SERBIAN JOURNAL OF ELECTRICAL ENGINEERING
Vol. 19, No. 3, October 2022, 291-302
UDC: 004.85:513.767 **DOI: https://doi.org/10.2298/SJEE2203291J**

Data Augmentation Methods for Semantic Segmentation-based Mobile Robot Perception System

Aleksandar Jokić[1], Lazar Đokić[1], Milica Petrović[1], Zoran Miljković[1]

Abstract: Data augmentation has become a standard technique for increasing deep learning models' accuracy and robustness. Different pixel intensity modifications, image transformations, and noise additions represent the most utilized data augmentation methods. In this paper, a comprehensive evaluation of data augmentation techniques for mobile robot perception system is performed. The perception system based on a deep learning model for semantic segmentation is augmented by 17 techniques to obtain better generalization characteristics during the training process. The deep learning model is trained and tested on a custom dataset and utilized in real-time scenarios. The experimental results show the increment of 6.2 in mIoU (mean Intersection over Union) for the best combination of data augmentation strategies.

Keywords: Mobile robot perception system, Deep learning, Data augmentation, Semantic segmentation.

1 Introduction

Advanced mobile robot systems utilize sensors (e.g., cameras or lidars) that provide a large amount of data about the environment. However, the way the data is transformed into useful information has been an active and challenging area of research. Convolutional Neural Networks (CNN) have tremendous potential to enable mobile robots to comprehend and interact with their environment more intelligently [1]. However, two prerequisites for using CNNs are a large amount of labeled data and substantial computational power. Fortunately, several novel single-board computers or hardware accelerators have enough computing power to deploy efficient CNNs and utilize them in real-time applications. The hardware utilized for testing the developed methods is presented in [2].

[1]University of Belgrade, Faculty of Mechanical Engineering, Department of Production Engineering, Laboratory for industrial robotics and artificial intelligence (ROBOTICS&AI), Kraljice Marije 16, 11120 Belgrade 35, The Republic of Serbia.
E-mails: ajokic@mas.bg.ac.rs; ldjokic@mas.bg.ac.rs; mmpetrovic@mas.bg.ac.rs; zmiljkovic@mas.bg.ac.rs

To train deep learning models, a large amount of data is required. Raw data generated by cameras can be acquired relatively fast; however, to label that data for semantic segmentation is challenging and time-consuming. Even the largest and most popular datasets for semantic segmentation have only several thousand densely labeled images [3]. One of the primary ways to generate more data is to utilize data augmentation techniques. Utilizing these techniques makes the dataset larger and covers a broader distribution of possible real-world scenarios. Therefore, in this paper, the authors propose to analyze the effect of 17 different data augmentation techniques on the accuracy of the deep learning model implemented for mobile robot perception system purposes.

The rest of the paper is structured as follows. Section 2 is devoted to the *state-of-the-art* survey. Section 3 defines the problem and analyzes data augmentation methods. Section 4 describes the utilized CNN model and training process. The experimental results and perception system evaluation with data augmentation techniques are presented in Section 5, while concluding remarks are presented in Section 6.

2 The State-of-the-Art

One of the most comprehensive *state-of-the-art* surveys regarding the data augmentation for deep learning was presented in [4]. The authors divided data augmentation techniques into two groups (i) image manipulation techniques and (ii) deep learning-based data augmentation methods. Image manipulation techniques include three types of manipulations i.e., geometric image transformation, image manipulation, and color transformation. The geometric transformation includes horizontal flips, random crops, image translation, and image rotation. Image manipulation techniques are random erasing, kernel filtering, noise injection, and image mixing. Lastly, the color transformation includes color space transformation and color jittering.

In the paper [5], the authors proposed to augment each object in the image individually. By using labeled data, each object in the images was firstly extracted and then augmented. Secondly, the augmented object was copied back to the original image, and the void parts (created due to the difference between original and augmented objects) were filled by using another CNN trained to fill missing pieces of the image. The authors argued that this process is more realistic for the image augmentation process since the individual object can usually be found in different places in the image. The proposed augmentation method improved the accuracy of all considered models compared to the original augmentation strategy.

