

Evaluative Study of PSO/Snake Hybrid Algorithm and Gradient Path Labeling for Calculating Solar Differential Rotation

Ehsan Shahamatnia^{1,2(✉)}, André Mora^{1,2}, Ivan Dorotovič^{1,3},
Rita A. Ribeiro^{1,2}, and José M. Fonseca^{1,2}

¹ Computational Intelligence Group of CTS/UNINOVA, Caparica, Portugal
{ehs, atm, id, rar, jmf}@uninova.pt

² FCT/NOVA University of Lisbon, 2829-516 Monte de Caparica, Portugal

³ Slovak Central Observatory, Hurbanovo, Slovak Republic

Abstract. PSO/Snake hybrid algorithm is a merge of particle swarm optimization (PSO), a successful population based optimization technique, and the Snake model, a specialized image processing algorithm. In the PSO/Snake hybrid algorithm each particle in the population represents only a portion of the solution and the population, as a whole, will converge to the final complete solution. In this model there is a one-to-one relation between Snake model snaxels and PSO particles with the PSO's kinematics being modified accordingly to the snake model dynamics. This paper provides an evaluative study on the performance of the customized PSO/Snake algorithm in solving a real-world problem from astrophysics domain and comparing the results with Gradient Path Labeling (GPL) image segmentation algorithm. The GPL algorithm segments the image into regions according to its intensity from where the relevant ones can be selected based on their features. A specific type of solar features called coronal bright points have been tracked in a series of solar images using both algorithms and the solar differential rotation is calculated accordingly. The final results are compared with those already reported in the literature.

Keywords: Particle swarm optimization · Snake model · PSO/Snake hybrid algorithm · Gradient path labeling · Image processing · Image segmentation · Object tracking · Solar images

1 Introduction

Particle swarm optimization (PSO), first introduced by [1], is a collective, anarchic, iterative method, with the emphasis on cooperation [2]. PSO is a general search method that can be used to solve a wide range of problems, but it is particularly useful for solving difficult problems, as there are often specific methods for solving easier problems, which are more effective. One such example is the resolution of linear systems, where PSO is not at all the best tool [2]. PSO is a stochastic algorithm based on the analogy of collective behavior of birds' swarms. PSO consists of a population of particles, each similar to a bird searching for the best place to find food. Each particle in PSO is a candidate solution. In PSO, particles are governed under their cognitive and

social behaviors, which make them able to exchange information and share their experience of explored space, and finally converge towards the optimum of search space, which is the solution to the formulated problem.

One of the main problems in digital image processing is image segmentation, for which more than a thousand different algorithms have been developed [3]. Deformable models are popular spatial segmentation techniques for outlining object boundaries using contours [4]. Active Contour Model (ACM), a technique based on deformable model, was introduced by Kass et al. [5] for 2D image segmentation. The basic idea of ACM, also known as snake model, is to evolve a contour (a curve or a surface) under some constraints to match certain image features. Snake model has been successfully employed in a variety of problem domains such as object tracking, shape modeling image segmentation and stereo vision [5–10].

To drive the snake control points we can use PSO, although limiting the particle search space to avoid premature convergence to the global optimum. There are already several solutions published, for instance Tseng *et al.* [11] and Li *et al.* [12] defined multiple PSO populations, in which each control point is confined to a sub-swarm spatially distinct from other sub-swarms. A polar coordinate system that limits the snake control points search space was proposed by Ballerini [13] and Nebti and Meshoul [14]. Another alternative solution proposed by Zeng and Zhou [15] was to iteratively rank the best particles position and by analytical calculations prevent particles from intersecting.

Most of the aforementioned methods act only as a general problem solver and take the approach of formulating the snake model calculations as a minimization problem and then just solving this optimization problem. In this paper, we take the hybrid PSO/Snake approach introduced in [16] and show its versatility by further extending it to solve a real world problem from the astrophysics domain. The PSO/Snake algorithm has already been successfully tested for detection and tracking of small deformable structures such as endothelium cells from cornea microscopic images [17] and tracking sunspots [18].

