UDC 519.6
https://doi.org/10.31891/csit-2022-3-3

LESIA MOCHURAD
Lviv Polytechnic National University, Lviv, Ukraine

A NEW APPROACH FOR TEXT RECOGNITION ON A VIDEO CARD

An important task is to develop a computer system that can automatically read text content from images or videos with a complex background. Due to a large number of calculations, it is quite difficult to apply them in real-time. Therefore, the use of parallel and distributed computing in the development of real-time or near real-time systems is relevant. The latter is especially relevant in such areas as automation of video recording of traffic violations, text recognition, machine vision, fingerprint recognition, speech, and more. The paper proposes a new approach to text recognition on a video card. A parallel algorithm for processing a group of images and a video sequence has been developed and tested. Parallelization on the video-core is provided by the OpenCL framework and CUDA technology. Without reducing the generality, the problem of processing images on which there are vehicles, which allowed to obtain text from the license plate. A system was developed that was tested for the processing speed of a group of images and videos while achieving an average processing speed of 207 frames per second. As for the execution time of the parallel algorithm, for 50 images and video in 63 frames, image preprocessing took 0.4 seconds, which is sufficient for real-time or near real-time systems. The maximum acceleration of image processing is obtained up to 8 times, and the video sequence – up to 12. The general tendency to increase the acceleration with increasing dimensionality of the processed image is preserved, which indicates the relevance of parallel calculations in solving the problem.
Key words: CUDA technology, real-time system, machine learning, acceleration factor, Gaussian filter, optical character recognition.

ЛЕСЯ МОЧУРАД
Національний університет «Львівська політехніка»

НОВИЙ ПІДХІД ДО РОЗПІЗНАВАННЯ ТЕКСТУ НА ВІДЕОКАРТІ

Важливою задачею є розробити таку комп'ютерну систему, яка б могла автоматично зчитувати текстовий вміст із зображень чи відео зі складним фоном. Через велику кількість обрахунків досить важко застосовувати її в реальному часі. Тому актуальним є застосування паралельних та розподілених обчислень при розробці real-time або near real-time систем. Останні набувають особливої актуальності в таких сферах як автоматизація відеореєстрації порушень правил дорожнього руху, розпізнавання тексту, машинний зір, розпізнавання відбитків пальців, мови, тощо. У роботі запропоновано новий підхід щодо розпізнавання тексту на відео карті. Розроблено та протестовано паралельний алгоритм обробки групи зображень та відеоряду. Розпаралелення на відео ядрі у роботі забезпечують фреймворк OpenCL та технологія CUDA. Не зменшуючи загальності, розглянуто задачу обробки зображень на яких присутні транспортні засоби, що дозволило отримати текст з номерного знаку. Розроблено систему, яку було протестовано на швидкість виконання обробки групи зображень та відео при цьому досягнуто середньої швидкості обробки - 207 кадрів на секунду. Щодо часу виконання паралельного алгоритму, то для 50 зображень та відео в 63 кадри препроцесинг зображень зайняв 0.4 секунди, що є достатнім для real-time або near real-time систем. Отримано максимальне прискорення обробки зображень до 8 разів, а відеоряду – до 12. Збережено загальну тенденцію щодо збільшення прискорення при збільшені розмірності оброблюваного зображення, що свідчить про актуальність проведення паралельних обчислень при вирішенні розглядуваної у роботі проблеми.
Ключові слова: технологія CUDA, real-time system, машинне навчання, коефіцієнт прискорення, фільтр Гаусса, оптичне розпізнавання символів.

Introduction

One of the very important, but non-trivial tasks of machine learning is the problem of text recognition from the image [1-3]. Due to the large number of calculations it is quite difficult to apply it in real time. However, the calculation can be significantly accelerated by using parallelization, which will allow you to perform operations independently for each image.

The main characteristic of the video is the number of frames per second, that the number of consecutive full-screen images that are displayed every second. On average, the human eye can process 12 frames per second. Most smooth motion cameras record at 30 or 60 frames per second [4]. Each of these photos can be used to find or recognize specific objects, process them for a specific purpose, and perform other tasks.

Assuming that the frame rate in the video is 30, you can extract 1800 images from the minute video. If you analyze a video that lasts more than an hour, the number of images will exceed 100,000. On the CPU, the processing of all these images can take a long time, and therefore there is a problem to speed up this process.

To solve the problem described above, it is proposed to use CUDA technology. Its use allows you to get a great acceleration over time, given the presence of drivers and libraries that allow you to run code on a graphics processor [5].

For optical character recognition, three main methods of image processing are used: morphological filter, low-pass filter and classification algorithm [6]. For this study, a sequence of processing was chosen by adding to the original image of its Top Hat transformation and subtracting from the original image of its Black Hat transformation. This sequence allows you to select the contours of the characters, which allows you to find and further process license plates.

