AUT Journal of Modeling and Simulation

AUT J. Model. Simul., 54(1) (2022) 3-18
DOI: 10.22060/miscj.2022.20649.5261

Fuzzy Content-Based Image Retrieval Speed-up Using the Multi-Agent Platform

M. Azimi Hemat

Department of Computer Engineering and Information Technology, Payame Noor University, Iran.

ABSTRACT: Parallelization is a technique that increases the speed of tasks by processing them simultaneously. Distributed multi-agent systems are one of the cases in which parallel processing techniques can be used. In this paper, first, the multi-agent model of a fuzzy content-based image retrieval system is designed to distribute it. Next, the corresponding parallel multi-agent model is designed. Afterwards, the parallel image retrieval system is implemented on reconfigurable hardware. This method is based on a multi-agent paradigm which is suitable for various parallelisms within applied problems. In this study, by using parallelism techniques, a method is presented that considerably decreases the image retrieval systems consumption time. This paper focuses on how to implement a fuzzy content-based image retrieval system in the form of a multi-agent model. Since reconfigurable hardware is appropriate to support software agents, I also show how these agents use the inherent parallelism of reconfigurable Hardware for parallel image retrieval and increase the speed of this new system greatly. The two sequential and parallel systems are tested on a data set containing 1000 images. The results signify the increase of almost 3-times speed for the proposed parallel system by software agents, and 400-times speed by hardware agents. In order to evaluate the retrieval efficiency of the proposed system compared to the previous works, two other image retrieval systems have been implemented and the efficiency and memory consumption of the systems have been compared. The results indicate better performance of the proposed system than other methods studied.

Review History:

Received: Sep. 09, 2021
Revised: Jul. 07, 2022
Accepted: Jul. 05, 2022
Available Online: Sep. 01, 2022

Keywords:

Image Retrieval
Multi-Agents
Parallelism
reconfigurable hardware

1- Introduction

Since 1970, image retrieval is an active field of research, and researchers in two major areas are working in this field: database management and machine vision. The view of the first group is Text-based and the vision of the second group is based on visual characteristics. Text-based images retrieval began in early 1970, in which a general framework of image retrieval was presented first by indexing images with keywords, then by using database management systems for image retrieval.

There are two major problems in this kind of retrieval, especially when the size of the database is large. The first is the time spent on indexing and the second is the large size of images and the different human perceptions of the same images. In the early 1990, due to the rapid increase in high-volume collections of images and the lack of accountability of systems based on text, image retrieval based on visual features were introduced [1,2].

In content-based image retrieval, the user describes an arbitrary image in the form of desired visual features, and the image retrieval system retrieves the closest image to what the user describes. Generally, visual features are of a few tens or hundreds of orders, thus in the large database, searching in low-level features to find the nearest images would be very time-consuming. With growing database images, database search time increases linearly and at large databases, spending this time makes the system slow. Researchers in this field have used multidimensional techniques and Quick indexing to reduce the dimension of feature vectors.

Ponomarev et al. have proposed a new CBIR system based on the integration of color, texture, and shape [3]. To extract these 3 features, automatic color correlation, Gabor transforms, and wavelet transform was used. The measure to calculate the similarity between the query image and the data set images is Manhattan distance. The mean values for Corel, Li, and Caltech 101 datasets were 0.8300, 0.8800, and 0.7000, respectively. The system also had drawbacks, including increased computational complexity due to the combination of several features.

Image analysis at a resolution level may lose valuable detail. Therefore, Srivastava and Khare proposed a new method [4]. This method is based on extracting texture and shape features using a Local Binary Pattern descriptor (LBP) to extract texture features and Legendre moments to extract

*Corresponding author's email: m.azimihemat@pnu.ac.ir

shape features from texture features at multi-resolution levels. Although LBP is used to extract local attributes, when local attributes are combined with global attributes, they also create an effective attribute vector. Their technique was tested against five datasets, which improved accuracy and sensitivity, but achieved higher resolution with increasing computational cost due to the use of multi-analysis.

Sajjad et al. have proposed an invariant CBIR system for texture rotation and color change [5]. The proposed system is based on the integration of colors and textures in the form of a feature vector with a size of 360. The extraction of color features is done by converting images to HSV color space and quantizing through the color histogram. Illumination changes are a problem in image retrieval systems. To solve this problem, only Hue and Saturation channels are used. Rotate Local Binary Pattern (RLBP) is used to extract fixed texture features. Their system is evaluated through implementation on the Zurich Building (ZB), 1K Corel and 10K Corel.

A multi-stage CBIR technique was introduced by Pavithra and Sharmila [6]. In the first step, the color feature was extracted using color moment, which reduces the cost of calculations. In the second step, texture and edge features are extracted from the images in the new data set created from the first step. LBP was used to extract texture information, while the Canny Edge Detector was used to extract edge information. Although the proposed multistage system improves performance by increasing accuracy and decreasing runtime, the runtime required depends on the number of images in the data set.

Pavithra and Sharmila also proposed a new method for color-based dominant image retrieval [7]. Four image datasets were used in the experiments to evaluate the proposed dominant color descriptor. To improve the proposed method, it should be combined with other feature extraction methods (shape, texture, and spatial information) to reduce the semantic gap.

Ashraf et al. developed a subjective approach to the CBIR system based on a fusion of low-level features (texture and color) [8]. Color moments in the HSV color space are used to extract color features, and DWT and Gabor wavelet are used to extract texture features. In the field of CBIR, local structure descriptors have been proposed to represent the local spatial structure of information, which makes these descriptors more semantic. MTSD is a new description for CBIR, which uses local and multi-trend structures [9]. It is based on the integration of edge, color, and intensity information. Using several trends, it identifies information about both the local spatial structure and low-level features (i.e. color, shape, and texture). The proposed descriptor is evaluated against the Caltech and Corel datasets (1K, 5K, and 10K), and the results show that the proposed descriptor performs better than many advanced descriptors. The disadvantage of this descriptor is that it does not describe the correlation between local spatial information, intensity, texture, and color [10].

In this paper, a different method is presented for decreasing the effect of the number of feature vectors. In this method, I design a new distributed image retrieval system using parallelism techniques.

In the past, large computers with multiple processors were used in large parallel processing systems. Today, with the advent of intelligent systems, multi-agent systems provide a suitable and general template for modular parallel design. Using multi-agent systems, the complex problems are broken into easier and smaller components and implemented by different agents. So far, many researchers have tried to design fast image retrieval systems using parallel and multi-agent techniques. In [11], a parallel content-based image retrieval system using spark and tachyon frameworks is presented. This system is composed of two steps: (i) Image indexing step, in which using Map Reduce distributed model on Spark in order to speed up the indexation process. Additionally, a memory-centric distributed storage system, called Tachyon, is using to enhance the write operation. (ii) Image retrieving step which speeds up by using a parallel k-Nearest Neighbors (k-NN) search method based on Map Reduce model implemented under Apache Spark. In [12], a content-based Image Retrieval through a Multi-Agent Meta-Learning Framework is designed. In this paper, viewing specialized image retrieval algorithms as agents, a general-purpose image retrieval system that uses a new multi-agent meta-learning framework is proposed. In [13], a parallel method in graphics processing unit was presented for image indexing referred as plane semantic ball.