Snakes model experiences several problems, namely, snake initialization, concave boundaries, sensitivity to noise and local minima. In this work we present a method to customize PSO to overcome these problems, maintaining its simple structure. The solution has a low order of complexity and consequently a fast processing time, while precisely calculating the differential rotation of solar features.

In previous paper [19] we presented the PSO/Snake hybrid algorithm for tracking CBPs and calculating solar differential rotation. In this article the PSO/Snake approach is compared with Gradient Path Labeling (GPL) segmentation method, proposed by Mora et al. [20], associated with a region matching process to identify the relevant solar features. The GPL segmentation method uses the image gradient as the basis for a pixel labeling procedure which groups ascending paths that belong to the same regional maximum. Its segmentation result is comparable to the Watershed Transform, with the advantage of having a lower over-segmentation effect, good computation efficiency and customizable segmentation effect. The method produces an image segmented in several intensity regions that are then filtered to match the relevant solar features.

In this paper we address the problem of calculating solar differential rotation to evaluate the PSO/Snake algorithm performance in comparison with the GPL method.

We calculate the solar differential rotation as a function of coronal latitude, by tracking some solar feature as markers. We use a dataset of consecutive images taken by the AIA instrument on board the SDO spacecraft. As solar feature markers for tracking we use Coronal Bright Points (CBPs) which are small and bright structures observed in the extreme ultraviolet (EUV) and the X-ray part of the solar spectrum [21]. They are known to have a mean lifetime of about 8 h. CBPs can reach up to $2 \times 10^8 \text{ Km}^2$ in size but still they look like a tiny shape on the solar images. CBPs are associated with bipolar magnetic features and a large quantity of them (several thousands) emerge over the surface of the Sun per day and thereby in total they bring up huge magnetic fluxes. Physicists and space weather scientists will benefit from a CBP tracking system, since these automatic tools allow them to precisely process large amounts of solar data and consequently improve their solar models [19].

The aim of this paper is to compare the result of applying the PSO/Snake and GPL algorithms for tracking coronal bright points. The tracking results are used for calculating the coronal differential rotation and are cross-referenced with pertinent results reported in the literature.

The remainder of this article is presented in the following order: Sect. 2 looks at the PSO/Snake hybrid algorithms and underlying concepts. The GPL algorithm is described in Sect. 3. The results and discussions are provided in Sect. 4 and in Sect. 5 conclusions and future work are presented.

2 PSO/Snake Hybrid Algorithm

We considered three main reasons to apply the customized bio-inspired PSO/Snake algorithm to the problem of tracking CBPs and calculating solar differential rotation:

- It can overcome the problem of noisy data [16]. CBPs are small in size and the solar images that are used to study them have a fair amount of pixel-size structures that make detecting CBPs more difficult. A noise-tolerant algorithm can improve the accuracy of the detection.
- For precise detection of CBPs, while finding and tracking their shape boundaries, several potential local-optima can be encountered in the process. Bio-inspired solutions, in general, are well suited for multimodal problems [22].
- PSO/Snake algorithm uses PSO dynamics. It is an adaptive algorithm, and can be applied in real world problems of dynamic and uncertain natures. CBPs, similar to sunspots [23], are deformable features on the solar disk that evolve during their life span and because of their loose definition, measuring their evolution bears inherent uncertainty.

As mentioned before, two main concepts driving the hybrid PSO/Snake algorithm are snake model, with its parametric contour concept and PSO with its particle movement mechanism. Snake model is driven by an energy minimization concept. It comprises of an energy function which should be minimized in order to find the optimal contour (snake). The function considers the similarity between the contour and the image features (e.g. object boundaries, image gradient, image intensity, texture, color, etc.) as well as the similarity of the contour to a prior model contour [3].

User interaction can also be involved to define the regions of interest for the snake algorithm [5]. After that whereabouts of the Region Of Interest (ROI) is approximated, the snake will evolve to latch to the exact boundary object edges. Snake model is in essence an optimization algorithm. Canonical snake model proposed by Kass et al. [5] implements the minimization process by solving Euler-Lagrange models of the problem. Details of the canonical snake model and a survey on snake model is provided in [24].

