Biyosistem Mühendisliği Dergisi
Journal of Biosystems Engineering

Biyosistem Müh Derg 3(1):1-17, 2022
J Biosystems Eng 3(1):1-17, 2022

Araştırma Makalesi/Research Article
ISSN: 2757-8100

Application of Parallel Background Algorithm on Radar Images to Detect Sudden Weather Changes

Ahmet ÇELİK[1*]

Kütahya Dumlupınar University, Tavşanlı Vocational School, Computer Technology, Tavşanlı, Kütahya

Received: 14 June 2021 Received in revised: 14 March 2022 Accepted: 08 June 2022

Abstract: Today, depending on the development of technology, large and diverse data are formed. It will be very useful to process these data quickly and to make decision-making process by an automated system. Decision-making process by machine learning and many calculations processes can be performed quickly and easily with the help of programs written on computer systems. If the programs written are parallel, fast results will be obtained in a shorter time. Paralleling can be performed using a common memory structure or using a distributed memory structure. In this study, an application that can be used in a fast early warning system indicating weather forecast or rapidly changing weather conditions has been implemented. For this, a frame difference background extraction algorithm was applied on the infrared satellite images of a specified region at different time intervals. This algorithm was used to compare the cloudiness levels of the images obtained at time t and the images obtained at time t-30 min. Subsequently, the results obtained are shown in color. In addition, in this study, the Background subtraction algorithm was applied by paralleling it on images of different resolutions in order to detect the speed gain. According to the results, it was determined that the high resolution images were processed 38.47 times faster with the parallel program than the serial program

Key words: Weather forecast, early warning system, image processing, background subtraction, parallel system, machine learning

Ani Hava Değişimlerini Tespit Etmek için Radar Uydu Görüntüleri Üzerinde Paralel Arkaplan Çıkarma Algoritmasının Uygulanması

Alınış tarihi: 14 Haziran 2021 Düzeltilme tarihi: 14 Mart 2022 Kabul tarihi: 08 Haziran 2022

Özet: Günümüzde teknolojinin gelişmesine bağlı olarak büyük ve çok çeşitli veriler oluşmaktadır. Bu verilerin hızlı bir şekilde işlenerek karar verme işleminin otomatik bir sistem tarafından gerçekleştirilmesi oldukça faydalı olacaktır. Bilgisayar sistemleri üzerinde yazılan programlar yardımıyla birçok hesaplama ve makine öğrenme yöntemleriyle de karar verme işlemleri hızlı ve kolay bir şekilde gerçekleştirilebilmektedir. Yazılan programların paralel olması durumunda daha kısa sürede hızlı sonuçlar elde edilecektir. Paralelleştirme, ortak bellek yapısı kullanılarak veya dağıtık bellek yapısı kullanılarak gerçekleştirilebilmektedir. Bu çalışmada hava tahmini veya hızlı değişen hava durumlarını belirten hızlı bir erken uyarı sisteminde kullanılabilecek bir uygulama gerçekleştirilmiştir. Bunun için belirlenen bir bölgenin, farklı zaman aralıklarında elde edilmiş kızıl ötesi uydu görüntüleri üzerinde çerçeve farkı arka plan çıkarma algoritması uygulanmıştır. Bu algoritma t zamanında elde edilmiş görüntü ile t-30 dakika zamanında elde edilmiş görüntülerin, bulutlanma düzeylerini kıyaslamak için kullanılmıştır. Daha sonra elde edilen sonuçlar renklendirilerek gösterilmiştir. Ayrıca bu çalışmada hız kazanımını tespit etmek için farklı çözünürlükteki görüntüler üzerinde Arka plan çıkarma algoritması paralelleştirilerek uygulanmıştır. Elde edilen sonuçlara göre yüksek çözünürlüklü görüntülerin paralel program ile seri programdan 38.47 kat daha hızlı işlendiği tespit edilmiştir.

Anahtar kelimeler: Hava tahmini, erken uyarı sistemi, görüntü işleme, arka plan çıkarma, paralel sistem, makine öğrenmesi

To Cite: A. Çelik, 2022. Application of Parallel Background Algorithm on Radar Images to Detect Sudden Weather Changes. Journal of Biosystems Engineering 3 (1):1-17

*Corresponding author(Sorumlu yazar): ahmet.celik@dpu.edu.tr

1. Introduction

Background subtraction is a basic method for detecting moving objects from videos. In this method, the differences between the reference frame and the active frame are determined. The reference frame is accepted as the background image or background model. The background model should be of high quality and free from noise. Poor background subtraction results are obtained when using a low quality background model. In this case, it causes false results to be obtained. The extraction of foreground objects from video images is performed by computer image processing methods. Background subtraction algorithms differ from each other with their mathematical calculation methods (Deepjoy and Sarat, 2014).

In some cases, it can be difficult to reveal objects in movie frames. Shadows, lighting variables, dynamic backgrounds in the images make it difficult to extract the object from the video images. For this, it is useful to predetermine the pixel colors of the objects as a reference. These pixels can make sense in video images (Braham et al., 2017)

Analyzing and interpreting video images is a current research area. Background models are used for different applications to detect the foreground object from the background image by detecting the foreground. Moving objects in video images are called “foregrounds”. In some cases, erroneous results may be encountered in detecting objects from the background model when there are lighting changes. Detecting the ambient change and motion detection in the background extraction method are two main problems (Sakpa and Sabnis, 2018).

In recent years, especially in military fields, background extraction methods have been used to detect moving targets from video and image applications. However, most methods are not successful enough due to dynamic background and noise in the image (Zuo et al., 2019).

Hu et al. (2018) designed an algorithm in which the background image is automatically renewed in urban areas when there is no moving object. This method has been a simple and powerful model that can adapt quickly to different lighting conditions. By means of this method, it is ensured that only useful data are obtained in the video images of urban surveillance systems where very large data are formed.

Yannick et al. (2010) indicated Basic Motion Detection (Basic), One Gaussian (1-G), Minimum, Maximum and Maximum Inter-Frame Difference (Min Max), Gaussian Mixture Model (GMM), Kernel Density Estimation (KDE), Codebook (CBRGB), Eigen Backgrounds (Eigen) performed a comparison of the background subtraction methods. This benchmark has been in terms of their success in different video images, their memory usage requirements and the processor requirements they need for computing. The study revealed that both low memory and low CPU usage cannot be performed at the same time. However, the One Gaussian model was found to be better in terms of speed.

