JAM CM Journal of Applied Mathematics and Computational Mechanics 2022, 21(4), 60-72
www.amcm.pcz.pl p-ISSN 2299-9965
DOI: 10.17512/jamcm.2022.4.06 e-ISSN 2353-0588

ESTIMATING THE DISTANCE TO AN OBJECT FROM GRAYSCALE STEREO IMAGES USING DEEP LEARNING

Joanna Kulawik

Department of Computer Science, Czestochowa University of Technology
Czestochowa, Poland
joanna.kulawik@icis.pcz.pl

Received: 16 October 2022; Accepted: 30 November 2022

Abstract. This article presents an innovative proposal for estimating the distance between an autonomous vehicle and an object in front of it. Such information can be used, for example, to support the process of controlling an autonomous vehicle. The primary source of information in research is monochrome stereo images. The images were made in compliance with the laws of the canonical order. The developed convolutional neural network model was used for the estimation. A proprietary dataset was developed for the experiments. The analysis was based on the phenomenon of disparity in stereo images. As a result of the research, a correctly trained model of the CNN network was obtained in six variants. High accuracy of distance estimation was achieved. This publication describes an original proposal for a hybrid blend of digital image analysis, stereo-vision, and deep learning for engineering applications.

MSC 2010: *68T40, 68T45*
Keywords: *estimating distance, stereo-vision, convolutional neural network, deep learning*

1. Introduction

In recent years, we have seen rapid development in the field of autonomous vehicles. Scientists are conducting intensive research on vehicle autonomy in various environments: water [1,2], ground [3–6], and air [7–10]. The vast majority of vehicle autonomy is based on meters and sensors. In systems, such as displacement or braking autonomy, the key information is the distance to objects. In practice, radars [11], lidars [6, 12, 13], or GPS [14] are mainly used to obtain this information. Each of the listed ones has some (its) limitations [15]. Radars and lidars are very sensitive to the properties of the shells of objects (obstacles). Luminance changes, including flash flares, are a significant problem. GPS systems require the object (obstacle) to be equipped with a transmitter. In addition, they do not work (they lose the signal) in closed rooms and places such as caves and tunnels. Devices consisting of an image recorder and another distance meter, such as a rangefinder (Kinect-V2 [16]) or optical sensors [17], are also used. Their unquestionable advantage is the speed of operation,

and the limitation is the lack of versatility (they are dedicated to strictly defined purposes). Combining various individual devices into sets is also practiced [18, 19]. In such a case, it is necessary to develop implementations integrating the obtained data from various sources. An unquestionable advantage is the possibility of extensive use of the information obtained.

A good solution seems to be using images from two cameras as a source of information. With their help, stereo pairs of images can be obtained. Each of the images individually can be used as a source of information for various functionalities. For example, semantic segmentation of the recorded scene, classification of objects, and a stereo system for distance detection [20]. The use of stereo-vision requires a good knowledge of the geometry of the assembled camera set and the optical parameters of each of the devices used [21]. Mounting such a set of cameras in a canonical layout simplifies the calculation method. In the classical analysis of stereo images, the information about the depth in the scene is obtained based on the disparity phenomenon. The difference in the position of objects on the surface of the left and right image allows the distance to a given object to be calculated. For this purpose, a dense disparity map is computed for each stereo image pair. The accuracy of such calculations strictly depends on the spatial resolution of the images. Therefore, the greater the distance from the object, the less accurate the results. Moreover, the computational complexity for each pair of frames of recorded video streams makes it challenging to apply stereo-vision in real-time.

Excellent results in image analysis are obtained using artificial intelligence (AI), in particular, convolutional neural networks (CNN) [22, 23]. Modern deep learning networks [24, 25] are used with great success to classify objects (images) and to predict specific values based on an image. It is only necessary to prepare a sufficiently large and representative data set for a given issue. In article [20], the authors obtained good results using R-CNN. They proposed a method to obtain information about the position of objects in the scene based on stereo images. Their solution is based on finding the object and classifying it. Then research is carried out only based on a 3D box estimator and a dense region-based photometric alignment method.