The authors of [6] presented a GAN-based (Generative Adversarial Network) methodology for semantic segmentation dataset augmentation. The central idea behind this paper was that the improvement of segmentation should

be oriented toward the better classification of low-represented classes. The authors calculate the frequency of all classes in the dataset and augment it to increase the overall class balance. GANs were used to generate whole new semantic maps and image pairs or to add the low represented classes to the semantic labels where they did not previously exist. The highest improvement in IoU (Intersection over Union) was achieved when around 50% of data was synthetically generated. The proposed methodology yielded an improvement of around 2.1% mIoU on the Cityscape dataset, which was higher than the previous *state-of-the-art* style transfer technique.

In the paper [7], the authors proposed a smart data augmentation sampling and search strategy. They utilized the Bayesian optimization framework to determine five optimal parameters regarding magnitude, type, and probability of augmentation performed at each training epoch. The general proposal was to utilize non-augmented images at the beginning of the training process and increase the probability of using augmented images with the number of epochs. Hyperparameter importance testing was performed with fANOVA to find and tune the most influential hyperparameter. The proposed method for smart augmentation outperformed the *state-of-the-art* method, while the smart sampling method provided an efficient sampling algorithm with competitive performance compared to other analyzed methods.

The authors of the paper [8] proposed a simple yet effective automatic data augmentation strategy entitled as TrivialAugmentation (TA). The proposed method applied only one augmentation type to each image, unlike other approaches. Their methodology included 14 standard augmentation types utilized for the image classification task. Another essential advantage of this approach was a significantly smaller search space. The extensive experimental results showed that TA method outperformed other state-of-the-art strategies compared on different datasets and deep learning models.

In this paper, different from other approaches, the authors consider the effect of data augmentation for semantic segmentation training on a small-scale dataset. Four groups of data augmentation methods are individually tested, and their performance is benchmarked.

3 Data Augmentation Methods

In this paper, 17 data augmentation methods are utilized. The image obtained with the mobile robot perception system is represented with $\boldsymbol{X} \in \mathbb{R}^{H \times W \times C}$, where H is image height, W is image width, and C is a number of channels (in this paper $C = 3$). The deep learning model is utilized to perform semantic segmentation and produce the semantic map $\boldsymbol{Y} \in \mathbb{R}^{H_1 \times W_1 \times N}$, where H_1 and W_1 are the height and width of the semantic map, and N is a number of possible classes. The basic proposition

of this paper is that it is possible to improve the accuracy of semantic segmentation by adding the image transformation τ to the input image or more formally (1):

$$\sum_{i}^{M} \ell\left(f\left(\tau\left(\boldsymbol{X}_i\right),\boldsymbol{\theta}\right),\boldsymbol{Y}_i\right) < \sum_{i}^{M} \ell\left(f\left(\boldsymbol{X}_i,\boldsymbol{\theta}\right),\boldsymbol{Y}_i\right),\ i=1,...,M\ , \tag{1}$$

where f represents the deep learning model that performs semantic segmentation for input image $\boldsymbol{X}_i$ and model parameters $\boldsymbol{\theta}$, $\boldsymbol{Y}_i$ is a semantic map, ℓ is the loss function, and i represents dataset input-output pair. The different methods of data augmentation are shown in Fig. 1. Images b), c), d), e), f) represent geometrical transformations; g), h), i) are noise addition methods; j), k), l), m), o) represent pixel intensity modification methods; and p), q), r) are other methods.

The standard and commonly used data augmentation methods such as random crops, horizontal flips, fancy PCA, rotation, and translation will not be discussed in detail, and the reader is referred to [10] and [11]. Now, we will discuss in detail a few non-standard data augmentation methods. The random erase method includes replacing a patch in the image with random noise. Some modern industrial cameras in bright environments can produce wave-like noise; to ensure robustness, we add a wave-like noise to the image. Wave noise is added according to (2):

$$\begin{aligned} &\boldsymbol{X}_w\left(x,y\right) = \boldsymbol{X}\left(x,y\right) + k\sin\left(\boldsymbol{q}\left(x\right)\cdot r_1\right), \\ &x=1,...,H, y=1,...,W, \boldsymbol{q}=\left[0,...,6\,\mathrm{p}\right], \end{aligned} \tag{2}$$

where r_1 is the random number drawn from a uniform distribution in range (0, 0.5) sampled for each image, k is sinus amplitude, and $\boldsymbol{q}$ is a linearly spaced vector between 0 and 6π with the H elements.