Wang et al. (2016) used two background models; it is intended to detect moving objects. Updates are performed by combining background contents with each other. Objects remaining in the foreground are detected by comparing the active video image with two backgrounds. This method is mentioned in the literature as a dual class background model.

Sakpal and Sabnis (2018) used the selective background extraction technique to remove unnecessary areas from the foreground object and background. In the study; Instead of removing the entire background from the stage, black areas are obtained by removing unnecessary areas in the image. Black areas are also eliminated by using the neighboring background. In the study, both gray images and HSV images were examined. The percentage of success achieved was 99.99% correct and 0.61% false.

Chen et al. (2020) developed a method for detecting the fall event by using the background and feature classification methods on the images they obtained from RGB cameras. With the increase in the rate of elderly population in the World, the safety of the elderly living alone has become a social problem. Falling is one of the most common and potential dangers faced by elderly people living alone in confined spaces, as elderly people experience impaired physical function, slow sensory response, and loss of balance. Generally, these falls cause injury, loss of mobility and worse health problems. In the study, the fall event detection of elderly people was successfully performed by using background extraction and feature classification methods on images taken from computerized vision based surveillance systems.

Zuo et al. (2019) used an advanced Gaussian Mixture Model (GMM) to detect moving objects within the dynamic background. Mathematical morphological operations were applied to reduce the noise on the images of the film frame containing moving objects. It has obtained more successful results than the classical Gaussian Mixture Model.

Huynh-The et al. (2019) proposed the locally statistical dual-mode (LSD) method to detect moving objects in video-based security camera systems. They used Wallflower and CDnet2014 as data set. The foreground local region pixel densities of the objects in the image were calculated by means of average and standard deviation methods. In the study, images with 31 frames per second (fps) speeds were successful.

It is not possible to make physical changes to achieve high performance from a system using single processor hardware. However, high performance can be achieved by designing a system with common memory or distributed memory in software. Structurally, multi-core processors are used in common memory systems during the production phase. Performance gain is achieved when job sharing is performed on threads on a software basis. CUDA software on graphic processors and OpenMP software on central processing units are widely used. In distributed memory systems, systems using a single processor structure are created by communicating with each other with the help of network equipment. PVM (Parallel Virtual Machine) and MPI (Message Passing Interface) are widely used in these systems as software (Akçay et al., 2011).

Today's operating systems use the multi-thread model structurally in multi-core or single-core hardware. Structurally, the one-to-one core and thread method in the processor has higher success. However, in a limited processor structure, performance decreases when multi-threads are run. This is because many threads compete to use processor resources. In addition, messaging between threads also causes a waste of time (Lim et al., 2020).

Orak and Çelik (2017) applied the COLMSTD algorithm to detect defects on the rail surfaces during the hot rolling process by paralleling it with the support of the CUDA interface on the graphics processor that includes a common memory parallel system. Paralleling has been accomplished by sharing each image among the threads, and by sharing the parts of each image among the threads. Faster results were obtained by running each image on separate cores. In addition, in this study, when the rail images are processed in a parallel system, it has been observed that they give 25 times faster results than the serial system.

Uçkan and Dal (2016) indicated robot animation using C++ and OpenGL libraries was performed both serially and in parallel using CUDA architecture on the graphics processor. The number of image frames formed per unit time was compared and it was seen that parallel system performance using CUDA gave good results between 33% and 164%.

Akçay and Erdem (2013) performed the multiplication of two matrices of 1000*1000 dimensions both in series and in parallel. The application was written on the Intel Parallel Studio interface and its performance was measured on the CPU. The serial process took place in 6844 milliseconds, and in parallel it was realized in 2266 milliseconds.

In this study; the application was carried out on the infrared satellite images obtained from the international weather forecast system by using serial and parallel background subtraction image processing algorithm. In practice, the infrared images obtained according to different clouding movements were processed in series and parallel and the changes that occurred were shown by color. The processor in the hardware structure used has a frequency of 2.4 GHz, 4 cores and 8 logical processors. Work division has been realized on the parallel system threads.

2. Material and Method

In this study, the cloudiness levels were determined by applying serial and parallel background extraction algorithm on the infrared satellite images obtained at intervals of 30 minutes over a website that performs weather forecast. Frame difference background subtraction algorithm was used to determine the cloudiness levels on the images. The flowchart of the work performed is shown in figure 1.