This article presents a hybrid solution, which is a way to estimate the distance of an autonomous vehicle from an obstacle (object). The only source of information is stereo pairs of dynamic grayscale images. The CNN deep learning network regression model was used as the solution method. It is a proposal that can be successfully applied to engineering solutions as a preliminary operation to provide data for the vehicle's steering. The article consists of 5 sections: 1. Introduction – presents an overview of currently conducted related research and an introduction to the research topic; 2. Materials and methods – what was used in the study was presented: 2.1. Dataset – how the set of images was developed, and 2.2. Deep Learning Network – the architecture of the prepared CNN network was discussed; 3. Research – the assumptions and the course of the conducted research are presented; 4. Results and discussion – shows the study's results and analysis; 5. Conclusions – Short conclusions from the research work were presented.

2. Materials and methods

The research was conducted taking into account their potential use in the autonomy of vehicle movement. Therefore, assumptions were made that the autonomous vehicle would be a moving ground vehicle and objects on its path would be potential obstacles. These are everyday objects such as cardboard boxes (cuboids of various proportions) and packaging (irregular shapes), all in the size range of 0.09 m to 0.6 m. Images recorded from two cameras located at a certain distance from the recorded scene and located at a small distance from each other will allow for obtaining two images of the same scene but with some differences [26]. As previously assumed, distance estimates were made based on the disparity phenomenon. The research method was based on the use of CNN, which, as we know, needs large data sets. Additionally, for learning the CNN model and verifying its operation, a correct answer is needed for the tested sample. Thus, in addition to the set of samples (stereo images), measuring the actual answer (distance from the vehicle to the obstacle) is necessary. Based on these assumptions, an original dataset was developed.

2.1. Dataset

A prototype test stand has been prepared, shown in Figure 1. As a symbolic autonomous vehicle, a traveling platform made of OSB with dimensions of 0.62 $\times$ 0.68 m and four wheels were used. On its surface, two identical LAMAX X9.1 cameras with $\alpha = 170^{\circ}$ angles were placed in the front. One method of recording video streams was adopted for the research: 30 fps in Full HD quality with a size of 1920 $\times$ 1080 pixels, and H.264 recording compression. The depth measurement was performed with the Benewake model CE30-D lidar using the ToF measurement method. According to the technical data, the device works from 0.4 m to 28.0 m. The laser beam scans the area in the range of 60° horizontally and 4° vertically. The result of the measurements is a point cloud with a resolution of 320 $\times$ 20 pixels and recorded with a frequency of 30 fps. A DELL laptop, model Inspirion 5567, was used to save the downloaded data. From the components mentioned above, a stand was built that met the conditions of the canonical system. The cameras and lidar have been arranged in the front place: their lenses and the shoreline of the platform form one plane. The distance between the cameras (the base of the canonical layout) is 0.49 m, and the lidar is halfway there.

Video streams and lidar readings were recorded on the prepared test stand. In the central part of the scene, an object was placed. Toward the object, the vehicle was moved at an average speed not exceeding 0.05 m/s. Surveys were usually done for distances from 9 m to 1 m with measurement accuracy up to 0.01 m. The use of three devices required their calibration and synchronization. Thanks to their canonical arrangement, it was possible to skip the process of rectifying images (frames). Lidar provides clouds of points found at a given moment and their location in the analyzed space. On its basis, the distance to the object (obstacle) was determined based on the

minimum number of points located with a sufficiently small distance between them. It was assumed that the most important aspect for each object is the distance to its front surface, the point closest to the autonomous vehicle. The first tests showed that there were delays in the lidar's work. Therefore, it was necessary to synchronize these devices by introducing an additional tag.

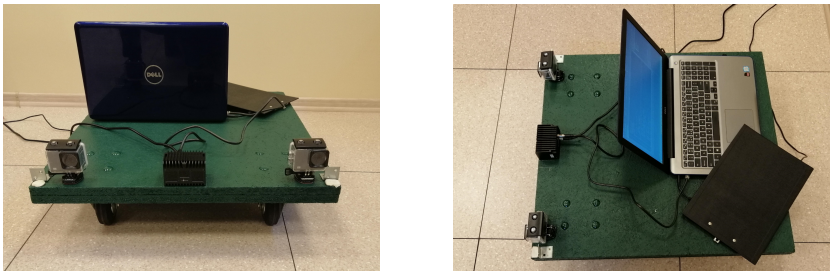


Fig. 1. Prototype test stand for recording video streams

After decomposing both video streams into frames and synchronizing them, the three data are dataset: two RGB images and the numerical value corresponding to the distance to the object on a given pair of images. It was assumed that detecting objects in the image was a preliminary operation for the conducted research. It was carried out by making a semantic segmentation of the image based on the levels of the color of the objects. Then operations such as erosion, noise removal, and dilatation were applied. Eventually, for each image pixel, all background except the subject was replaced with black. This was to rule out the background structure's influence on the model's learning. In addition, the images have been converted to grayscale, significantly reducing computational complexity. This set (two grayscale images and one value) is one data sample. The developed proprietary dataset includes 5,890 such original samples; they are intended for the process of training, validation, and testing of the CNN model.

