Computer Science Journal of Moldova, vol.**22**, no.2(65), 2014

Parallel Processing and Bio-inspired Computing for Biomedical Image Registration

Invited Article

Silviu Ioan Bejinariu, Hariton Costin, Florin Rotaru,
Ramona Luca, Cristina Niţă, Camelia Lazăr

Abstract

Image Registration (IR) is an optimization problem computing optimal parameters of a geometric transform used to overlay one or more source images to a given model by maximizing a similarity measure. In this paper the use of bio-inspired optimization algorithms in image registration is analyzed. Results obtained by means of three different algorithms are compared: Bacterial Foraging Optimization Algorithm (BFOA), Genetic Algorithm (GA) and Clonal Selection Algorithm (CSA). Depending on the images type, the registration may be: area based, which is slow but more precise, and features based, which is faster. In this paper a feature based approach based on the Scale Invariant Feature Transform (SIFT) is proposed. Finally, results obtained using sequential and parallel implementations on multi-core systems for area based and features based image registration are compared.

Keywords: image registration, clonal selection algorithm, bacterial foraging algorithm, genetic algorithm, parallel computing.

1 Introduction

Image registration is the process of geometric overlaying or alignment of two or more images of the same scene taken at different times, from different viewpoints, and/or by different sensors [1]. Image registration

(IR) is the first step in image fusion procedures, which combine relevant information from one or more images to create a single image with more informational content. Image registration and fusion methods are used in remote sensing applications, geographic information systems, multispectral image analysis, medical image analysis and other domains. Image fusion may be applied at pixel, feature or decision levels. In the first case, when pixel level image fusion have to be applied, the input images must be registered, because they may differ by the view angle, subject position and also some geometric distortions may be added by the capture device.

There are two different approaches in IR: area (pixel intensity) based methods and feature-based methods [1]. The geometric transform that must be computed may be global (for the entire image) or local in case the images are locally deformed. The most frequently used transforms are the shape preserving mappings (rotation, translation, scaling and the affine transform).

In this paper three different optimization methods are used for the geometric transform parameters estimation: Bacterial Foraging Optimization Algorithm (BFOA), Genetic Algorithm (GA) and Clonal Selection Algorithm (CSA). The foraging model is suitable for optimization problems because animals search for nutrients and try to avoid noxious substances in a way that maximize their energy intake per unit time spent foraging [2]. Computational methods can provide decision models for optimal foraging. The Bacterial Foraging Optimization Algorithm (BFOA) proposed by Passino uses the Escherichia coli bacteria model because it is the most understood microorganism [2], [3], [4]. BFOA is used in image processing to solve also other optimization problems: edge detection in combination with a probabilistic derivative technique [5]; fuzzy entropy based image segmentation [6]. A modified version of BFOA used for multilevel thresholding segmentation was compared to genetic algorithms and particle swarm optimization algorithm [7]. Image registration BFOA based methods were proposed in [8], [9] and [10] Parallel implementations of BFOA were proposed in [11] and [12]. Genetic Algorithms are search techniques that emulate evolutionary processes to solve optimization problems [13]. Like BFOA,

GAs start with a population of individuals (points) in the problem domain and use these points to approximate the optimal solution. The difference is that instead moving in the problem domain, GAs use the recombination of two or more parents to produce offspring [14]. GAs are often used in biomedical or remote sensing image registration [13]. The Clonal Selection Algorithm (CSA) belongs to the field of Artificial Immune Systems that include computational methods inspired by the mechanisms of the biological immune system [15]. Like GAs, CSA may use binary solution coding and real coding. In [16] a real coded clonal selection algorithm is used in electromagnetic design optimization. It is also suitable for high dimensional optimization problems. CSA is effective, in terms of accuracy, capable of solving large-scale problems [17] and is comparable to other optimization algorithms. A performance comparison of CSA and GA is presented in [18] and conclusion is that each one has better performance depending on the function to optimize.

The paper is organized as follows. In the second section the measures used for area based and features based IR methods are described. In case of features based IR, a short description of SIFT transform and procedure to find the SIFT key points correspondences are also included. In the third section, the optimization algorithms – BFOA, GA and CSA – are shortly presented. In the fourth section the proposed parallel versions of BFOA and GA are presented. In the fifth section the results obtained by applying the optimization procedures for biomedical image registration are shown in both sequential and parallel versions. The last section concludes the paper.

2 Image Registration

There are two different approaches in IR: area (pixel intensity) based methods and feature-based methods [1]. Almost all methods consist of four steps: feature detection, feature matching, transform estimation and image resampling. The feature detection step is specific to feature based registration methods and distinctive and stable features (points, lines, contours, regions) have to be detected. Because the transform es-

timation is performed while looking for the correspondent features, the second and third steps are usually combined. In the image resampling step, different interpolation methods are used: the nearest neighbor function, the bilinear and bicubic functions, quadratic splines, cubic and higher-order B-splines [1].

2.1 Area Based IR

In case of area based IR methods, to evaluate the similarity between images the normalized correlation (NCC), the Fourier representation or normalized mutual information (NMI) are used. In this study, the Mutual Information is used to evaluate the similarity in case of area based registration.