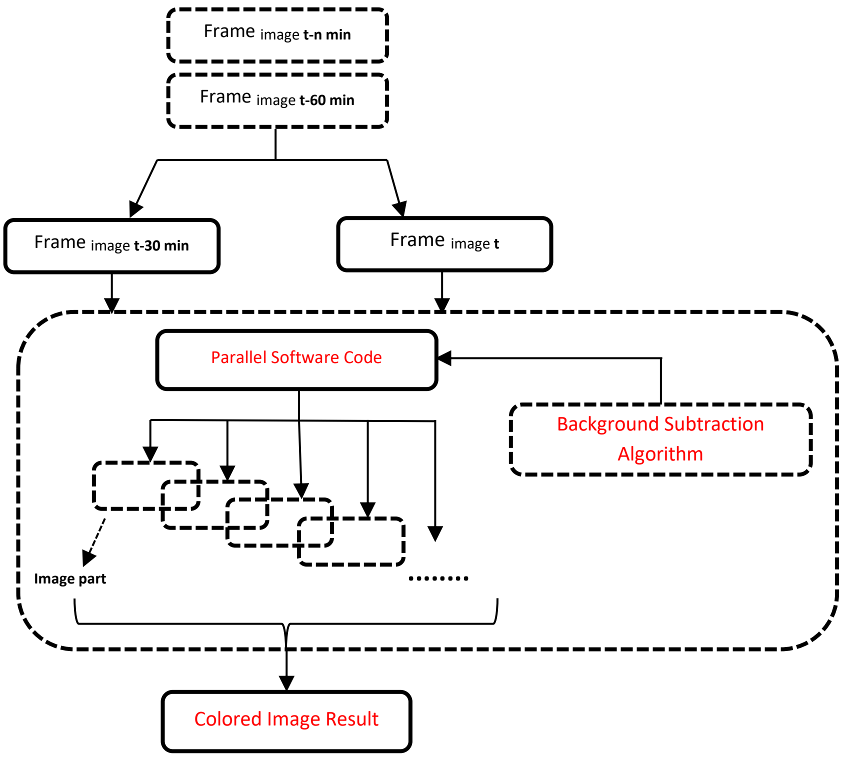


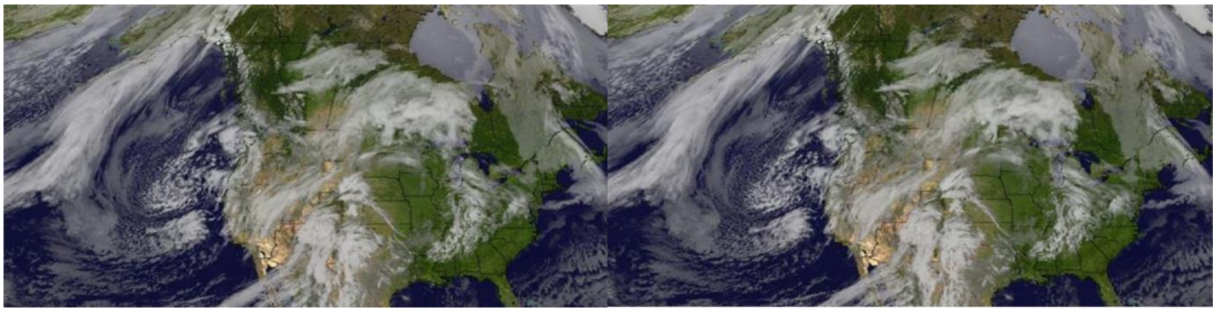
Figure 1. Flow chart of the used method.

Images obtained via satellite and published on the website at 30-minute intervals are read by the software. As shown in the flow chart of the study, the image at the time t is compared with the image at the t-30 min time. The last frame image and the previous frame image are compared live. The comparison is performed using the background subtraction algorithm, and then the cloud regions are determined. In order to perform this process quickly, images are divided into parts within the central processing unit. Then, these image pieces were processed as threads on the central processor unit cores.

In this study, infrared satellite images published on i-weather.com website, including the regions of USA, Canada and Mexico, North Pacific Ocean and North Atlantic Ocean, were used. In the study, a background extraction algorithm was applied on the images taken from the satellite at 30-minute intervals. The reason for choosing this region is that the infrared satellite images are high resolution. Thanks to its flexible structure, this study can be applied on high resolution live images of different regions. The map of the processed area obtained from the Google Maps satellite is shown on Figure 2.

Figure 2. Map of the imaged region (Google Maps, 2021).

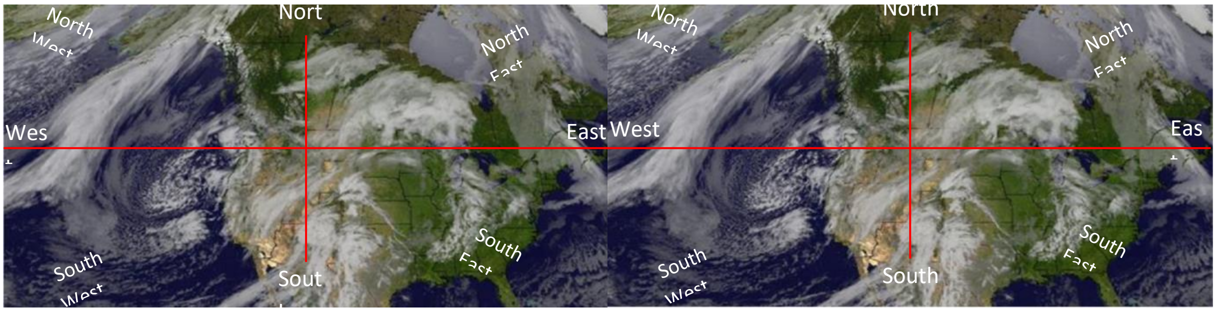
The cloudiness levels on the infrared satellite images of the regions at t-30 min and t times in the study are shown in figure 3. The clouding levels and locations of the images in the figure are actually different from each other. It is very difficult and slow to distinguish these differences by the human eye. This process can be performed accurately and quickly by using computer-aided image processing.



a) b)

Figure 3. Satellite radar images a) Image at t-30 minute time b) Image at t time (Freemeteo, 2021).

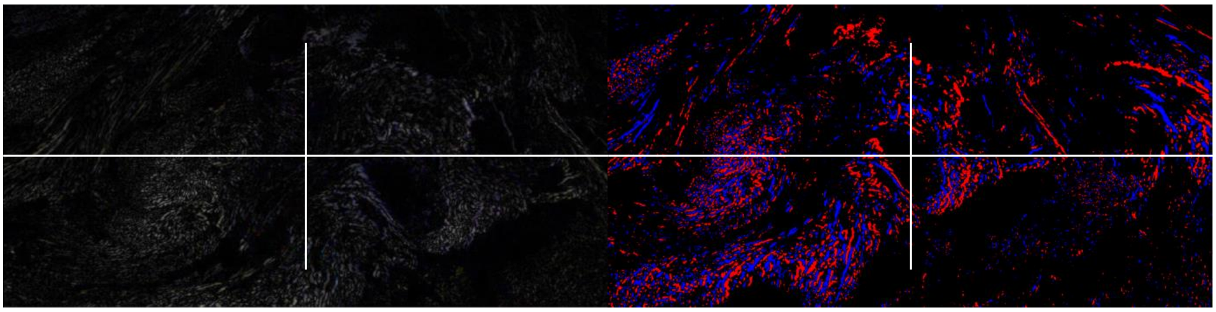
Figure 3 a) shows radar image at t-30 minute time and figure 3 b) shows radar image at t time. The image in figure 4 was created by entering the direction information on the satellite images. Thus, clouding movements that occur as a result of the application can be interpreted using directional information.



a) b)

Figure 4. Radar satellite images containing direction information a) Image containing direction information at t-30 minute time b) Image containing direction information at time t (Freemeteo, 2021).

