Research article
DOI: https://doi.org/10.18721/JCSTCS.15403
UDC 004.032

TECHNIQUE FOR AUTOMATING CHARGING OF AN ELECTRIC VEHICLE BASED ON A RASPBERRY PI CONTROLLER USING NEURAL NETWORKS

N.A. Vlasenko[1], *A.I. Dusaeva*[2], *I.V. Nikiforov*[3] ✉, *D.S. Prelovskii*[4]

[1,2,3,4] Peter the Great St. Petersburg Polytechnic University, St. Petersburg, Russian Federation

✉ igor.nikiforovv@gmail.com

Abstract. The expansion of Russian market of electric and autonomous vehicles leads to an increase in demand for automation of contactless charging (without driver participation). The article proposes a method of contactless charging of electric vehicles, which involves automatically determining the type of car charging connector, selecting the appropriate charger and connecting it to the charging connector of an electric vehicle through the use of a robot manipulator. A feature of the technique is the determination of the type and coordinates of the location of the charging connector of the car by reading images obtained from the camera of a gas station in real time and processing them with a convolutional neural network model. A study was conducted, and a function was selected that allows optimally solving the problems of classification of charging connectors, which ensures maximum accuracy of the result. The volume of the training sample for the neural network was used in the amount of 10,000 images from a synthetic data set, which was created on the basis of three types of the most popular three-dimensional models of charging connectors on various backgrounds. The proposed technique is implemented in a prototype of a software and hardware control complex for a manipulative robot based on a Raspberry Pi controller.

Keywords: electric vehicle, computer vision, convolutional neural network, robotic arm, charging

Citation: Vlasenko N.A., Dusaeva A.I., Nikiforov I.V., Prelovskii D.S. Technique for automating charging of an electric vehicle based on a Raspberry Pi controller using neural networks. Computing, Telecommunications and Control, 2022, Vol. 15, No. 4, Pp. 37–50. DOI: 10.18721/JCSTCS.15403

Научная статья
DOI: https://doi.org/10.18721/JCSTCS.15403
УДК 004.032

МЕТОДИКА АВТОМАТИЗАЦИИ ЗАРЯДКИ ЭЛЕКТРОМОБИЛЯ НА БАЗЕ КОНТРОЛЛЕРА RASPBERRY PI С ИСПОЛЬЗОВАНИЕМ НЕЙРОННЫХ СЕТЕЙ

Н.А. Власенко[1], *А.И. Дусаева*[2],
И.В. Никифоров[3] ✉ , *Д.С. Преловский*[4]

[1,2,3,4] Санкт-Петербургский политехнический университет Петра Великого,
Санкт-Петербург, Российская Федерация

✉ igor.nikiforovv@gmail.com

Аннотация. Расширение российского рынка электрических и автономных транспортных средств ведет к увеличению спроса на автоматизацию бесконтактной зарядки (без участия водителя). В статье предложена методика бесконтактной зарядки электромобилей, которая предполагает автоматическое определение типа зарядного коннектора автомобиля, выбор соответствующего зарядного устройства и его подключение в зарядный коннектор электромобиля благодаря использованию робота-манипулятора. Особенностью методики является определение типа и координат расположения зарядного коннектора автомобиля за счет считывания изображений, получаемых с камеры автозаправочной станции в режиме реального времени и обработанных при помощи модели свёрточной нейронной сети. Проведено исследование и выбрана функция, позволяющая оптимально решать задачи классификации зарядных коннекторов, обеспечивающая максимальную точность результата. Объём обучающей выборки для нейронной сети использован в размере 10 000 изображений из синтетического набора данных, созданного на основе трёх типов наиболее популярных трёхмерных моделей зарядных коннекторов на различных фонах, приближенных к реальным условиям использования роботов, обслуживающих зарядные станции. Предложенная методика реализована в прототипе программно-аппаратного комплекса управления манипуляционным роботом на основе контроллера Raspberry Pi.

Ключевые слова: автоматизация, компьютерное зрение, свёрточная нейронная сеть, робот-манипулятор, зарядное устройство

Для цитирования: Vlasenko N.A., Dusaeva A.I., Nikiforov I.V., Prelovskii D.S. Technique for automating charging of an electric vehicle based on a Raspberry Pi controller using neural networks // Computing, Telecommunications and Control. 2022. T. 15, № 4. C. 37–50. DOI: 10.18721/JCSTCS.15403

Introduction

To date, one of the most acute environmental problems in developed countries is the pollution of the atmosphere by exhaust gases from motor vehicles. The total level of environmental pollution by exhaust gases from total emissions of harmful substances in Europe is 72.9 % [1]. One of the possible ways to solve this problem is the widespread use of electric cars since they are more environmentally friendly than cars with an internal combustion engine and contribute to reducing the consumption of fossil fuels. This trend leads to the need to build a modern infrastructure for the vehicles maintenance [2].

Often, the construction of infrastructure for vehicles is carried out using robotic tools and systems, which makes it possible to automate processes and minimize human involvement [3]. One of the parts of such infrastructure is a contactless charging system with the use of robotic manipulators, which should provide a high-quality, reliable, and fast connection of the charging station with the charging socket of the car. Accordingly, with the increase of electric vehicles in the Russian market [4, 5], the need for automatic contactless charging systems using robotic manipulators also increases, which is why *the relevance* of the chosen topic is determined.