Low-spectrum filters in optical character recognition are used mainly for image blur. With the help of the convolution operation and the lower spectrum filter as the core, the image blur effect is achieved. For blurring use averaging filtration, Gaussian filter, median filtration, bilateral (two-way) filter. Averaging filtering is performed by a convolution operation with a normalization filter. This kernel takes the average of all the pixels below it and replaces the central element. The Gaussian filter uses the Gaussian core.

This is a standard and effective blur method that is often used by default. Also, this filter is highly effective in cleaning the image from Gaussian noise. The median filtering occurs by finding the median of the values under the nucleus and replacing the central element with the found median. Effective in cleaning pulse noise (a common example is noise in analog television) [7]. Bilateral filtering is effective in eliminating noise and keeping edges clear, but this operation is very slow compared to other filters.

Among these options, a Gaussian filter was chosen because of the best ratio of execution speed and blur efficiency. The averaging filter and median filtering blurred the edges too much, which could lead to the elements of the number merging with its edges and, consequently, the loss of information. The bilateral filter relative to the analogues is slower, which can slow down the algorithm.

Various algorithms are available for the classification problem, from classification algorithms (Random Forest, Decision Tree, KNN and others) to neural networks (LaNet5, AlexNet, VGG, ResNet and others) [9-13]. Due to the fact that the paper considers image processing, the classification algorithm – KNN [14] was chosen. This algorithm is implemented in the OpenCV library, which is used for image processing, which allows you to use it with the appropriate data type. Because the test system supports CUDA technology, it was decided to use the integrated module in the OpenCV image processing library itself to work with this technology. This solution allows you to seamlessly run image processing algorithms on the video core.

A set of car license plates from kaggle.com was used for testing [15]. This dataset contains 433 photos of cars with license plates. To test the video stream, video recordings of different lengths were created based on the selected dataset.

The relevance of the topic is due to the fact that the methods of optical character recognition are not sufficiently studied. Also, many tasks that use character recognition require high performance for real-time use. This performance, in turn, can be provided by image processing concurrency, for which parallelization on the video core is best, allowing many simple image operations to be performed simultaneously.

The purpose of this work is to develop an algorithm for the parallel processing of a group of images or a video sequence. At the same time, without reducing the generality, the problem of processing images on which there are vehicles is considered, which will allow us to obtain the text from the license plate.

To achieve this aim, we set the following tasks:

1. Define algorithms that will highlight the desired text in the image.
2. Apply these algorithms to find first the license plate and then the characters in it.
3. Classification of found characters, which will allow us to get the text from the image.
4. Parallelization of the algorithm for processing images of vehicle numbers, and analysis of its efficiency in comparison with the sequential version.

The object of research is the algorithm for selecting and retrieving text from images and its parallel implementation.

The subject of the research is the application of sequential and proposed parallel algorithms for obtaining text from images of vehicle numbers and comparing their efficiency.

The main contribution of this article can be summarized as follows:

1. We have for the first time proposed a computationally efficient approach to the parallel processing of a group of images and a video sequence. The problem of image processing on which there are vehicles, allows to obtain text from the license plate, which is especially relevant for the development of real-time and near real-time systems to automate video recording of traffic violations.
2. We have developed an algorithmic implementation of the proposed approach. The OpenCL framework and CUDA technology were used.
3. We have demonstrated an increase in the processing speed of a group of images and a video sequence. Thus, within the acceleration indices 8 were obtained – for image processing and 12 – for video. We reduced computational complexity using the proposed algorithm. The effectiveness of the proposed approach with increasing the number and dimensionality of the processed images is confirmed.

The rest of this article is organized as follows: literature analysis is presented in Section 2. A detailed description of the sequential and proposed parallel algorithm is given in Section 3. Section 4 contains the results of numerical experiments of algorithmic implementation of the proposed approach, and their discussion is presented in Section 5. Studies are presented in the last section.

Related works

In most cases, OCR (optical character recognition) technology is used to recognize text, which allows you to convert images, PDF files or scanned documents into editable formats. The technology itself consists of two stages: image pre-processing, character recognition for a certain part of the image [16].

An example of the implementation of this technology is the work [17], the purpose of which was to extract text from images to automate the input of nutrition data into fitness applications. The article gives an example of image processing and achieved an accuracy of over 80% for standard font recognition.

Proper pre-processing of the image is of great importance for the success of text recognition. For example, in [18] describes in great detail the preparation of the image for extracting text from it and as a result for different alphabets (Chinese, Arabic and Urdu) was achieved an accuracy from 86 to 99%.