The parallel and multi-agent techniques are used in many purposes to design faster systems. For example, in a paper, a parallel multi-agent real-coded genetic algorithm for large-scale black-box single-objective optimization is proposed [14]. The writers claimed that the individualization of heuristic operators at the level of agent-processes that implement independent evolutionary searches facilitates the improved likelihood of obtaining the best solutions in the fastest time. Based on this property, a parallel multi-agent single-objective real-coded genetic algorithm for large-scale constrained black-box single-objective optimizations (*LSOP s*) is proposed. In [15], a multi-agent based model with parallel computing to discrete-event simulations for metro train operation under emergencies is presented. In this research a discrete-event simulation method based on a multi-agent model with parallel computing is proposed to estimate the effects of emergencies efficiently. Additionally, the parallel multi-agent platform is used for hierarchical path finding in [16]. In this paper, two approaches to solve the HNA* bottleneck is present, and thus obtain a performance boost for all hierarchical configurations. The first method relies on further memory storage, and the second uses parallelism on the GPU. The comparative evaluation shows that both approaches offer speed-ups as high as 9x faster than A*, and show no limitations based on hierarchical configuration.

The fast image retrieval using various methods is of interest to researchers in many other papers. In [17], a speedup procedure for image retrieval systems was proposed on RDISK machine. In [18], an efficient CBIR system based on speeded up robust features referred as SURF is presented, followed by an optimization method. The trends of recent

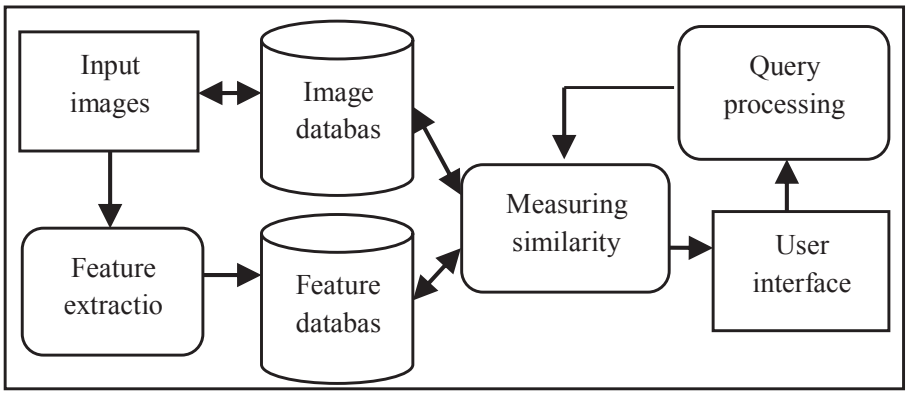


Fig. 1. Content-Based Image Retrieval system

image retrieval research concentrate on the use of deep learning to improve accuracy at the cost of increasing running time [19].

In [20], a fast solution for Manifold-Ranking (MR), a ranking method for CBIR systems, was introduced that exploited two important properties shared by many real graphs, including linear correlations and block-wise community-like structure. In [21], a fast calculation method of cosine similarity with L2 norm indexed in advance on Elastic search was presented for CBIR systems constructed using CNNs. In [22], a fast and efficient image retrieval scheme was proposed for searching images among JPEG2000 compressed image databases. In [23], a CBIR approach was presented to solve high computational time, handling high dimension data, and comparing images consistent with human perception. In [24], an efficient and effective CBIR technique was applied directly to the compressed domain, and thus did not need full decompression for feature extraction. In [25], a fast retrieval scheme was designed for big data applications such as images to be retrieved from large image databases. It used reasonable elements ranking, and appropriate distance metric to decrease retrieval time.

In [26], a fast image retrieval procedure was presented by classifying image features into different levels. Levels are considered as features and retrieval is done by similarity comparison between query images and database images. In [27], a fast and simple content access method and retrieving JPEG images with DCT coefficients of coded blocks were presented without needing to complete decompression of coded images. In [28], a fast medical image retrieval system based on wavelet features and image signature calculated using Kurtosis and standard deviation were proposed to help physicians in medical images analysis and identification.

In [29], the indexing method for color images was proposed with Error Diffusion Block Truncation Coding (EDBTC) feature extraction followed by an unsupervised clustering to decrease the required time for comparing the target and query image. In [30], a fast CBIR system was introduced with a Bayesian logistic regression model. This model was used to compute pseudo-metric weights and led to increasing discriminatory capacity and retrieval accuracy. In [31], a novel and fast CBIR model was proposed based on Dual-Cross Patterns (DCP). These patterns encoded second-order information of the local surrounding region of every center pixel in the vertical, horizontal, and diagonal directions.

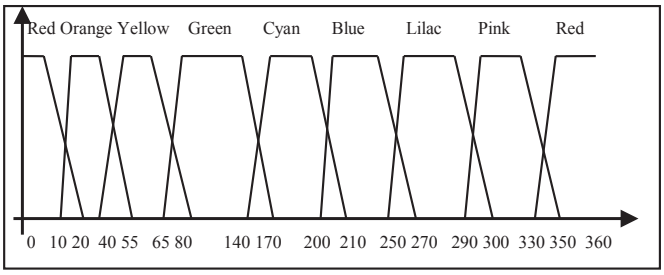
Additionally, with the emersion and development of reconfigurable hardware, such as FPGA, it is possible that flexibility which has been a property of software in the past also be applied on parallel and high-speed hardware. In this research, I combine multi-agent technology and reconfigurable hardware to design a parallel system. In the following, a fuzzy image retrieval system based on the content of the image is presented in section 2. In section 3, multi-agent systems, and in section 4, reconfigurable hardware are explained. Parallel processing using Agents is explained in section 5. Section 6 with 2 sub-sections, explains the proposed method. Finally, in implementation results the software and hardware agent speed-up are calculated.

2- Fuzzy Content-Based Image Retrieval

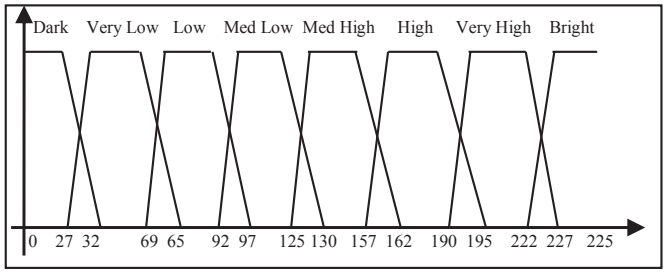
The structure of a content-based image retrieval system is shown in Figure 1. The image database is including images among which retrieval is done.

Various visual features extracted from these images and features are saved on the feature database. Query processing subsystem extracts the query image features and sends it to the similarity measurement unit. In this unit, the similarity of query image features with features of each image in the database is calculatedو and most similar images to query image are determined and sent to the user interface.

Feature extraction is the key part of a content-based image retrieval system. The main features are divided into two general and specific categories. General features usually include color, texture, shape, and spatial relationships. These features that is applied for different images in a general retrieval system are not so applicable. General features that are divided into four general categories include color, texture,



(A)



(B)

Fig. 2. (A). fuzzy partitioning of hue component. (B). fuzzy partitioning of intensity component

shape, and color layout.

