Advances in Science and Technology Research Journal

Advances in Science and Technology Research Journal, 18(6), 247–258
https://doi.org/10.12913/22998624/191924
ISSN 2299-8624, License CC-BY 4.0

Received: 2024.06.24
Accepted: 2024.08.15
Published: 2024.09.08

Application of Deep Learning Methods for Trajectory Planning Based on Image Information

Artur Babiarz[1*], Małgorzata Kustra[1], Shuhuan Wen[2]

[1] Department of Automatic Control and Robotics, Silesian University of Technology, Akademicka 16 St., 44-100 Gliwice, Poland

[2] Department of Engineering Research Center Ministry of Education for Intelligent Control System and Intelligent Equipment and the Key Laboratory of Industrial Computer Control Engineering of Hebei Province, Yanshan University, Qinhuangdao, China

* Corresponding author's e-mail: artur.babiarz@polsl.pl

ABSTRACT

This work aims to develop a mobile robot utilizing neural network technology. The algorithm, programmed in Python on a Raspberry Pi 4B platform, is detailed across four main chapters. These chapters cover the fundamental assumptions of deep learning, the construction of the platform, and the research validating pattern recognition accuracy under various disturbances. The mobile platform employs a neural network to analyze selected traffic signs and translates the recognized patterns into corresponding motor movements.

Keywords: mobile robot, deep learning, image, neural network.

INTRODUCTION

Navigation challenges for mobile robots have been extensively studied, particularly in planning collision-free trajectories, as documented in [1]. Recently, the application of artificial intelligence methods to plan mobile robot movements has garnered significant interest [2]. A critical task in mobile robotics involves recognizing objects in the environment and making informed movement decisions. For this purpose, digital image processing using deep learning (DL) is used (see e.g. [3, 4]). Currently, deep learning usually functions as a stand-alone element of the system. The authors of the article [5] proposed an autonomous landing system for unmanned aerial vehicles, in which deep learning is only used to classify the terrain. Additionally, deep learning was also used to model the scene labels for each pixel as described in the research presented in the works [6, 7], which led to the semantic mapping results.Deep learning is widely used in computer vision for recognition. Considering the receptive recognition field, these tasks can be divided into patch-wise and pixel-wise classification [8, 9, 10]. A typical example of piecewise classification is image classification. Its purpose is to assign a label to each image. In recent years, canonical image classification datasets have emerged to validate new algorithms. These datasets consist of handwritten digits such as mnist [11], street view house numbers such as the svhn dataset [12], and objects such as cifar-10 [13]. Another example of point classification is the so-called detection, location and recognition of objects. In this task, one has to first detect the possible location of an object and then recognize it.

Taking advantage of the wide application of deep learning, researchers use these methods, among others, to plan the movements and navigation of mobile robots. Authors of [14] describe an indoor exploration algorithm for mobile robots using a hierarchical structure. The proposed structure combines several layers of a convolutional neural network with the decision-making

process, and the planning system is trained from start to finish, taking only visual information (RGB-D information) as input. Then it generates a sequence of the main direction of movement as a control signal, as a result of which the robot achieves the ability of autonomous exploration. A very similar solution is presented in [15], where the authors also proposed a method of exploring the environment by a mobile robot based only on the input signal from the RGB-D camera. In [16] authors described an internal analysis algorithm for mobile robots that hierarchically connects various layers of a convolutional neural network (CNN) with the decision-making process. The entire system is comprehensively trained on the basis of data recorded by a depth camera (RGB-D). The results include a proposed model for expanding the robot's critical movement directions to achieve autonomous analysis capability. The combination of deep learning with the RRT (eapidly-exploring random tree) method is presented in the paper [17]. The authors proposed an algorithm for planning the optimal trajectory of a mobile robot equipped with a digital camera in an environment with static and dynamic obstacles. The discussion about path-planning methods that use neural networks, including deep reinforcement learning, and its different types, such as model-free and model-based, Q-value function-based, policy-based, and actor-critic-based methods is presented in [18].

The practical implementation of deep learning in real robotic tasks poses significant problems. Thus, the ultimate goal of this paper is to find solutions to these problems, with the vision of providing practical DL methods for autonomous navigation that are effective in training, generalizable, and can be implemented in real mobile robots. The main idea is to use low-budget elements to build a research station and an actual robot. Moreover, the research results presented below can also be used in real-world applications. As is known, road sign recognition is an essential part of perception in autonomous vehicles, which is currently performed almost exclusively using deep neural networks (DNN). However, it is known that DNN are susceptible to adversarial attacks. Road signs are particularly promising for adversarial attack research due to the ease of conducting real-world attacks using printed signs or stickers. Therefore, it can be concluded that research on the use of neural networks for road sign recognition under various interferences is a significant problem in the current world.

This study has the following structure. The second section presents a design of the research station. The third section includes the assumptions of the studying. The fourth section presents results and their analysis. Finally, discussion about obtained results and further work are shown.