Mutual information is a robust measure used in image registration [1], [10]. It evaluates the relative independence of two images and does not depend on the specific dynamic range or intensity scaling of the images. High values of mutual information indicate high dependence between images. It is defined as

$$MI(A,B) = H(A) + H(B) - H(A,B), \tag{1}$$

where $H(.)$ is the image entropy and $H(A,B)$ is the joint entropy of the two images. Because mutual information based registration methods are sensitive to changes that occur in the distributions as a result of difference in overlapping regions, normalized mutual information can be used:

$$NMI(A,B) = \frac{H(A) + H(B)}{H(A,B)}. \tag{2}$$

Registration of two images A and B requires maximization of mutual information, thus maximization of the entropies $H(A)$ and $H(B)$, and minimization of the joint entropy $H(A,B)$.

Usually, optimization in image registration means to maximize similarity. If the optimization algorithm is oriented on cost function minimization, then the value of $(-1) * MI$ is used to evaluate the cost of the transform for a certain solution.

2.2 Feature based IR

In case of feature based IR methods, spatial relations, invariant descriptors, relaxation methods and multiresolution transforms (pyramids and wavelets) are used. In this paper, the features based IR will use key points determined using the Scale Invariant Feature Transform (SIFT) [19], [20]. SIFT is used to select distinctive features, used in pattern recognition, localization, 3D mapping, tracking and image registration. It allows scale and rotation invariant features detection, with good results for affine distortions. The SIFT algorithm has 4 distinctive stages: extrema detection in the scale space of the image, key points selection and localization, key points orientation assignment and description generation. The identified features have to be distinctive.

a. *Scale-space extrema detection.* Key point candidates selection is performed by finding the extrema of the Difference of Gaussians (DOG) function computed as the difference of two scaled images separated by a multiplicative factor k.

$$\begin{aligned} D(x,y,\sigma) &= L(x,y,k\sigma) - L(x,y,\sigma) = \\ &= (G(x,y,k\sigma) - G(x,y,\sigma)) * I(x,y), \end{aligned} \tag{3}$$

where $L(x,y,\sigma)$ is the scale space of the image $I(x,y)$ obtained by convolving it with the Gaussian kernel $G(x,y,\sigma)$. Extrema points depend on the frequency sampling in the scaled space and the initial value of σ.

b. *Key points localization.* Key points are selected from the most stable and accurately localized candidates. Key point candidates having low contrast or strong edge response in one direction only are removed. Because the candidates obtained in higher scales correspond to several pixels in the original image, for an exact localization is performed by computing the extrema points of the Taylor expansion up to quadratic terms of the scale space function $D(x,y,\sigma)$ [19].

c. *Orientation assignment.* To make key point descriptions invariant to rotation, their orientations are computed using the orientation histogram of local gradients of the closest smoothed image $L(x,y,\sigma)$.

The gradient magnitude and orientation are computed using pixel differences:

$$m(x,y) = \\ = \sqrt{(L(x-1,y)-L(x+1,y))^2+(L(x,y-1)-L(x,y+1))^2}, \quad (4)$$

$$\theta(x,y) = arctg\frac{L(x,y+1)-L(x,y-1)}{L(x+1,y)-L(x-1,y)}. \quad (5)$$

Each point is added to the histogram weighted by the gradient magnitude $m(x,y)$ and by a circular Gaussian. To obtain a more accurate orientation, the dominant peaks in the histogram are interpolated with their neighbors.

d. *Key point descriptor computing.* The key point descriptor contains $128 = 4 \times 4 \times 8$ values obtained using 16 orientation histograms computed in a 4×4 grid. Each histogram contains 8 orientation bins. The descriptor is computed in a support window of 16×16 pixels around the key point [19].

To evaluate the similarity between two images the key points correspondences have to be established. The Euclidian distances between SIFT descriptors of each key point from source image and those of the model image are computed. By sorting the computed values for source images key points, a match is established when the minimum computed distance is less than a certain percentage from the second distance. In our experiment a percent of 30% is used [20].

For IR, in the similarity evaluation step, the coordinates of the key points in the source image are transformed accordingly to the values of transform parameters and the sum of Euclidean distances between positions of key points in the model and transformed source image is used as similarity value [12].

3 Bio-inspired computing in IR

In this paper three different bio-inspired optimization methods are used in order to compute the optimal geometric transform that allows the

source image to overlay the model image: Clonal Selection Algorithm, Bacterial Foraging Optimization Algorithm and Genetic Algorithm.

3.1 Clonal Selection Algorithm

The Clonal Selection Algorithm (CSA) belongs to the field of Artificial Immune Systems which includes computational methods inspired by the mechanisms of the biological immune system. A simplified description of the immune system is an organ system intended to protect the host organism from the threats posed to it from pathogens and toxic substances.

CSA is inspired by the Clonal Selection theory of acquired immunity. It is a population based stochastic method with binary representation of variables [16] which may be used for multimodal optimization. In some cases, also real encoding of variables may be used to solve numerical problems.