In this paper, a method is presented based on the color features, and HSI (Hue-Saturation- Intensity) color space is used instead of RGB (Red-Green-Blue) which is usually used for color image modeling. My reason to use HSI color space is its ability to design a retrieval system using only two Hue and Intensity components of space instead of using all three components of RGB space. Therefore, I reduce the dimensions of feature vectors, and then the time consumed by the system is reduced.

The HIS color space describes the color intuitively and is based on direct perception [32, 33]. The hue component describes the color according to wavelength. The saturation component is the color value that is displayed. This component describes the differences between each specific color with white light. The intensity component shows the amount of brightness of color. HSI color space can be modeled to a cylindrical coordinates. The hue component is represented with an angle that is variable from 0 to 360 degree. The saturation component is also by the radius of the cylinder which is variable from 0 to 1 (or from 0 to 255). The intensity component along the z-axis can change, and therefore 0 is black and one (or 255) is white.

In the following, I first describe the fuzzy modeling of color images for indexing them, and then investigate the similarity of these fuzzy feature vectors.

2- 1- Fuzzy Color Feature Extraction (Indexing Phase)

In this study, I divide the Hue and Intensity components of the HSI color space by the trapezoidal fuzzy functions [34] to 8 sections. These membership functions are shown in Figure 2.

In this modeling, the trapezoidal fuzzy numbers are used. A trapezoidal fuzzy number A [35], is a fuzzy set with a trapezoidal membership function. The trapezoidal membership function is usually depending on four scalar parameters a, b, c and d, as follow:

$$\mu_A(x;a,b,c,d)=\begin{cases}0 & if\ x\le a,\\ \dfrac{x-a}{b-a} & if\ a<x<b,\\ 1 & if\ b\le x\le c,\\ \dfrac{d-x}{d-c} & if\ c<x<d\\ 0 & if\ d\le x\end{cases} \quad (1)$$

First, to compute the Hue image feature vector, according to the 8 colors that are modeled in Figure 2a, the membership degree of all image pixels is calculated. For image A_j and a pixel p this fuzzy number is $\mu_{A_j}^{p}$:

$\mu_{A_j}^{p} = (c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8)$ Where each numerical c_i in [0, 1] is the membership degree of pixel p in hue segment i where i= 1 to 8.

The average of these fuzzy numbers is a fuzzy number with 8-length that called $\mu_{A_j}^{H}$, which is the Hue feature vector.

To extract the feature vector of the intensity component of the image, the membership degrees of each pixel in the 8 functions are shown in Figures 2b.

Other calculations are done like the Hue component. In this way, I obtain a fuzzy number for the Intensity component. For an image A_j and an intensity value I, the obtained fuzzy number are called $\mu_{A_j}^{I}$.

2- 2- Fuzzy Retrieval (Determining the Similarity of Two Images)

In the indexing phase, two fuzzy feature vectors with length 8 are extracted for each image. In the retrieval step, the main problem is determining the similarity of two images (two feature vectors). In this paper, as the feature vector of images is extracted in the form of fuzzy sets, a fuzzy similarity measure is used. The descriptions and types of fuzzy similarities are mentioned in [36]. Here, the similarity measure separately using for each feature, and finally the results is combined to calculate the overall similarity of the two images.

To calculate the similarity of the Hue component, two images A_0 and A_j are utilized in a color c, using equation (2). As mentioned above, this relationship is a fuzzy similarity measure. Here A_0 is the query image and A_j is an image in the dataset.

$$S\left(\mu_{A_0}^{c} . \mu_{A_j}^{c}\right) = \frac{\left|\mu_{A_0}^{c} \cap \mu_{A_j}^{c}\right|}{\left|\mu_{A_0}^{c} \cup \mu_{A_j}^{c}\right|} = \frac{\sum_{x \in S_c} \min\left(\mu_{A_0}^{c}(x) . \mu_{A_j}^{c}(x)\right)}{\sum_{x \in S_c} \max\left(\mu_{A_0}^{c}(x) . \mu_{A_j}^{c}(x)\right)} \quad (2)$$

In this regard, S_c is the same fuzzy function range for the color c.

I have obtained eight numbers and merge them as follow to calculate the overall similarity of two images for the components of hue:

$$S^{h}\left(A_0, A_j\right) = \frac{\sum_{c} S\left(\mu_{A_0}^{c}, \mu_{A_j}^{c}\right)}{8} \quad (3)$$

The average similarity of two images A_0 and A_j in Intensity component is $S^{I}\left(A_0, A_j\right)$, which is calculated with the equation 2 and 3.

The overall similarity of two images is obtained in terms of the similarity of two images in hue and intensity components, as follows:

$$S\left(A_0, A_j\right) = \frac{S^{h}\left(A_0, A_j\right) + S^{i}\left(A_0, A_j\right)}{2} \quad (4)$$

3- Multi-Agent Systems

An agent is an autonomous and independent entity that can process. It receives data from the environment through sensors, does the work according to its program in the environment by using some actuators [37]. Moreover, these agents can achieve their goals. They have interaction with the environment and other agents in the environment.

Agents should not be confused with objects or entities. An agent, especially an intelligent agent, is more than one object. An agent, unlike an object, is autonomous. It means that the autonomous agent follows series of actions that should reach its goals, and is capable of making decisions independently. The agents are an integral component in processing environments. They receive information from their environment continuously and always carry out the actions whose result is reforming and changing the environment to achieve their goals.

Intelligent agents have an important feature, adaptability to the environment. They can contribute with other agents to increase the probability of events that bring them to their goals continually. In a traditional software environment, multi-agent structures are used as a method for implementing adaptable and flexible modular systems [38]. In such systems, software agents are highly autonomous and they react based on received input and often are capable of learning and inference.

Most of these agents have been designed to react to unpredictable events and they are doing this action by comparing their operation with environmental changes [39,40]. Such intelligent software agents have their own goals and have the ability to do a series of actions independently to reach these goals. They can observe the environment and use the perimeter events to learn, inference and decision making.

In addition, these agents are designed some way that they can work together. In other words, they can cooperate with other agents. Often, this cooperation is based on breaking large and complex problems into smaller and simpler tasks. Each of the agents can do these tasks easily and quickly. As mentioned before, a multi-agent system is a distributed system in which several agents are responsible for doing their duty. In the past, implementation of such agent-based systems only has been limited to the software implementation, and it has been done in the old distributed processing environment.

The complexity of agent-based systems and lack of flexibility of hardware systems prevent direct agent implementation in hardware. Recently, advances happen in hardware with the ability of re-configuration, such as Field Programmable Gate Arrays (FPGA) [41-43]. These mutations improve the capabilities and capacity of the hardware in such

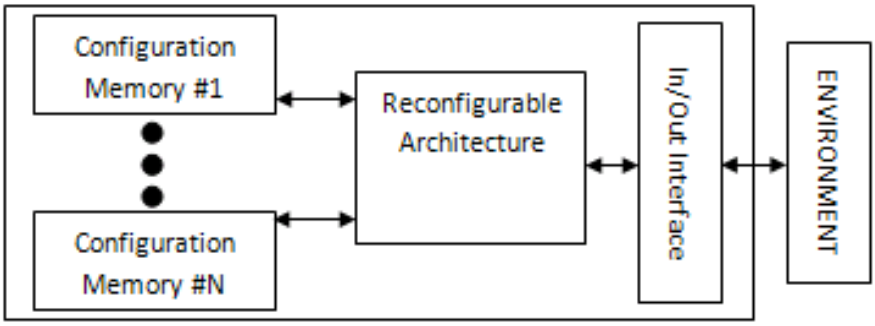


Fig. 3. Reconfigurable hardware general environment [44]

a manner that one or more processing elements are placed in them. This means that I have no structural constraints for implementing multi-agent systems in the reconfigure hardware [44]. These new agents, which can be implemented in reconfigurable hardware, are called hardware agents [45].