PREPARING A STAND FOR ALGORITHM TESTING

In order to carry out tests, a simple mobile platform was created, consisting of a chassis with two DC motors and controller. The entire system was connected to a Raspberry Pi 4B. The camera was connected to the robot using a USB interface. The algorithm based on image registration by a camera and the recognized pattern. As a result, it is important to have a constant preview of the camera image, along with information about what character is currently visible and how accurately the algorithm identifies it (i.e. probability expressed in percentage). This requires a permanent connection of the Raspberry Pi microcomputer to the monitor. Since the research aims to evaluate deep learning quality in robot navigation, controlling the mobile platform through sign recognition without physical movement was deemed the best solution.

The stationary platform allows continuous monitoring of the neural network's character recognition probability via the monitor. The research stand presented in Figure 1 consists of mobile robot with controller (1), digital camera (2), Raspberry Pi 4B (3), monitor with a view from a digital camera (4). The written movement algorithm allows the robot to drive through the maze, but it is not a physical route. However, there is a standard examination of the quality of the algorithm by recognizing characters and responding appropriately. This is a compromise between a physical robot moving through a maze and a kind of "simulation" that allows the algorithm to be tested in reality, rather than in an ideal environment without disruptions.

If the probability of recognizing a particular character is higher than the value set as the limit (e.g. above 95% recognition certainty), the mobile platform changes its state. The condition is related to the movement of the engines, i.e. appropriate response to certain road signs. Table 1 shows the dependence of the movement of the motors on the state in which the robot is located.

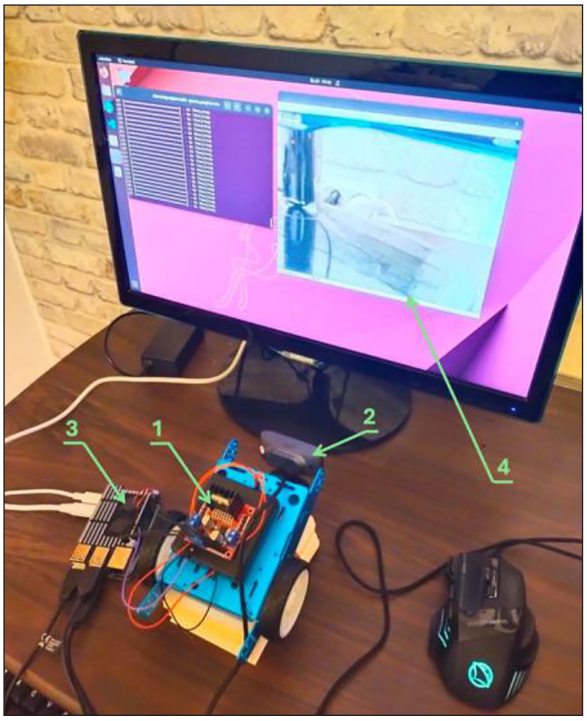

Figure 1. A research stand

ASSUMPTIONS AND PURPOSE OF THE RESEARCH

Using the knowledge related to current solutions in terms of machine learning and the growing popularity of artificial intelligence, an attempt was made to create a mobile device that uses deep learning in navigation. It was decided to create a real robot with an implemented neural network, programmed in Python. Thanks to a real device, it is possible to study the influence of external disturbances, such as a shadow created by the sun's rays. A model in a simulator can make a neural network work under ideal conditions, but not in real life. The construction of the mobile device consists of a chassis on wheels and a camera, which, in conjunction with the neural network on the Raspberry Pi 4B, is able to recognize road sign objects and, based on them, drive through the maze. The main task of the constructed robot is to examine how effective deep learning is in real conditions and whether complete vehicle autonomy is possible based on neural networks.

The neural network was programmed using Python 3.10 and the OpenCV library for image processing. It was used primarily to read images for pre-processing data for learning the neural network, as well as for the preparation of patterns. In addition, thanks to the combination of a computer camera with OpenCV, it was possible to use them to record the image from the computer on an ongoing basis to check whether the network is properly taught the patterns. In addition, the OpenCV package will also be used in the mobile platform itself for road sign detection.

Preparation of patterns for the use of deep learning

The first stage of starting work on the neural network was to consider how to prepare patterns for its learning. Deep learning is based on learning patterns through a convolutional network based on pictures of images. Therefore, it was necessary to prepare appropriate folders for each type of patterns, and then assign photos to them, on which the algorithm would learn. In order for the network to be properly trained, many images had to be generated to train it.