Also, the result of the image processing stages is illustrated by the authors in the work [19] and describes the methods of implementation in C ++.

In [20], the K-Nearest Neighbor algorithm was used for character recognition. The OpenCV library was used for both image processing and classification. After analyzing this work, we used the KNN algorithm for character recognition.

An example of using KNN for our model problem, namely license plate recognition, is also described in the article [21]. The difference in the stage of image preparation is the use of the principal components method to remove the features of reducing the dimensionality of the image set.

After processing the source data, it was noticed that there was no parallel processing of the preprocessing stages for a large number of images, as well as a small amount of data on the method of extracting characters (mostly mentions of the Tesseract package). Our work is an attempt to solve this problem.

Methodology

Sequential image processing algorithm

1. Apply morphological transformations to one image.

A combination of Top Hat and Black Hat operations is used as morphological transformations. Top Hat (T_w) operation is defined as

$$T_w(f) = f - f \circ b,$$

where f this image is grayscale, $\circ$ − indicates the opening operation, b − is a structural element. A structural element is a binary matrix that defines the structure in units.

For Black Hat (T_b), the operation will look like this:

$$T_b(f) = f \cdot b - f,$$

where $\cdot$ − is a closing operation.

The opening operation is defined as:

$$A \circ B = (A \ominus B) \oplus B,$$

where A is the original set of values, and B is a structural element; $\ominus$ − erosion operation, $\oplus$ − expansion operation.

Closing is the operation inverse to opening and is defined as follows:

The erosion operation is mathematically represented as: $(f \ominus b)(x) = inf_{y \epsilon B}[f(x+y) - b(y)]$, where *inf* is an infinum, $f(x)$ is an image, $b(x)$ is a structural element, B is the space in which $b(x)$ is defined.

The expansion operation then looks like this:

$$(f \oplus b)(x) = sup_{y \epsilon E}[f(y) + b(x-y)],$$

where sup is the supremacy, $f(x)$ is the image, $b(x)$ is the structural element, E is the space in which it is defined.

2. Apply to one image of a Gaussian filter.

Gaussian blur is an image convolution operation with a Gaussian kernel:

$$I_\sigma = I \cdot G_\sigma,$$

where I is the original image, $\cdot$ − convolution operation, G_σ − two-dimensional Gaussian kernel with standard deviation σ, I_σ − the resulting image after applying the filter with σ.

The Gaussian nucleus with normal distribution is defined as:

$$G_\sigma = \frac{1}{2\pi\sigma} e^{\frac{-(x^2+y^2)}{2\sigma^2}},$$

where x and y are the coordinates of the image value on the respective axes OX and OY.

Thus, the two-dimensional Gaussian function will look like this:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}}.$$

3. Repeat steps 1 and 2 for all images.

Parallel image processing algorithm

1. A separate stream is allocated for each image.
2. A parallel algorithm of morphological transformations is applied to all images.

Morphological transformations are performed by convolution operation with specific operation cores. Unlike a sequential algorithm, where the convolution operation is applied using a floating window, that the value of each element is alternately calculated on the processor core, CUDA technology allows you to calculate the value of each image element simultaneously. That is, there is a division of the algorithm by memory, which allows you to perform one operation on different data, which, in turn, corresponds to the purpose of the video core.

Thus, the time complexity of the algorithm will be reduced by simultaneously calculating the resulting convolution values for the image or part thereof.

For every element $(i, j)|i \in m, j \in n$ matrix of the original image A_{mn} we take the matrix of the dimension of the nucleusB_{wh} B_{wh} where the center of the matrix will be the selected element (i, j) and multiply it by the kernel. The center of the resulting matrix will be an element of the matrix, which will represent the image A with the applied filter.

3. A parallel filtering algorithm is used for the image.

As with the sequential algorithm, a Gaussian filter with an identical core and function is applied to the image, but thanks to CUDA technology, this operation is performed on the entire image at once. The approach is identical to accelerating the application of morphological transformations, because the Gaussian filter is also used by the convolution operation, only with the Gaussian kernel.

Because GPU cores are not completely independent, but can only perform the same calculations, each action on the matrix becomes equivalent to the action on one matrix element. That is, if we greatly simplify the idea of CUDA architecture, then a matrix of size 320×2 when adding 1 will perform on each of the 640 cores an operation equivalent to adding to the (i, j)–th element of the matrix unit.

Since image processing operations are mainly the use of a convolution operation with a certain core that returns images of the same size, all these actions can be reduced to the use of a filter multiplied by a constant value equal to the number of filters used in the algorithm, and therefore the complexity of the algorithm can be generalized to the following formula:

$$O(w_{img} \cdot h_{img} \cdot w_{kernel} \cdot h_{kernel}).$$

Accordingly, when paralleling with CUDA, the complexity over time will be equal to the complexity of the algorithm divided by the number of threads:

$$\frac{O(w_{img} \cdot h_{img} \cdot w_{kernel} \cdot h_{kernel})}{N}.$$

where N – number of threads.