Figure 4 a) shows radar image containing direction information at t-30 minute time and figure 4 b) shows radar image containing direction information at time t. The results shown in figure 5 were obtained when the images obtained at t-30 minute and t times were analyzed using the background subtraction algorithm.



a) b)

Figure 5. Results when background subtraction algorithm is applied a) Colorless clouding differences of images b) Color clouding differences of images.

Figure 5 a) shows the differences in cloudiness between the two images. Here, clouds that appear at time t-30 minute but disappear at time t and Clouds that are not in t-30 minute time but newly formed at t time cannot be distinguished from each other. However, this distinction is shown by using blue and red colors on figure 5 b). The clouds that formed in t-30 minute time but disappeared at time t are also shown with red pixels but also the clouds that disappeared at t-30 minute time, but formed new clouds at time t are shown with blue pixels.

2.1. Frame Differences Background Subtraction Method

The frame difference is applied by subtract the previous frame from the active frame. The results obtained as a result of subtraction are compared with a threshold value. Pixels greater than this threshold value are consequently displayed on the image. The advantage of this method is that requires less calculation and is faster (Deepjoy and Sarat, 2014).

Vehicle tracking, human identification, wildlife monitoring, security camera analysis, and target tracking can be performed quickly within the image obtained from a fixed camera using background extraction methods. This method is known as Basic Motion Detection (Basic) in the literature (Yannick et al., 2010). In this method, moving objects are detected by revealing the difference of pixel values in the same coordinates when the active image and the background image are compared. The applying method of the background subtraction algorithm is shown in equation 1.

$$B_s = \begin{cases} 1 & \text{if } |d(I_a, B_m)| > T_h \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

B_s, indicates the application's result of the background subtraction method. The image of the active video film frame is indicated by I_a, the background model is indicated by B_m and the distance of the active image to the background is indicated by d. When T_h (threshold) value which determined and d (distance) value is compared, the pixels larger than the threshold are shown white color (1) and the other pixels are shown black (0) (Yannick et al, 2010). Equation 2 shows the distance calculation.

$$d(I_a, B_m) = \begin{cases} I_{Pixel_{Red_a}} - B_{Pixel_{Red_m}} \\ I_{Pixel_{Blue_a}} - B_{Pixel_{Blue_m}} \\ I_{Pixel_{Green_a}} - B_{Pixel_{Green_m}} \end{cases} \quad (2)$$

The red color of the active frame image pixel is $I_{Pixel_{Red_a}}$, its blue color is $I_{Pixel_{Blue_a}}$, and its green color is $I_{Pixel_{Green_a}}$. The red color of the background model image pixel is $B_{Pixel_{Red_m}}$, its blue color is $B_{Pixel_{Blue_m}}$, and its green color is $B_{Pixel_{Green_m}}$. The calculation of the colored background subtraction algorithm is shown in equation 3.

$$B_{s_{colored}} = \begin{cases} \text{if } d(I_a, B_m) > Th \; {-}{-}{-}{>} \; \text{Red} \\ \text{else if } d(I_a, B_m) < -Th \; {-}{-}{-}{>} \; \text{Blue} \\ \text{else} \; {-}{-}{-}{>} \; \text{Black} \end{cases} \quad (3)$$

$B_{s_{colored}}$, indicates the color scheme of the image with its background removed. Th, indicates the positive threshold value, $-$Th the negative threshold value. It is colored red when it is greater than the positive value, blue when it is smaller than the negative value and black when it is lower than the negative value.

2.2. Shared Memory Parallel Systems

Parallel systems are encountered in two ways: shared memory or distributed memory. Distributed memory parallel systems are created by communicating different computers with the help of network devices. But in shared memory systems, many processor cores with the same features use common memory areas by working with the same frequency values. In the software department, work sharing is done and then the threads are run in parallel on the cores (Çelik and Özmen, 2009).

Processor cores can process a single thread in a period of time, as well as process two threads (Hyper-Threading) or more threads (Multi-Threading) at the same time. However, the most optimum number of threads should be selected to run on the core (Silberschatz et al., 2012). Because too many generated threads can cause performance loss when they use too much displacement or messaging while using kernel resources. The common memory parallel system infrastructure of the processor used in this study is shown in Figure 6. The processor has 4 independent cores and 2 logical operations within each core have one Arithmetic Logic Unit (ALU) Therefore, a total of 8 Logical Processors (LCPU) work in the processor structure. By sending threads to logical processors or cores at the same time, work execution is performed in parallel. The common core and logical processor structures on the central processor unit are shown Figure 6.