It is well known that in deep learning, large data sets are required to train a network model properly; therefore, it was decided to enlarge the prepared set. Thus, three separate datasets have been developed. The first set named *Db*1 with 5,890 original stereo pairs of grayscale images. The second set, named *Db*2, contains 11,780 stereo pairs of grayscale images, consisting of the *Db*1 set and its copies rotated 180°. The third set, named *Db*3, of 23,560 stereo pairs of grayscale images, consisting of set *Db*2, and images of *Db*1 with the first 200 image rows removed and their copies rotated by 180°. With these three data sets available, it will be possible to perform comparative tests and analyze the obtained results.

2.2. Deep Learning Network

A CNN model was developed for research purposes. It is the task to predict one value (distance to an object) based on a stereo pair of digital gray images (left frame/image and right frame/image) containing this object. The original images are

1920 × 1080 pixels. They are scaled to the size required by the model when loading into the model.

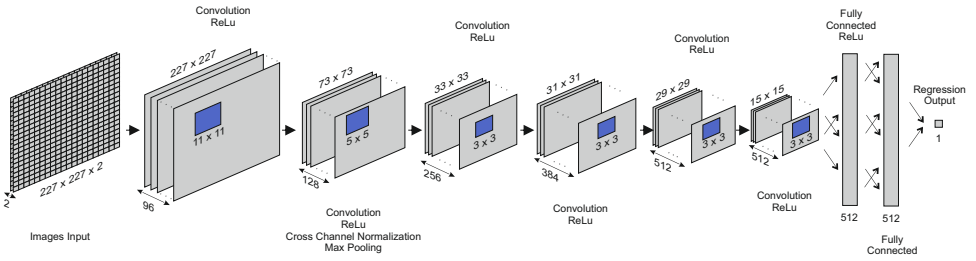


Fig. 2. Convolutional Neural Network architecture

A simple regression CNN model consisting of 19 layers was used, as shown in Figure 2. The input image layer is the matrix 227 × 227 × 2, where the first component of the range 227 × 227 is the left image and the second component is the right image. The second is a convolution layer with 96 filters of size 11 × 11, with their offset stride of [3 3] (stride every 3 places horizontally and every 3 places vertically) and [0 0 0 0] padding (no top, bottom left and right padding), after which has a relu (Rectified Linear Units) layer. The fourth and fifth layers are again a pair of convolution and relu layers, but this time 128 filters of the size 5 × 5 are used, and [2 2] stride. Then cross-channel normalization with five channels per element was applied. After which, max pooling was performed in the 3 × 3 range with a shift by [1 1] stride. Then a series of four pairs of layers of convolution and relu were used. Each one uses 3 × 3 size filters. The first two pairs had 256 and 384 filters, a [1 1] shift stride and [0 0 0 0] padding. Two consecutive pairs had 512 filters each, a [2 2] shift stride, and [1 1 1 1] padding was applied. In the following part, a series of fully connected layers were used in the scheme. First, fully connected with size 512, next a relu layer, and next a fully connected layer. The regression output layer was used as the last one. As a result, the network model returns one numerical value, the distance from the object measured in meters.

3. Research

The research aimed to teach the developed CNN model to estimate the distance from an object/obstacle. The input data are the sample sets presented earlier. It was assumed that the trained model could be used for distance estimation based only on two images without using other measuring devices such as lidar, rangefinder, etc. The research was carried out in the Matlab environment. Training, validation, and testing of the developed network model were performed on a computing station with the following parameters: Windows 10 Pro, AMD Ryzen 9 3950X 16-Core @3.5 GHz processor, 32 GB RAM installed, NVIDIA GeForce RTX 3080 graphics card.

The research work is divided into three independent parts (*Part*1, *Part*2, *Part*3), one for each of the prepared datasets. In each case, the entire set was randomly divided into three data types: 20% of the samples were separated as a test set, and the rest of the data was divided into validation and training sets. Due to the small dataset, eight times cross-validation was used in the studies. The process was repeated eight times. In each of them, one part was successively a validation set, and the remaining seven were a training set. The given test set was used only to test the correctness of the learned CNN model, so it did not take any part in training the model.