Experiments

The tests were performed on a system with the following characteristics:

CPU: core i5-8300H

RAM: DDR4 2667 MHz 24 Gb

GPU: NVIDIA GeForce GTX 1050

GPU RAM: GDDR5 4096 MB

CUDA Cores: 640

Memory Interface: 128-bit

GPU Interface: PCI Express x8 Gen3

The results of the software implementation of OCR are recognized text. An example of text recognition together with the stages of image transformation obtained on the basis of the developed information system is presented in Figure 1.

Fig. 1. Example of the result of OCR software implementation

When recognizing characters in an image, identical transformations are performed for the entire image (Origin) and part of the recognized as a license plate (Plate). Since this preprocessing occurs for different types of data, the processing time was calculated separately for them. This partition allows us to compare the effectiveness of the proposed parallel algorithm to different sizes of data.

In Tables 1 – 4 shows the values of the execution time of the sequential image processing algorithm are presented for each group, the number of measurements – 5. These tables allow us to find the average execution time of the sequential algorithm. A total of 100 tests were performed on the CPU for comparative image processing performance.

Table 1

Measurements of the execution time of a sequential algorithm depending on the number of images

Number of images	Origin processing time, ms					Average time, ms
50	906	896	899	910	898	901.8
100	2564	1915	1832	1843	1833	1997.4
433	10463	7990	7972	7915	8117	8491.4
1000	18621	19120	20670	19187	19251	19369.8
2000	38595	37678	40455	43235	38835	39759.6
Plate processing time, ms						
50	365	358	365	400	361	369.8
100	740	716	755	776	757	748.8
433	3392	3722	3300	3328	3409	3430.2
1000	7709	7901	7756	8078	8037	7896.2
2000	16594	16410	16543	16748	16185	16496

From the results of Table 1 you can see that photo processing is much faster for license plates. This is due to the fact that the license plate is a cut part of the overall image and its dimension is much smaller.

Table 2

Measurements of the execution time of a sequential algorithm depending on the number of frames in the video

Number of images	Origin processing time, ms					Average time, ms
63	1301	1359	1390	1346	1326	1344.4
150	6681	5991	6120	6145	6000	6187.4
300	10277	8619	8738	8995	8807	9087.2
600	20774	17662	16153	16407	16994	17598
1200	40441	34648	32917	34037	34037	35216
Plate processing time, ms						
63	332	372	337	339	333	342.6
150	1247	1107	1089	1075	1072	1118
300	3647	3359	3296	3182	3157	3328.2
600	6440	6349	6378	6370	6459	6399.2
1200	14249	14283	15150	14283	14719	14536.8

From the results of Table 2 it can be seen that the trend of decreasing processing time for general images and license plate images persists. But because for video, unlike a set of images, each frame has the same resolution, the average execution time increases more smoothly.

In Tables 3 – 4 show the data processing time for the proposed parallel algorithm using GPU and CUDA technology.

Table 3

Measurements of execution time of the proposed parallel algorithm for image processing on the video core

Number of images	Origin processing time, ms					Average time, ms
50	399	429	412	404	415	411.8
100	506	491	517	497	496	501.4
433	1165	1152	1152	1145	1129	1148.6
1000	3151	3373	3196	3280	3186	3237.2
2000	5180	5193	5011	4973	4730	5017.4
Plate processing time, ms						
50	188	184	183	180	179	182.8
100	308	314	312	308	316	311.6
433	1341	1364	1354	1337	1328	1344.8
1000	3575	3895	3508	3515	3827	3664
2000	7076	7046	6785	6730	6742	6875.8

In Table 3 in comparison with the results of Table 1, it is seen that the average processing speed of the general images increases from 2 to 12 times. This is due to the fact that the GPU-parallel algorithm applies each operation simultaneously to all data. The speed of processing license plate images was initially lower than the processing of the full image, but later it became higher. This is due to the fact that in this case the small dimension of the license plate image does not allow to fully obtain the acceleration of parallel image processing on the GPU and this indicates the reliability of the results.