Hardware agents vary from old software agents that generally are placed in a computer program memory (RAM) and can be implemented by using a microprocessor. In hardware designing of environments in which tasks are divided among agents, object-oriented languages, such as VHDL, should be used to describe the hardware.

4- Reconfigurable Hardware

In the past, agents only were implemented in software because the software is flexible and is easily implemented. Today, FPGA technology and other reconfigurable hardware are improved, and this allows us to enjoy the flexibility of hardware [44,45]. So far, this property was limited to software.

A major advantage for implementing multi-agent systems in hardware is the production of systems in which agents work in parallel together and economize in system time. Furthermore, with this hardware reconfiguring, a piece of hardware at different times can have different functions. In other words, one part of hardware from beginning to the end of system life is not limited to just one agent and one function. This also leads to saving hardware resources.

I can change the configuration of each hardware section in different ways and at different times. In some cases, the hardware configuration is done before implementing the system as static and remains unchanged until the system works.

In other cases, it is possible to configure the hardware dynamically when the system is working. In other words, the hardware designing may be changed in response to the system demanding at the time of its execution, and the adaption of the system to the environment changes or changing the conditions of the system itself. In both cases, the reconfiguration logic acts similar to an executive engine for different hardware functions. Some functions work in parallel and others are consecutive.

In some systems, to provide a balance between performance and limited hardware resources, I can use the combination of hardware and software agents in the system. As mentioned, the hardware agents have capabilities that allow them to communicate with other hardware and software agents. In Figure 3, an overview of a reconfigurable hardware environment is shown [44]. In such a structure, the system designer controls hardware performance by placing data that is related to system designing directly into configuration memory.

In this environment, the reconfigurable architecture gives the power of dynamic change of operations to the system via the production of new agents into the configuration memory. Also reconfigurable logic employs partial reconfiguration. In a partial re-configuration, different parts of the logic can be changed without affecting other sections. In each system, some input/output connections support interaction with the environment. These are directly connected with configurable logic and let the reconfigurable logic directly controls high-speed sensor and actuator operations without processor intermediacy.

5- Parallel Processing Using Agents

In distributed systems processes are assigned to different systems to run. In this research, I use hardware agents to provide a distributed system. One of the main advantages of distributed systems is their high speed which can be achieved by performing tasks in parallel. The first step of processing tasks within a distributed system is to choose the method of parallelism. In general, three methods for basic parallelism can be used (i.e. tasks parallelism, data parallelism, pipelining), and the combination of these three methods. These methods are shown in Figure 4 [46].

In the next section, I describe the image retrieval multi-agent model using these three parallelism methods.

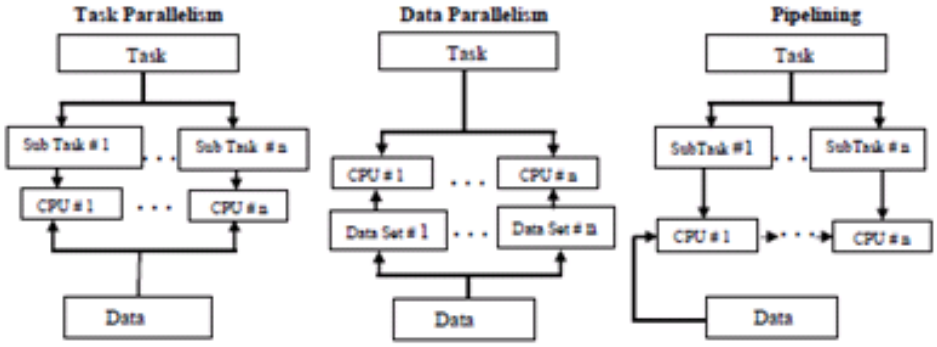


Fig. 4. Parallel processing methods [46]

Table 1. The time required to run the four proposed models

The time of indexing phase	T_1 = Sequential model time = **[100000 × A] + 16 B**
	T_2 = Parallel model time = **[250 × P($A1_H$, $A2_H$, $A1_I$, $A2_I$)] + [8× p (B_H , B_I)]**
The time of retrieving phase	T_3 = Sequential model time = **[1000 × C] + D**
	T_4 = Parallel model time = **[333 × P(C1, C2, C3)] + D**

6- Multi-Agent Image Retrieval Model (Sequential and Parallel)

In the system that was introduced in section 2, in the indexing phase, after receiving the three components RGB of color space for each pixel, the Hue and Intensity components for each pixel are calculated. Afterwards, two fuzzy numbers for these components for each picture are extracted. The sequential multi-agent model of this phase is shown in Figure 5a.

In this phase, the hue and intensity components are calculated separately and those fuzzy feature vectors are extracted separately. These operations are independent of each other and can be done in parallel (task parallelism). Additionally, half of pixels can be processed in parallel to other half (data parallelism). The parallel multi-agent model of this phase is shown in Figure 5b. In this model, agent A_H performs the same operation as agent A to calculate Hue component, and agents $A1_H$ and $A2_H$ each work on half of the pixels. This explanation also applies to other factors.

In these figures, the data are in rectangle and the agents are in the circle and the agent name is written beside it.

In the retrieving phase, the similarity of the query image feature vector with the feature vector of all database images must be calculated. The sequential model for the retrieving phase is shown in Figure 6a. In most cases, the number of database images is very large and data parallelism is used. The parallel model of this phase is also shown in Figure 6b. Agents C1, C2, and C3 are all from the agent C category.

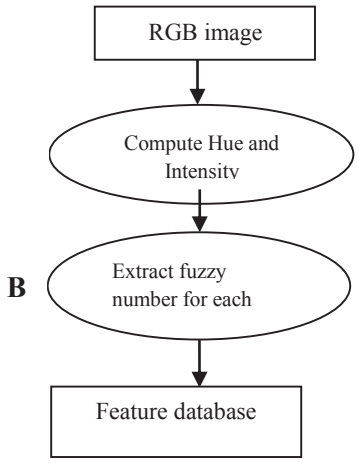
7- Implementation Results

In order to compare the proposed parallel method with the sequential method, each two system were implemented and the processing time of them is calculated. This processing time is the time which systems are required to retrieve images a database of 1000 images taken from the Corel collection. These images are arranged in 10 different semantic groups: people, lions, elephants, horses, flowers, foods, mountains, monuments, interior design and buses. All images are in JPEG format and of sizes 256*384 and 384*256 [47].

