of Łukasiewicz – PIAP and a Ph.D. student at Warsaw University of Life Sciences – SGGW. His major research interests are: 3D scanning and environment reconstruction; perception for situational awareness; deep learning; pattern recognition; image detection and segmentation; machine learning; reinforcement learning; robotic manipulation.