Table 4

Measurements of execution time of the proposed parallel algorithm for video processing on the video core

Number of images	Origin processing time, ms					Average time, ms
63	457	477	457	468	488	469.4
150	877	883	876	882	914	886.4
300	1238	1246	1192	1198	1188	1212.4
600	1920	1752	1738	1749	1719	1775.6
1200	3340	3100	3115	3065	3091	3142.2
Plate processing time, ms						
63	352	350	369	351	363	357
150	933	901	907	893	905	907.8
300	1670	1674	1672	1681	1668	1673
600	5702	5232	5272	5234	5259	5339.8
1200	7228	7102	6956	7060	6897	7048.6

According to the data presented in Table 4, video stream frame processing with more than 150 frames is performed more efficiently for larger images. Which corresponds to the general trend. The best result for processing license plate images at a video frame length of 63 frames may be due to the fact that the number of images transmitted for processing was small enough that the resources allocated to the image processing stream sufficiently affect the efficiency of the algorithm.

Based on the data obtained from the above tables, the following graphs were compared comparing the results for further analysis.

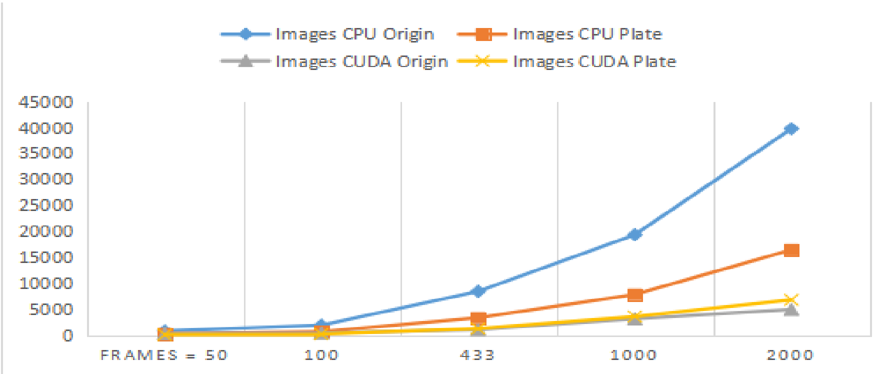

Fig. 2. Graph of comparison of image processing time for different technologies and sizes, time, ms

From Figure 2 shows that with increasing number of images that are processed, the difference in execution time between serial and parallel versions of the algorithm – increases. It is also noticeable that the processing of larger images gives a greater gain in time than the processing of smaller ones.

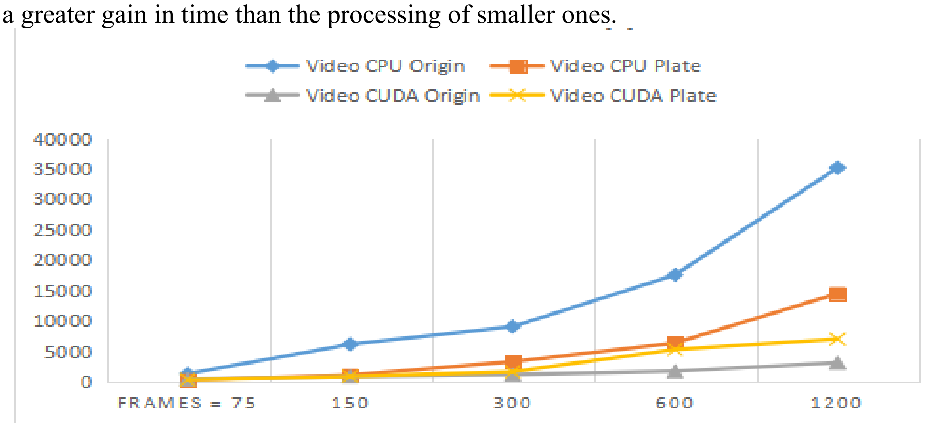

Fig. 3. Graph of comparison of video processing time for different technologies and sizes, time, ms

From the graph (see Figure 3) of the speed comparison it is seen that with increasing number of frames in the video, which is processed, the difference in execution time between consecutive parallel processing options increases. There is also the same trend as in previous tests - processing larger images gives more gain. The processing of smaller images has a less stable trend than in the previous version, which can be caused by different image complexity.

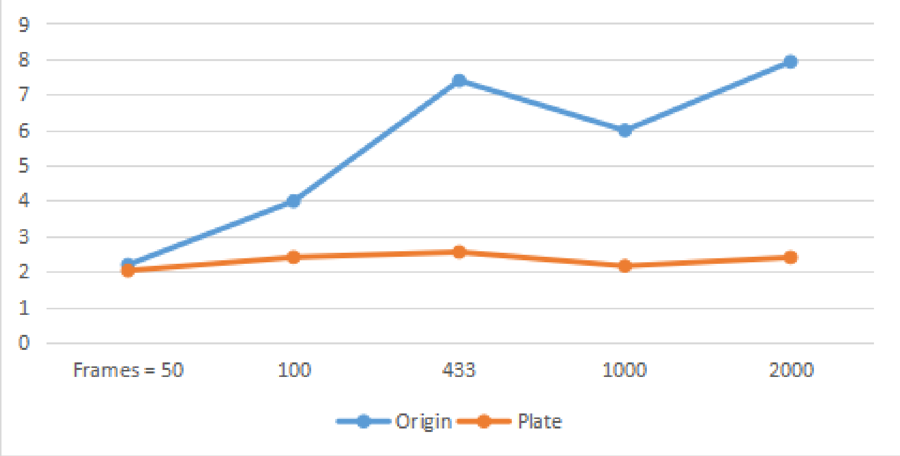

Fig. 4. Graph comparing the acceleration of image processing on the GPU depending on the number of frames and image size