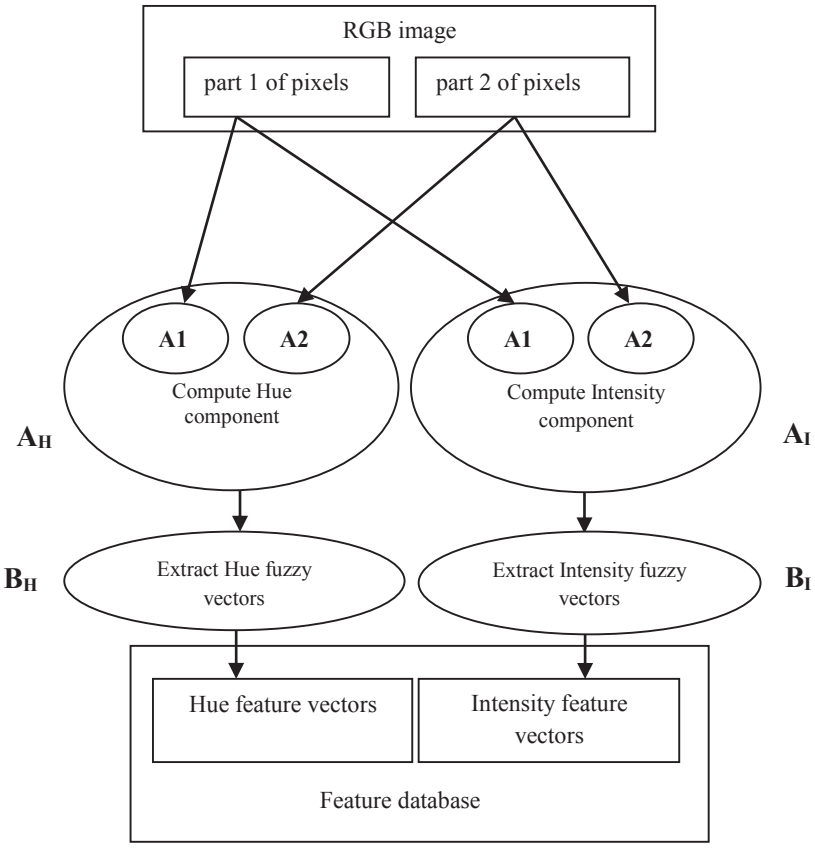
The methods of calculating the consumption time of the systems are shown in Table 1. In this table the time which is required for run the agent X, is written X and time of run parallel agents X and Y is written p (X,Y) briefly.

In this table, 100,000 is the number of image pixels and 16 is the length of fuzzy number which must be calculated. 25,000 is a quarter and 8 is the half of those numbers. 1000 is the number of image database and 333 is one third of them.

In the end, increasing the speed of proposed parallel

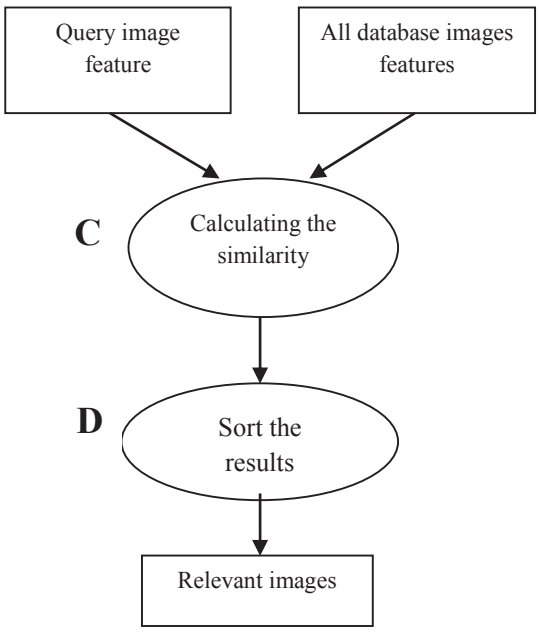


(a)

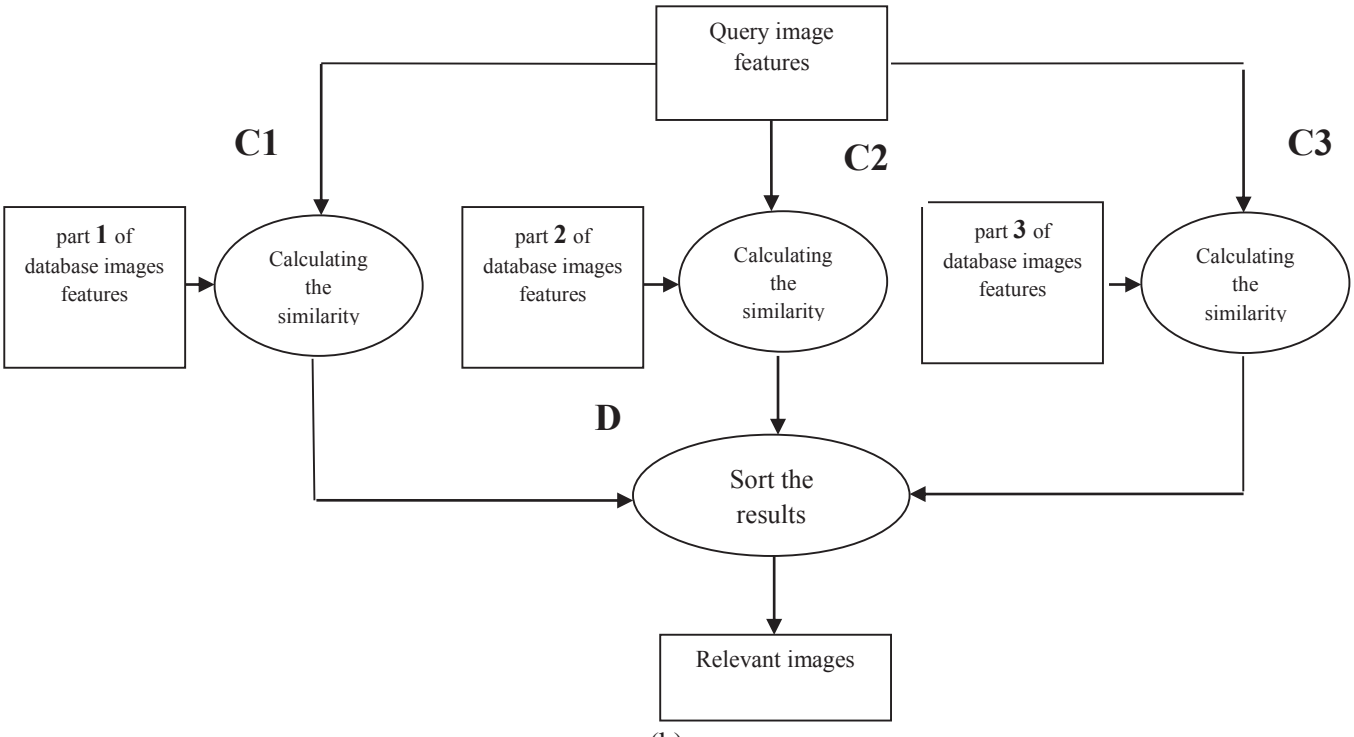


(b)

Fig. 5. (a). Sequential multi-agent model for indexing phase (b). Parallel multi-agent model for indexing phase



(a)



(b)

Fig. 6. (a). sequential multi-agent model for retrieving phase. (b). parallel multi-agent model for retrieving phase

Table 2. Results of software implementation

Agent	A	B	P ($A1_H$, $A2_H$, $A1_I$, $A2_I$)	P (B_H , B_I)	C	D	P(C1, C2, C3)
Time (ms)	10	1000	13	1200	25	15	30