In addition, the situation in Russia differs from other electric vehicle markets [4], since there are cars in our country that meet both European and Japanese standards. Guided by these considerations, the paper considers the most popular charging sockets of electric vehicles in the territory of the Russian Federation [5].

Research

Since the issue of the introduction and creation of automatic charging stations is relevant almost for the whole world, many research groups are developing automated charging systems for electric vehicles.

For example, M. Bell, J. Duro, T. Flynt et al. in their article [6] consider a robot manipulator for refuelling electric cars. Their research is aimed at developing a navigation system for the robot using the LiDar system, as well as a system of movement towards the charging connector option.

Researcher E.H.C Harik in the article [7] considers the possibility of introducing autonomous electric tractors into agriculture and proposes an autonomous charging station system that uses visual navigation and detection to connect a power cable to an outlet. In the article, E.H.C Harik uses only one type of charging connectors, which is a generalized model and is not used in real conditions.

A group of authors, B. Walzel, C. Sturm, J. Fabian, and M. Hirz, in their work [8] give a brief overview of both existing charging systems from large companies such as Tesla, Volkswagen, and various research projects, for example, a charging system from the Technical University of Dortmund. In addition, the article highlights the main problems associated with the development of such an automatic system: the location of the car in the parking space at the time of connecting the charger, various types of connectors. They also proposed the concept of an automated charging system.

The works considered in the study provide a solution to their task, but they also have several disadvantages associated with a low degree of automation, low connection quality (below 70 %), which leaves room for improvement of the car charging automation. Some solutions describe a generalized model and are not applied in real conditions.

The article suggests a different approach for charging electric vehicles with a robot manipulator. Its main task is to automatically determine the type of charging connector of the car and build the optimal trajectory from the charging station to the socket of the car using a robot manipulator. Also, a distinctive feature is the creation, training and testing of a neural network for classifying various charging sockets on a physical model of an automated charging system with the function of visual navigation of a robot manipulator.

The aim of the work is to develop a technique that provides contactless charging of electric vehicles by means of implementing a software and hardware complex based on the use of neural networks and a Raspberry Pi controller.

The methodology consists of three main steps presented below.

Step 1. Automatic detection of the type of car charging connector by reading the image from the video camera and its recognition by means of a neural network.

Step 2. Determining the coordinates of the charging socket and calculating the optimal trajectory for moving the selected charger to the socket of the electric vehicle.

Step 3. Connecting the charger to the charging connector of the electric vehicle by using a robot manipulator.

A feature of the technique is the determination of the type and coordinates of the location of the charging connector of the car by reading images obtained from the camera of a gas station in real time and processing them using a convolutional neural network model. The technique proposed in the paper is implemented in a software tool based on the Raspberry Pi 4 controller.

Finite automaton behaviour of the robot manipulator

The technique proposed in the paper was brought to implementation in a hardware and software complex, the basis of which is the behaviour described by a finite automaton. Fig. 1 shows a model of its behaviour in the form of a Mile machine [9], consisting of a finite number of states. Based on certain input data, there is a transition from one state to another and data transfer.

In a finite state machine, the following basic states can be distinguished:

– "take photo" – the state in which the camera located at the gas station takes a picture of the charging socket of an electric vehicle;

– "recognize socket" – the state in which the type of car charger is recognized by the image created by the camera;

– "change charger" – the state in which the robot manipulator changes the active charger to a suitable one for the current car;

– "calculate coordinates" and "calculate angle" – states in which, based on photos of the charging socket of an electric vehicle, the coordinates of its location and angle are calculated, respectively;

– "plug" – connection of the charger by the robot manipulator to the charging socket of the electric vehicle;

– "charge" – a condition that characterizes the process of direct physical charging of an electric vehicle;

– "idle" – the state of the end of the charging process of the electric vehicle.

The most interesting states that will be considered in the paper are: "recognize socket", "calculate coordinates" and "calculate angle".

Automation of determining the type of car charging connector

In the "recognize socket" state, the type of car charging socket is determined based on image recognition [10].

There are a large number of charging sockets for electric cars. They differ in charging speed. According to statistics [4] the most popular cars for 2020–2021 are Nissan Leaf, Porsche Taycan, Audi e-tron, Tesla Model 3, Mitsubishi i-MiEV and others. These models use the following types of charging connectors:

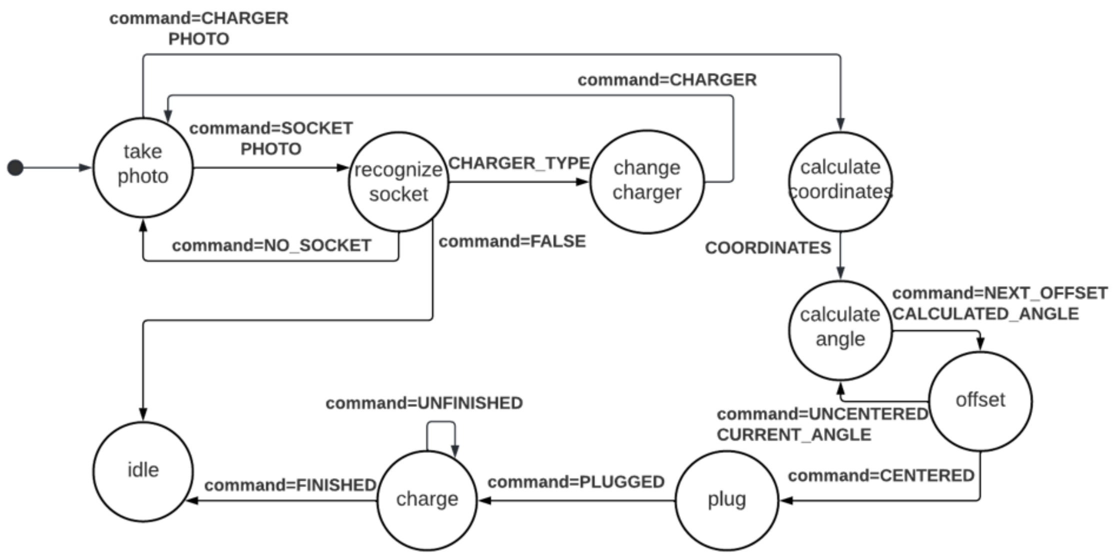

Fig. 1. Behavior model of robotic arm system

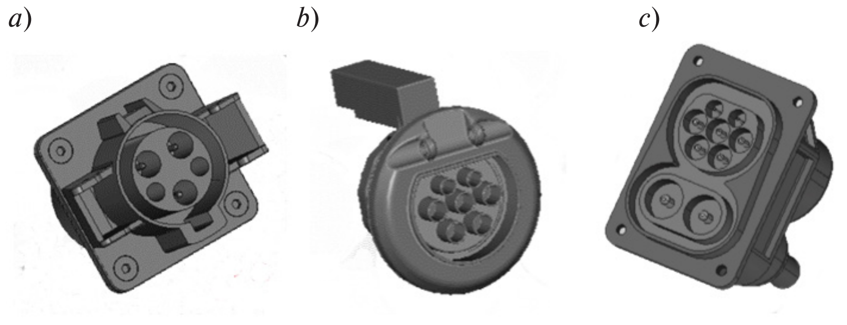

Fig. 2. Models of charging sockets in use

– standard connector of the first type (SAE J1772, J-PLUG, J-plug), shown in Fig. 2*a* is a North American standard, common among American and Japanese vehicles. It is single-phase and includes 5 contacts;

– connector of the second type (Mennekes), shown in Fig. 2*b* is a European standard with 7 contacts, at the moment it is the most common among electric vehicles in Europe and China, can be single-phase or three-phase;

– combined type (CCS Combo 2), shown in Fig. 2*c* is an improved version of the second type with 9 contacts. The second type of the cable can also be used for this connector.

Thus, the work selected the above tips for recognition automation as the most common in Russia. Their example shows the applicability and effectiveness of the methodology. At the same time, the applicability of the technique is not limited to the considered types of electric vehicles and their charging sockets. It is universal and can be used, among other things, for other electric vehicles. To do this, it will be necessary to expand the data set for training the neural network with images with additional photos of charging connectors and retrain the neural network.

Figure 3 shows the architecture of a neural network for determining the type of electric car charger, which consists of four convolution layers [11] (Conv2D with the number of neurons 8, 12, 16 and 18, respectively), performing the operation of multiplying the image matrix (28×28) by the convolution core matrix (3×3). Convolution layers alternate with four layers of subdiscretization (MaxPooling2D), which are necessary to compress the size of the extracted image feature map [12]. The work uses a convolutional neural network, because this architecture has proven itself well in image recognition tasks, and the number of layers is justified by the highest efficiency and quality of recognition in comparison with other numbers of layers.

Consider the layers of a neural network. The thinning layer (Dropout) is used to solve the problem of retraining the network, converting to a one-dimensional vector (Flatten). Next is a fully connected layer with a linear activation function (Dense 128 + ReLU), a thinning layer (Dropout) and the last fully connected layer with a sigmoid activation function (Dense 4 + sigmoid), which determines independent probabilities for determining the coordinates of the object in the frame. At the output of the neural network, we get a two-dimensional array array [samples] [4], where samples is the number of images submitted to the input of the neural network, and 4 is the number of coordinates received.

To classify the connectors, the Softmax activation function was used, which determines the dependent probabilities of distribution over three possible classes for classifying objects. At the output of the neural network, we get a two-dimensional array array [samples] [3], where samples is the number of images submitted to the input of the neural network, and 3 is the probability distribution by class.

To train a neural network that implements the recognition mechanism of the charging socket of an electric vehicle, a data set [13] from CAD models (Computer-Aided Design) was prepared.

