2022 Smart Information Systems and Technologies (SIST)
28-30 April, 2022, Nur-Sultan, Kazakhstan

An Intelligent Automatic Number Plate Recognition System Based on Computer Vision and Edge Computing

Y. Seitbattalov
Department of Computer and Software Engineering
L. N. Gumilyov Eurasian National University
Nur-Sultan, Kazakhstan
sbtl.jeks@gmail.com

Hüseyin Canbolat
Department of Electrical and Electronics Engineering
Ankara Yıldırım Beyazıt University
Ankara, Turkey
huseyin.canbolat@gmail.com

Zhanar S. Moldabayeva
Department of Computer and Software Engineering
L. N. Gumilyov Eurasian National University
Nur-Sultan, Kazakhstan
zh.moldabaeva@gmail.com

Abzal E. Kyzyrkanov
Department of Computer and Software Engineering
L. N. Gumilyov Eurasian National University
Nur-Sultan, Kazakhstan
abzzall@gmail.com

2022 International Conference on Smart Information Systems and Technologies (SIST) | 978-1-6654-6790-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/SIST54437.2022.9945787

***Abstract*—The current traffic monitoring system accumulates a large amount of data and apply algorithms to process the received photo and video images through cameras.**

These data are used to regulate traffic flows, recognize number plate and record traffic collisions. The collected data are traditionally sent to data centers for processing, analyzing, and then sent to the operator. Thus, a large amount of incoming data to the data center and to the traffic monitoring center reduces the network bandwidth and has a negative impact on the speed of decision-making by the operator.

To solve this problem, we proposed to use the edge computing paradigm by placing edge nodes in the near vicinity of the cameras and conducting data preprocessing on edge devices.

This paper considers the Raspberry Pi as an edge device, as well as computer vision algorithms for pre-processing data at the edge device and the process of transferring data to the data center with the following prompt decision-making by the operator.

As a result of usage the edge computing paradigm helps to optimize network traffic and offload the computing power of the data center.

Keywords—internet of things; edge computing; image processing; video analytic; raspberry pi; network technology

I. INTRODUCTION

As the number of Internet of Things (IoT) devices in the world has increased, a huge amount of data has been generated over the past decade [1]. The growth trend in the number of IoT devices will continue. According to the forecast from Statista, by 2030 the number of IoT devices will reach more than 25.4 billion [2]. In this regard, the volume of generated and transmitted data over the Internet will increase. It creates an issue for highly sensitive systems to time delays. Among of all transmitted data types over the Internet, the most capacious in terms of volume are video and photographic images.

One such system that is sensitive to network delays is a traffic monitoring system (TMS). Those systems are engaged in the following time-consuming tasks: calculating the speed of a vehicle, recognizing the number plate of cars, detecting traffic accidents, traffic jams and violations, regulating traffic flow, making a decision to call a fire brigade and ambulance in case of traffic accidents [3] - [5]. It should also be noted that the one of listed functions of the system, the process of recognizing a number plate seems simple at first glance, but it can still take a long time for recognition and data transmission over the network.

The aim of this research is to implement the process of recognizing the number plate of a car based on computer vision, and then provide a prompt transfer data to a data center and traffic monitoring center (TMC) due to the edge computing.

II. RELATED WORKS

In accordance with the related works, the following stages were identified as the main stages of number plate recognition: number plate localization (NPL), character segmentation (CS) and optical character recognition

(OCR) [4]. At the first stage, the next approaches such as image binarization or grayscale are used under adverse weather conditions or low light. Then the number plate must be localized and segmented through Connected Component Analysis (CCA) and ratio analysis. And finally, character recognition comes through Support Vector Classifier (linear, polygonal, rbf), k Nearest Neighbor (KNN), Extra Tree Classifier, Logistic Regression (LR), Random Forest (RF) and Support Vector Classifier (SVC) plus KNN [5]. Despite the high data processing speed, the implementation of such systems based on field-programmable gate arrays (FPGA) is quite expensive.

As an alternative technical solution, we propose to use a Raspberry Pi, single-board computer, which will reduce the cost of system development and make it easy to track the packet of data sent over the network. To solve the problem of long-term data transfer, it is proposed to apply the edge computing paradigm. A number of articles have shown the successful implementation of the transition from traditional cloud architecture to the edge computing paradigm with the final result of increasing the speed of data transfer over the network [3], [6] - [9]. This approach has found its success in Smart House, Smart Traffic Monitoring System, Smart city, recognizing human emotions, monitoring indicators of the patient's health, i.e. in those systems where the speed of processing and data transmission is critical. However, it should be noted that when performing time-consuming tasks and a situation where the Internet connection is stable, it is advisable to use a hybrid paradigm that allows the use of both cloud and edge computing by choosing the optimal network through an intelligent task offload system based on fuzzy logic [10, 11].