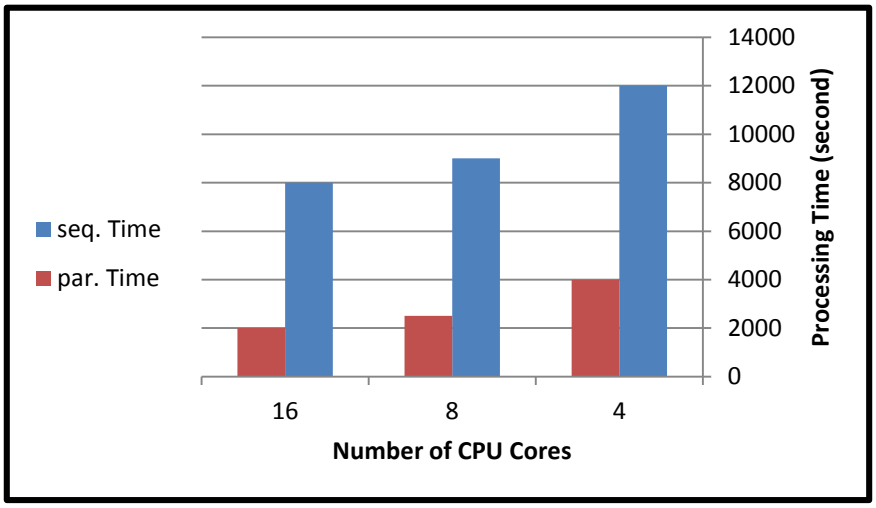


Fig. 7. The time of two parallel and sequential processing using various cpu

system is calculated as following (5):

$$Speed\ up = \frac{sequentiol\ time}{paralel\ time} = \frac{T_1+T_3}{T_2+T_4} \qquad (5)$$

7- 1- Software Implementation Results

Systems were implemented using MATLAB and C# programming environments in a four-core system with a 3 GHz hour pulse. To measure the performance of each system, all 1000 selected Corel database images are indexed and their features are stored in a separate database. In the next step, which is the retrieving phase, images are randomly selected as the query image, and similar images are retrieved. I execute the program in two parallel and sequential methods (by using threads) and compute the consumed time of different parts. This operation was performed for 10 semantic groups of the Corel dataset, and the average of the obtained results is shown in Table 2.

Speed up = 3.

This means parallel implementation is almost three times faster. Due to the effect of the number of CPU cores on the power of the computer for parallel processing, the systems implemented on computers with different cores were tested. The results are shown in Figure 7. As can be seen from the results, the parallel processing power of processors has a direct impact on the performance of parallel multi-agent systems.

7- 2- Hardware Implementation Results

To increase the speed of the software multi-agent system, I have implemented the multi-agent system on reconfigurable hardware. For hardware implementation, I first simulate agents that were implemented in the previous section, by hardware description languages like VHDL. After this primary design, I am using synthesis tools to convert high-level hardware descriptions created by VHDL to a low-level bit string. The result of hardware implementation is shown in table 3.

In this table the speed up is: $\frac{parallel\ \ software\ agent\ time}{parallel\ \ hardware\ agent\ time}$.

Table 3. Results of hardware implementation

FPGA type	Vertex 2	Vertex 4	Spartan 3
Device family	Xc2v4000	Xc4vfx100	Xc3s5000
P ($A1_H$, $A2_H$, $A1_I$, $A2_I$) (ns)	350	275	445
P (B_H , B_I) (ns)	149085	102327	175719
P (C1, C2, C3) (ns)	1465	1085	1795
D (ns)	805	495	905
Speed up	**4×10^2**	**5×10^2**	**3×10^2**

7- 3- Efficiency and Space Evaluation Results

To compare this proposed system with the previous works, this paper and two other papers are implemented and tested on a database of 1000 images taken from the Corel collection [7, 48].

Since this proposed method is fuzzy and based on color content, we compared it with non-fuzzy methods based on color content to prove the superiority of the proposed method. Pavithra et al. [7] proposed a method based on color content that is based on dominant color descriptors. Here, similar colors are clustered and the average value of each cluster represents the dominant color. In our proposed method, the color content is modeled based on the predominant fuzzy colors, and in experiments, better results are obtained in terms of retrieval efficiency. Additionally, the length of the extracted feature vector is less than the mentioned method.

The combination of color and other content features of the image usually produced better results than using only the color feature [48]. In this paper, Jun et al. present a method based on color and texture features. In this method, color histograms and texture properties are extracted based on a concurrency matrix to form feature vectors. Color histogram and texture features are then analyzed. This method acts better than the reference [7]. Due to the different texture of different samples of images in some semantic groups of query, results close to the proposed method were obtained, but due to the comprehensiveness of fuzzy sets, the average results show better performance of the proposed method than it. Additionally, in terms of the length of the extracted feature vector, the power of the fuzzy method is observed.

To measure the performance of each system, all 1000 selected Corel database images are indexed, and their features are stored in a separate database. In the retrieval phase, images are randomly selected as the query images, and similar images are retrieved. The query images are the same for comparing all three image retrieval systems.

The most common evaluation measures are different types of precision and recall. In this paper, we used the efficiency measure presented in [49]. If the number of retrieved images is lower than the number of relevant images, retrieval efficiency represents the precision, otherwise the recall (6).

$$\left(\begin{matrix}\text{Retrieval}\\ \text{Efficiency}\end{matrix}\right) = \begin{cases} \dfrac{\text{No. of relevant images retrieved}}{\text{Total no. of images retrieved}}; & \\ \text{If No. of retrieved images} < \text{No. of relevant images} & \\ \dfrac{\text{No. of relevant images retrieved}}{\text{Total No. of relevant images}}; & \\ \text{Otherwise} & \end{cases} \quad (6)$$

The considered query set includes 500 images of 10 various groups. In order to test the proposed system, it is compared with two other systems. The obtained results are tabulated in table 4 and graphed in figure 8, where the average retrieval efficiencies for 500 queries for the 1, 5, 10, 20, 30, 40, 50, 60, 70, 80, 90, and 100 first retrieved images are reported. Figure 8 shows that the proposed method makes better results than other systems. The retrieval results for one sample queries are given in figure 9, where the most similar images retrieved are presented.

As to the performance of the retrieval systems, the storage space for the feature vectors is of concern. When all images in a dataset are indexed, the extracted feature vectors should be stored in a separate database. One of the advantages of indexing images in the form of fuzzy sets is a significant decrease in the dimension of the feature vector. The lengths of the extracted vectors of the three systems are tabulated in Table 5, where the proposed system requires less storage space than other systems. When the count of the images in the database is high, the value of this reduction will be greater as well.

Table 4. Average retrieval efficiency versus the number of images retrieved, computed over 500 queries.