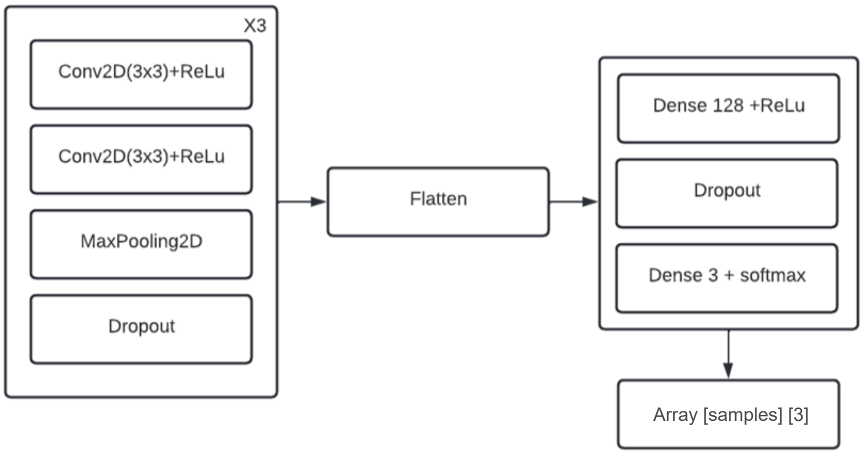

Fig. 3. Architecture of classification charging sockets neural network

The approach used in the work makes it possible to obtain many images of the required objects in various positions, lighting and environment, and automatically mark up the data.

10,000 images were obtained in the work. In the Blender program, a program was created using the bpy library, where a random angle and position of the part on the scene are calculated within acceptable limits, since it should be facing the camera and should not go beyond the stage. After that, the background is applied, the image is saved with the settings set.

Choosing an optimizer for classification of charging connectors

The study and selection of the best optimizer for the task of classifying charging connectors was carried out. These optimizers are used in most machine-learning tasks.

Stochastic gradient descent (SGD) [14] is one of the simplest methods of minimizing the loss function, on the basis of which other optimizers are built:

$$\theta_{t+1} \leftarrow \theta_t - \eta \nabla_\theta J(\theta_t),$$

where θ – network parameters (weights), η – learning rate, $J(\theta_t)$ – loss function.

In the connector classification problem, this optimizer is at $\eta = 0.001$ (this value is optimal, because with an increase in this hyperparameter, the final classification accuracy will decrease due to a large number of local minima omissions, convergence may not be achieved, with a decrease, the convergence rate will decrease significantly, especially when entering the plateau zone), it has the lowest convergence, the classification accuracy on the test set was 35.64 %. This problem arises due to the impossibility of adaptive changes in the learning rate and, accordingly, the step length.

The Root Mean Square Propagation Algorithm (RMS Pro) is used for batch optimization, in which data is processed in blocks. Its main idea is to preserve the moving average of the squares of the gradients for each weight. The learning rate is adapted by dividing it by an exponentially decreasing average of the squares of the gradients:

$$v_t = \beta v_{t-1} + (1-\beta)\left(\nabla_\theta J(\theta_t)\right)^2,$$

$$\theta_{t+1} \leftarrow \theta_t - \frac{\eta}{\sqrt{v_t} + \epsilon} \nabla_\theta J(\theta_t),$$

where v_t – moving average, β – accumulation coefficient, ϵ – smoothing parameter.

In this case, the loss function for the validation set and the classification accuracy are unstable, the accuracy on the test set was 81.54 %.

Adaptive Moment Estimation (Adam) is an advanced RMS Pro algorithm using momentum and one of the most effective and most frequently used optimizers in machine learning [15]. It calculates the adaptive learning rate for each parameter with a slight change in weights for characteristic features. To ensure this, it is necessary to preserve the exponentially decreasing average of the squares of the gradients in previous iterations and the exponentially decreasing average of the past gradients. Instead of using the entire dataset to calculate the actual gradient, this optimization algorithm uses a randomly selected subset of the data to create a stochastic approximation:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla_\theta J(\theta_t)$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2)\left(\nabla_\theta J(\theta_t)\right)^2$$

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1-\beta_1^t}$$

$$\theta_{t+1} \leftarrow \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon}\hat{m}_t = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon}\left(\beta_1 \hat{m}_{t-1} + \frac{1-\beta_1}{1-\beta_1^t}\frac{\Delta loss}{\Delta\theta_t}\right),$$

where $\eta = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$; $0 \le \beta_1 < 1{,}0 \le \beta_2 < 1$ – accumulation coefficient; v_t – average non-centered variance; m_t – exponential moving average.

At the end of the training, the classification accuracy, and the loss function for the training set and for the validation set begin to diverge, which entails retraining, so the accuracy with this optimizer was 84.15 %.

The algorithm of adaptive estimation of moments with infinite norms (Adamex) uses the inertial moment of gradient distribution:

$$v_t = \beta_2^k v_{t-1} + \left(1-\beta_2^k\right)\left|\nabla_\theta J(\theta_t)\right|^k$$

$$u_t = \sqrt[\infty]{v_t} = \max\left(\beta_2^{t-1}\left|\nabla_\theta J(\theta_1)\right|,\ \beta_2^{t-2}\left|\nabla_\theta J(\theta_2)\right|,\ \ldots,\ \beta_2\left|\nabla_\theta J(\theta_{t-1})\right|,\ \left|\nabla_\theta J(\theta_t)\right|\right)$$

$$\theta_{t+1} \leftarrow \theta_t - \frac{\eta}{u_t}\hat{m}_t$$

$$\eta = 0.002, \quad \beta_1 = 0.9, \quad \beta_2 = 0.999.$$

At the beginning of the training, there is a big difference and instability of the results of functions on the test and validation datasets, however, by the hundredth epoch, the results gradually converge; nevertheless, the recognition accuracy of the test set was 77.23 %.