The process of training, validation, and testing was performed identically in each of the three parts of the study. The root mean square propagation algorithm (RMSProp) [27] was used to optimize the parameters in a given learning process. The size of the mini-batch was set to 30 samples. For comparative purposes, each part of the research was performed twice as independent learning processes lasting 50 and 100 epochs. In the learning process, a half-mean-squared-error was used as a loss function. After each epoch, the hyperparameters were fine-tuned based on the validation set. Then, the order of samples in the training set was mixed, and the next epoch of the learning process was performed, which reduced the risk of the model incorrectly learning the schema of consecutive images.

In each of the conducted learning processes, the course of the graphs was correct, and there were no sudden deviations (peaks). In the first five epochs, there has been a sharp decline in loss and *RMSE* figures. The course of the plot with successive epochs indicated a decrease in the oscillation range of the *RMSE* function and a gradual decrease in the value of the loss function.

4. Results and discussion

The research has resulted in many trained network models. Additional markings were introduced to systematize the analysis of the results. The process designations *Part*1, *Part*2 and *Part*3 refer to the datasets used. The letter *A* or *B* indicates that the number of epochs used in a given learning process is *A* – 50 epochs, *B* – 100 epochs. The correctness of the operation of each of the learned network models was verified on the previously separated test sets. Each test set sample has one real/correct answer. Thus, each test set has a vector of real responses $V(n)$, where $n = 1,..,N$ are consecutive responses for the samples of a given test set, and N is its number. Since each model is a regression model, it returns one numeric value as a response for each sample in the test set; this is the so-called predicted value. The set of all predicted values for a given test set is a value vector denoted as $Vp(n)$. For the analysis of the obtained results, the error value $B(n)$ was calculated for each sample, the formula (1):

$$B(n) = V(n) - Vp(n). \tag{1}$$

In the analysis of the results, the root mean square error (*RMSE*) parameter was taken into account, which was calculated for each training process, the formula (2):

$$RMSE = \sqrt{\frac{1}{N}\sum_{n\in N} B(n)^2}. \tag{2}$$

The real value is the distance to the object in the range of 1 m to 9 m. So it seems evident that the greater the distance to the subject, the greater the blur in the images. Thus, the risk of error in the distance estimation is also greater. Likewise, the shorter the distance, the greater the chances of higher accuracy (lower error). Therefore, the error value expressed in meters, as well as the *RMSE* seems to be insufficient. So it was decided to count the following parameters. First, for each sample $n \in N$, the percentage of error $Pb(n)$ to its real value was calculated in the formula (3):

$$Pb(n) = \frac{B(n) * 100}{V(n)}. \tag{3}$$

In regression models, to calculate the accuracy of their operation, it is necessary to define when the predicted answer is correct. By definition, the predicted and actual response should not be equal because it would indicate over-training. Hence, the next step is determining the thresholds p for the permissible error $Pb(n)$. Three thresholds are defined: $p1$, when $Pb(n) \leq 5\%$; $p2$, when $Pb(n) \leq 10\%$; and $p3$, when $Pb(n) \leq 20\%$. According to these thresholds, a given predicted value is classified into individual sets of results. Then, the number of individual sets of network model responses that meet the given threshold was counted, which was marked as follows: $N1$ for $p1$, $N2$ for $p2$, and $N3$ for $p3$. Then, the accuracy ($Acc1$, $Acc2$, $Acc3$) for individual sets was calculated as the quotient of the size of the set of correct answers relative to the set threshold ($N1$, $N2$, $N3$) to the size of the entire test set (N).

Six independent CNN model learning processes were carried out in the research. Three processes lasted 50 epochs ($Part1.A$, $Part2.A$, and $Part3.A$), and three lasted 100 epochs ($Part1.B$, $Part2.B$, and $Part3.B$). Each of the mentioned processes was performed in 8-fold cross-validation. 48 CNN learned models were obtained. According to the previously presented formulas, calculations were performed independently for each of the 48 testing processes.

As the aim of the research, it was assumed that a correctly functioning model should return values with an error within 10% (threshold $p2$). For the analysis, the number and accuracy for the 5% threshold ($p1$) were also checked, assuming that these are perfect results. These statistics were also made for 20% (threshold $p3$), but it was only to check the convergence of the results returned by the network model.