Clonal Selection Algorithm can be listed as follows [18]:

1. Randomly generate a set of solution candidates: antibodies.
2. Compute the affinity values of each candidate solutions.
3. While the minimum error criterion is not met
   - 3.1 Sort the antibodies starting from the lowest affinity. The lowest affinity means better matching between antibody and antigen.
   - 3.2 Clone the better matching antibodies more with some predefined ratio.
   - 3.3 Mutate the antibodies with some predefined ratio. This ratio is obtained in a way that better matching clones mutated less and weakly matching clones mutated much more in order to reach the optimal solution.
   - 3.4 Compute affinity values of each antibody.

It starts with an initial set of adaptive units: the general immune cells. Each cell represents a possible solution of the problem and participates in a competitive selection process. The algorithm involves the selection of antibodies based on affinity against a pattern, computed by a cost function. Selected antibodies are cloned and resulted clones are subject of hypermutation. The hypermutation is inverse proportional to computed clone affinity. The resulted set competes with the already existing antibodies in the next generation of the evolution process. The low-affinity population members are replaced by new randomly generated antibodies.

3.2 Bacterial Foraging Optimization Algorithm

The Bacterial Foraging Optimization Algorithm belongs to the field of Bacteria Optimization Algorithms and Swarm Optimization. There have been many extensions of the approach that attempt to hybridize the algorithm with other Computational Intelligence algorithms and Metaheuristics such as Particle Swarm Optimization, Genetic Algorithm.

The bacterial foraging paradigm [2], [3], [4] is suitable as model for optimization algorithms because animals / bacteria behavior is to search for nutrients and avoid noxious substances to maximize their energy. BFOA is based on a colony of evolving bacteria which are replicated if have a good strategy to find nutrients or die in the other case. Each bacterium is characterized by its position and quantity of accumulated nutrients or healthy status.

In optimization problems, the possible solutions are encoded in the bacteria position and the movement of the colony members tends to approximate the optimal solution. The final solution is specified by the position in which a bacterium is in the best healthy state or the nutrients amount is the highest.

According to BFOA approach, the bacteria colony moves in the n-dimensional space, where n is the optimization problem dimension and the quantity of nutrients / healthy status is described by a cost function defined according to the optimization problem. Dur-

ing its evolution, the bacteria colony proceeds through four foraging steps: chemotaxis, swarming, reproduction and elimination-dispersal. In the following paragraphs, the colony consists of S individuals; $P(j,k,l) = \{\theta^i(j,k,l), i = 1...S\}$ is the position of colony members in the j^{th} chemotactic step, k^{th} – reproduction step and l^{th} – elimination-dispersal step; $J(i,j,k,l)$ – the cost of the i^{th} bacterium in position $\theta^i(j,k,l)$.

Chemotaxis. In the chemotactic step, a bacterium can move in two ways: tumble and swim. First, a tumble is executed in a random direction. The new position of the i^{th} bacterium is:

$$\theta^i(j+1,k,l) = \theta^i(j,k,l) + C(i)\varphi(i), \tag{6}$$

where $C(i)$ is the size of the chemotactic step and $\varphi(i)$ is a unit length of randomly generated direction [4]. The movement continues in the same direction while the value of the cost function decreases but not more than a maximum number of steps.

Swarming. The bacteria tend to swarm together if they have the possibility to signal to each other the presence of a favorable or poisonous environment (social behavior). The cell to cell attraction or rejection is modeled by modifying the value of the cost function $J(i,j,k,l)$ by a value that depends on the status of all the other bacteria in the colony.

Reproduction. After a number of chemotactic steps, all bacteria accumulate a quantity of nutrients that is usually expressed as the cost function computed in the current position. Those which accumulated a greater quantity of nutrients are in a healthier state and split into two bacteria. Those which accumulated a smaller amount of nutrients die. In BFOA, to keep constant the size of the colony, the number of bacteria which split is equal to the number of bacteria which die. The new bacteria are created without mutation in the same position as the parent bacteria [4].

Elimination and Dispersal. After a number of reproduction steps, with a specified probability P_{ed}, some bacteria are removed from colony

(elimination) regardless their healthy state and new bacteria are created in random positions (dispersal) [4].

The optimization algorithm starts with a colony of S bacteria placed in randomly generated positions. The evolutionary process consists of N_{ed} elimination-dispersal steps, each of these consists of N_{re} reproduction steps and each reproduction step consists of N_C chemotactic steps. In each chemotactic step a bacterium may do at most N_S swarming steps while the cost function value decreases.

Bacterial Foraging Optimization Algorithm can be listed as follows:

```
Initialize bacteria colony
for l = 1 to N_ed (elimination dispersal loop)
  for k = 1 to N_re (reproduction loop)
    for j = 1 to N_C (chemotaxis loop)
      for i = 1 to S (each bacterium)
        perform tumble and change bacteria position to θ^i(j+1, k, l)
        compute cost function in new position
        m = 0
        while cost function value decreases and m < N_S
          perform swarm and change bacteria position
          compute cost function in new position
        end while
      end for
    end for
  end for
end for
```

The position in which a bacterium reaches the lowest value of the cost function (greatest healthy status) is the solution of the optimization problem. In case of image registration, the size of the search space is equal to the number of parameters of the geometric transform.