Adaptive estimation of moments with Nesterov acceleration (NAdam) is an improved Adam algorithm. In this optimizer the next position of the exponential running average $(\hat{m}_t)$ is not predicted in advance, and the gradient update formula compared to the Adam algorithm changes accordingly as follows:

$$\theta_{t+1} \leftarrow \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon}\hat{m}_t = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon}\left(\beta_1 \hat{m}_t + \frac{1-\beta_1}{1-\beta_1^t}\frac{loss}{\theta_t}\right),$$

$$\eta = 0.002, \quad \beta_1 = 0.9, \quad \beta_2 = 0.999, \quad \epsilon = 10^{-7}.$$

Thanks to this algorithm, significant success was achieved: instability of functions was eliminated, the results of their work converged, the recognition accuracy of the test set was 91.59 %.

Table 1 shows the final comparison of the accuracy of the classification of optimizers for each charging connector. The rows of the table are the tests performed to determine accuracy by class (Type 1, Type 2, Type 3), the total accuracy on training data (Train_accuracy) and on test data (Test_accuracy), and the columns are the optimization functions under consideration. The cells show the results of calculating the recognition accuracy on the test data set.

Table 1

The resulting classification accuracy for different optimizers

	Adam	NAdam	RMS Pro	Adamax	SGD
Type 1	0.7554	0.8785	0.6677	0.7862	0.0615
Type 2	0.9031	0.8985	0.8062	0.8367	0.9978
Type 3	0.8661	0.9708	0.9723	0.7228	0.0077
Train_accuracy	0.8583	0.8938	0.8982	0.7632	0.5408
Test_accuracy	0.8415	0.9159	0.8154	0.7723	0.3564

Automation of determining the trajectory of the robot manipulator from the charger to the charging connector of the car

In the "calculate coordinates" and "calculate angle" states (see Fig. 1), the trajectory of the robot manipulator [16] with the selected active charger to the charging socket of the electric vehicle is determined by using images from the camera.

One of the main physical characteristics of the camera, which takes pictures of the charging connector of the car at the gas station, are its viewing angles, which are calculated by the formulas:

$$\alpha = 2\arctan\left(\frac{h}{2f}\right),$$

$$\beta = 2\arctan\left(\frac{v}{2f}\right),$$

where $\alpha,\ \beta \in (0;360)$, rad; h, v – sensor dimensions, mm; f – focal length, mm.

To create an algorithm for the movement of the robot manipulator from the charger to the charging connector of the car, it is necessary to enter the definitions listed below.

Definition 1. $F_x,\ F_y \in Q \middle| 0 < F_x < 0.5\alpha \wedge 0 < F_y < 0.5\beta$ we will call the final shifts, the relative values by which the object in the image is shifted relative to the optical axis of the camera. Let $x,\ y \in (0;1)$, then:

$$F_x^{k+1} = \left(x_0^k + \frac{x_1^k - x_0^k}{2} \right) * \alpha, \quad F_y^{k+1} = \left(y_1^k + \frac{y_1^k - y_0^k}{2} \right) * \beta,$$

where α, β – camera viewing angles; x_0, x_1 – diagonal coordinates along the X-axis, the fraction relative to the total width of the image; y_0, y_1 – diagonal coordinates along the Y-axis, the fraction relative to the total height of the image.

Definition 2. γ_x^{k+1}, $\gamma_y^{k+1} \in Q$ we will call the rotation angles (rad), the values by which the servomotor will rotate at each new iteration, based on its current position and the coordinates of the object in the resulting image:

$$\gamma_x^{k+1} = \left(\gamma_x^k \pm F_x^k \right), \quad \gamma_y^{k+1} = \left(\gamma_y^k \pm F_y^k \right),$$

where F_x, F_y – final shifts.

Thus, we get:

$$\gamma^{k+1} = \left(\gamma^k \pm \left(x_0^k + \frac{x_1^k - x_0^k}{2} \right) * \alpha \right).$$

The algorithm for calculating the trajectory of movement can be represented as a sequence of steps.

Step 1. Take the initial values h, v – sensor dimensions, f – focal length, x – coordinate vector.

Step 2. Accept $\alpha := 2\arctan\left(\frac{h}{2f}\right)$, $\beta := 2\arctan\left(\frac{v}{2f}\right)$, $k = \overline{0,n}$.

Step 3. Accept $F_x^{k+1} := \left(x_0^k + \frac{x_1^k - x_0^k}{2} \right) * \alpha$.

Step 4. If $F_x^{k+1} < 0.5$, accept $\gamma_x^{k+1} := \left(\gamma_x^k - F_x^k \right)$, if $F_x^{k+1} > 0.5$, accept $\gamma_x^{k+1} := \left(\gamma_x^k + F_x^k \right)$ otherwise $\gamma_x^{k+1} := \left(\gamma_x^k \right)$.

Step 5. Return γ_x^{k+1}, accept $k = k + 1$.

In this case, the local minimum of the function is achieved when the optical axis of the camera and the calculated center of the recognized object coincide, since in the interval between iterations there is a possibility of changing the position of the tracked object or changing the selection results in case the object does not fully enter the frame during the first iteration. The algorithm is carried out before the condition of changing axes.