Many different approaches to how samples should be prepared have been considered. At the very beginning, it was decided that the network would recognize five different road signs. These will be signs indicating the end of the route, stop and three arrows responsible for moving to the right, straight and left (Figure 2). It was found that five different objects to be identified would be a sufficient number for the robot to be able

Table 1. Dependence of the state on the movement of the motors

State	Condition description	Left engine speed	Right engine speed	Movement of robot
1	Reading the end of the road sign	100% max speed counterclockwise	100% max speed counterclockwise	Platform rotation
2	Reading a left turn sign	0% of maximum speed	100% max speed clockwise	Turn left
3	Reading a right turn sign	100% max speed counterclockwise	0% of maximum speed	Turn right
4	Reading the go straight sign	100% max speed counterclockwise	100% maximum speed clockwise	Moving straight
5	Stop sign reading	20% of maximum speed counterclockwise	20% of maximum speed clockwise	Moving straight at reduced speed

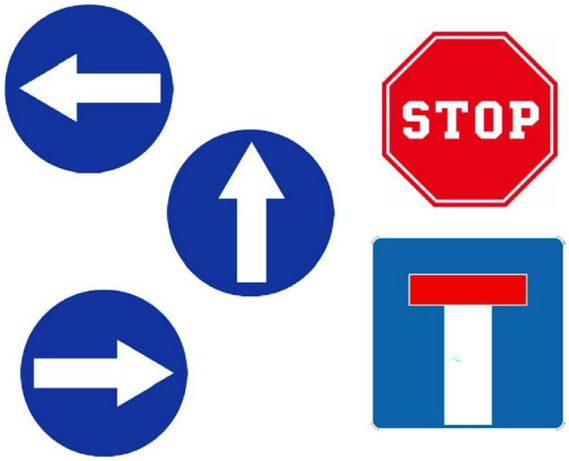

Figure 2. Road signs that the algorithm will recognize

to successfully navigate the maze, as well as to check the quality of the algorithm.

Implementation and learning of a neural network

The most popular neural network for image recognition processing is the convolutional network (of the convert type). Within the Keras package is a combination of two main layers, Conv2D and MaxPooling2D. In the designed model, they play the main function as the basis for the created algorithm, but other, additional layers have also been added. So that the neural network does not require too much data processing and the Raspberry Pi 4B platform can process it, all images have been reduced to a size of 50 by 50 pixels. It was found that, depending on the results achieved, it would be possible to enlarge the image. For relatively simple elements, such as road signs, this neural network size will be sufficient to be able to recognize images well.

The basic element of a neural network is a layer. In the Keras package, it consists of a calculation function and some stored state that is updated in the subsequent ones layers. The library used contains many built-in layers that are used in deep learning, including: image processing.

Programming the algorithm began by specifying a sequential neural network model using the keras.Sequential() command. This involves creating a stack of layers, where each layer has only one output and input tensor. Thanks to this model, it is possible to gradually arrange subsequent layers using the model.add() expression. This is especially useful in the case of a convolutional network, which requires the sequential application of two layers.

The next pair of layers is a stack of Conv2D() and MaxPooling2D() layers. At the very beginning, the network accepts an object (tensor) with a specific structure. This shape is determined by three parameters – the height and width of the image and its depth. In the case of this neural network and prepared images values are equal to input_shape = (50, 50, 3). Images for learning the network have dimensions of 50×50 pixels and are in RGB colors, hence the depth is three. The tensor specified in the declaration is accepted only in the first layer, in subsequent layers the arguments are transferred automatically. Except for the first layer, where there was a need to enter an argument, all Conv2D() layers are similar in their structure. They are defined by two main parameters, the size of the patches extracted from the input data and the depth of the output feature map. Patches are matrices with dimensions usually 3 × 3, depth is the number of filters. Here, convolution works by moving a matrix of a specific dimension on the feature map and extracting a three-dimensional patch. This patch is then transformed to obtain a vector with a length equal to the depth of the output feature map. All vectors are later rebuilt to create a new three-dimensional output map.

The MaxPooling2D() layer complements Conv2D() by reducing the feature map size. It samples the input using a 2×2 extraction window, which moves along the height and width dimensions, capturing the maximum value of each channel within the input window. This mechanism is used to reduce the number of processed feature map elements and to implement a hierarchy of spatial filters. After implementing a few image processing layers, the next class was layers.Flatten(). Its task is to flatten the input data into one dimension. This layer is necessary to be able to add layers unrelated to convolution or scaling. Layers.Dropout() prevents overfitting. This is a phenomenon that means that the model is only able to recognize the object correctly in the case of input images, and any slightest change causes the result to deviate from the correct solution. This phenomenon may occur when there are few examples in the training set. To avoid overfitting, a Dropout() layer was introduced, where 1/3 of the input units are discarded. The last layers are regular Dense() layers with a specific number of output units. The layer accepts a two-dimensional tensor with the length

returned from the Dropout() layer, and returns a tensor with a specific length specified in the function, in this case first with length 500, and finally 5 (number of labels) [19]. An activation function has been defined in the Conv2D() and Dense() layers. In most layers it is relu, only in the last one it is sigmoid. This allows you to generate values that can be interpreted as probability. The relu function, or rectified linear unit, zeroes negative values, and sigmoid rescales the values so that they are in the range from 0 to 1 [19]. Ultimately, the resulting neural network consists of 12 layers. Its entire appearance is presented in Listing 1. All network transformations in sequence and all parameter values established are visible there.