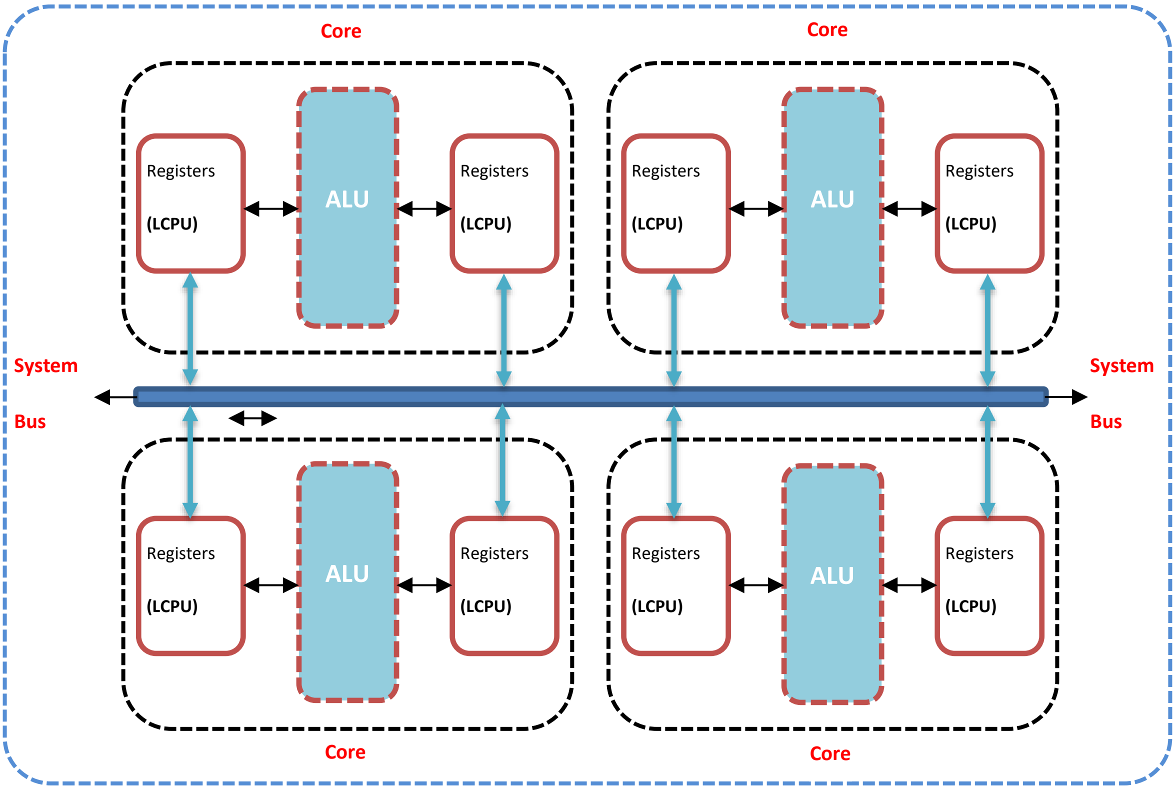


Figure 6. The cores and logical processors of CPU structure.

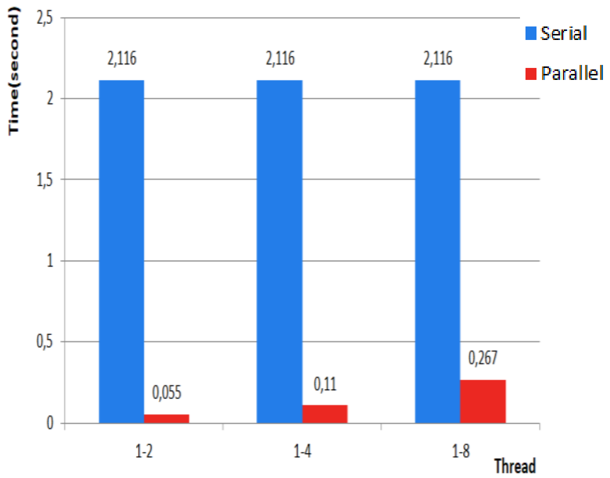
3. Experimental Results

In this study; clouding movements have been demonstrated on the radar images with 1280x640, 1024x512, 768x384, 512x256, 256x128 and 128x64 resolutions using the background subtraction algorithm. Satellite images are published live for weather forecasts all day long. It will be very useful to process the images obtained by an automatic system quickly. In this study, when high resolution images are processed on a parallel system, a very fast analysis is performed from the serial system. Performance comparison was performed by applying the background subtraction algorithm using serial and parallel on the satellite images. Threads were worked in parallel on the cores in the central processor unit. Parallel system design was realized by changing the thread count between 2 and 8. Processing times and performance values when images with different resolutions are run using different threads are shown in table 1. The central processing unit (CPU) used in this study has 4 cores. Each of the cores can perform 8 logical processor tasks thanks to its Hyper Threading capability. In the study, the highest speed and performance was achieved when an image with a resolution of 1280x640 was processed. Accordingly, 38.47 times higher performance was obtained with parallel code than serial code. The lowest performance was achieved when an image with 128x64 resolution was processed. Accordingly, 0.54 times slower performance was obtained with parallel code than serial code.

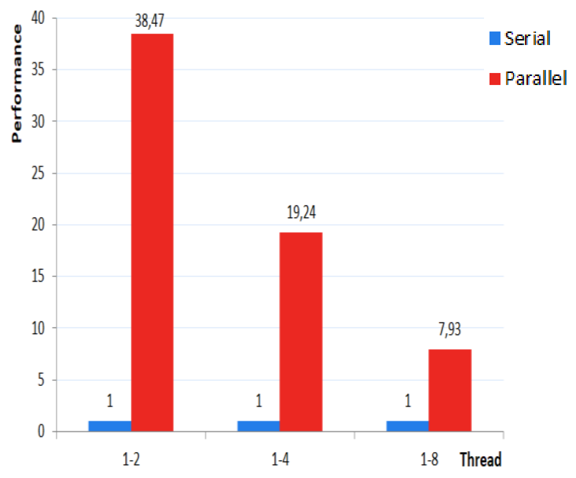
Table 1. Processing performance according to different image resolutions of threads.

Thread Counts	Time (Second)	Performance (Serial/Parallel)	Resolution
1	2,116	1,00	1280x640
2	0,055	38,47	1280x640
4	0,11	19,24	1280x640
8	0,267	7,93	1280x640
1	1,366	1,00	1024x512
2	0,037	36,92	1024x512
4	0,075	18,21	1024x512
8	0,21	6,50	1024x512
1	0,751	1,00	768x384
2	0,023	32,65	768x384
4	0,048	15,65	768x384
8	0,18	4,17	768x384
1	0,332	1,00	512x256
2	0,013	25,54	512x256
4	0,027	12,30	512x256
8	0,103	3,22	512x256
1	0,109	1,00	256x128
2	0,007	15,57	256x128
4	0,018	6,06	256x128
8	0,046	2,37	256x128
1	0,02	1,00	128x64
2	0,005	4,00	128x64
4	0,015	1,33	128x64
8	0,037	0,54	128x64

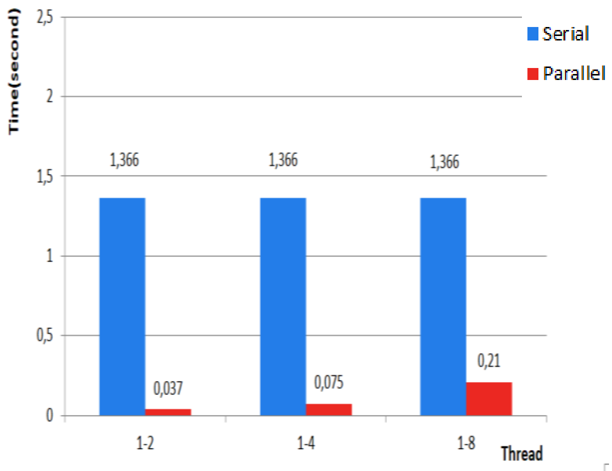
The graphs in figure 7 were obtained by using the values on table 1. The speed and performance results obtained when images with different resolutions are processed both in series and in parallel are shown in the graphics. When parallel code using 2, 4 and 8 threads and serial code run on image, the speed and performance values obtained are shown in graphics. According to all test results, the highest speed was obtained when images were processed on 2 threads. The speed and performance of the serial code with the blue columns, and the speed and performance of the parallel code with the red columns is showing in the graphs.