This graph (see Figure 4) of image processing acceleration showed acceleration up to 8 times. You can also see a sharp increase in the processing of 433 photos and a small decline in 1000. The main reason for this is the different resolution ability of photos, because the implementation of high-quality image processing takes much longer.

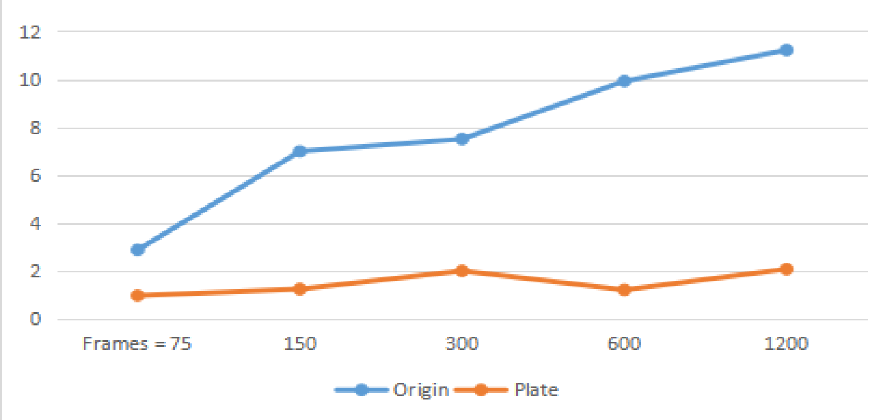

Fig. 5. Graph comparing the acceleration of video processing on the GPU depending on the number of frames and image size

From Figure 5 shows that the acceleration in this case reaches 12, and increases much more smoothly. This is because, unlike a set of many photos of vehicles with different resolutions, video frames have an identical extension. Therefore, this graph describes the acceleration for real-time image processing much more accurately.

Discussion of research results

The algorithmic implementation of the proposed parallel approach is performed in the work. As a result, a system was developed for which the average image processing speed is 207 frames per second. This speed allows you to build realtime or near-realtime systems for OCR. Such systems allow automating the video recording of violations and omissions in the road sector, which is especially relevant today.

To compare the execution speed of the serial and the proposed parallel image processing algorithm, 5 tests were performed for 5 different image sizes and 5 videos for 2 groups of image sizes for both serial and parallel algorithms. For further analysis, the average results of 5 tests for each of the groups were taken.

Based on the results of testing sequential and parallel image processing algorithms for OCR, a comparison chart of the average acceleration for image processing was constructed (see Figure 6).

According to this graph, we can conclude that when processing video, where a larger type of image (from 600x400) is more efficiently processed on the GPU. Also from this graph it is seen that at small images of parallelization on GPU by means of SUDA technology practically does not give an increase, however still almost 1.5 times more effectively than the consecutive algorithm.

According to the results of numerical experiments, the feasibility of the proposed parallel algorithm based on CUDA technology for GPUs is confirmed. To improve the results, you can transfer the calculation of small images to the CPU, which can achieve much greater acceleration.

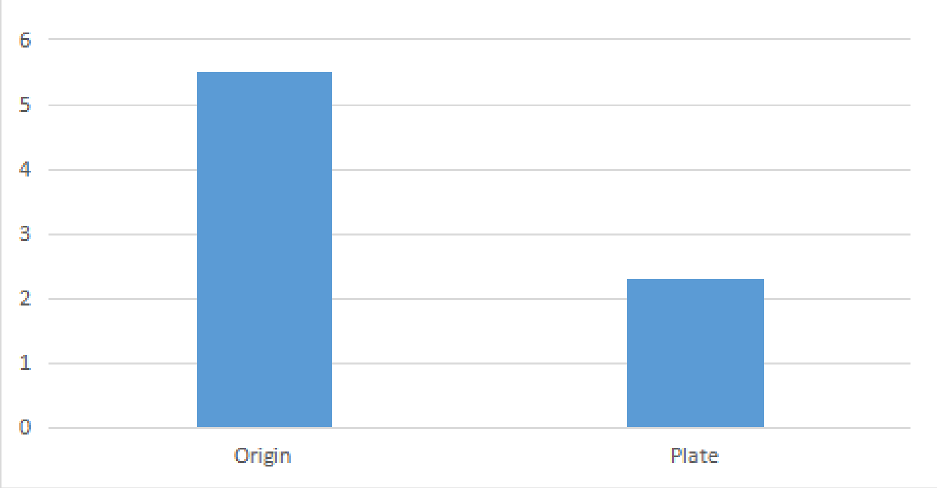

Fig. 6. Diagram comparing the average acceleration of image processing of different dimensions on the GPU (using CUDA technology)