Due to a large number of different values, the obtained calculation results were systematized. Table 1 shows the average values for each of the six processes. In addition, the value of the standard deviation (*SD*) obtained within individual processes is also included. The table also consists of the average learning and validation time in seconds.

Table 1. Average results of CNN learned model testing across all six processes: *Part*1.*A*, *Part*2.*A*, *Part*3.*A*, *Part*1.*B*, *Part*2.*B* and *Part*3.*B*

		N	Time	Acc1	Acc2	Acc3	RMSE
Part1.A	Aver.	1178	**733**	77.4	97.8	**99.9**	0.246
	SD		12.33	7.79	1.59	0.16	0.033
Part2.A	Aver.	2300	1505	82.8	96.8	99.6	0.208
	SD		45.49	11.19	3.57	0.50	0.041
Part3.A	Aver.	4600	1837	**92.6**	**98.8**	**99.9**	**0.133**
	SD		39.22	4.35	0.96	0.19	0.035
Part1.B	Aver.	1178	**1456**	91.8	**99.1**	99.9	0.157
	SD		18.38	4.62	0.89	0.09	0.039
Part2.B	Aver.	2300	2952	90.6	97.7	99.6	0.149
	SD		41.82	8.66	3.10	0.55	0.039
Part3.B	Aver.	4600	5909	**96.3**	**99.7**	**100.0**	**0.121**
	SD		77.45	2.92	0.34	0	0.033

Moreover, for better visualization of numerical statements, individual values are presented in the form of charts. Figure 3 shows the precision values obtained for the individual processes of each Test Part. Figure 4 shows the average accuracy values obtained from each Part of the test.

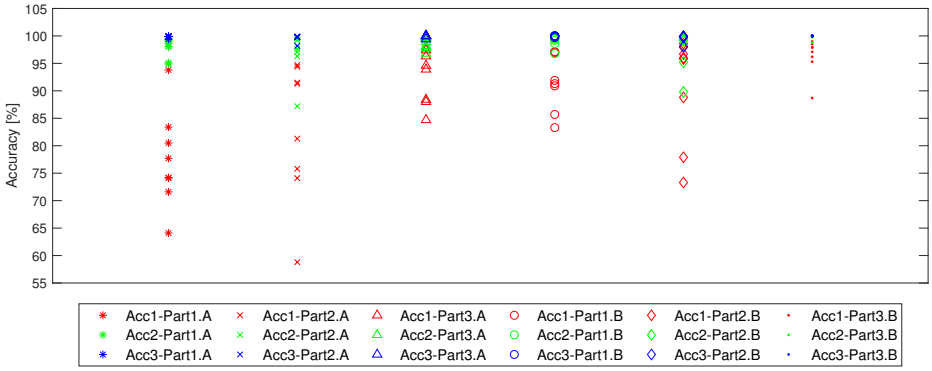


Fig. 3. Accuracy values obtained for the individual processes of each Part of the research. The values for the acceptable thresholds are marked with colors: red for the 5% threshold, green for the 10% threshold, and blue for the 20% threshold

It is easy to notice that the most significant discrepancies in the accuracy of individual processes are found for the $p1$ threshold (marked in red). From the processes lasting 50 epochs, the lowest average accuracy was the *Part*1.*A* process (only 77.4% with $SD = 7.79$). *Part*2.*A* obtained a slightly better average accuracy value (82.8% with $SD = 11.19$). The process *Part*3.*A* obtained the highest average accuracy (92.6% with the lowest value $SD = 4.35$). At the same time, better results are obtained for the tests run for 100 epochs. The best (*Part*3.*B*) achieved an average accuracy value of 96.3% with $SD = 2.92$. But it is also the process with the longest learning time (average 5909 s).