Tools and technologies for implementing the methodology

The set of tools and technologies for the implementation of each individual module of the software product includes:

– to write the motor control module for the Arduino Uno board, an ATmega328P-based microcontroller, the Arduino IDE 1.8.19 development environment and the C++ programming language with the Wiring framework and an extensive set of libraries were used;

– the motion vector calculation module for the Raspberry Pi 4 B board (Cortex-A72 [17]) was developed using Python 3.6;

– training and testing of neural networks was conducted in the Jupiter Notebook 6.4.8 interactive development environment using open libraries for Tensorflow 2.2.0 and Keras 2.2.1 machine learning and Python 3.6 programming language;

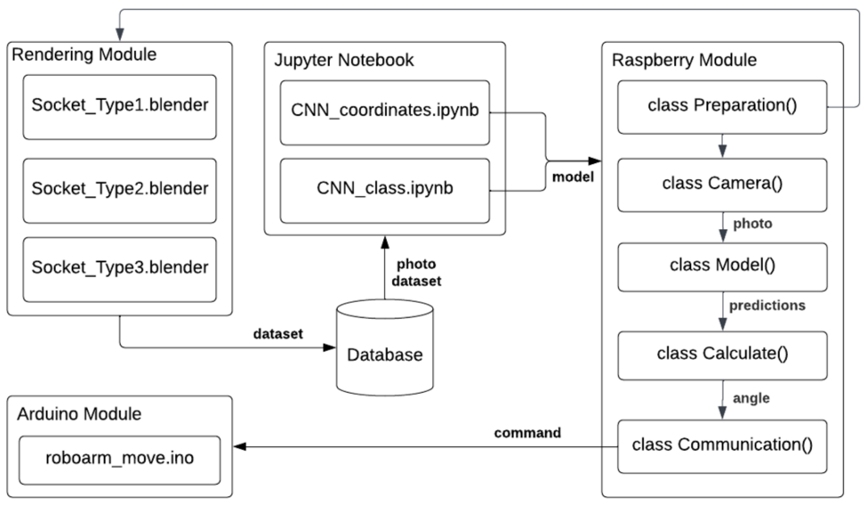

Fig. 4. Software module diagram that implements the proposed technique

– writing a program for rendering images and creating a data set for processing by neural networks was carried out on the Ubuntu 21.04 operating system in open-source 3D modelling software Blender 3.0.1 in Python.

In accordance with the stages of the methodology proposed in the work, Fig. 4 shows a diagram of the modules of the software tool. It consists of 5 main modules.

The "Rendering" module is necessary to compile a data set for training a neural network: the components "Socket_Type1", "Socket_Type2", "Socket_Type3" start the process of creating images with three types of charging connectors on different backgrounds.

The module "Jupyter Notebook" trains neural networks to determine the coordinates and type of the charging connector of the car: the component "CNN_coordinates" implements the definition of the coordinates of the required object in the frame; the component "CNN_class" implements the definition of the connector class.

The Raspberry module is the main one, it starts the initialization of the camera and the process of creating photos in real time loading neural network models from the Jupyter Notebook module, basic calculations and data transfer to the Adruino module: the "Preparation" component prepares directories, checks the existence of the necessary directories, creates them if necessary; the "Camera" component starts the process of creating a photo; the "Model" component processes the received photo using a neural network model; the "Calculate" component performs basic calculations with the received data; the "Communication" component establishes a connection to the Arduino Uno board, sends a command to the ttxAXMx port.

The "Arduino module" controls the motors, changing the position of the manipulator in space.

The "Dataset" module stores data sets for training, validation, and testing of neural networks.

Testing of software and hardware

The prototype of the gas station robot manipulator created in the work, shown in Fig. 5, meets the following requirements:

– recognizes the type of charging socket of the vehicle with an accuracy of at least 80 % before instructing the robot arm to grab the charger and start moving it to the charging socket of the vehicle (average recognition accuracy of 91.59 %);

– determines the coordinates of the car's charging socket and selects the optimal trajectory;

– starts moving when the coordinates of the object are received;

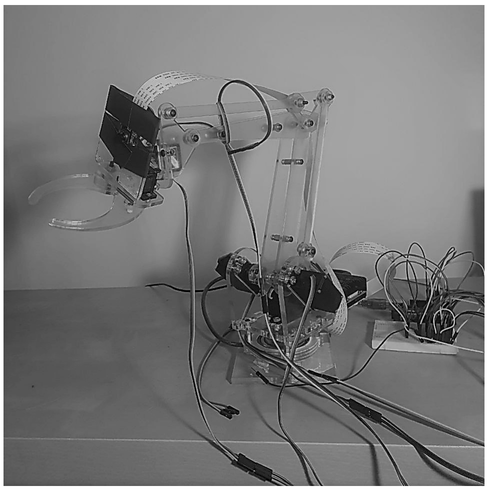

Fig. 5. Prototype of robotic arm

– at each iteration, searches for the center of the selected object for subsequent calculations of the rotation angles of the motors of which it consists.

With the number of images in the training dataset equal to 1500 images and an acceptable error of $p = 0.01$ and $p = 0.05$ 45 % and 69 % of images are recognized correctly, respectively, while with an error of $p = 0.2$ % – 96 %. When increasing the size of the input dataset to 10,000 images and with an acceptable error of $p = 0.01$, 96.41 % of the images are recognized correctly.