Edge enhancement is performed by first converting the image to grayscale and extracting the edge mask with the standard Sobel filter image. Then the initial image is converted to HSV color space, and a mask of edge pixels is used to scale saturation and value by a factor of 5.

The random HSV method is performed by scaling each color channel in HSV images with a factor drawn from a uniform distribution in range (0, 255). The output represents a diverse, colorful image that can be looked at as a type of domain randomization technique proposed in [12]. The parameters for each data augmentation type are given in **Table 1**. The groups are set to produce the same number of additional augmented images.

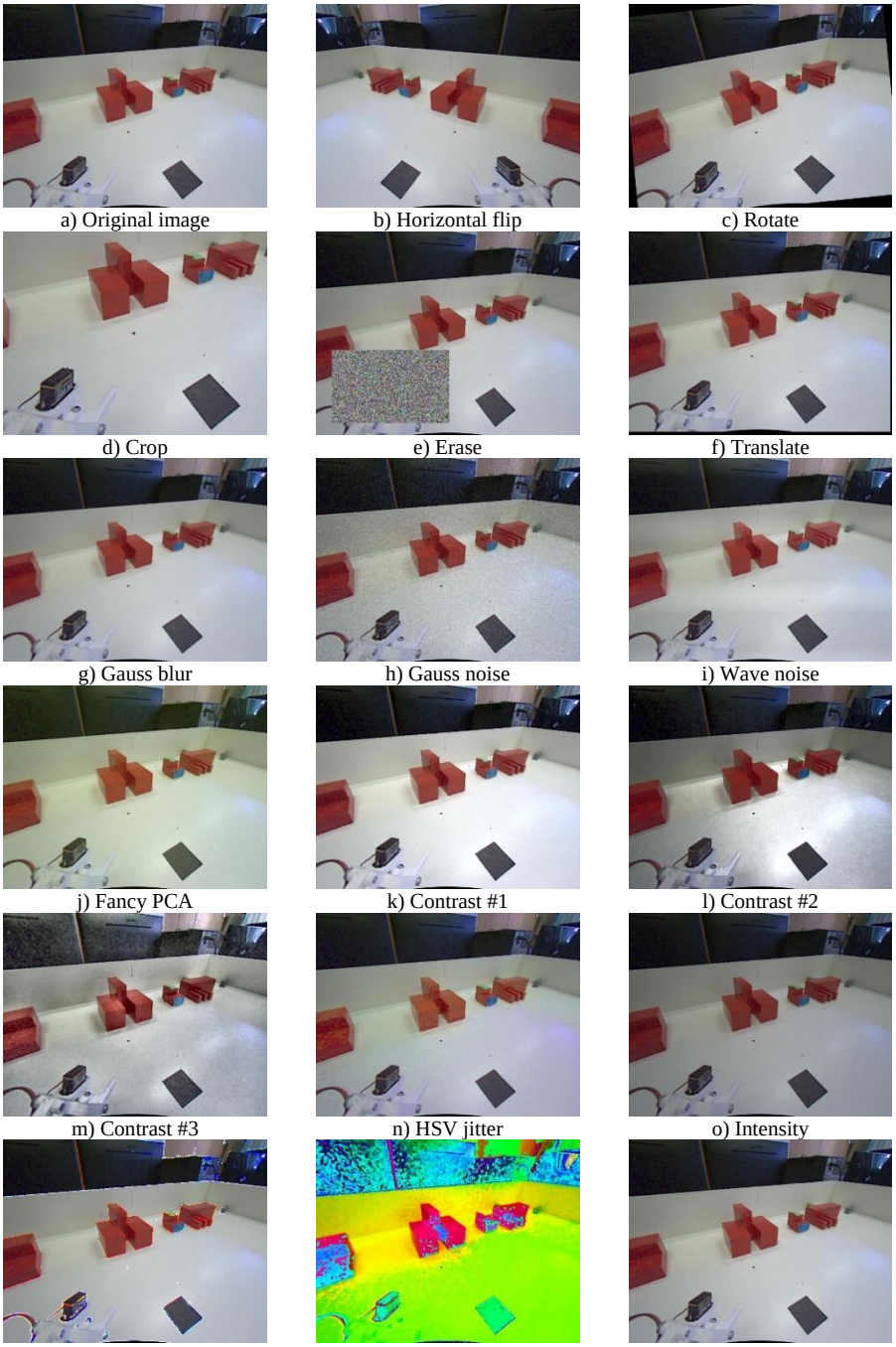

Fig. 1 – *Examples of data augmentation methods.*

Table 1
Parameters for different augmentation types. U(x, y) represents a random number drawn from a uniform distribution with a range from x to y.

Group	Augmentation type	Dataset percentage	Method-specific parameters
Geometric transformation methods	Horizontal flip	0.7	/
	Rotate	0.5	$rot\_angle = U(-10,10)$
	Crop	0.8	$window\_size = 0.7 \cdot [W, H]$
	Erase	0.5	$err\_window = [W \cdot U(0,0.5), H \cdot U(0,0.5)]$
	Translate	1	$[t_x, t_y] = [U(-20,20), U(-40,40)]$
Noise addition methods	Gauss blur	0.6	$\mathbb{N}(0,1)$
	Gauss noise	0.9	$\mathbb{N}(0,0.005)$
	Wave noise	2.0	$k = 15$
Pixel intensity manipulation methods	Fancy PCA	0.9	$color\_factor = \mathbb{N}(0,0.3)$
	Contrast #1	0.3	$max\_luminosity = 100$
	Contrast #2	0.3	$max\_luminosity = 100$
	Contrast #3	0.3	$max\_luminosity = 100$
	HSV jitter	0.9	$HSV = [U(0,0.1), U(0,0.1), U(0,0.1)]$
	Intensity	1.0	$intensity\_factor = U(-0.7,1.3)$