The structure of the modeled neural network:

Listing 1

```
model = keras.Sequential()
model.add(layers.Conv2D(64, (3, 3),
activation=”relu”, input_shape=(50,50,3)))
model.add(layers.MaxPooling2D((2,2)))
model.add(layers.Conv2D(128,
(3, 3), activation=”relu”)) model.
add(layers.MaxPooling2D((2,2)))
model.add(layers.Conv2D(255,
(3, 3), activation=”relu”)) model.
add(layers.MaxPooling2D((2,2)))
model.add(layers.Conv2D(255,
(3, 3), activation=”relu”)) model.
add(layers.MaxPooling2D((2,2)))
model.add(layers.Flatten())
model.add(layers.Dropout(0.33))
model.add(layers.Dense(500, ac-
tivation = ”relu”))
model.add(layers.Dense(5, ac-
tivation = ”sigmoid”))
```

After creating the deep learning algorithm, the loss function, optimizer and metrics needed to be defined. Due to the fact that the network, through the last layer with the sigmoid activation function, returns values interpreted as probability, the most appropriate choice for the loss function is binary crossentropy, and the best choice of the optimizer will be Adam() with a learning_rate value of 0.0001. Accuracy was chosen as the metric to monitor the accuracy of the neural network [19].

RESULTS

This section is divided into four subsections regarding the results of the experiments performed. Individual experiments demonstrate the quality of the designed and programmed neural network.

Experiment no. 1 – the influence of the size of the pattern

The first study was to check the influence of the pattern size on the quality of road sign identification. The study aimed to determine what size of individual characters is sufficient for the algorithm to recognize correctly pattern and rotate the motors according to the desired effect. Checking the influence of size is directly related to determining the distance from which the mobile platform is able to detect a road sign and behave in accordance with the programmed instructions. The probability expressed as a percentage directly translates into the quality of the results obtained. To create conditions enabling the tests to be performed, it was necessary to prepare a field with road signs of various sizes and a previously constructed stand for testing the algorithm. Patterns of three signs were made, i.e. a go straight sign and a stop sign and the end of the road, with six different sizes. Individual signs were placed next to each other, at an equal distance from the camera.

Three measurement series were performed, one for each character. First, tests were carried out for the order to go straight, then for the stop sign, and finally for the end of the road. The results are presented in the Tables 2, 3, 4.

The tests were performed over a short period of time, under the same light source. Thanks to this, stable conditions were ensured with similar disturbances for each sign. The camera was positioned stably for each test. Only the shadows

Table 2. Recognizing the straight move sign

Pattern size (diameter expressed in mm)	48 mm	33 mm	28 mm	23 mm	18 mm	13 mm
Recognized sign	Move straight	Move straight	Move straight	Move left	Move left	Stop
The probability with which the sign was recognized	100%	88%	66%	47%	70%	54%

Table 3. Recognizing the stop sign

Pattern size (diameter expressed in mm)	48 mm	33 mm	28 mm	23 mm	18 mm	13 mm
Recognized sign	Stop	Stop	Stop	Stop	Stop	Stop
The probability with which the sign was recognized	100%	100%	100%	98%	95%	72%

Table 4. Recognizing the end of the road sign

Pattern size (diameter expressed in mm)	48 mm	33 mm	28 mm	23 mm	18 mm	13 mm
Recognized sign	The end of the road	The end of the road	The end of the road	The end of the road	Move left	Move left
The probability with which the sign was recognized	100%	99%	94%	84%	78%	89%

falling on individual road signs could influence the algorithm's evaluation of the patterns. Road signs were recognized by the camera of a mobile platform placed 20 cm from the model. The first thing it can notice when analyzing the results is the fact that for a stop sign, regardless of the size of the pattern, the algorithm correctly recognizes the sign. This may be due to the fact that the stop sign is very distinctive compared to other road signs that the network recognizes. Due to the presence of a red background tint, the neural network is able to distinguish it from others, even with a small pattern size, around 12 mm in diameter. In the case of the sign indicating straight movement, there is a significant deterioration in the quality of sign recognition. Up to a diameter of 28 mm, the neural network was able to correctly identify the pattern. In the case of a signpost with a diameter of 23 mm and 18 mm, the algorithm classified it as a left turn. This confirms insufficiency training the neural network, but the deep learning program was still able to correctly recognize the background on which the arrow was located. In the case of the 13 mm diameter, the classification was completely incorrect, by confusing the pattern with a stop sign. The end of the road sign was identified by the deep learning algorithm better than when moving straight. The neural network correctly recognized the sign for diameters from 45 mm to 25 mm. The algorithm achieved erroneous results at sizes 20 mm and 12 mm, where it classified the pattern as moving to the left. This is a permissible error due to the background colour, but the program completely ignored the red fragment.

Experiment no. 2 – the influence of changing light

Another test of the quality of the deep learning algorithm was to test how light affects the quality of signpost recognition. On roads in natural environments, light intensity is constantly changing due to changing weather conditions. For this reason, the neural network should recognize individual patterns well regardless of the lighting. The experiment was carried out for all five characters that the program can identify. The signs were placed at the same distance from the camera against the same background.