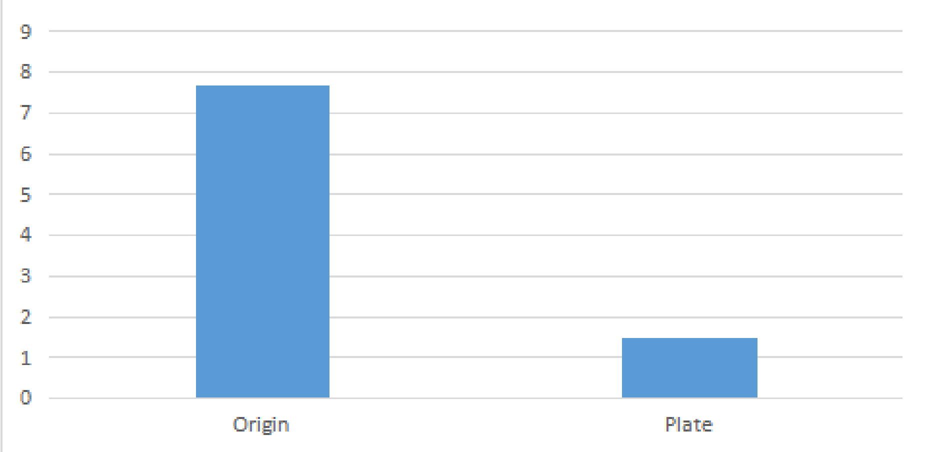

Fig. 7. Graph comparing the acceleration of video processing on the GPU for different image sizes

Conclusions

This paper analyzes image processing algorithms that allow you to select and retrieve text from photos, and identifies 3 main stages: the use of a morphological filter, the use of a blur filter to eliminate noise and the classification of the results.

First, the structure of the morphological filter was studied. The next step was to choose one of the blurring algorithms and the algorithm for classifying the results. The Gaussian filter was chosen as the blur and noise reduction algorithm because it is the highest quality and most effective for most conventional images. Among the classification algorithms, in turn, was chosen KNN because of the simplicity of its implementation.

Next, numerous experiments were conducted, in which there was a comparison of sequential and parallel implementation of the algorithm for processing images of cars. The results showed that the acceleration when processing large images in parallel is much greater than small ones. This is because GPU parallelization applies one operation to the entire image at a time, which is effective for high-resolution photos. In terms of execution time, image reprocessing took 0.9 seconds for 50 images and more than 1.3 seconds for 63-frame videos. Such results are not sufficient for real-time or near real-time systems to automate video recording of violations. In turn, the algorithm paralleled by the GPU showed a time of 0.4 seconds for both cases. In this case, the acceleration is doubled when processing images, and 3 times when processing video. And this value can increase up to 12 times if you increase the number of frames.

Prospects for further research are the selection and comparison of the application of other classification algorithms, as well as the maximum increase in the dimensions of the processed images. Also, the system developed in the work can be implemented as a system of automation of video recordings of traffic violations and passes in the road sector.

References

1. Chen X. and Jin L., Zhu Yu., Luo C., Wang T. Text Recognition in the Wild: A Survey. *J. ACM*, 1(1), 2020, 34 p. doi: 10.1145/nnnnnnn.nnnnnnn.
2. Nadav Ben-Haim. Task Specific Image Text Recognition. *UC San Diego Electronic Theses and Dissertations,* 2008, 39 p.
3. Coates A., Carpenter B., Case C., Satheesh S., Suresh B., Wang T., Wu D., and Ng A. Text Detection and Character Recognition in Scene Images with Unsupervised Feature Learning. *In: ICDAR : IEEE Computer Society*, 2011, 440-445.
4. FPS (Frames Per Second) Definition - TechTerms. https://techterms.com/definition/fps.

5. Mochurad L., Shchur G. Parallelization of Cryptographic Algorithm Based on Different Parallel Computing Technologies. *Proceedings of the Symposium on Information Technologies & Applied Sciences (IT&AS 2021)*. Bratislava, Slovak Republic, March 5, 2824, 2021, 20-29.