Table 2 shows a comparison of the hardware boards that can be used in robotic arm for neural networks processing. Raspberry Pi 4 B occupies a leading position in image processing for machine learning [18] and image compression [19]. It provides fast enough speed of data processing and image recognition in order to produce output in a reasonable amount of time. So Raspberry Pi 4 B was used to build a robotic arm model.

At the same time, Arduino Uno controller was chosen for conversion of calculated coordinates into rotational output (to servomotors), since it has good reputation in the similar tasks solutions [20].

Fig. 6 shows an example of recognition and allocation of charging connectors, as well as accuracy for each type.

Conclusion

The paper proposes a method of contactless charging of electric vehicles, which involves automatically determining the type of car charging connector, selecting the appropriate charger, and connecting it to the charging connector of an electric vehicle using a robot manipulator.

A prototype robot manipulator was developed that implements the proposed technique and has the following properties:

– determines the type of vehicle charging connector with an accuracy of 91.59 % due to the use of a convolutional neural network. The data collection method used to train a neural network based on 3D models of charging sockets allows you to create data sets in different areas (at gas stations, in urban areas, parking lots, on federal highways, etc.) with different natural and weather conditions that affect lighting and visibility (rain, fog, blizzard, night time) at a lower cost;

– determines the coordinates for the movement of the robot manipulator with an accuracy of 96.41 % due to the use of the optimal loss minimization function.

Based on wide and deep comprehensive analysis Raspberry Pi 4 B board was selected as the most powerful and fast enough board to handle and process input data and provide relevant output in a reasonable amount of time.

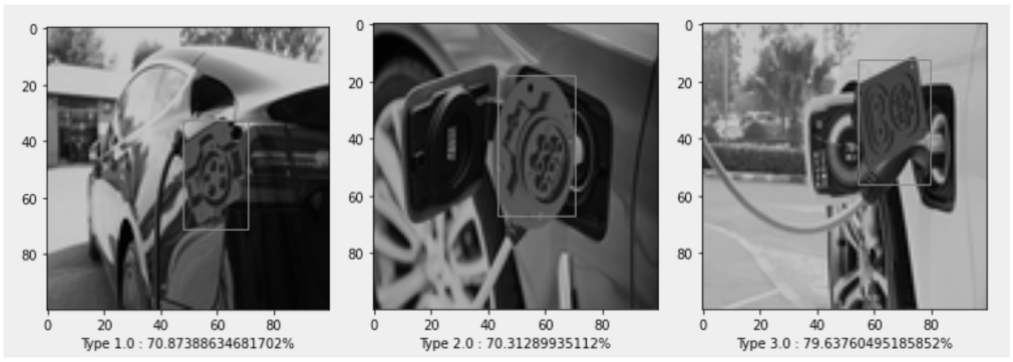
Type 1.0 : 70.87388634681702%
Type 2.0 : 70.31289935112%
Type 3.0 : 79.63760495185852%

Fig. 6. Demonstration of neural networks output

Table 2

Comparison of hardware boards for data processing in robotic arm

Board name	CPU	GPU	RAM (GB)	Memory bandwidth (MBps)		Machine Learning (images/sec)	Image Compression (Mpixels/sec)
				Read	Write		
Raspberry Pi 4 B	1.5GHz Quad core Cortex-A72 (ARM v8) 64-bit	VideoCore VI OpenGL ES 3.x	8	4129	4427	7.30	33.2
Orange PI 4	Rockchip RK3399(Cortex-A72+Cortex-A53) 1.8 GHz	Mali-T860	4	2972	3106	3.80	13.4
LeMaker Banana Pi	ARM Allwinner @ 960 MHz	ARM Mali-400 MP2 GPU dual-core	1	805	1358	0.30	2.54
BeagleBone AI-64	Texas Instruments Jacinto TDA4VM 2x ARM Cortex-A72 2 GHz	PowerVR® Rogue™ 8XE GE8430 3D	4	3679	3843	5.59	17.0

The developed prototype of the software and hardware solution allows reducing human involvement in the process of automating the charging of vehicles.

REFERENCES

1. **Gómez-Vilchez J.J., Julea A., Peduzzi E., Pisoni E., Krause J., Siskos P., Thiel C.** Modelling the impacts of EU countries' electric car deployment plans on atmospheric emissions and concentrations. *European Transport Research Review*, 2019, vol. 11, No. 40, 1 p. DOI: 10.1186/s12544-019-0377-1

2. **Zhang H., Sheppard C., Lipman T., Zeng T., Moura S.** Charging infrastructure demands of shared-use autonomous electric vehicles in urban areas. *Transportation Research Part D Transport and Environment*, 2020, No.78, 102210 p. DOI: 10.1016/j.trd.2019.102210

3. **Sanatov D.V. et al.** *Electric vehicle market and charging infrastructure development perspectives in Russia: Expertise and analysis report*. POLYTECH-PRESS, 2021, 44 p.

4. **Labudin A.V., Eshenkova N.S., Burdukova A.O.** Current trends and prospects for the development of the electric vehicle market in Russia. *Economics and Management of the National Economy* (*Saint Petersburg*), 2021, No. 14 (16), 143 p.