3.3 Genetic Algorithms

Genetic Algorithm is an adaptive strategy used for global optimization problems. Inspired by population genetics, GA is based on a set of individuals in which the possible solutions of the problem are encoded as chromosome strings. The general structure of GAs is: (a) selection of the appropriate encoding method and fitness function, (b) generation of a random initial population and (c) the evolution loop of the algorithm: fitness function evaluations, application of genetic operators and creation of the new generation. After a number a generations, the population is expected to contain chromosomes that approximate the global maximum value of the fitness function. In each generation chromosomes with best fitness values are retained and generate offspring that replaces chromosomes with the lowest values of the fitness function. Genetic operators used for new generation creation are: selection, crossover and mutation.

1. Randomly generate a set of individuals.
2. Compute fitness for all individuals.
3. While the stop criterion and maximum generation number are not met
   (Evolution loop)
   3.1 Apply reproduction
   3.2 Mutation
   3.3 Crossover.

In [13] it is proposed an IR procedure using the string encoding of chromosomes. The parameters of the geometric transform are encoded as bit fields in a 32 bit value. In the procedure described below, the real encoding is used and each chromosome is characterized by a number of real values equal to the number of geometric transform parameters. Discrete, average and simplex crossover operators are used depending on user defined probabilities (p_d, p_a and p_s).

4 Parallel approach for bio-inspired IR

Analyzing the optimization procedures execution, it must be noticed that most of the processing time is spent in the cost function evaluations. In case of BFOA based registration, about 99% of the execution time is spent in the cost evaluation function and more detailed, about 83% for mutual information computing and 16% of total time is spent in when the geometric transform is applied to source image. The same, in case of GA optimization: 96% of the execution time is spent in the cost evaluation (81% to compute mutual information and 15% to apply the geometric transform). Because both BFOA and GA procedures were executed using the sequential implementation, only about 25% of the computing power is used in case of a Core i5 processor.

To optimize the IR procedure parallel implementations based on the computing power of multi-core processors were proposed in [12].

A closer look at BFOA reveals that it contains 4 nested loops: elimination/dispersal, reproduction and chemotaxis for each bacterium in the colony. The body of the inner loop is executed $N_{ed} \times N_{re} \times N_C \times S$ times, which may be a fairly large number. In fact, the cost function evaluation is performed more than two times this number due to the fact that each bacterium may perform more swim steps in a single chemotactic step. While the calculations performed for each individual bacterium in the inner loop are independent, the bacteria colony may perform the chemotactic steps simultaneously.

In case of GA optimization, the cost function evaluation is called from two different places. First, it is called from the main evolution loop of the algorithm (about 41% of execution time) only for the new created chromosomes evaluation, and second, in the simplex crossover function (about 53% of execution time). In the first case, the cost function is called for all not already evaluated chromosomes, so this task is easily parallelized. In case of simplex crossover that involves more than one chromosome, the crossover function will be executed in parallel for each group of chromosomes [12].

The IR procedure that uses the Clonal Selection Algorithm was not parallelized because there are few tasks completely independent,

suitable for parallel execution.

The parallel implementation was evaluated on an Intel Core i5 3.10 GHz processor and is detailed in the next section.

5 Experiments

In this section a comparison of results obtained using image registration procedures based on Bacterial Foraging Optimization Algorithm, Clonal Selection Algorithm and Genetic Algorithm is presented. The optimization procedures were applied for area based registration and features based registration. In the second case the SIFT features are used.

The image registration procedure was tested on a large set of DICOM medical images from a database available at http://www.osirix-viewer.com/datasets/ [21]. Below, only the results obtained using the *Brainix* image as model are described. It is a gray level image (8 bits per pixel) and size 256×256 pixels. *Brainix* is a MR image of a brain tumor.

The image registration procedure was applied also to the image after it was modified by adding "salt & pepper" noise.

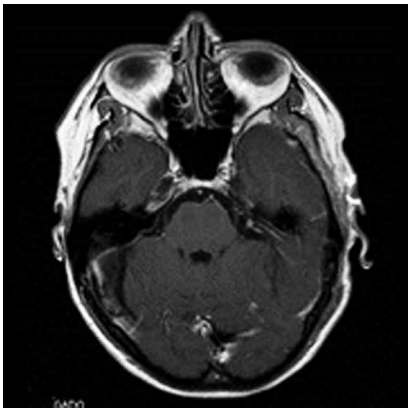


Figure 1. Original image BRAINIX (from [21])

The source images were obtained by applying a rotation (angle $\theta = 10°$) against the rotation center ($c_x = -20$ and $c_y = 20$) followed by an isotropic scaling ($scale = 1.2$). While the transform is defined by 4 parameters, the search space in the optimization problem is R^4. The actual value of the transform matrix is

$$T = \begin{bmatrix} \alpha & \beta & (1-\alpha)\,c_x - \beta\,c_y \\ -\beta & \alpha & \beta\,c_x + (1-\alpha)\,c_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1.1818 & 0.2084 & -0.5322 \\ -0.2084 & 1.1818 & -7.8029 \\ 0 & 0 & 1 \end{bmatrix}, \quad (7)$$

where $\alpha = scale \cdot \cos\theta$ and $\beta = scale \cdot \sin\theta$.