Number of retrieved images	Pavithra L. K. [7]	Jun Y. [48]	Proposed system
1	0.823	0.901	0.920
5	0.801	0.873	0.912
10	0.789	0.832	0.901
20	0.734	0.801	0.887
30	0.705	0.770	0.843
40	0.687	0.760	0.812
50	0.659	0.712	0.800
60	0.636	0.684	0.785
70	0.611	0.645	0.743
80	0.588	0.612	0.729
90	0.579	0.601	0.703
100	0.550	0.587	0.690

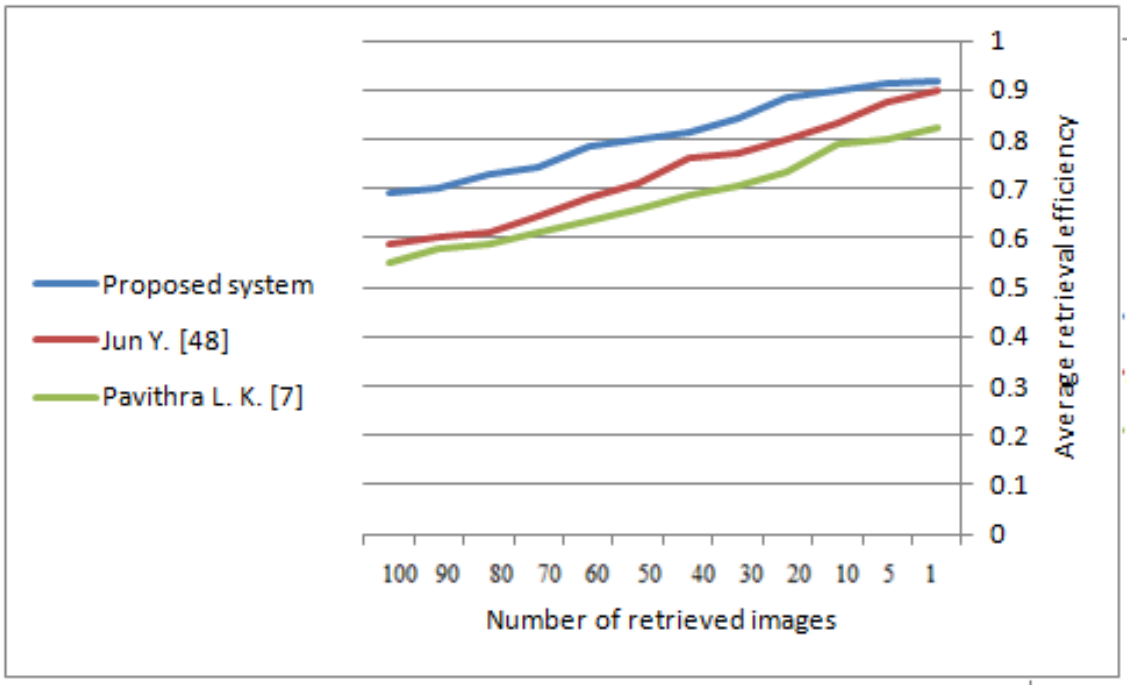


Fig. 8. The graph in terms of the number of retrieved images for the 3 systems examined

Fig. 9. Sample query: The image on the top left is the query. Ordered from left to right and top to bottom are the images retrieved

8- Conclusions

Parallelization is a technique that increases the speed of tasks by processing tasks simultaneously. Distributed systems are one of the cases in which parallel processing techniques can be used. Reconfigurable hardware is also a strong platform for implementing distributed systems.

In this article, an image retrieval system is upgraded. First, the multi-agent model of this system is designed to distribute the system. Afterwards, the corresponding parallel multi-agent model is designed. These multi-agent models were implemented on the software and the speed-up was obtained about 4. Next, by implementing these agents on the reconfigurable hardware, the speed-up was obtained about 400 times more than the software model, which indicates the high power of the reconfigurable hardware. In this way, the challenge of time-consuming image retrieval systems is solved.

In addition to increasing the speed, analyzing the retrieval efficiency and storage space required for the proposed system is also important. For this reason, these issues have been studied in comparison with previous works, and the results indicate that our system performs better than the systems studied.

I applied this method to a simple fuzzy image retrieval system, while this method can be generalized to other image retrieval methods and is an idea for all researchers in this field.

References

[1] Y.Rui and T.S.Huang, 1999, Image retrieval: current technique promising directions and open issuesî , Journal of Visual Communication and Image Representation, vol.10, pp.39-62.

[2] Y.Li, X.Wan and C.C.J.Kuo, 2001, Introduction to content-based image retrieval-overview of key techniquesî , in Image Database: Search and Retrieval of Digital Imagery, Edited by Bergman and Castelli, John Wiley & Sons.

[3] Ponomarev, A., Nalamwar, H. S., Babakov, I., Parkhi, C. S., & Buddhawar, G. (2016, February). Content-based image retrieval using color, texture and shape features. Key Engineering Materials, 685, 872–876

[4] Srivastava, P., & Khare, A. (2017, January). Integration of wavelet transform, Local Binary Patterns and moments for content-based image retrieval. Journal of Visual Communication and Image Representation, 42, 78–103.

[5] Sajjad, M., Ullah, A., Ahmad, J., Abbas, N., Rho, S., & Baik, S. W. (2018, February). Integrating salient colors with rotational invariant texture features for image representation in retrieval systems. Multimedia Tools and Applications, 77(4), 4769–4789.

[6] Pavithra, L. K., & Sharmila, T. S. (2018, August). An efficient framework for image retrieval using olor, texture and edge features. Computers & Electrical Engineering, 70, 580–593.

[7] Pavithra, L. K., & Sree Sharmila, T. (2019, December).

An efficient seed points selection approach in dominant color descriptors (DCD). Cluster Computing, 22(4), 1225–1240.

[8] Ashraf, R., Ahmed, M., Ahmad, U., Habib, M. A., Jabbar, S., & Naseer, K. (2020, April). MDCBIR-MF: Multimedia data for content-based image retrieval by using multiple features. Multimedia Tools and Applications, 79(13–14), 8553–8579.

[9] Zhao, M., Zhang, H., & Sun, J. (2016, July). A novel image retrieval method based on multi-trend structure descriptor. Journal of Visual Communication and Image Representation, 38, 73–81.

[10] Raza, A., Dawood, H., Dawood, H., Shabbir, S., Mehboob, R., & Banjar, A. (2018). Correlated primary visual texton histogram features for content base image retrieval. IEEE Access, 6, 46595–46616.

[11] S. Mezzoudj, A. Behloul, R. Seghir, Y. Saadna , A parallel content-based image retrieval system using spark and tachyon frameworks , Journal of King Saud University – Computer and Information Sciences 33 (2021) 141–149.

[12] A. Bagherjeiran, R. Vilalta, C. F. Eick, Content-Based Image Retrieval Through a Multi-Agent Meta-Learning Framework, Conference: Tools with Artificial Intelligence, 2005. ICTAI 05. 17th IEEE International Conference on.

[13] Zhu, L., "Accelerating content-based image retrieval via GPUadaptive index structure", The Scientific World Journal, Vol. 2014, (2014), 1–11.

[14] A.S. Akopov, L.A. Beklaryan, M. Thakur et al., Parallel multi-agent real-coded genetic algorithm for large-scale black-box single-objective optimisation, Knowledge-Based Systems (2019).

[15] [15] Y. Li, X. Yang, J. Wu, H. Sun , X Guo, L. Zhou, , Discrete-event simulations for metro train operation under emergencies: A multi-agent based model with parallel computing, Physica A 573 (2021) 125964.