5. **Komarova M.V.** Analiz rynka elektromobiley v Rossii. *Innovatsii. Nauka. Obrazovaniye*, 2020, No. 21, Pp. 276–281. (rus)

6. **Bell M., Duro J., Flynt T., Hameed I., Lang G., Park F.** Autonomous electric vehicle charging system. *Systems and Information Engineering Design Symposium* (*SIEDS*), 2019, Pp. 1–6. DOI: 10.1109/SIEDS.2019.8735620

7. **Harik E.H.C.** Design and implementation of an autonomous charging station for agricultural electrical vehicles. *Applied Sciences*, 2021, vol. 11, No. 13: 6168. DOI: 10.3390/app11136168

8. **Walzel B., Sturm C., Fabian J., Hirz M.** Automated robot-based charging system for electric vehicles. *International Stuttgarter Symposium*, 2016. DOI: 10.1007/978-3-658-13255-2_70

9. **Salle D., Herrero C.H., Outon J., Esnaola U., Lopez-de-Ipiña K.** State machine based architecture to increase flexibility of dual-arm robot programming. *Bioinspired Computation in Artificial Systems*, 2015, No. 2, 98 p. DOI: 10.1007/978-3-319-18833-1_11

10. **Al-Saffar A.M., Tao H., Talab M.A.** Review of deep convolution neural network in image classification. *International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications* (*ICRAMET*), 2017, Pp. 26-31. DOI: 10.1109/ICRAMET.2017.8253139

11. **Cuiping S., Tan C., Wang T., Wang L.** A waste classification method based on a multilayer hybrid convolution neural network. *Applied Sciences*, 2021, vol. 11, No. 18: 8572. DOI: 10.3390/app11188572

12. **Parasich A.V., Parasich V.A., Parasich I.V.** Formirovaniye obuchayushchey vyborki v zadachakh mashinnogo obucheniya. *Informatsionno-Upravlyayushchiye Sistemy*, 2021, No. 4, Pp. 61–70. (rus). DOI: 10.31799/1684-8853-2021-4-61-70

13. **Voinov N., Drobintsev P., Kotlyarov V., Nikiforov I.** Distributed OAIS-Based digital preservation system with HDFS technology. *Proceedings of the 20th Conference of Open Innovations Association FRUCT*, St. Petersburg, 2017, Pp. 491–497. DOI: 10.23919/FRUCT.2017.8071353

14. **Chernorutskiy I., Drobintsev P., Kotlyarov V., Voinov N.** A new approach to generation and analysis of gradient methods based on relaxation function. *2017 UKSim-AMSS 19th International Conference on Computer Modelling & Simulation* (*UKSim*), Cambridge, UK, 2017, Pp. 83–88. DOI: 10.1109/UKSim.2017.14

15. **Postalcioglu S.** Performance analysis of different optimizers for deep learning based image recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 2019, vol. 34, No. 2. DOI: 10.1142/S0218001420510039

16. **Rostova E.N., Rostov N.V., Yan Z.** Neural network compensation of dynamic errors in a position control system of a robot manipulator. *Computing, Telecommunications and Control*, 2020, vol. 13, No. 1, Pp. 53–64. DOI: 10.18721/JCSTCS.13105

17. **Pankov P.A., Nikiforov I.V., Drobintsev D.F.** Hardware and software data processing system for research and scientific purposes based on Raspberry Pi 3 microcomputer. *Proceedings of the Institute for System Programming of the RAS*, 2020, vol. 32, No. 3, Pp. 57–70. DOI: 10.15514/ISPRAS-2020-32(3)-5

18. **Ozkan Z., Bayhan E., Namdar M., Basgumus A.** Object detection and recognition of unmanned aerial vehicles using Raspberry Pi platform. *5th International Symposium on Multidisciplinary Studies and Innovative Technologies* (*ISMSIT*), Ankara, Turkey, 2021, Pp. 467–472. DOI: 10.1109/ISMSIT52890.2021.9604698

19. **Sahitya S., Lokesha H., Sudha L.K.** Real time application of Raspberry Pi in compression of images. *IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology* (*RTEICT*), Bangalore, India, 2016, Pp. 1047–1050, DOI: 10.1109/RTEICT.2016.7807990

20. **Oza V., Mehta P.** Arduino robotic hand: Survey paper. *International Conference on Smart City and Emerging Technology* (*ICSCET*), Mumbai, India, 2018, Pp. 1–5, DOI: 10.1109/ICSCET.2018.8537312

INFORMATION ABOUT AUTHORS / СВЕДЕНИЯ ОБ АВТОРАХ

Власенко Наталия Андреевна
Nataliia A. Vlasenko
E-mail: nt.vlasenko@yandex.ru

Дусаева Анеля Ильясовна
Anelya I. Dusaeva
E-mail: an.dusaeva@gmail.com

Никифоров Игорь Валерьевич
Igor V. Nikiforov
E-mail: igor.nikiforovv@gmail.com

Преловский Дмитрий Сергеевич
Dmitrii S. Prelovskii
E-mail: dimaprelovski@gmail.com

Поступила: 29.11.2022; Одобрена: 17.12.2022; Принята: 12.01.2023.
Submitted: 29.11.2022; Approved: 17.12.2022; Accepted: 12.01.2023.