In our model, contour or snake has an energy associated with it, which correlates with the location of the snake in the image and its geometrical characteristics. The idea is to minimize the integral measure which represents the total snake energy, by evolving the snake over time. The original algorithm that controls the snake model iteratively solves a pair of Euler equations on the discrete grid, seeking the minimization of the total snake energy [10]. One of drawbacks is that it is time consuming to achieve good snake models.

To represent snakes we used a parametric approach, since it is computationally efficient and easy to interact with users [25]. Here, the snake is defined as a curve $p(s) = (x(s), y(s))$, composed by arc lengths where s is the snaxel point. As it is shown in Eq. (1), a number of discrete points called control points or snaxels characterize the snake [5]. In our implementation we use the parametric snake (curve) presentation as follows:

$$p(s, t) = (x(s, t), y(s, t)), \quad s \in [0, 1] \quad (1)$$

In this equation, time step for each snaxel point s is denoted by t . Equation (2) shows how the total snake energy is calculated as the sum of integrals of the snake internal energy and external energy coming from the image. In the PSO/Snake hybrid algorithm, the objective function calculates the total snake energy. Since in this implementation the whole population altogether represents one candidate solution to the problem, the objective is to find the contour with the least total snake energy. The lesser the total snake energy, the better it matches the ROI or moves towards it.

$$E_{snake} = \int_0^1 E_{int}(p(s))ds + \int_0^1 E_{ext}(p(s))ds \quad (2)$$

The snake model is considered to be a controlled continuity spline under the influence of internal and external forces, which induce the snake energy. Internal energy consists of two terms, the first and the second derivatives of the snake with respect to s . First term coerces the spline to act like a membrane and the second term makes the snake act like a thin plate [5]. The external energy determines the snake relationship to the image. It is formulated in a way that its local minima corresponds to the image features of interest. Various external energies can be employed such as image intensity, image gradient, object size or shape. One common definition used for gray-level images is the gradient of Gaussian.

PSO is a class of evolutionary optimization algorithms, based on a population of particles (swarm), where each one is a potential solution to the optimization problem. It is the leading part of proposed PSO/Snake hybrid algorithm. Besides the information related to the optimization problem, which represents his position in the search space,

each particle is also characterized with a speed. Iteratively, the particles' positions are recalculated according to their velocity, their best solution and also their neighbors' solutions. Each particle position and corresponding fitness score are stored as their best solution and form the cognitive aspect of particle evolution. The influence of neighbors in the position update process is related to their social behavior, which can be defined with various topologies such as ring, star, Von Neumann and random.

If the particle neighborhood is restricted to a subset of swarm it is called local best PSO (*lbest*), while if the neighborhood equals the whole swarm it is called global best PSO (*gbest*). The proposed PSO/Snake hybrid algorithm uses *lbest* with ring structure and radius of 3. The following equations show the dynamics of the canonical PSO algorithm for updating particle velocity and position:

$$v_i(t+1) = \omega(t)v_i(t) + c_1r_1(y_i(t) - x_i(t)) + c_2r_2(\hat{y}_i(t) - x_i(t)) \quad (3)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (4)$$

where $x_i(t)$ and $v_i(t)$ are position and velocity of i -th particle at time t , $y_i(t)$ and $\hat{y}_i(t)$ denote the best solutions discovered by the i -th particle and its neighborhood up to the time t , i.e. *pbest* and *lbest* respectively. $\omega(t)$ is the inertia weight which controls the impact of the previous velocity and prevents radical changes. Usually, inertia weight is decreased dynamically during the iterative process to balance between exploration in the initial iterations and exploitation when converging to a good solution. Coefficients r_1 and r_2 are uniform random numbers in the range $[0,1]$ to introduce stochastic movement to the PSO particles. Weights of cognitive and social aspects of the algorithm are represented by acceleration factors c_1 and c_2 respectively. As it is shown in [26] regulated values for inertia and acceleration weights can be used to achieve guaranteed convergence. Pseudo-code of a typical standard PSO algorithm is presented in Table 1.