Three series of tests were performed for each sign. First, it was checked how well the algorithm works in a well-lit research stand. Then, the quality of the algorithm was checked in twilight, when no light source was present, and the experiment was performed in the afternoon. Finally, tests were performed in a dark room, with side light present. The achieved results are presented in Table 5, which shows the probability with which the sign was recognized depending on the light.

Table 5. Probability of recognizing the sign expressed as a percentage for experiment no. 2

Light/sign	Move straight	Move right	Move left	Stop	The end of the road
Top light	100%	99%	99%	100%	100%
Dusk	100%	96%	82%	100%	100%
Side light	100%	97%	79%	100%	100%

The tests performed showed that for straight ahead, end of road and stop signs, variable light has no effect on pattern recognition. This may be due to characteristic elements that are present in road signs (e.g. red background of a stop sign). The identification probability was 100% regardless of lighting changes, which means that the neural network in these cases excluded other possible solutions. It is possible that the algorithm in the case of a larger database of recognized elements (e.g. increased by a no-entry sign) would have difficulty assessing the Stop sign in poorer light. Different results were achieved for the right and left movement sign. In the case of these patterns, it can be assumed that the neural network does not recognize the character correctly in all cases and has some problems in distinguishing left-right arrows. This conclusion can be drawn from the fact that if the signs changed to a right or left movement during the study, the neural network needed a short time to stabilize the type of the recognized signpost. For the top light, the results of the driving order to the right and left were equal to 98% and 99%, respectively. When the lighting changed, the probability of identification changed. In the case of movement to the right, these results were high, and the different light intensity did not drastically affect the evaluation of the pattern, the difference was of the order of 1-2%. Moving to the left, however, achieved much worse results when the light changed. In twilight, the recognition quality dropped to 82%, and in side light to 79%. This means that in both cases the probability decreases by approximately 20%. With the default limit value, the neural network would not be able to respond appropriately to the sign. Even though the training and validation sets were equal for each signpost, there were problems with correct recognition of right-left signs. This may be due to the fact that the patterns are very similar to each other (the occurrence of a horizontal white line), and scaling the image to a smaller size distorts the estimation of which direction the arrow is pointing. Changing light affects the occurrence of errors even more drastically.

Experiment no. 3 – the effect of occluding the sign

In the third study, it was checked how much occlusion of the sign was allowed so that the neural network would be able to correctly identify the pattern. In fact, some acts of vandalism occur in the form of damage to the sign, which disturbs the correct assessment. Also occurring natural conditions, such as leaves or snow, may partially obscure the signpost. Checking the extent to which a sign can be hidden so that the neural network recognizes it correctly is related to perception distortions occurring in the natural environment. The same stand was used for the experiment as in the case of testing variable light. The covering effect was achieved by partially covering the sign. Four measurement series were performed for each of the signs with occlusions of 0%, 25%, 50% and 75%. Covering was done by dividing each sign into four equal parts and gradually covering more and more quarters in the signpost. The obtained results of the probability of recognizing a specific pattern are presented in Table 6.

The lighting and the distance at which individual characters were recognized were constant for each pattern. Therefore, the influence of other elements that could interfere with the tests was minimized. The assessment of which sign was currently visible was influenced only by the degree of occlusion. During the examination, care was taken to ensure that the characteristic element was still visible on each of the signs. This was to prevent mistakes in the event of covering the element that distinguishes a given sign from others. For example, if the arrowhead of a traffic sign to the right is obscured, it would be very difficult for the created neural network algorithm to distinguish it from an order to go left. Analyzing the results in Table 6, it can be seen that for no and 25% occlusion, the neural network achieves very good results in recognizing all characters. The exception, however, is the straight move sign, for which the probability is only 92%. Is this is still

Table 6. Probability of recognizing the sign expressed as a percentage for experiment no. 3

Sign	Move straight	Move right	Move left	Stop	The end of the road
No obscuration	100%	100%	100%	100%	100%
25% occlusion	92%	99%	100%	100%	100%
50% occlusion	88%	93%	99%	100%	99%
75% occlusion	Not recognized	Not recognized	95%	99%	98%

a significantly dominant value, but no longer sufficient to respond appropriately with the default limit value. With other scores equal to 99% or 100%, this figure stands out greatly from the rest of the scores.

For an occlusion of 50%, the results obtained for straight and right movement are much worse than for the other three signs. The decrease in probability may be caused by the fact that for these signs the elements influencing the identification of the shape of the white arrow in the signs are obscured. However, this phenomenon does not occur in the case of the left turn sign, which is recognized with a probability of 99%. For a 75% occlusion, straight and right movement signs were not recognized. In both cases, the neural network confused them with movement to the left. The probability of the sign moving to the left was estimated at 95%. This is slightly lower than for smaller occupancies, but still very high (higher than for the straight motion sign at 25% occlusion). The stop sign and the end of the road sign were recognized with a probability of 99% and 98% respectively. This may be due to the presence of very characteristic colour arrangements. By analyzing the table with the results of recognizing individual signs, it was decided to extend the experiment. For signs moving straight, right and left, an occlusion of 50% was used, including covering the direction of the arrow. There is such a significant disturbance in direction assessment that the mobile platform should not react to the change in pattern. The results are presented in Table 7.