Raspberry Pi is a single board computer that can easily be used in both paradigms. Another advantage of the Raspberry Pi is the medium speed processor, the provision of peripheral interface and network support, which allows the collection and analysis of data from IoT devices [12]. There is the Message Queue Telemetry Transport (MQTT) as a messaging protocol solution on the Raspberry Pi. It is considered one of the most used messaging protocols in the IoT industry due to its low power consumption [13].

III. Materials and Methods

As we mentioned in the previous section, the Raspberry Pi 4 Model B (4 Gigabyte) with an OmniVision OV5647 camera have been used as an edge device for number plate recognition based on computer vision algorithms (Fig. 1).

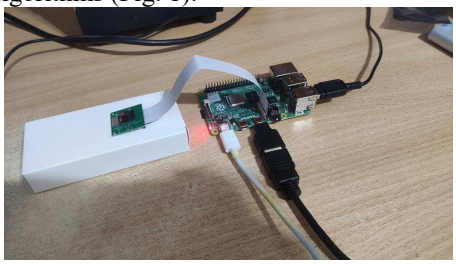

Figure 1. Edge node (Raspberry Pi and OmniVision OV5647 Camera)

The MQTT protocol based on Node Red and MQTTBox have been used to organize a message between the publisher (Raspberry Pi) and subscribers (TMC operators). The scheme for organizing interaction through the MQTT protocol is shown in Fig. 2.

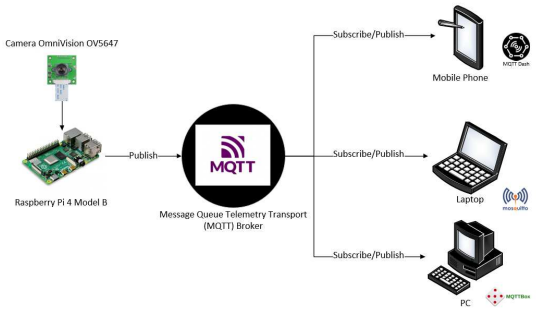

Figure 2. The MQTT message broker interaction scheme

A. Internet of Things Network Configuration

We have installed Node Red on the Raspberry Pi to organize the data flow and visual programming for IoT devices, and also configured the MQTTBox message broker on the side of client. The recommended operating system for the Raspberry Pi is Raspberry Pi OS (Raspbian), but for the server could be any compatible operating system with a message broker based on the Linux core or Windows. In our case, Windows and MQTTBox message broker have been chosen.

The IoT network scheme is shown in Fig. 3.

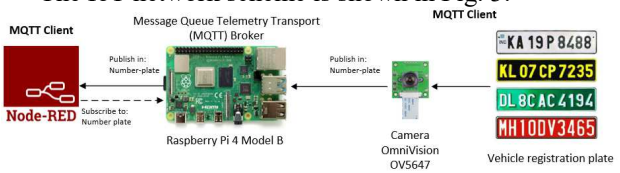

Figure 3. IoT network scheme

The OmniVision OV 5647 camera transmitted photo and video images to the Raspberry Pi. A single-board computer has used computer vision algorithms to detect the number plate of the car and transferred it to the server side in text format via the MQTT message broker based on Red Note. After that the TMC operator has received the processed data via MQTTBox and should decide to enter the number plate of the car into the database, for instance, in case of traffic violations.

B. Development of a Computer Vision Algorithm

For the convenience process of developing a computer vision algorithm, it is highly recommended to configure and deploy the VNC Viewer on the side of the developer, as well as run the Secure Shell (SSH), Virtual Network Computing (VNC) protocols on the Raspberry Pi.

Nano editor and the OpenCV, Tesseract, PYTTSX3, Imutils, PIL import Image, picamera, and Tesseract OCR libraries were used to write the algorithm in the Python programming language and localize the number plate, provide a character segmentation and optical character recognition [14]. Fig. 4 shows a flowchart of computer vision algorithm for recognizing the number plate of a car.