Figure 1 shows the block diagram for a typical scenario where PSO/Snake technique is used for object detection and tracking. In [27] it is discussed how a modular design for solar image processing applications could boost the extendibility and reusability of the applications. As the block diagram shows, PSO/Snake model is implemented in a modular way, making it possible to be customized for specific applications. The PSO/Snake hybrid algorithm integrates the snake model mechanisms with PSO dynamics. While most of swarm intelligence approaches in the literature used in conjunction with snake model tries to optimize the snake model equations, PSO/Snake hybrid does not employ PSO algorithm only as a general problem solver to optimize snake energy minimization, but it also customizes the standard PSO to better solve this specific type of image processing problems. Early experiments on medical image segmentation [16] and sunspot tracking [18] reported promising results. The hybrid model helps to overcome the major drawbacks of traditional snakes: initialization and poor convergence to the boundary concavities, while benefitting from PSO robustness and simplicity. In the Hybrid PSO/Snake model we use a population of particles where each particle is a snaxel of the contour. All particles together form the contour and hence the population itself is the final solution. As the algorithm runs, each

particle updates its position and its velocity according to its personal best experience, local best experience, and also according to the internal force of the snake and external force of the image. This gives the PSO/Snake dynamics a wider range of informative guides to update the particle position so that it converges to the ROI.

PSO/Snake hybrid explores the search space according to PSO trajectory disciplines. This eliminates the need to have a separate searching window around each particle as many swarm based snake optimization algorithms do [11, 14, 25]. These methods consider a searching window around each particle and evaluate every position inside that window to determine the snaxels' next position. Since this local search is performed for each particle per iteration, it is a computationally expensive operation that is avoided in the PSO/hybrid model. The velocity update equation in PSO/Snake is as follows:

Table 1. Particle Swarm Optimization pseudo-code

Local-best PSO algorithm	
1:	Initialize_all_particles();
2:	
3:	Do
4:	For each particle
5:	Calculate fitness value
6:	If the fitness value is better than the best fitness value (<i>pbest</i>) achieved so far
7:	Set current value as the new <i>pbest</i>
8:	End For
9:	
10:	For each particle
11:	Set <i>lbest</i> as the particle with the best fitness value in the defined neighborhood
12:	Calculate particle velocity
13:	Update particle position
14:	End For
15:	While maximum iterations or minimum error criteria is not attained.
16:	

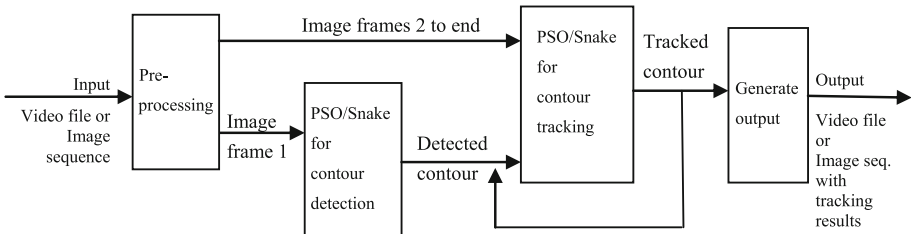


Fig. 1. Block diagram of PSO/Snake algorithm for object detection and tracking

$$\begin{aligned}
v_i(t+1) = & \omega v_i(t) \\
& + c_1 r_1 (pbest_i(t) - x_i(t)) \\
& + c_2 r_2 (lbest_i(t) - x_i(t)) \\
& + c_3 r_3 (\bar{x}(t) - x_i(t)) \\
& + c_4 (f.Image_i)
\end{aligned} \tag{5}$$