Time(second)
2,5
2
1,5
1
0,5
0
2,116
2,116
2,116
0,055
0,11
0,267
1-2
1-4
1-8
Thread
Serial
Parallel

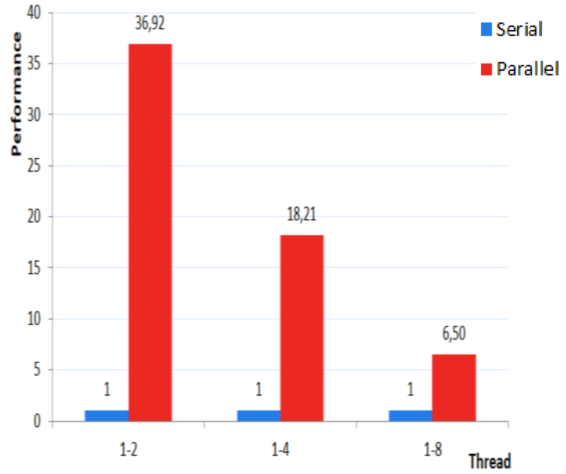
a)


Performance
40
35
30
25
20
15
10
5
0
38,47
19,24
7,93
1
1
1
1-2
1-4
1-8
Thread
Serial
Parallel

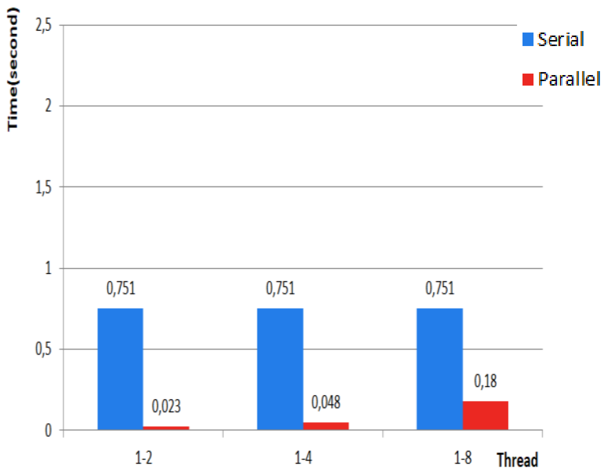
b)


Time(second)
2,5
2
1,5
1
0,5
0
1,366
1,366
1,366
0,037
0,075
0,21
1-2
1-4
1-8
Thread
Serial
Parallel

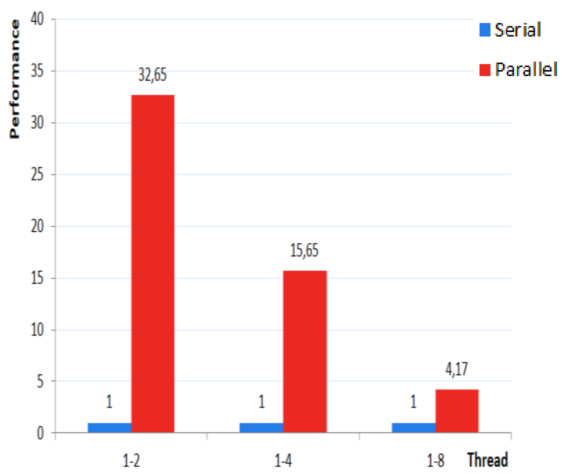
c)


Performance
40
35
30
25
20
15
10
5
0
36,92
18,21
6,50
1
1
1
1-2
1-4
1-8
Thread
Serial
Parallel

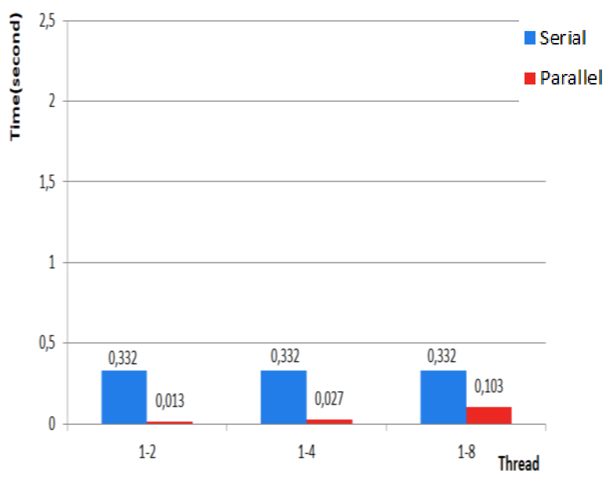
d)


Time(second)
2,5
2
1,5
1
0,5
0
0,751
0,751
0,751
0,023
0,048
0,18
1-2
1-4
1-8
Thread
Serial
Parallel

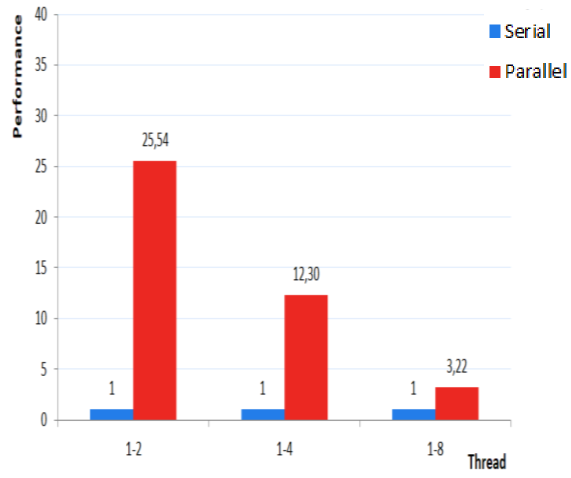
e)


Performance
40
35
30
25
20
15
10
5
0
32,65
15,65
4,17
1
1
1
1-2
1-4
1-8
Thread
Serial
Parallel