From the conducted study, it can be concluded that covering the arrowhead significantly affects pattern recognition. The results achieved vary greatly. In the case of right and left traffic when the direction of the arrows visible on the road sign was covered, the results deteriorated significantly. The opposite effect was obtained when moving straight. In this case, covering the arrowhead increased the efficiency of sign recognition. However, when the arrowhead was covered, none of the characters exceeded the limit value, which is actually desirable when the mobile platform is moving. Covering the direction of the arrow causes the person navigating the vehicle to rely more on their guesses than on objective judgment. In case the individual is not sure what the outcome should be, the robot should not make the decision on its own.

Experiment no. 4 – the effect of changed colours

During previous experiments, it was noticed that the recognition of a sign is greatly influenced by the presence of characteristic elements, such as a red horizontal rectangle for the end of the road sign. In some cases, their presence made the road sign recognizable with a very high probability, even when it was small or largely obscured. Therefore, it was concluded that it should be checked to what extent the absence of these elements will affect the probability check. It was decided to perform a series of tests for three signposts whose colours were changed. For the research, it was decided to take into account the stop sign, the end of the road and the traffic sign. Their appearance has been subject to modifications, such as a completely changed colour palette, brightening colours or presenting signposts in gray scale. Eliminated this allows for distinctive details to occur in many ways. Examples of colour modifications of signs are presented in Figure 3. Tests were performed separately for each road sign pattern. In each case, it was examined whether the neural network

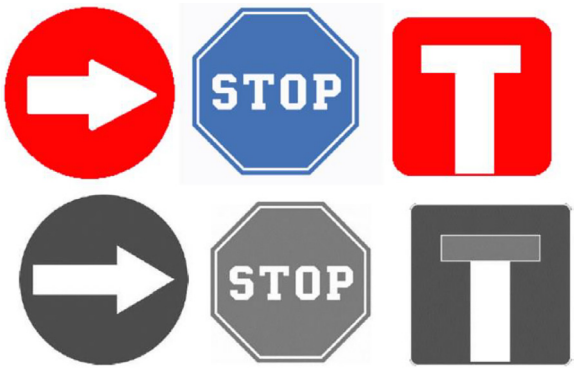

Figure 3. Examples of changed sign colors

Table 7. Probability of recognizing the sign expressed as a percentage for covering the arrowheads

Sign	Move straight	Move right	Move left
No obscuration	100%	100%	100%
50% occlusion (arrowhead visible)	88%	93%	99%
50% occlusion (arrowhead obscured)	93%	78%	72%

Table 8. Recognizing the straight move sign

Pattern color	Original	Faded original colors	Green	Orange	Yellow	Red	Gray	Black
Recognized sign	Move straight	Move straight	Move left	Stop	Stop	Stop	Stop	Move straight
The probability with which the sign was recognized	100%	100%	40%	100%	56%	100%	39%	46%
Was it recognized correctly?	Yes	Yes	No	No	No	No	No	No

Table 9. Stop sign recognition

Pattern colour	Original	Faded original colours	Green	Orange	Yellow	Red	Gray	Black
Recognized sign	Stop	Stop	Stop	Stop	Stop	Stop	Stop	Stop
The probability with which the sign was recognized	100%	99%	39%	79%	83%	75%	75%	53%
Was it recognized correctly?	Yes	Yes	Yes	Yes	Yes	No	Yes	No

Table 10. Recognizing the end of the road sign

Pattern colour	Original	Faded original colours	Green	Orange	Yellow	Red background, white element	Gray	Black without red element	Black with red element	Blue background, white element
Recognized sign	The end of the road	The end of the road	Stop	Stop	Stop	Stop	Stop	Move straight	The end of the road	Move straight
The probability with which the sign was recognized	100%	100%	65%	94%	47%	100%	92%	84%	92%	98%
Was it recognized correctly?	Yes	Yes	No	No	No	No	No	No	Yes	No

recognized the correct sign and with what probability the evaluation was made. The results are presented in the Tables 8, 9, 10.