The inverse transform matrix is

$$T^{-1} = \begin{bmatrix} 0.8207 & -1.1447 & -0.6924 \\ 1.1447 & 0.8207 & 6.4807 \\ 0 & 0 & 1 \end{bmatrix}$$

that corresponds to an affine transform with the following parameters: $\theta' = -10°$, $c'_x = -20$, $c'_y = 20$ and $scale' = 0.8333$.

To evaluate the similarity between model image and registered source image, the normalized mutual information is used.

The BFO parameters values used in the experiment are: bacteria colony size $S = 400$; number of chemotactic steps $N_c = 20$; maximum number of swim steps $N_s = 10$; number of reproduction steps $N_{re} = 16$; number of elimination / dispersal steps $N_{ed} = 2$; probability of dispersal $P_{ed} = 0.25$; length of the move step $C_i = 0.005$.

In case of GA optimization, the real encoding is used, and each chromosome is characterized by four real values representing the number of geometric transform parameters. Discrete, average and simplex crossover operators are used depending on user defined probabilities (p_d, p_a and p_s). The GA parameters are: number of generations $nGen = 500$ and number of chromosomes $nCr = 1500$. The crossover probabilities are: $p_d = 0.05$, $p_a = 0.15$ and $p_s = 0.2$.

Two different source images were used. The first one was obtained by applying the transform described above to the model image. The

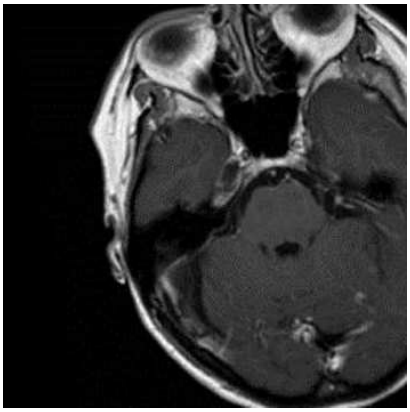
second source image was obtained from the first one by applying "salt & pepper" noise. The signal-to-noise ratio in these images is about -1 dB.

In Figure 2, source images and some samples of registered images are presented.

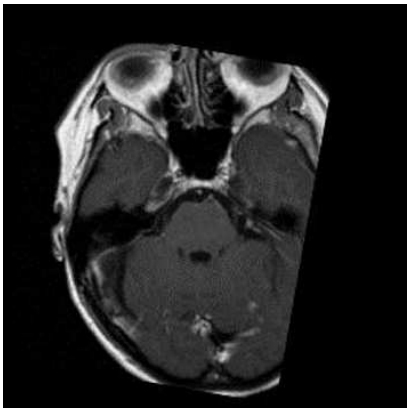
In Tables 1 and 2 the results of both sequential and parallel area based registration are presented. The column '# cost eval' shows the total number of cost function evaluations; the column 'best cost eval' shows the cost function evaluation in which the best value was obtained; column 'ex_MI' (expected MI) shows the expected cost value obtained by measuring the similarity between model image and the source image after the computed inverse transform was applied; column 'c_MI' (computed MI) shows the cost value obtained by measuring the similarity between model image and source image after the approximated inverse transform was applied. In both tables 1 and 2, 'Brainix' denotes the source image and 'Brainix+SP' denotes the source image altered by adding the "salt & pepper" noise.

Table 1. Results of area based image registration, sequential version

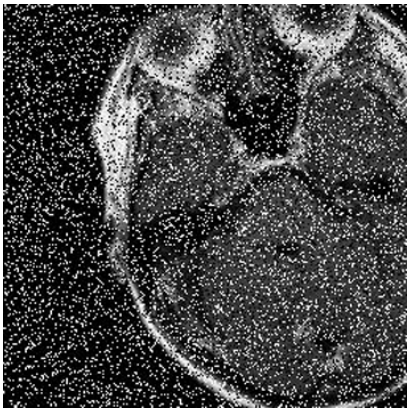
Mode	Image	Opt.	Time (sec)	# cost eval	best cost eval	ex_MI	c_MI
Seq	Brainix	BFOA	811.3	591686	469054	1.3218	1.3202
		GA	135.8	99148	63176	1.3218	1.3215
	Brainix+ SP	BFOA	820.9	563561	546678	1.1394	1.1382
		GA	140.2	95371	60.348	1.1402	1.1387
Paral- lel	Brainix	BFOA	253.4	595365	249742	1.3218	1.3177
		GA	46.6	98894	79889	1.3218	1.3191
	Brainix+ SP	BFOA	254.9	566433	269008	1.1398	1.1390
		GA	47.4	95042	86093	1.1398	1.1388



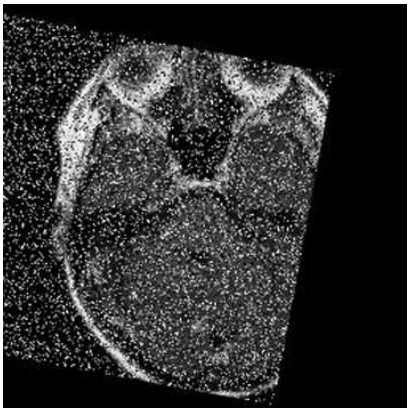
a. Source image, no noise added



b. Registered image



c. Source image, salt and pepper noise added, $SNR = 1.03$ dB



d. Registered image

Figure 2. Results of image registration procedure

As noted in Table 1, the GA optimization requires about 6 times less cost function evaluations and this is the reason for which the IR procedure is faster in this case. In case of area based IR, the Clonal Selection Algorithm was not used for optimization because it requires about 100 times more cost function evaluations, consequently the registration is not achieved within a reasonable time interval. In fact, in case CSA is used as optimization method, the duration of features based registration is comparable to duration of area based registration using BFOA / GA as optimization algorithm.