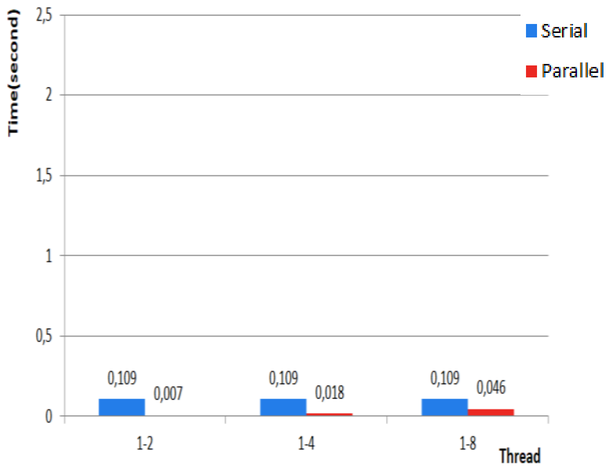
f)



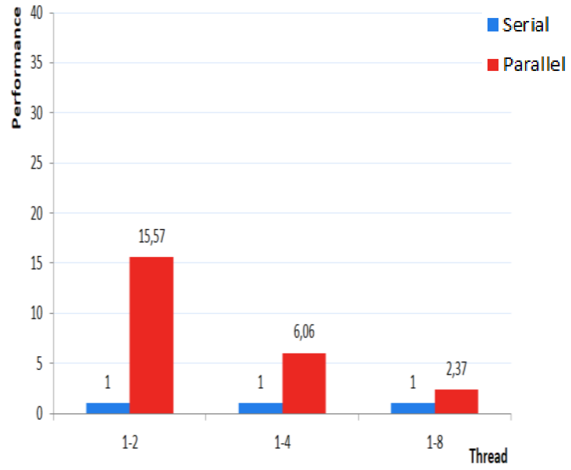
g)



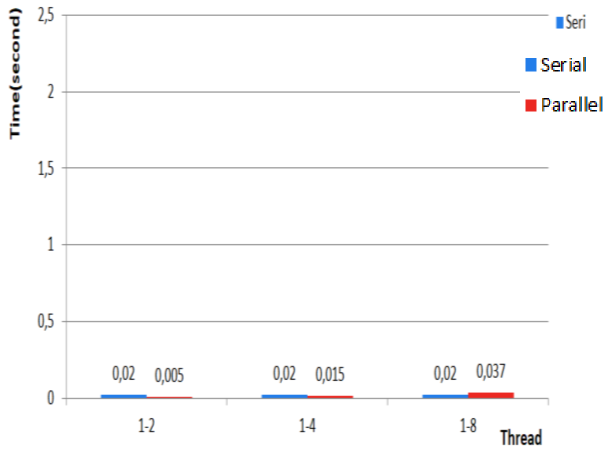
h)



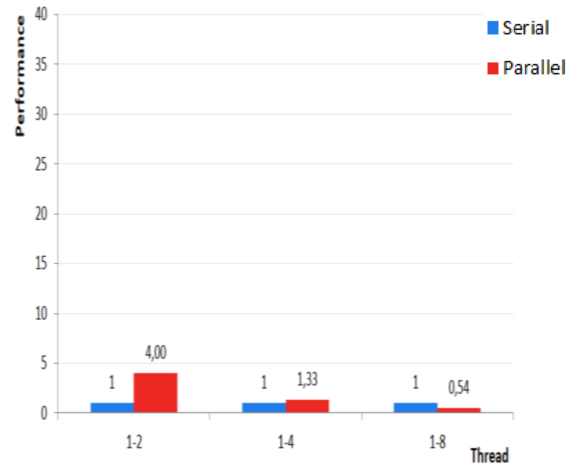
i)



k)



m)



n)

Figure 7. Speed and performance graphics obtained when images with different resolutions are processed on Serial and Parallel systems a)Processing time of 1240x640 resolution image on serial and parallel system b)Performance comparison of 1240x640 resolution image on serial and parallel system c)Processing time of 1024x512 resolution image on serial and parallel system d)Performance comparison of 1024x512 resolution image on serial and parallel system e)Processing time of 768x384 resolution image on serial and parallel system

f)Performance comparison of 768x384 resolution image on serial and parallel system g)Processing time of 512x256 resolution image on serial and parallel system h)Performance comparison of 512x256 resolution image on serial and parallel system i)Processing time of 256x128 resolution image on serial and parallel system k)Performance comparison of 256x128 resolution image on serial and parallel system m)Processing time of 128x64 resolution image on serial and parallel system n)Performance comparison of 128x64 resolution image on serial and parallel system.

According to the results obtained, when the image with a resolution of 1280x640 was performed in 2.113 seconds using serial code but was performed 0.055 seconds on 2 threads, 0.110 seconds on 4 threads and 0.210 seconds on 8 threads using parallel code. When the serial program and the parallel program are compared, 38.47 times faster on 2 threads, 19.24 times higher on 4 threads and 7.93 times faster on 8 threads were obtained.

When the image with a resolution of 1024x512 was performed in 1.366 seconds using serial code but was performed 0.037 seconds on 2 threads, 0.075 seconds on 4 threads and 0.267 seconds on 8 threads using parallel code. When the serial program and the parallel program are compared, 36.92 times faster on 2 threads, 18.21 times higher on 4 threads and 6,60 times faster on 8 threads were obtained.

When the image with a resolution of 768x384 was performed in 0.751 seconds using serial code but was performed 0.023 seconds on 2 threads, 0.048 seconds on 4 threads and 0.180 seconds on 8 threads using parallel code. When the serial program and the parallel program are compared, 3.65 times faster on 2 threads, 15.65 times higher on 4 threads and 4,17 times faster on 8 threads were obtained.

When the image with a resolution of 512x256 was performed in 0.332 seconds using serial code but was performed 0.013 seconds on 2 threads, 0.027 seconds on 4 threads and 0.103 seconds on 8 threads using parallel code. When the serial program and the parallel program are compared, 25.54 times faster on 2 threads, 12.30 times higher on 4 threads and 3.22 times faster on 8 threads were obtained.