[16] V. Rahmani, N. Pelechano, Multi-agent parallel hierarchical path finding in navigation meshes (MA-HNA□), Computers & Graphics , 2019.

[17] Noumsi, A., Derrien, S. and Quinton, P., "Acceleration of a content-based image-retrieval application on the RDISK cluster", In Proceedings 20th IEEE International Parallel & Distributed Processing Symposium, IEEE, (2006), 1–10.

[18] Wasson, V., "An efficient content based image retrieval based on speeded up robust features (SURF) with optimization technique", In 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), IEEE, (2017), 730–735.

[19] Markowska-Kaczmar, U., & Kwaśnicka, H. (2018). Deep learning—a new era in bridging the semantic gap. Intelligent Systems Reference Library, 145, 123–159..

[20] He, R., Zhu, Y. and Zhan, W., "Fast Manifold-Ranking for contentbased image retrieval", In 2009 ISECS International Colloquium on Computing, Communication, Control, and Management (Vol. 2), IEEE, (2009), 299–302.

[21] Tanioka, H., "A Fast Content-Based Image Retrieval Method Using Deep Visual Features", In 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW) (Vol. 5), IEEE, (2019), 20–23.

[22] Zargari, F., Mosleh, A. and Ghanbari, M., "A fast and efficient compressed domain JPEG2000 image retrieval method", IEEE Transactions on Consumer Electronics, Vol. 54, No. 4, (2008), 1886–1893.

[23] Park, M., Jin, J.S. and Wilson, L. S., "Fast content-based image retrieval using quasi-gabor filter and reduction of image feature dimension", In Proceedings Fifth IEEE Southwest Symposium on Image Analysis and Interpretation, IEEE, (2002), 178–182.

[24] Schaefer, G., "Fast Compressed Domain JPEG Image Retrieval", In 2017 International Conference on Vision, Image and Signal Processing (ICVISP), IEEE, (2017), 22–26.

[25] Yang, J., Jiang, B., Li, B., Tian, K. and Lv, Z., "A fast image retrieval method designed for network big data", IEEE Transactions on Industrial Informatics, Vol. 13, No. 5, (2017), 2350–2359.

[26] Sreedevi, S. and Sebastian, S., "Fast image retrieval with feature levels", In 2013 Annual International Conference on Emerging Research Areas and 2013 International Conference on Microelectronics, Communications and Renewable Energy, IEEE, (2013), 1–4.

[27] Mehrabi, M., Zargari, F., Ghanbari, M. and Shayegan, M. A., "Fast content access and retrieval of JPEG compressed images", Signal Processing: Image Communication, Vol. 46, (2016), 54– 59.

[28] Anwar, S.M., Arshad, F. and Majid, M., "Fast wavelet based image characterization for content based medical image retrieval", In 2017 International Conference on communication, computing and digital systems (C-CODE), IEEE, (2017), 351– 356.

[29] Devi, S. and Mathew, A., "Fast image retrieval using Error Diffusion Block Truncation Coding and unsupervised clustering", In 2016 International Conference on Emerging Technological Trends (ICETT), IEEE, (2016), 1–6.

[30] Ksantini, R., Ziou, D., Colin, B. and Dubeau, F., "Logistic Regression Models for a Fast CBIR Method Based on Feature Selection", In Proceedings of the 20th international joint conference on Artifical intelligence, (2007), 2790–2795.

[31] Kakde, B. and Okade, M., "A Novel Technique for Fast ContentBased Image Retrieval sing Dual-Cross Patterns", In 2018 3rd International Conference for Convergence in Technology (I2CT), IEEE, (2018), 1–5.

[32] J.C. Russ, "The image processing handbook", CRC Press, 1999.

[33] G. Sharma, "Digital color imaging handbook", CRC Press, 2003.

[34] J. Chamorro-Martnez, J.M. Medina, C. Barranco, E. Galn-Perales, and J.M. Soto-Hidalgo, "An approach to image retrieval on fuzzy objectrelational database using dominant color descriptors", in Proceedings of the 4th

Conference of the European Society for Fuzzy Logic and Technology, EUSFLAT, 2005, pp. 676–684.

[35] R. Fuller, "On product-sum of triangular fuzzy numbers", Fuzzy Sets and Systems, vol. 41(1), pp. 83–87, 1991.

[36] D. Van der Weken, M. Nachtegael, E. Kerre, "Using similarity measures for histogram comparison," , International Fuzzy Systems Association World Congress, pp. 396-403, 2003.

[37] G. Weiss, Multiagent Systems—A Modern Approach to Distributed Artificial intelligence, MIT Press, Cambridge, MA, 1999.

[38] M.A. Hale, J. Craig, Preliminary development of agent technologies for a design integration framework, in: Proceedings of the Fifth Symposium on Multidisciplinary Analysis and Optimization, Panama City, FL, 1994.

[39] N. Jennings and M. Wooldridge, "Intelligent Agents: Theory and Practice," The Knowledge Eng. Rev., vol. 10, no. 2, 1995, pp. 115–152.

[40] G. Weiss, Multiagent Systems—A Modern Approach to Distributed Artificial Intelligence, MIT Press, 1999.

[41] Guccione, S.A. "Reconfigurable computing at Xilinx ",Proceedings. Euromicro Symposium on Digital Systems Design, Page(s): 102 , 2001.

[42] Becker, J.; Pionteck, T.; Glesner, M. ,"Adaptive systems-on-chip: architectures, technologies and applications" ,14th Symposium on Ingegrated Circuits and Systems Design, 2001.

[43] Hartenstein, R. ,"Coarse grain reconfigurable architectures ",Design Automation Conference, 2001. Proceedings of the ASP DAC 2001. Page(s): 564 –569, Asia and SouthPacific,2001.

[44] Hamid R. Naji ,Letha Etzkorn, Reza Adhami, B. Earl Wells, "Parallel Image Processing with Agent-based Reconfigurable Hardware," Proceedings of the 15th International Conference on Parallel and Distributed Computing Systems (PDCS 2002), September 2002, Louisville, KY.

[45] Hamid R. Naji ,B. Earl Wells,"On Incorporating Multi Agents in Combined Hardware /Software based Reconfigurable Systems, A General Architectural Framework," Proceedings of the 2002 Southeastern Symposium on System Theory, Huntsville, AL , March 2002.

[46] Hamid R. Naji ,B. Earl Wells, M. Aborizka, "Hardware Agents," Proceedings of the ISCA 11th International Conference on Intelligent Systems on Emerging Technologies(ICIS-2002),Boston, MA, July 2002.

[47] https://sites.google.com/site/dctresearch/Home/content-based-image-retrieval

[48] Jun Yue , Zhenbo Li , Lu Liu , Zetian Fu, "Content-based image retrieval using color and texture fused features", Mathematical and Computer Modelling 54 (2011) 1121–1127.

[49] Mehtre, B.M., M.S. Kankanhalli, A.D. Narasimhalu and G.C. Man,. Color matching for image retrieval, Pattern Recognition Letters 16,(331-325 , 1995 .

HOW TO CITE THIS ARTICLE

M. Azimi Hemat, Fuzzy Content-Based Image Retrieval Speed-up Using the Multi-Agent Platform. AUT J. Model. Simul., 54(1) (2022) 3-18.

DOI: 10.22060/miscj.2022.20649.5261

This page intentionally left blank