On the Intel Core i5 processor which was used in experiments, the parallel implementations of the optimization procedures are about three times faster than the sequential versions, while the number of cost evaluations in close.

The expected value of similarity measure in Table 2 has different values due to the "salt & pepper" noise randomly added into images. The best values were obtained using sequential GA optimization in case of *'Brainix'* image and parallel BFOA optimization in case of *'Brainix+SP'* image.

In Table 2 the parameters of the approximated geometric transform are presented for both sequential and parallel implementation. The values determined by computing the inverse geometric transform are: $\theta' = -10^\circ$, $c'_x = -20$, $c'_y = 20$ and $scale' = 0.8333$.

In Table 3 the results of features based IR are presented. While in this case the cost function evaluation requires applying the approximated inverse transforms to a small number of pixel coordinates, the registration process is faster. The last column of Table 2 contains the number of correspondent key points between model and source images, i.e. the number of key points for which the geometric transform must be applied. If source image is not altered by noise, there are 188 correspondent key points. In case the source image is randomly altered by noise, the number of correspondences is between 11 and 16. But, features based registration is not a solution for noisy images, by increasing the noise in the source images, it is possible to don't find any correspondent key points pairs. This is the case for images obtained using different types of sensors or acquisition methods.

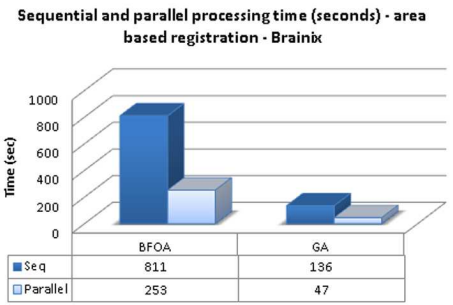
Table 2. Parameters of inverse affine transform computed using area based registration

Mode	Image	Opt.	c'_x	c'_y	θ'	$scale'$
Seq	Brainix	BFOA	-19.69	20.17	-10.02	0.83
		GA	-20.10	20.12	-9.99	0.83
	Brainix+SP	BFOA	-20.40	18.79	-9.98	0.83
		GA	-19.12	19.35	-10.07	0.83
Parallel	Brainix	BFOA	-20.16	19.90	-9.98	0.83
		GA	-20.26	20.30	-9.98	0.83
	Brainix+SP	BFOA	-19.85	19.41	-9.99	0.83
		GA	-19.10	20.13	-10.06	0.83

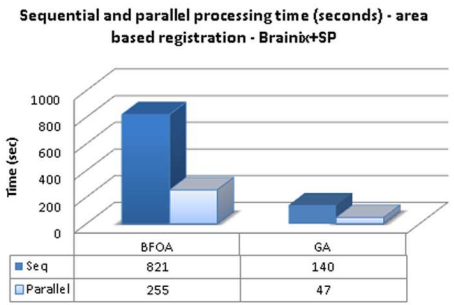
In Table 4 the values of the approximated transform parameters are presented. It must be noticed that in two cases the results are not so near to expected values: when the noisy image 'Bainix+SP' is used as source and in case CSA is used as optimization method.

The charts presented in Figure 4 show compare the sequential and parallel execution time for all the experiments presented before. It is obvious that for long tasks, as area based IR, the gain obtained by using the parallel versions is greater. For short tasks, as features based IR, the speedup is lower. This happens because the processing time becomes comparable to that of the synchronization tasks required by the parallel implementation and also because shorter time intervals are affected by all other processes and events that occur in the operating system. The results obtained in case of CSA optimization are not included in charts because in this case the duration was too long.

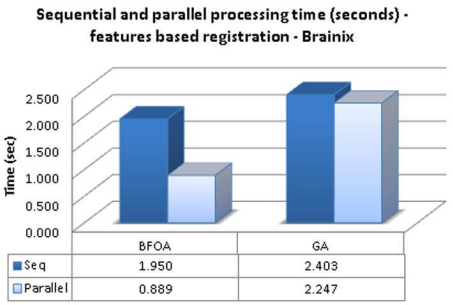
In Figure 4, the parallelization evaluation is presented. The most common evaluation of parallel algorithms is performed using the parallel efficiency $E = \frac{t_s}{t_p \times n}$, where t_s is the time used by the sequential version of the algorithm, t_p is the processing time for the parallel version and n is the number of used processors. As it was already said, better efficiency ($> 74\%$) is obtained for area based IR parallelization