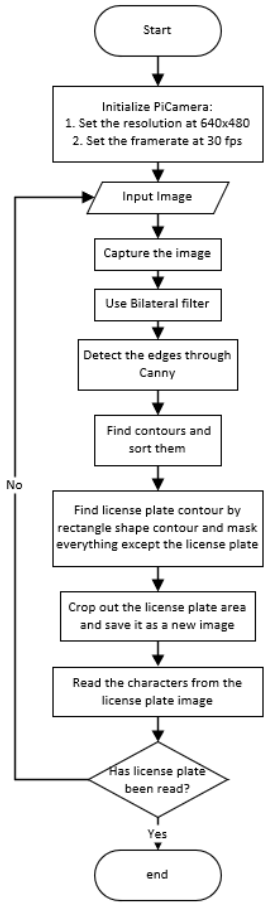

Figure 4. The flowchart of computer vision algorithm

At the start of flowchart, the camera was initialized. Its resolution was set at 640x480 and frame rate was set at 30 frames per second.

The capture continuous function was used to start capturing frames from the Raspberry Pi camera. After capturing a frame, a bilateral filter function was applied to remove unwanted details from the captured image.

After removing unnecessary details, the Canny Edge method was applied to perform edge detection. For the next step, we search for contours in the obtained images and sort them from big to small.

Since the Raspberry Pi can find multiple contours, we have to filter the number plate contour by looking for a rectangular shape. After that all areas were cropped except for the number plate. The resulting image was saved as a new one.

At the last stage, characters were read from the number plate image through the Tesseract library and the recognized characters were saved to a variable. Fig. 5 shows an example of displaying the number plate as the text (output in the terminal) and as a cropped image.

Figure 5. The process of identifying the number plate of a car through computer vision

C. *Development of the Message Queue Telemetry Transport Message Broker Algorithm*

The aedes broker, inject, exec, function, mqtt out, mqtt in, and debug nodes were used from the Node Red's menu to build the number plate recognition flow and organize the message flow. The flow diagram is shown in Fig. 6.

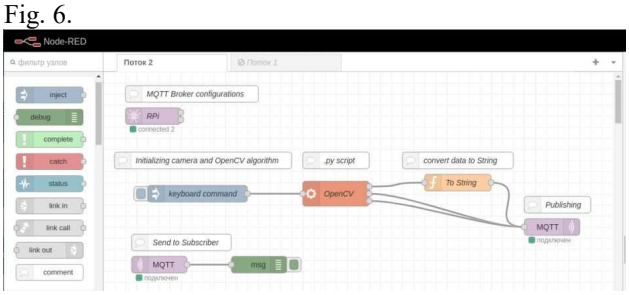

Figure 6. Flow in Node Red

The aedes broker (RPi) node allows user to configure the message broker for the Raspberry Pi and specify its port. The inject node (keyboard command) initialized the work of the flow, the camera, and launches the OpenCV node (computer vision algorithm), which implemented in the Python programming language for number plate localization, character segmentation and optical character recognition. The results are transferred to the function node (To string) in order to translate the text into a readable form. The mqtt out and mqtt in nodes (MQTT) publish data to the address of subscribers (TMC operators). The msg node allows user to display the result.

IV. Experimental Results

Fig. 7 shows the results of processing photo images in the Cropped and terminal windows as the text. The image captured by the OmniVision OV 5647 is displayed in the Frame window.

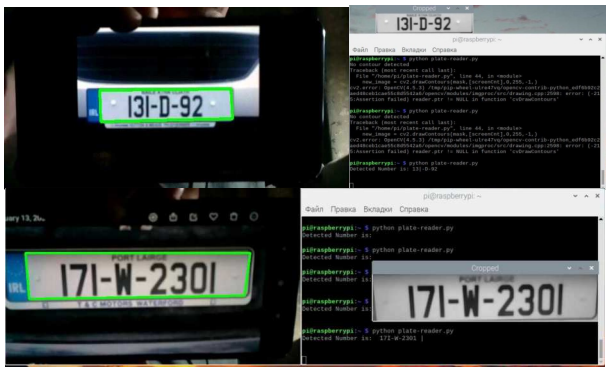

Figure 7. The result of processing photographic images with the number plate of the car

According to the results as the text, we can note some error in the identification of number plates of cars, which is associated with a small database for training computer vision.

In Fig. 8 the "Debug" tab of Node Red console (the right side of the window) present the results of number plate recognition. The result of data transfer to the TMC operator and the execution time of computer vision algorithm are indicated in the MQTTBox message broker window after the text "run time".