When the image with a resolution of 256x128 was performed in 0.109 seconds using serial code but was performed 0.007 seconds on 2 threads, 0.018 seconds on 4 threads and 0.046 seconds on 8 threads using parallel code. When the serial program and the parallel program are compared, 15.57 times faster on 2 threads, 6.06 times higher on 4 threads and 2.37 times faster on 8 threads were obtained.

When the image with a resolution of 128x64 was performed in 0.020 seconds using serial code but was performed 0.005 seconds on 2 threads, 0.015 seconds on 4 threads and 0.037 seconds on 8 threads using parallel code. When the serial program and the parallel program are compared, 4.00 times faster on 2 threads, 1.33 times higher on 4 threads and 0.54 times slower on 8 threads were obtained.

According to the results, the higher the resolution of the image to be processed, the higher the performance of parallel systems created with multiple threads. However, parallel systems can be slower in cases where the resolution of the image to be processed is very low.

4. Result and Discussion

Today, weather forecasts are carried out based on observations. Generally, weather forecasts are made by meteorology experts according to the data obtained from the meteorology stations. Since these data are daily, the estimates are limited. In addition, weather predictions can be made by observing satellite images. However, the interpretation of these satellite images cannot be made completely and precisely by meteorologists. In addition, weather changes can occur in a short time throughout the day. It is also not possible to monitor these changes continuously by experts.

Computer aided image processing systems are used in many areas. Analyzing satellite images using image processing techniques and making predictions with machine learning methods will provide more accurate results. However, these images need to be processed and interpreted quickly. Image processing requires a lot of workload on computer systems. Therefore, image processing processes can take a lot of time. However, when this workload is processed in parallel, it is performed faster in a shorter time.

In this study; parallelization method using threads inside the processor is used for fast processing of satellite images. Satellite images of different resolutions were processed using serial and parallel background subtraction method. The highest resolution image of 1280 x 640 was processed in 0.055 seconds using 2 threads. This result shows that the parallel code processes the image 38.47 times faster than the serial code.

In this study; it is aimed to realize a warning system that automatically, rapidly and continuously analyzes satellite images by using computerized image processing method and thus detects rapid air changes. In addition, according to the results obtained from this study, weather forecasts can be made more accurately by seeing the cloudiness levels and using machine learning methods. In future studies, when high resolution images of shorter time intervals and different locations are obtained, and when machine learning methods are used in the prediction phase, more useful results will be obtained.

Financial Disclosure: No financial support was received in the preparation of this article.

Author contribution: The authors contributed equally to the article.

Conflict of interest statement: As the authors, we declare that there is no conflict of interest.

Ethics committee: The article does not require ethics committee approval.

References

Akçay M, Sen B, Orak, IM, Çelik, A, 2011. Paralel Hesaplama ve CUDA. 6th International Advanced Technologies Symposium 2011 (IATS'11), 16-18 Mayıs, Elazığ.

Akçay M, Erdem HA, 2013. Intel Parallel Studio ile Paralel Hesaplama. XVIII. Türkiye'de İnternet Konferansı, 9-11 Aralık 2013, İstanbul.

Braham M, Piérard S, Droogenbroeck MV, 2017. Semantic background subtraction. 2017 IEEE International Conference on Image Processing (ICIP), 17-20 September, Beijing.

Chen Y, Li W, Wang L, Hu J, Ye M, 2020. Vision-Based Fall Event Detection in Complex Background Using Attention Guided Bi-Directional LSTM. in IEEE Access, 8:161337-161348.

Çelik A, Özmen A, 2009. Dağıtık Paralel Sistemler Hakkında Kıyaslamalı Bir Çalışma: PVM ve MPI. 5. Uluslararası İleri Teknolojiler Sempozyumu (IATS'09), 13-15 Mayıs Karabük.

Deepjoy D, Sarat S, 2014. Implementation And Performance Evaluation Of Background Subtraction Algorithms. International Journal on Computational Science & Applications, 4(2): 49-55.

Freemeteo, 2021. i-weather.com- USA Live Satellite Weather Image, https://i-weather.com/weather/england/live-satellite/?gid=4110001&language=english&country=us-united-states#type=Infrared (21.02.2021).

Google Maps, 2021. USA, https://www.google.com.tr/maps/@36.2115201,-113.7164387,4z (15.03.2021).

Hu L, Ni Q, Yuan F, 2018. Big data oriented novel background subtraction algorithm for urban surveillance systems. in Big Data Mining and Analytics, 1(2):137-145.

Huynh-The T, Hua C, Tu NA, Kim D, 2019. Locally Statistical Dual-Mode Background Subtraction Approach, in IEEE Access, 7: 9769-9782.

Lim G, Kang D, Eom YI, 2020. Thread Evolution Kit for Optimizing Thread Operations on CE/IoT Devices. in IEEE Transactions on Consumer Electronics, 66(4):289-298.

Orak IM, Çelik A, 2017. Üretim Aşamasında Ray ve Profilde Oluşan Kusurların Tespitine Yönelik Bir Paralel Kusur Algılama Algoritması. Gazi Üniversitesi Mühendislik-Mimarlık Fakültesi Dergisi, 32(2):439-448.

Sakpa NS, Sabnis M, 2018. Adaptive Background Subtraction in Images. 2018 International Conference On Advances in Communication and Computing Technology (ICACCT), 8-9 February, Sangamner.

Uçkan T, DAL D, 2016. OpenGL Tabanlı Animasyonlarda Görüntü Kalitesinin CUDA Mimarisi ile Attırılması. Uludağ Üniversitesi Mühendislik Fakültesi Dergisi, 21(1):79-96.

Silberschatz A, Gagne G, Galvin PB, 2012. Operating System Concepts. John Wiley & Sons, Inc. No:9, s. 160-198, Bridgewater, USA.

Wang B, Zhu W, Tang S, Zhao Y, Zou W, 2016. Background subtraction using dual-class backgrounds. 2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV), 13-15 November, Phuket.

Yannick B, Pierre-Marc J, Bruno E, Hélène L, Christophe R, 2010. Comparative study of background subtraction algorithms. Journal of Electronic Imaging, 19(3):1-31.

Zuo J, Jia Z, Yang J, Kasabov N, 2019. Moving Target Detection Based on Improved Gaussian Mixture Background Subtraction in Video Images. in IEEE Access, 7:152612-152623.