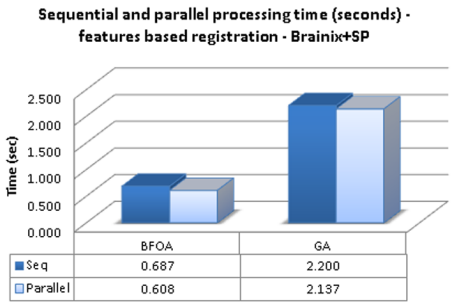
a. Area based IR of Brainix image



b. Area based IR of Brainix+SP image



c. Features based IR of Brainix image



d. Features based IR of Brainix+SP image

Figure 3. Processing time for different IR approaches

Table 3. Results of features based image registration, sequential version

Mode	Image	Opt.	Time (sec)	# cost eval	best cost eval	ex_MI	c_MI	key-points
Seq	Brainix	BFOA	1.9	660726	391414	1.3218	1.3145	188
		GA	2.4	118359	17701	1.3218	1.3173	188
		CSA	353.3	73240500	-	1.3218	1.2948	188
	Brainix	BFOA	0.7	662137	378711	1.1400	1.1365	15
	+SP	GA	2.2	111775	40274	1.1405	1.1240	11
		CSA	191.5	65687973	-	1.1392	1.1187	12
Paral-lel	Brainix	BFOA	0.9	658684	636958	1.3218	1.3171	188
		GA	2.2	115568	39971	1.3218	1.3168	188
		CSA	139.9	71315426	-	1.3218	1.2916	188
	Brainix	BFOA	0.6	640345	528443	1.1398	1.1234	12
	+SP	GA	2.1	104900	26898	1.1389	1.1174	16
		CSA	102.9	70095946	-	1.1405	1.1305	11

which has a longer execution time. In case of CSA optimization used for features based IR, the efficiency is between 47% and 68% which may lead to the conclusion that the procedure was not completely parallelized. In all other cases (BFOA and GA for features based IR), the lower parallelization efficiency is not relevant, while the execution time is too short.

The image registration procedures were implemented and tested in an image processing framework developed by authors of this paper. It is implemented in C++ as a Windows application and uses OpenCV library [22] for images manipulation and the parallel programming support available in Microsoft Visual Studio 2010 [23].

6 Conclusions

This paper is focused on the use of some bio-inspired optimization methods for medical images registration. Three different approaches

Table 4. Parameters of inverse affine transform computed using features based registration

Mode	Image	Opt.	c'_x	c'_y	θ'	$scale'$
Seq	Brainix	BFOA	-20.85	20.14	-9.93	0.83
		GA	-19.81	20.10	-9.99	0.83
		CSA	-20.35	19.15	-9.97	0.83
	Brainix+SP	BFOA	-20.01	20.20	-9.97	0.83
		GA	-22.48	19.33	-9.96	0.84
		CSA	-17.99	16.93	-10.22	0.84
Parallel	Brainix	BFOA	-19.88	19.54	-9.99	0.83
		GA	-19.82	20.01	-9.99	0.83
		CSA	-17.33	20.11	-10.16	0.83
	Brainix+SP	BFOA	-25.43	21.66	-9.45	0.83
		GA	-11.73	19.72	-10.77	0.83
		CSA	-20.31	20.64	-9.85	0.83

are presented: Bacterial foraging optimization algorithm, genetic algorithm and clonal selection algorithm. Since image registration may be a time consuming task, different optimization strategies were applied: the use of scale invariant features transform key points and full usage of computing power of multi-core processors. The obtained results may be summarized as follows:

- BFOA and GA allow to obtain comparable results in terms of registration precision;
- GAs perform faster the image registration about three times faster than BFOA;
- CSA is too slow for features based registration (comparable to area based IR combined with BFOA and GA) and also with lower precision, provided that algorithm's parameters were not enough tuned;

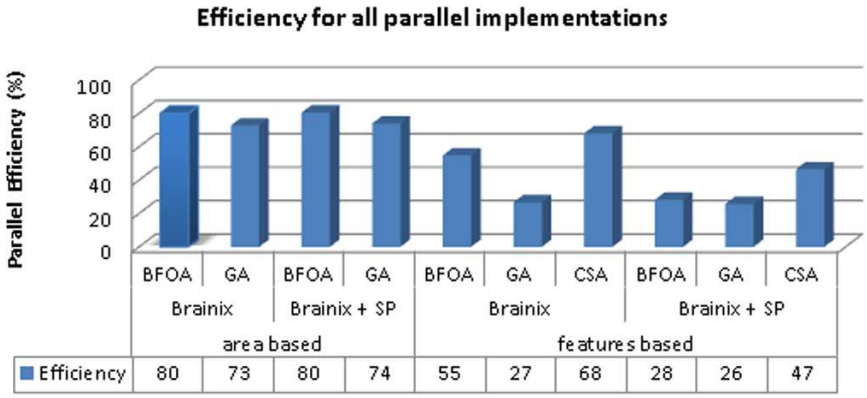


Figure 4. Parallel efficiency obtained in all experiments

- Even the features based IR performs faster, it's usage for multimodal images is limited by the procedure's capability to find common and stable features in the images to be registered;
- Parallel implementations are suitable in image registration, while cost function evaluations are independent and time consuming tasks.

References

[1] B. Zitova, J. Flusser. *Image registration methods: a survey*, Image and Vision Computing, 21, Elsevier, 2003, pp. 977–1000.