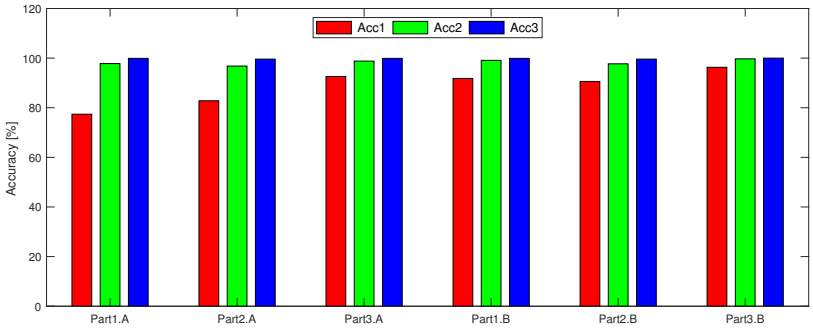
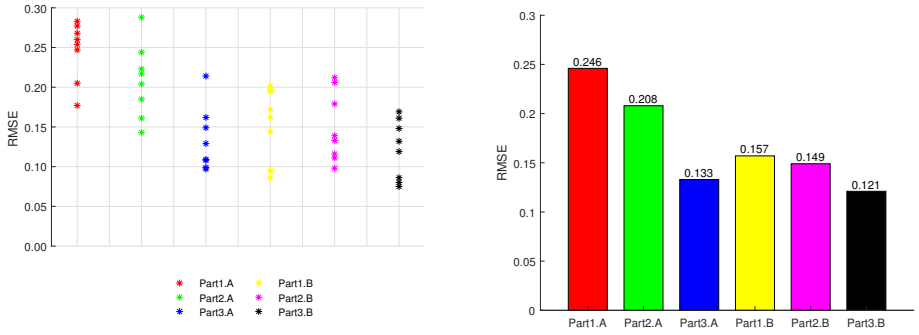


Fig. 4. Average accuracy values obtained from each Part of the test. The values for the permissible thresholds are marked with colors: red for the 5% threshold, green for the 10% threshold, and blue for the 20% threshold

When analyzing the results for the $p2$ threshold (marked in green), it should be stated that a much greater convergence characterizes them. In each of the tested six parts, the average accuracy exceeded 96.8% (the lowest is for $Part2.A$ where SD = 3.57). The best accuracy was achieved again for $Part3.B$ (as much as 99.7% with SD = 0.34). It is worth emphasizing the result obtained for $Part1.B$, where the average accuracy was 99.1% and SD = 0.89. As a reminder, it is a process with the most miniature data set, a relatively short learning time (second in sequence), and a learning process lasting 100 epochs.

The results obtained for the $p3$ threshold (marked in blue) confirm the high convergence of all parts. The average accuracy value for each of them exceeded 99.5%. In practice, for almost all of the samples, an error does not exceed 20%.



(**a**) (**b**)

Fig. 5. Summary: a) *RMSE* values in individual processes of each Part; b) averaged *RMSE* values for each Part

The summary of *RMSE* values is presented in Figure 5. It is easy to notice that better results were achieved for longer learning processes (B) than for shorter learning

processes (*A*). In addition, as expected, there is a relationship between the size of the data set and the results. The best results obtained in this metric were for *Part*3.*B*.

It should also be mentioned that, as predicted, the larger the dataset used in the training process, the more time it took to train the CNN model. A similar relationship exists between the number of epochs and the training time. The shortest learning time was achieved for the smallest set and 50 epochs, while the longest time was obtained when training on the set *Db*3 and 100 epochs.

Figure 6 shows the correlation graphs of the true distance values and the values predicted by the model.

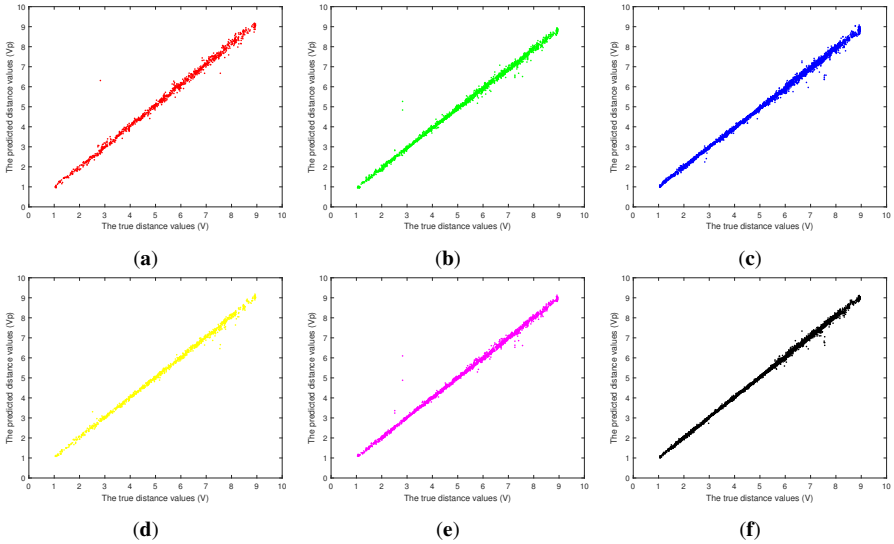


Fig. 6. Graphs showing the correlation of the true distance values and the values predicted by the model for sample processes from: a) Part1.A4; b) Part2.A7; c) Part3.A8; d) Part1.B4; e) Part2.B3; f) Part3.B7