Other methods	Edge enhancement	1.3	$scale = 5$
	Random HSV	1.2	$HSV\_factors = [U(0,1), U(0,1), U(0,1)]$
	Patch shuffle	1	$p = 0.5$

4 Deep learning model – description and training procedure

The details regarding utilized CNN architecture are presented in Fig. 2. The CNN model is based on ResNet18 architecture [13] with separated convolutional layers inspired by ones utilized in [14] and [15]. Different blocks of layers are presented with rectangles of different colors, while the parameters for those layers are presented within the rectangle. ***W*** represents the weight matrix dimensions, S is the stride value, and P represents the padding value. Blue blocks represent the combination of the following layers: Convolution-ReLU-Convolution-BatchNormalization. The green rectangle is the MaxPooling layer, while the brown block represents Convolution and BatchNormalization. The addition layer with the ReLU activation is presented with orange. Input images have 800×600×3 resolution, while the output semantic mask has the dimension of 100×75×5. The prediction for each pixel is calculated by utilizing the Softmax activation function (3), and the loss function is Cross-entropy (4),

$$s_j = \frac{e^{Y_j}}{\sum_{j=1}^{N} e^{Y_j}}, \tag{3}$$

$$\ell(s,c) = -\sum_{j=1}^{N} c_j \log(s_j), \tag{4}$$

where ***Y*** represents the output vector of the neural network, *j* is the current element of the output vector, *N* is the total number of classes (and also the number of elements in the output vector for each element), s_j is the output of the Softmax function for each element, ***c*** represents $H_1 \times W_1 \times N$ one-hot vector for the correct class of the current input vector, and ℓ represents the loss function value.