[2] Y. Liu, K. M. Passino. *Biomimicry of Social Foraging Bacteria for Distributed Optimization: Models, Principles, and Emergent Behaviors*, Journal of Optimization Theory and Applications, Vol. 115, No. 3, 2002, pp. 603–628.

[3] K.M. Passino. *Biomimicry of Bacterial Foraging for Distributed Optimization and Control*, IEEE Control Systems Magazine, June 2002, pp. 52–67.

[4] K.M. Passino. *Biomimicry for Optimization, Control, and Automation*, Chapter 18: Cooperative Foraging and Search, Springer Verlag, 2005.

[5] O. P. Verma, M. Hanmandlu, P. Kumar, S. Chhabra, A. Jindal. *A novel bacterial foraging technique for edge detection*, Pattern Recognition Letters, 32, Elsevier, 2011, pp. 1187–1196.

[6] N. Sanyal, A. Chatterjee, S. Munshi. *An adaptive bacterial foraging algorithm for fuzzy entropy based image segmentation*, Expert Systems with Applications, 38, Elsevier, 2011, pp. 15489–15498.

[7] P.D. Sathya, R. Kayalvizhi. *Modified bacterial foraging algorithm based multilevel thresholding for image segmentation*, Engineering Applications of Artificial Intelligence, 24, Elsevier, 2011, pp. 595–615.

[8] Z. Yudong, W. Lenan. *Multi-resolution rigid image registration using bacterial multiple colony chemotaxis*, 5th International Conference on Visual Information Engineering, 2008, VIE 2008, pp. 528–532.

[9] H. Costin, C. Rotariu. *PET and CT images registration by means of soft computing and information fusion*, Proceedings of the 1st WSEAS international conference on Biomedical electronics and biomedical informatics, 2008, pp. 150–161.

[10] H. Costin, S. Bejinariu. *Medical Image Registration by means of a Bio-Inspired Optimization Strategy*, Computer Science Journal of Moldova, vol. 20, Nr. 2 (59), 2012, pp. 178–202.

[11] H. Costin, S. Bejinariu. *Medical Signal Processing by Means of Immune Algorithms*, 4th IEEE International Conference E-Health and Bioengineering – "EHB 2013", Iasi, Romania, 21-23 Nov. 2013.

[12] S. Bejinariu. *Image Registration using Bacterial Foraging Optimization Algorithm on Multi-core Processors*, Electrical and Elec-

tronics Engineering (ISEEE), 2013 4th International Symposium on, October, 11-13, 2013. Galaţi, România.

[13] R. Singhai, J. Singhai. *Registration of Satellite Imagery using Genetic Algorithm*, Proc of the World Congress on Engineering, WCE 2012, London, UK, Vol II.

[14] S. Tsutsui, M. Yamamura, T. Higuchi. *Multi-parent Recombination with Simplex Crossover in Real Coded Genetic Algorithms*, Proc. of the Genetic and Evolutionary Computation Conference, pp. 657–664. Orlando, Florida, USA, 2000.

[15] J.Brownlee. *Clever Algorithms. Nature-Inspired Programming Recipes*, lulu.com; 1st edition, 2012.

[16] F.Campelo, F.G. Guimarães, H.Igarashi, J.A. Ramírez. *A Clonal Selection Algorithm for Optimization in Electromagnetics*, IEEE Transactions on Magnetics, Vol. 41, no. 5, 2005, pp.1736–1739.

[17] M. Pavone, G. Narzisi, G. Nicosia. *Clonal selection: an immunological algorithm for global optimization over continuous spaces*, Journal of Global Optimization, Springer Science+Business Media, LLC. 2011.

[18] E.D. Ülker, S. *Ülker. Comparison Study for Clonal Selection Algorithm and Genetic Algorithm*, International Journal of Computer Science & Information Technology (IJCSIT) Vol 4, No 4, 2012.

[19] D. Lowe. *Distinctive Image Features from Scale-Invariant Keypoints*, International Journal of Computer Vision, 60(2), 2004, pp. 91–110.

[20] S. Bejinariu, M.Costin, A. Ciobanu, S. Cojocaru. *Similarities Identification in Logo Images*, Proceedings IIS 2013, International Workshop on Intelligent Information Systems, Chisinau, Republic of Moldova, 2013, pp. 53–59.

[21] *DICOM sample image sets*, http://www.osirix-viewer.com/data sets/, last accessed on 1.06.2014.

[22] G. Bradski, A. Kaehler. *Learning OpenCV. Computer Vision with the OpenCV Library*, O'Reilly Media, Inc., 2008.

[23] C. Campbell, A. Miller. *Parallel programming with Microsoft Visual C++*, Microsoft Corporation, 2012.

Silviu Ioan Bejinariu, Hariton Costin, Florin Rotaru, Ramona Luca, Cristina Niţă, Camelia Lazăr

Received June 2, 2014

Institute of Computer Science, Romanian Academy,
Iasi Branch, Romania
E–mails: {*silviu.bejinariu, hariton.costin, florin.rotaru, ramona.luca, cristina.nita, camelia.lazar*}*@iit.academiaromana-is.ro*