5. Conclusions

As part of the research described in this article, a distance estimation system was developed. The works were carried out to potentially use them in supporting the autonomy of vehicle traffic. For this purpose, the regression model of the CNN network was used. Testing the developed prototype achieved excellent results. Only pairs of stereo images are used as sources of information. The research is based on the phenomenon of disparity. The use of grayscale images significantly reduces computational complexity.

Each of the six learning parts was successful and had the right results. All of them can be as support systems. Finally, when considering which model to apply, one should guide one primarily by what is available and the assumed goals. If accuracy is the most important for the user and has many computing resources, then *Part*3.*B*

will be the best choice. For more than 96% of the samples, the estimated distance did not exceed the 5% difference from the true distance. Moreover, almost all (99.7%) results fell within the 10% error limit.

On the other hand, $Part1.B$ will be more than enough if the potential user has limited computing resources. Then the learning process is fast, and the model easily achieves accuracy above 99% for the p2 threshold. His learning time was over three times shorter than the $Part3.B$.

The better learning outcomes for processes using more extensive data sets were not surprising. All the results were correct, but the outcome analysis shows that a better solution is to use 100 epochs to train the model. In this case, the lower entropy in distance estimation, lower (better) *RMSE* value, and better (greater) accuracy were obtained for the processes named $Part1.B$, $Part2.B$, and $Part3.B$.

The hybrid solution presented in the article combines the following issues: stereovision, image analysis, and deep learning. It can be successfully used for engineering solutions as a source of information. E.g., systems support the autonomy of vehicle traffic.

Funding: The project financed under the program of the Polish Minister of Science and Higher Education under the name ”Regional Initiative of Excellence” in the years 2019-2023 project number 020/RID/2018/19 the amount of financing PLN 12,000,000.

References

[1] Boulares, M., & Barnawi, A. (2021). A novel UAV path planning algorithm to search for floating objects on the ocean surface based on object's trajectory prediction by regression. *Robotics and Autonomous Systems*, *135*, 103673.

[2] Yang, F., Qiao, Y., Wei, W., Wang, X., Wan, D., Damaševičius, R., & Woźniak, M. (2020). Ddtree: A hybrid deep learning model for real-time waterway depth prediction and smart navigation. *Applied Sciences*, *10*(8), 2770.

[3] Wang, P., Gao, S., Li, L., Sun, B., & Cheng, S. (2019). Obstacle avoidance path planning design for autonomous driving vehicles based on an improved artificial potential field algorithm. *Energies*, *12*(12), 2342.

[4] Yu, H., & Kong, L. (2018). Autonomous Mobile Robot Based on Differential Global Positioning System. *In Proceedings of the 2018 IEEE International Conference on Mechatronics and Automation (ICMA)*. Changchun, China, 5-8 August 2018; 392-396.

[5] Kumar, D., Malhotra, R.,& Sharma, S.R. (2020). Design and construction of a smart wheelchair. *Procedia Computer Science*, *172*, 302-307.

[6] Gao, H., Cheng, B., Wang, J., Li, L., Zhao, J., & Li, D. (2018). Object classification using CNN-based fusion of vision and LIDAR in autonomous vehicle environment. *IEEE Transactions on Industrial Informatics*, *14*(9), 4224-4231.

[7] Lee, D., & Cha, D. (2020). Path optimization of a single surveillance drone based on reinforcement learning. *International Journal of Mechanical Engineering and Robotics Research*, *9*, 12.

[8] Jones, E., Sofonia, J., Canales, C., Hrabar, S., & Kendoul, F. (2020). Applications for the hovermap autonomous drone system in underground mining operations. *Journal of the Southern African Institute of Mining and Metallurgy*, *120*, 49-56.

[9] Oh, D., & Han, J. (2020). Fisheye-based smart control system for autonomous UAV operation. *Sensors*, *20*(24), 7321.