Dataset is generated in the laboratory model of the manufacturing environment. Four machine tools models are labeled (classes M #1 – M #4) and a fifth background class (class B). Therefore, the images are segmented into up to five classes representing entities of interest in a manufacturing environment. Before any data augmentation is done, the dataset is split into two subsets; 80% of images are utilized for training and 20% for validation. The training is carried out using PyTorch v1.6.0 with Stochastic gradient descent and a momentum of 0.9. The initial learning rate is set to η=0.01 with the schedule defined in (5):

$$\eta^{new} = \eta^{old}\left(1 - \frac{current\_epoch}{max\_epoch}\right)^{0.9}. \tag{5}$$

The variable *current_epoch* is enumerated from 0 to (*max_epoch* – 1). Also, the maximum number of epochs is 100, while the mini-batch size is 4. Lastly, the additional regularization technique is utilized with a weight decay of 0.0001. Training is performed on Nvidia RTX 1660 GPU with 6GB of RAM. The deep learning model is later deployed to a single-board computer Nvidia Jetson nano with NVIDIA 128-core Maxwell GPU.

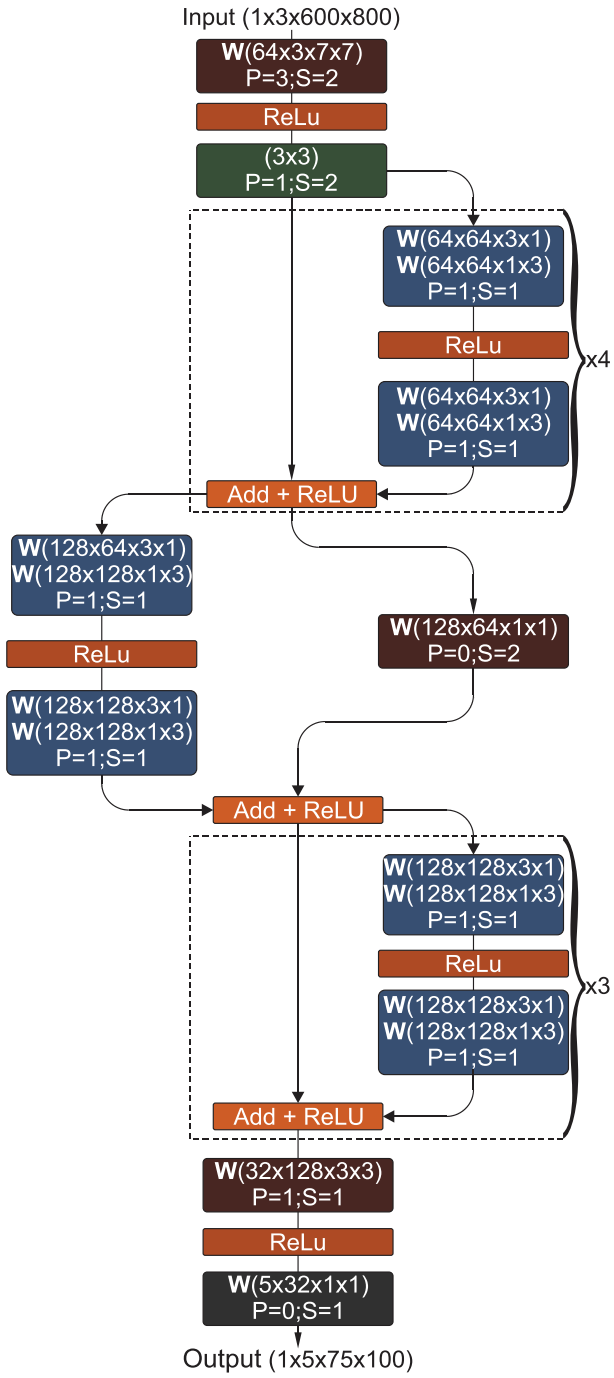

Fig. 2 – *ResNet18 architecture with separated convolutional layers.*

5 Experimental Results

The experimental plan is as follows. Data augmentation methods are grouped according to the type of performed transformation. Four groups include (i) geometric transformation methods, (ii) noise addition methods, (iii) pixel intensity manipulation methods, and (iv) other methods. A total number of 16 experiments are performed, where each group can be either included in the experiment or not, depending on the value of numbers in vector **b**. The vector **b** has 4 elements that correspond to each group; if the first value in vector b is 1, then the first group is included in the experiment, and so on (see **Table 2**). The group to which data augmentation technique belongs to can be seen in **Table 1**. The proposed system is experimentally tested for four machine tool models (M #1 – M #4) and background class (B). The experimental plan and results are shown in **Table 2**.