6. Chaudhuri A., Mandaviya K., Badelia P., Ghosh S.K. Optical Character Recognition Systems. In: Optical Character Recognition Systems for Different Languages with Soft Computing. *Studies in Fuzziness and Soft Computing.* Springer, Cham, 352, 2017, 9-41. doi: 10.1007/978-3-319-50252-6_2.

7. Liang, H.; Li, N.; Zhao, S. Salt and Pepper Noise Removal Method Based on a DetailAware Filter. *Symmetry*, 13, 515, 2021, 24 p. doi: 10.3390/sym13030515.

8. Morillas S., Gregori V., Sapena A. Fuzzy Bilateral Filtering for Color Images. In: Campilho A., Kamel M.S. (eds) Image Analysis and Recognition. *ICIAR 2006. Lecture Notes in Computer Science.* Springer, Berlin, Heidelberg, 4141, 2006, 138-145. doi: 10.1007/11867586 13.

9. Thanh Noi, P., Kappas, M. Comparison of Random Forest, K-Nearest Neighbor, and Support Vector Machine Classifiers for Land Cover Classification Using Sentinel-2 Imagery. *Sensors (Basel, Switzerland)*, 18(1), 2017, 18. doi: 10.3390/s18010018.

10. Boateng, E. , Otoo, J. and Abaye, D. Basic Tenets of Classification Algorithms K-Nearest-Neighbor, Support Vector Machine, Random Forest and Neural Network: A Review. *Journal of Data Analysis and Information Processing,* 8, 2020, 341-357. doi: 10.4236/jdaip.2020.84020.

11. Şanlı T., Sıcakyüz Ç., Yüregir O.H. Comparison of the accuracy of classification algorithms on three data-sets in data mining: Example of 20 classes. *International Journal of Engineering, Science and Technology*, 12(3), 2020, 81-89. doi: 10.4314/ijest.v12i3.8.

12. Maeda-Gutiérrez V., Galván-Tejada C.E., Zanella-Calzada L.A., Celaya-Padilla J.M., Galván-Tejada J.I., Gamboa-Rosales H., Luna-García H., Magallanes-Quintanar R., Guerrero Méndez C.A. and Olvera-Olvera C.A. Comparison of Convolutional Neural Network Architectures for Classification of Tomato Plant Diseases. *Appl. Sci.* 10(1245), 2020, 15 p. doi: 10.3390/app10041245.

13. Khan A., Sohail A., Zahoora U., Qureshi A.S. A Survey of the Recent Architectures of Deep Convolutional Neural Networks. *Artificial Intelligence Review*, 2019, 70 p. doi: 10.1007/s10462-020-09825-6.

14. Mochurad L.I. Optimization of numerical solution of model problems on the basis of parallel calculations. *Chapter 1: monograph.* Lviv: PE "BONA Publishing House", 2021, 208 p.

15. Kaggle. Car License Plate Detection dataset. https://www.kaggle.com/andrewmvd/car-plate-detection.

16. Memon J., Sami M., Khan R. A. and Uddin M. Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR). *In IEEE Access*, vol. 8, 2020, 142642-142668. doi: 10.1109/ACCESS.2020.3012542.

17. Matsunaga Nate and Sullivan Rick. Image processing for the extraction of nutritional information from food labels. *Computer Science and Engineering Senior Theses*, 42, 2015.

18. Karthick K., Ravindrakumar K.B., Francis R., Ilankannan S. Steps Involved in Text Recognition and Recent Research in OCR; A Study. *International Journal of Recent Technology and Engineering (IJRTE)*, 8(1), 2019, 3095-3100**.**

19. Harraj A., Raissoun N. OCR accuracy improvement on document images through a novel pre-processing approach. Signal & Image Processing : An International *Journal (SIPIJ)*, 6(4), 2015, 18 p. doi: 10.5121/sipij.2015.6401.

20. Ong V., Suhartono D. Using K-Nearest Neighbor in Optical Character Recognition. *ComTech: Computer, Mathematics and Engineering Applications*, 7(1), 2016. doi: 10.21512/comtech.v7i1.2223.

21. Gunawan D., Rohimah W. Rahmat R.F. Automatic Number Plate Recognition for Indonesian License Plate by Using K-Nearest Neighbor Algorithm. *IOP Conference Series: Materials Science and Engineering*, 648, 2019, 8 p.

Lesia Mochurad **Леся Мочурад**	PhD, Associate Professor of Department of Artificial Intelligence, Lviv Polytechnic National University, Lviv, Ukraine, e-mail: lesia.i.mochurad@lpnu.ua. https://orcid.org/0000-0002-4957-1512	Кандидат технічних наук, доцент, доцент кафедри систем штучного інтелекту національного університету "Львівська політехніка", Львів, Україна.