where $pbest_i(t)$ and $lbest_i(t)$ are personal best velocity and local best velocity terms respectively. $\bar{x}(t)$ is the average of positions at time step t , approximating the center of mass of the particles. This term pushes the snake to contract or expand with respect to the sign of its weighting factor, r_3 . This term speeds up the algorithm and is particularly useful when the snake is stagnated and there is no other compelling force. If the snake is initialized far from the ROI, this term allows the snake to either expand or shrink towards the ROI and hence it increases the convergence rate and speed. $f.Image_i$ is the normalized image force corresponding to external energy from snake model principles. For particle i , $f.Image_i$ gives the image force at the position specified by that particle. Image gradient or gradient of Gaussian functional are generally accepted for obtaining the image force with acceptable performance. Note that it does not vary along time is computed one single time. c_4 is the weighting factor to control the effect of image force. Inertia weight, ω , is taken to be a relatively small constant and r_1 , r_2 and r_3 denote random numbers. Coefficients c_1 , c_2 , c_3 and c_4 are determined dynamically in a way that, if there is a higher image force c_4 it always gets a higher value. This ensures that if a snaxel is next to the object boundary, it will latch to the object of interest. As Fig. 2 shows, this is implemented by a negative logarithmic function. For each pixel that a particle visits, the dynamic coefficients get negative logarithmic value for the corresponding image force in that pixel. Table 2 provides pseudo-code of the PSO/Snake algorithm used for CBP detection and tracking. A detailed description of the PSO/Snake algorithm is given in [19].

A main difference between PSO and PSO/Snake algorithm is that in the classical PSO population evolves over time, but in the end only one particle (or a limited subset of particles) embody the final solution, where in the PSO/Snake each particle of the population contributes to the solution and final solution is comprised of all particles of the population. In order to control the particle evolution in a tractable manner, velocity control strategies such as velocity strapping is used. PSO being an stochastic approach will provide different results in each because of random seed. PSO/Snake also has stochastic component, but since the initial particle positions are not random, and also because the velocity strapping mechanism along with cognitive and social components of the velocity update prevents particles from drastic changes, the result of several runs of PSO/Snake algorithm is the same. The randomness of the algorithm affects the speed (no of iterations) that it takes to converge to the final results.

Table 2. PSO/Snake pseudo-code

PSO/Snake algorithm for CBP detection and tracking	
1:	// Preparation of input images, i.e. normalization of contrast, resolution, sorting, etc.
2:	Input_Image_Prep ();
3:	
4:	// The region of interest for detecting CBP is selected
5:	Select_ROI ();
6:	
7:	// Based on the selected ROI x_i and v_i vectors are initialized
8:	Best_contour \leftarrow Initial_Contour ();
9:	
10:	
11:	// The weight parameters C_1 , C_2 , C_3 , C_4 , w , and random numbers r_1 , r_2 , r_3 are initialized.
12:	Parameters_Initialization ();
13:	
14:	For each input image
15:	// Calculate $f/Image_i$ with an operator appropriate to the problem, e.g. a GoG function
16:	Calculate_Image_Force ();
17:	
18:	Do
19:	For each particle
20:	Current_contour \leftarrow Best_contour;
21:	
22:	// Set the best velocity snaxel experienced so far
23:	Calculate_pbest ();
24:	
25:	// Set l_{best} as average of velocities of neighboring particles
26:	Calcualte_lbset ();
27:	
28:	// Update the snaxel velocity and position
29:	Move_Snaxels ();
30:	End For
31:	
32:	Best_contour \leftarrow the whole population of snaxels with the lowest energy;
33:	While convergence criteria is not attained.
34:	
35:	Calculate_CBP_Specs ();
36:	End For

3 GPL Algorithm

The Gradient Path Labeling (GPL) segmentation algorithm was designed and proposed to segment retinal images [20] and its accuracy and flexibility made it suitable to be applied in other domains, as it was the case of 2D ion mobility spectra analysis [28] and

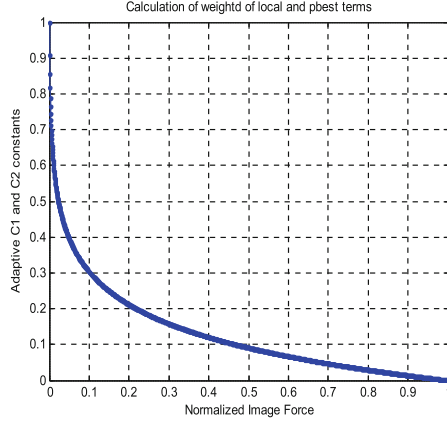


Fig. 2. Dynamic coefficients of C_1 , C_2 and C_3 . These coefficients control the cognitive, social and expansion behavior of the PSO/Snake population with a negative logarithmic functional of image force. Horizontal axis shows the normalized image force and the vertical axis shows the corresponding value for the dynamic coefficients for a pixel with the corresponding image force.