Table 2
Experimental results are generated by using images from the validation set. In bold are shown the best results per IoU metric.

No.	**b**	IoU measure					
		B	M #1	M #2	M #3	M #4	mIoU
1	[1 1 1 1]	**99.5**	90.8	**94.4**	**89.0**	92.1	**93.2**
2	[1 1 1 0]	**99.5**	88.4	93.6	82.6	**93.4**	91.5
3	[1 1 0 1]	99.4	83.5	90.5	85.2	92.1	91.4
4	[1 1 0 0]	99.4	83.5	90.5	85.2	92.1	90.2
5	[1 0 1 1]	99.3	89.2	93.0	84.9	90.3	91.4
6	[1 0 1 0]	99.4	**91.0**	93.8	84.1	91.6	92.0
7	[1 0 0 1]	99.2	84.7	89.6	84.2	87.8	89.1
8	[1 0 0 0]	99.3	86.2	86.7	85.4	87.2	89.0
9	[0 1 1 1]	99.4	83.9	89.2	84.3	82.0	87.8
10	[0 1 1 0]	99.4	80.4	90.9	82.9	88.1	88.4
11	[0 1 0 1]	99.4	80.9	89.6	88.0	82.2	88.0
12	[0 1 0 0]	**99.5**	88.1	91.5	88.7	86.7	90.9
13	[0 0 1 1]	99.4	77.9	86.7	86.2	79.8	86.0
14	[0 0 1 0]	99.3	82.8	91.2	82.2	85.8	88.3
15	[0 0 0 1]	99.1	61.5	73.8	68.4	85.2	77.6
16	[0 0 0 0]	99.1	81.5	88.5	77.5	88.6	87.0

The experimental results show that the considered data augmentation methods significantly improve the accuracy of the proposed mobile robot perception system based on deep learning. Individually, the most significant improvement is achieved by utilizing the second data augmentation group of methods, where mIoU increases by 3.9. It is also shown that when utilizing three out of four groups, it is best not to utilize group 4. Moreover, it can be seen that it is best to utilize all data augmentation groups of methods since the overall mIoU is increased by 6.2 compared to the standard non-augmented dataset. Even though the experiment with all utilized groups did not achieve the highest IoU for all the classes, the mIoU is the largest, and therefore, this experiment is considered to be the best. The output of the developed model for test images can be seen in Fig. 3. Test images are generated online by the stereo vision system of the mobile robot RAICO and utilized only for a visual representation of the accuracy of the considered system. It can be seen that CNN adequately semantically segments all the machines, with a small error regarding machines #1 (represented with blue) and #2 (shown in teal). The network can be run in near real-time with 24 FPS for a single camera system.

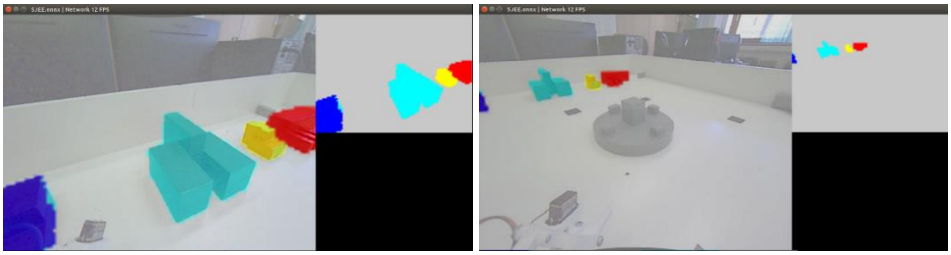

Fig. 3 – *The output of the perception system.*

6 Conclusion

In this paper, the authors analyze the impact of different groups of data augmentation methods on the accuracy of the deep learning semantic segmentation model utilized in mobile robot perception system. Resnet18 model is trained on a small dataset with only 159 images; however, when all data augmentation methods are applied, dataset contains 2607 images. Different groups of data augmentation methods are geometric image manipulation methods, noise addition methods, pixel intensity manipulation methods, and other methods. The experimental results show the usefulness of the proposed groups of data augmentation processes; their combination improves the mIoU by 6.2 and achieves an overall mIoU of 93.2. Moreover, the authors also conclude that the most valuable individual data augmentation group is noise addition methods contributing to the increase of around 63% of overall mIoU for all groups

considered. In particular, noise addition methods themselves achieved an increase of around 3.9 mIoU.