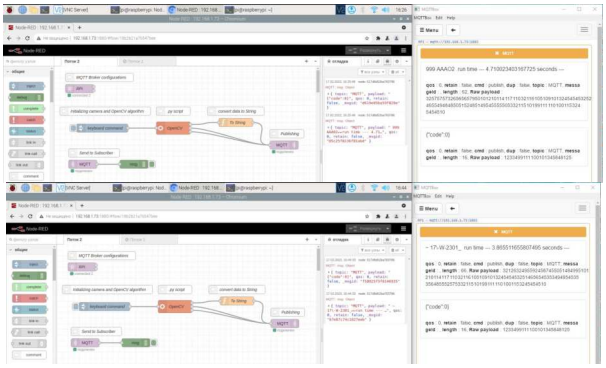

Figure 8. The result of processing photographic images in Node Red and transferring data to subscriber

The average time to execute the algorithm was 4 seconds. The presented results also have an error in number plate recognition.

A graphic diagram of the network traffic for transferring data to the message broker is shown in Fig. 9.

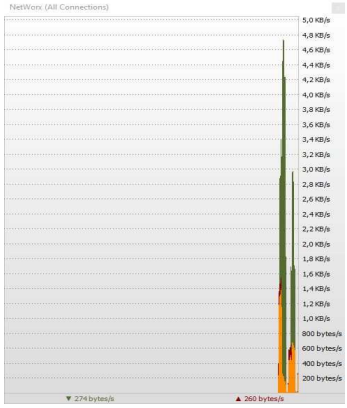

Figure 9. A graphic diagram of network traffic for transferring data to a message broker

The maximum value of the network traffic was about 4.78 kilobits per second. Thus, it can be noted that the process of data transfer using the edge computing paradigm contributes to the optimization of network traffic, since under the cloud paradigm, the amount of transmitted photo and video data is measured in megabits per second and gigabits per second.

Table 1 presents various image processing methods, platforms, resolution and execution time in milliseconds [4, 15].

TABLE I. ANALYSIS OF THE COMPARISON OF EXECUTION TIME OF DIFFERENT IMAGE PROCESSING ALGORITHMS IN CENTRAL PROCESSING UNIT

Image processing method	***Platform***	***Resolution***	***Execution time, milliseconds***
High Definition Number Plate Localization	Zynq-7000 SoC	960×720 (HD)	16.17
Standard Definition Number Plate Localization	FPGA Virtex-4	640×480 (SD)	6.7
Oily style NPR	CPU: Q9450 with 42.56 GFLOPS GPU: NVIDIA G92 (GeForce 9800 GTX) with 128 Cores and 512 MB Video Memory, GTX 280 for Speed UP Test	512×512	7172
Corners and edge detection	OS: Ubuntu 11.04 CPU: Dual Core 6600, 2.40 GHz, Mem: 2 GB GPU: GeForce GTX 280, 240 CUDA cores, Memory: 1GB GPU: Tesla C1060, 240 CUDA cores, Memory: 4GB	2048×2048	4006
Content authenticatio n	CPU Intel Xeon 5520 (2.26GHz) RAM 12GB DDR3 (1333MHz) GPU Architecture Tesla C1060 OS Centos 5.3 (64 bit) V2.3 CUDA	1024×1024	28877.66
Linear feature extraction	CPU: Q9450 with 42.56 GFLOPS GPU: NVIDIA G92 (GeForce 9800 GTX) with 128 Cores and 512 MB Video Memory, GTX 280 for Speed UP Test	1800×1400	500.958685
Inverse sinusoidal contrast transformati on	AMD Phenom II Quad-core to 3.2 GHz, 12 GB of RAM, Operating System: 64-bit Linux Fedora 14 GPU: GeForce 430 GT video card with 96 cores and 1 GB of RAM DDR3 is used	1024×1024	151.256079

Thus, the execution time of various image processing methods on field-programmable gate arrays (FPGAs) is actually faster than on a Raspberry Pi. However, it should be noted that the execution time of algorithm on a Raspberry Pi is slightly inferior to the computing power of a personal computer when its processing a photo image by a central processing unit.

V. CONCLUSION

In this paper, we proposed to apply the edge computing paradigm based on a Raspberry Pi 4 Model B with an OmniVision OV 5647 camera and conduct data pre-processing on this edge device, which helped to

optimize network traffic and offload the computing power of the data center. The applied edge computing approach with Raspberry Pi and data transmission as text helped to reduce the cost of the technical solution, in contrast to field-programmable gate arrays (FPGAs) and personal computers. In addition, it is worth noting that the proposed method allows engineer to increase the number of connected IoT devices to the network due to the minimal impact on network traffic, and also allows engineer to keep most of the data confidential.

Thus, in order to increase the speed and quality of recognition of number plate through computer vision algorithm, it is proposed to increase the training database. And also add the functionality of switching the edge device operation mode between cloud and edge architectures.