The above tables clearly show that colour has a very significant impact on the recognition of a road sign. It can be seen that some background shades strongly influence the reading of the correct pattern by the neural network. The algorithm performed best at stop signs, where it got the sign correct 75% of the time. For the straight ahead sign, the neural network gave the result correctly in 35.5% of cases, and for the end of the road sign in 30%. For the first character, i.e. straight movement, it is clear that the algorithm had major problems in distinguishing patterns. For the original and faded colours, the recognition probability was 100% and correct. However, in the case of orange and red shades, the neural network identified patterns as stop signs with full probability. Due to the very characteristic background of the stop sign, the algorithm did not take into account the position of the arrow and made incorrect decisions regarding movement. Additionally, the algorithm had trouble recognizing the colour yellow. He also assumed he "saw" the stop sign, but was only 56% confident in his decision. This means that changing the colour to yellow does not have such a significant impact on the incorrect identification of the pattern as in the case of orange and red. Incorrect results were also obtained when changing the background to green and gray. In the case of the green colour, 40% confidence in the assessment of the sign of movement to the left was obtained. This is an erroneous result, but low enough to prevent the mobile platform from reacting in an undesirable way. In the case of a gray sign, the neural network recognized the stop sign with 39% probability. This is also a low enough result, but very close to correct. A slight change in colour from gray to black allowed the algorithm to correctly recognize the sign. In the case of black, this is a result that does not allow the algorithm to respond appropriately (46%), but it is more desirable than in the case of gray.

The best results were achieved for the stop sign. The program was unable to recognize the pattern correctly only in two extreme cases - for blue and black backgrounds. Despite correct

recognition in the remaining colour variants, the results were different for each shade. For original and faded shades, the recognition result was around 100%. In the case of orange, the confidence in the assessment dropped to 79%. Comparing the result to an attempt to recognize a straight traffic sign on an orange background, it can be seen that the algorithm was more likely to be dealing with a stop sign in the case of the changing colour of straight traffic than for the changed original. Moreover, the identification probability was higher for the yellow colour than for the orange colour. In the case of gray scale, the stop sign was recognized with a probability of 75%. This is a high value, but it does not allow the algorithm to respond appropriately. Contrary to the case of moving straight, slightly increasing the colour of the black background automatically caused the sign to be misrecognized and identified as moving to the right with a probability of 53%.

The worst results were achieved for the end of the road sign. As with the previous tests, the recognition probability was 100% for the original and faded colours. The next and last correct result was a score of 92% for a signpost with a black background and a red horizontal element. This means that the end of the road sign is identified by the neural network only based on the red fragment. The green and yellow background meant that the signpost was misrecognized, but the probability was not high, only 65% and 47%. A higher value was achieved with an orange background. As in the previous cases, the sign was mistaken for a stop pattern, but this time the probability was 94%. This means that the closer the colour is to red, the greater the possibility of confusing it with the above. sign. For the gray and black patterns, a similar effect can be seen as in the case of the straight move sign. A slight change in colour to a darker one reduced the likelihood of an incorrect assessment. The background colour and the presence of characteristic elements greatly influence the recognition of the sign. This is very visible in the case of tests for red and blue backgrounds. Both patterns were misidentified with very high probability. In the case of a red background, the sign was identified as a stop sign and the presence of a white T-shaped element was irrelevant. Similarly, with a blue background, the signpost was recognized as a straight traffic sign by the white, vertical element.

Although there are no actual road signs with changed colours, significant colour fading may occur due to environmental influences. For all cases of the analyzed road signs, the recognition was almost 100% correct. The algorithm would respond correctly to the occurrence of this type of disruptions during autonomous driving.

CONCLUSIONS

From the above research, detailed conclusions can be drawn regarding the general operation of the resulting neural network in the event of various disturbances. Performing a series of tests for signs allows you to determine how well the algorithm has been trained and in what cases it makes incorrect decisions regarding movement. Moreover, the tests show what details the program pays most attention to, and the presence of specific elements can completely change the results obtained. In the case of the pattern size influence test, the correct recognition of a stop sign with a high probability of identification stands out very well. This is due to the presence of a very characteristic element, the red background of the road sign. This is a part that is not present in the other four signs, so it greatly influences the recognition of the signpost. The neural network algorithm directly attributed its occurrence to the stop sign. This is confirmed by tests performed for the changed colour palette, where in the case of a red tint in the field, the program recognized a stop sign regardless of what main element it contained (a vertical arrow or a T-shaped element). The algorithm also recognized the end of the road sign with a very high probability. This is due to the characteristic colour palette (the only sign with three colours) as well as the horizontal, bright element. This is confirmed by the fact that in the experiment with a changed colour palette, the algorithm correctly identified the character only when there was a red fragment. Additionally, for the small size of the pattern (which corresponds to recognition when the signpost is still far from the driver), the sign was assessed incorrectly. This may be due to the blue background, which, due to its small size, may be confused with traffic warning signs. The algorithm identified it as movement left. The most problems with correct recognition concerned signs indicating movement in a certain direction. These signs are very similar, differing only in the arrangement

of the arrow. For this reason, when the size of the pattern was reduced, the neural network did not recognize it correctly, and even with a small size reduction, the probability of correct identification was significantly reduced. It was also noticed that the greatest problems occurred when assessing the sign of moving to the left. When examining changing light, the probability of recognition was much lower than in the case of moving to the right and straight. Additionally, the occlusion had a very negative impact on identification, which in the case of stops and the end of the road did not contribute to obtaining incorrect results. Summarizing the research performed, it can be concluded that the algorithm would work well in real conditions. The degree of recognition of individual road signs is sufficient for the algorithm to correctly navigate, for example, a maze. In the case of distant road signs, their recognition probability is so low that the mobile platform algorithm will not react too early. However, some restrictions should be introduced. Adopting constant lighting conditions (optimal overhead light) would be necessary to successfully complete the maze. Additionally, signs should not be covered to a large extent, because in the case of a traffic order, even the slightest covering drastically affects recognition.