Future studies could include the analysis of the performance of different deep learning models trained on dataset where a combination of data augmentation methods is applied to each individual image in the dataset.

7 Acknowledgment

This work has been financially supported by the Ministry of Education, Science and Technological Development through the project "Integrated research in macro, micro, and nano mechanical engineering – Deep learning of intelligent manufacturing systems in production engineering" (Contract No. 451-03-68/2022-14/200105), and by the Science Fund of the Republic of Serbia, Grant No. 6523109, AI – MISSION 4.0, 2020 – 2022.

8 References

[1] J. Shabbir, T. Anwer: A Survey of Deep Learning Techniques for Mobile Robot Applications, arXiv:1803.07608 [cs.CV], March 2018, pp. 1 – 10.

[2] A. Jokić, L. Đokić, M. Petrović, Z. Miljković: A Mobile Robot Visual Perception System based on Deep Learning Approach, Proceedings of the 8th International Conference on Electrical, Electronics and Computing Engineering (IcETRAN), Ethno Village Stanisici, BiH, September 2021, pp. 568 – 572.

[3] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele: The Cityscapes Dataset for Semantic Urban Scene Understanding, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, USA, June 2016, pp. 3213 – 3223.

[4] C. Shorten, T. M. Khoshgoftaar: A Survey on Image Data Augmentation for Deep Learning, Journal of Big Data, Vol. 6, July 2019, p. 60.

[5] J. Zhang, Y. Zhang, X. Xu: ObjectAug: Object-Level Data Augmentation for Semantic Image Segmentation, Proceedings of the International Joint Conference on Neural Networks (IJCNN), Shenzhen, China, July 2021, pp. 1 – 8.

[6] S. Liu, J. Zhang, Y. Chen, Y. Liu, Z. Qin, T. Wan: Pixel Level Data Augmentation for Semantic Image Segmentation Using Generative Adversarial Networks, Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, May 2019, pp. 1902 – 1906.

[7] M. Negassi, D. Wagner, A. Reiterer: Smart (Sampling) Augment: Optimal and Efficient Data Augmentation for Semantic Segmentation, arXiv:2111.00487 [cs.CV], October 2021, pp. 1 – 10.

[8] S. G. Müller, F. Hutter: TrivialAugment: Tuning-Free Yet State-of-the-Art Data Augmentation, Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, Canada, October 2021, pp. 774 – 782.

[9] G. Kang, X. Dong, L. Zheng, Y. Yang: PatchShuffle Regularization, arXiv:1707.07103 [cs.CV], July 2017, pp. 1 – 10.

[10] A. Krizhevsky, I. Sutskever, G. E. Hinton: ImageNet Classification with Deep Convolutional Neural Networks, Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS), Lake Tahoe, USA, December 2012, pp. 1097 – 1105.

[11] K. Simonyan, A. Zisserman: Very Deep Convolutional Networks for Large-Scale Image Recognition, Proceedings of the 3rd International Conference on Learning Representations (ICLR), San Diego, USA, May 2015, pp. 1 – 14.

[12] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, P. Abbeel: Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, Canada, September 2017, pp. 23 – 30.

[13] K. He, X. Zhang, S. Ren, J. Sun: Deep Residual Learning for Image Recognition, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, USA, June 2016, pp. 770 – 778.

[14] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam: MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, arXiv:1704.04861 [cs.CV], April 2017, pp. 1 – 9.

[15] E. Romera, J. M. Alvarez, L. M. Bergasa, R. Arroyo: ERFNet: Efficient Residual Factorized ConvNet for Real-Time Semantic Segmentation, IEEE Transactions on Intelligent Transportation Systems, Vol. 19, No. 1, January 2018, pp. 263 – 272.