[10] Teixeira, M.A.S., Neves-Jr, F., Koubâa, A., De Arruda, L.V.R., & De Oliveira, A.S. (2020). A quadral-fuzzy control approach to flight formation by a fleet of unmanned aerial vehicles. *IEEE Access*, *8*, 64366-64381.

[11] Ort, T., Gilitschenski, I., & Rus, D. (2020). Autonomous navigation in inclement weather based on a localizing ground penetrating radar. *IEEE Robotics and Automation Letters*, *5*(2), 3267-3274.

[12] Tang, L., Shi, Y., He, Q., Sadek, A., & Qiao, C. (2020). Performance test of autonomous vehicle lidar sensors under different weather conditions. *Transportation Research Record*, *2674*(1), 319-329.

[13] Zhao, X., Sun, P., Xu, Z., Min, H., & Yu, H. (2020). Fusion of 3D LIDAR and camera data for object detection in autonomous vehicle applications. *IEEE Sensors Journal*, *20*(9), 4901-4913.

[14] Varun, G.M., Sunil, J., Saira, J., Paramjit, S., Mohammad, R.K., & Fadi, A.-T. (2022). An IoT-enabled intelligent automobile system for smart cities. *Internet of Things*, *18*, 100213.

[15] Chen, Y., Wu, Y., & Xing, H. (2017). A complete solution for AGV SLAM integrated with navigation in modern warehouse environment. *In Proceedings of the Chinese Automation Congress (CAC)*. Jinan, China, 20-22 October 2017; IEEE: Hoboken, NJ, USA, 6418-6423.

[16] Han, D., Nie, H., Chen, J., & Chen, M. (2018). Dynamic obstacle avoidance for manipulators using distance calculation and discrete detection. *Robotics and Computer-Integrated Manufacturing*, *49*, 98-104.

[17] Diwan, H. (2019). *Development of an obstacle detection and navigation system for autonomous powered wheelchairs*. University of Ontario Institute of Technology (Canada).

[18] Domínguez-Morales, M.J., Jiménez-Fernández, A., Jiménez-Moreno, G., Conde, C., Cabello, E., & Linares-Barranco, A. (2019). Bio-inspired stereo vision calibration for dynamic vision sensors. *IEEE Access*, *7*, 138415-138425.

[19] Wang, F., Lü, E., Wang, Y., Qiu, G., & Lu, H. (2020). Efficient stereo visual simultaneous localization and mapping for an autonomous unmanned forklift in an unstructured warehouse. *Applied Sciences*, *10*(2), 698.

[20] Li, P., Chen, X., & Shen, S. (2019). Stereo r-cnn based 3d object detection for autonomous driving. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7644-7652.

[21] Rzeszotarski, D., & Wiecek, B. (2008). Calibration for 3D reconstruction of thermal images. *In Proceedings of 9th International Conference on Quantitative InfraRed Thermography (QIRT)*. Krakow, Poland, 2-5 July 2008; 563-566.

[22] Eickenberg, M., Gramfort, A., Varoquaux, G., & Thirion, B. (2017). Seeing it all: Convolutional network layers map the function of the human visual system. *NeuroImage*, *152*, 184-194.

[23] Wenzel, M., Milletari, F., Krüger, J., Lange, C., Schenk, M., Apostolova, I., Klutmann, S., Ehrenburg, M., & Buchert, R. (2019). Automatic classification of dopamine transporter SPECT: deep convolutional neural networks can be trained to be robust with respect to variable image characteristics. *European Journal of Nuclear Medicine and Molecular Imaging*, *46*(13), 2800-2811.

[24] Kamsing, P., Torteeka, P., Boonpook, W., & Cao, C. (2020). Deep neural learning adaptive sequential Monte Carlo for automatic image and speech recognition. *Applied Computational Intelligence and Soft Computing*, 8866259.

[25] Woźniak, M., Siłka, J., & Wieczorek, M. (2021). Deep neural network correlation learning mechanism for CT brain tumor detection. *Neural Computing and Applications*, 1-16.

[26] Kulawik, J., & Kubanek, M. (2021). Detection of false synchronization of stereo image transmission using a convolutional neural network. *Symmetry*, *13*(1), 78.

[27] 1994-2021 The MathWorks, Inc. (2021) *MATLAB Documentation*; The MathWorks, Inc.: Natick, MA, USA.