However, also taking into account the above problem, it seems that it may be promising to create hybrid methods for recognizing number plates, using the advantages and eliminate the disadvantages of the various approaches.

REFERENCES

[1] M. Marjani, F. Nasaruddin, A. Gani, A. Karim, I. A. T. Hashem, A. Siddiqa, and I. Yaqoob, "Big IoT Data Analytics: Architecture, Opportunities, and Open Research Challenges," Ieee Access, Article vol. 5, pp. 5247-5261, 2017.

[2] A. Sulich, M. Rutkowska, A. Krawczyk-Jezierska, J. Jezierski, and T. Zema, "Cybersecurity and Sustainable Development," Knowledge-Based and Intelligent Information & Engineering Systems (Kse 2021), Proceedings Paper vol. 192, pp. 20-28, 2021.

[3] G. X. Liu, H. Shi, A. Kiani, A. Khreishah, J. Lee, N. Ansari, C. J. Liu, and M. M. Yousef, "Smart Traffic Monitoring System Using Computer Vision and Edge Computing," Ieee Transactions on Intelligent Transportation Systems, pp. 1-12, September 2021.

[4] A. Al-Zawqari, O. Hommos, A. Al-Qahtani, A. A. H. Farhat, F. Bensaali, XJ. Zhai, and A. Amira, "HD number plate localization and character segmentation on the Zynq heterogeneous SoC," Journal of Real-Time Image Processing, Article vol. 16, no. 6, pp. 2351-2365, Dec 2019.

[5] S. Babbar, S. Kesarwani, N. Dewan, K. Shangle, and S. Patel, "A New Approach For Vehicle Number Plate Detection," in 11th International Conference on Contemporary Computing (IC3), Jaypee Inst Informat Technol, Noida, INDIA, Aug 02-04 2018, NEW YORK: Ieee, in International Conference on Contemporary Computing, 2018, pp. 196-201.

[6] S. Y. Nikouei, R. Xu, and Y. Chen, "Smart surveillance video stream processing at the edge for real-time human objects tracking," Fog and Edge Computing: Principles and Paradigms, pp. 319-346, 2019.

[7] J. N. Yang, T. T. Qian, F. Zhang, and S. U. Khan, "Real-Time Facial Expression Recognition Based on Edge Computing," Ieee Access, Article vol. 9, pp. 76178-76190, 2021.

[8] H. Yar, A. S. Imran, Z. A. Khan, M. Sajjad, and Z. Kastrati, "Towards Smart Home Automation Using IoT-Enabled Edge-Computing Paradigm," Sensors, Article vol. 21, no. 14, p. 23, July 2021, Art no. 4932.

[9] S. S. Brimzhanova, S. K. Atanov, M. Khuralay, K. S. Kobelekov, and L. G. Gagarina, "Cross-platform compilation of programming language Golang for Raspberry Pi," 5th International Conference on Engineering and MIS (ICEMIS 2019), Proceedings Paper vol. 10, June 2019.

[10] S. K. Atanov, Z. Y. Seitbattalov and Z. S. Moldabayeva, "Development an Intelligent Task Offloading System for Edge-Cloud Computing Paradigm," 2021 16th International Conference on Electronics Computer and Computation (ICECCO), pp. 1-6, November 2021.

[11] Z. Y. Seitbattalov, S. K. Atanov, and Z. S. Moldabayeva, "An Intelligent Decision Support System for Aircraft Landing Based on the Runway Surface," 2021 IEEE International Conference on Smart Information Systems and Technologies (2021 IEEE SIST), pp. 495-499, April 2021.

[12] R. Mahmud, A.N. Toosi, "Con-Pi: A Distributed Container-based Edge and Fog Computing Framework," (2021) IEEE Internet of Things Journal.

[13] D. B. C. Lima, R. Lima, D. D. Medeiros, R. I. S. Pereira, C. P. de Souza, and O. Baiocchi, "A Performance Evaluation of Raspberry Pi Zero W Based Gateway Running MQTT Broker for IoT," 2019 Ieee 10th Annual Information Technology, Electronics and Mobile Communication Conference (Iemcon), Proceedings Paper pp. 76-81, 2019.

[14] Y. Arslan and H. Canbolat, "Performance of Deep Neural Networks in Audio Surveillance," 2018 6th International Conference on Control Engineering & Information Technology (Ceit), Proceedings Paper p. 5, 2018.

[15] S. Saxena,S. Sharma and N. Sharma, "Parallel Image Processing Techniques, Benefits and Limitations," Research Journal of Applied Sciences, Engineering and Technology, Article vol. 12, no. 2, pp. 223-238, January 2016.