In the future, the project may be developed in many different directions. The first one may be to expand the database of road signs recognized by the neural network. Currently, the database consists of five patterns, but it can be expanded to include more prohibition signs (such as speed limits), as well as examples of warning signs. Additionally, you can create an algorithm that would be able to recognize lines on the road using a camera and, as a result, stay within them. Another development option is to increase the number of patterns in the training and validation catalogs and subject them to additional modifications. As research has shown, the neural network recognizes some signs only on the basis of specific elements occurring within the surface of the signpost. Expanding the catalog with examples related to changed colours and different lighting would result in finding another element that would indicate what sign was identified (e.g. an inscription instead of a red background would be more important). Also, changing the scaling when training a neural network may bring different results, which could be the subject of research if the work continues. Another development option is to transfer the neural network to a new platform and further tests on a different microcomputer with greater computing power, e.g. on the Nvidia Jetson platform. This will allow you to learn about other hardware capabilities and give you a better idea of what technology is most suitable for building autonomous devices. Due to the algorithm being created in Python, the Linux-based Jetson microcomputer would be the most suitable option for further testing. Adding the option of manually controlling the robot via the operator panel is also being considered. This would be useful in case the platform reacts incorrectly or too late to a recognized pattern. Manual mode would be controlled using the so-called a pad on which it would be possible to switch between automatic mode (moving with the help of a neural network) and manual mode. Another option to consider is connecting a touch screen to Raspberry, which would create a graphical user interface to monitor and control the vehicle's movement.

REFERENCES

1. Siciliano, B., Khatib, O., Eds. Springer Handbook of Robotics; Springer, 2016.
2. Rybczak, M., Popowniak, N., Lazarowska, A. A survey of machine learning approaches for mobile robot control. Robotics 2024, 13.
3. Lee, M.F.R., Yusuf, S.H. Mobile robot navigation using deep reinforcement learning. Processes 2022, 10.
4. Bui, T.L., Tran, N.T. Navigation strategy for mobile robot based on computer vision and YOLOV5 network in the unknown environment. Applied Computer Science 2023, 19, 82–95.
5. Maturana, D., Scherer, S. 3D Convolutional Neural Networks for Landing Zone Detection from LiDAR. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2015, 3471–3478.
6. Nair, V., Hinton, G.E. 3D object recognition with deep belief nets. Advances in neural information processing systems 2009, 22.
7. Hinton, G.E., Osindero, S., Teh, Y.W. A fast learning algorithm for deep belief nets. Neural Computation 2006, 18, 1527–1554.
8. Shao, J., Kang, K., Loy, C.C., Wang, X. Deeply Learned Attributes for Crowded Scene Understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, 4657–4666.

9. Tai, L., Li, S., Liu, M. A Deep-Network Solution Towards Model-Less Obstacle Avoidance. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016, 2759–2764.
10. Li, H., Zhao, R., Wang, X. Highly efficient forward and backward propagation of convolutional neural networks for pixelwise classification. arXiv preprint arXiv:1412.4526 2014.
11. Deng, L. The MNIST Database of Handwritten Digit Images for Machine Learning Research. IEEE Signal Processing Magazine 2012, 29, 141–142.
12. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y. Reading digits in natural images with unsupervised feature learning. In: Proceedings of the NIPS workshop on deep learning and unsupervised feature learning. Granada, Spain, 2011, 7.
13. Coates, A., Ng, A., Lee, H. An Analysis of Single-Layer Networks in Unsupervised Feature Learning. In: Proceedings of the Fourteenth International Conference on Artificial Intelligence And Statistics. JMLR Workshop and Conference Proceedings, 2011, 215–223.
14. Tai, L., Li, S., Liu, M. Autonomous exploration of mobile robots through deep neural networks. International Journal of Advanced Robotic Systems 2017, 14.
15. Tai, L., Liu, M. Mobile robots exploration through cnn-based reinforcement learning. Robotics and Biomimetics 2016, 3, 24.
16. Sleaman, W.K., Hameed, A.A., Jamil, A. Monocular vision with deep neural networks for autonomous mobile robots navigation. Optik, 2023, 272, 170162.
17. Zhang, L., Zhang, Y., Li, Y. Path planning for indoor Mobile robot based on deep learning. Optik 2020, 219, 165096.
18. Singh, R., Ren, J., Lin, X. A review of deep reinforcement learning algorithms for mobile robot path planning. Vehicles, 2023, 494(5), 1423–1451.
19. Chollet, F. Deep learning with Python; Simon and Schuster, 2021.