microscopy image analysis [29]. The segmentation and tracking of CBPs in solar images is also a promising domain for the application of GPL, since CBPs are higher intensity regions with distinguishable boundaries. The GPL is a pixel-level segmentation algorithm that groups ascending paths belonging to the same regional maximum resulting in a segmented image where higher intensity regions are labeled individually. The results are comparable to the Watershed Transform although with better noise independence. Another advantage is that it follows a simple pixel labeling approach allowing it to get a fast segmentation with a complexity proportional only to the image size.

The approach to segment CBPs follows a three step process that starts by pre-processing the image in order to reduce noise, followed by the GPL segmentation. Finally, the generated segmentation regions are filtered to select the region that matches a CBP, and its center of mass location is determined. Table 3 presents the pseudo-code for the GPL algorithm.

3.1 Image Preprocessing

In order to get a more accurate segmentation, a preprocessing step is applied to the original image. The first step is to define the CBP initial position and create a squared ROI centered on this location with a predefined width that encompasses the CBP boundaries to limit the complexity of the GPL segmentation filtering process. Then the ROI, as shown in Fig. 3, is filtered by a 3×3 median filter and green channel is selected for the further processing steps (Fig. 4a-c). As it can be seen in Fig. 4b, the median filter is successful in smoothening the image and removing the salt and pepper noise. It should be noted that for this study we have used the JPEG images (accessible from the SDO image repository from NASA) to test the algorithms. However, for the

Table 3. GPL pseudo-code

Gradient Path Labelling	
1:	label_I = function Gradient_Path_Labelling(original_I) {
2:	
3:	// Initial configuration
4:	curlab = 1; // current label
5:	label_I = zeros(size of original_I); // empty matrix for storing labels
6:	regions = new_list(); // regions list
7:	equiv = new_list(); // label equivalences list
8:	
9:	// Get Sobel gradient directions and normalize them to 8-connectivity ($\overrightarrow{G_{norm}}$)
10:	$\overrightarrow{G_{norm}}$ = get_norm_directions (original_I);
11:	
12:	// Iteration over all image pixels
13:	for all $p \in \text{label_I}$ {
14:	if (label_I(p) is empty) { // propagate only from empty labels
15:	label_I(p) = curlab; // set new label
16:	$p_{next} = p + \overrightarrow{G_{norm}}(p)$; // get next pixel coordinates
17:	
18:	// label propagation
19:	while ((p_{next} is inside boundaries) and (label_I(p_{next}) is empty)) {
20:	label_I(p_{next}) = curlab; // set to the current label
21:	$p_{next} = p_{next} + \overrightarrow{G_{norm}}(p)$; // get next pixel coordinates
22:	}
23:	
24:	// stop condition: outside the image boundaries or ended on the same label
25:	if (p_{next} is not inside boundaries or (label_I(p_{next}) == curlab))
26:	regions.add(p_{next}); // add a new regional maximum
27:	
28:	else // ended on a different label
29:	equiv.add(curlab, label_I (p_{next})); // define them as compatibles
30:	
31:	curlab = curlab + 1; // set next label value
32:	}
33:	}
34:	
35:	label_I = apply_equivalences(label_I, equiv); // get final labelled image
36:	label_I = merge_labels(label_I, regions) // label merging stage
37:	}
38:	
39:	
40:	

astrophysical applications the use of the FITS file format, that preserves the raw observation data, is common. In the AIA images used in this study, solar feature locations and boundaries are clearer in the green channel; in the blue channel only the more intense events can be perceived; and in the red channel high intensity CBPs location can usually be observed but less intense ones cannot be perceived (Fig. 3). To obtain the CBP boundaries using the GPL segmentation, it should be applied to the gradient image using the Sobel operator. Comparing the three segmentation results (see

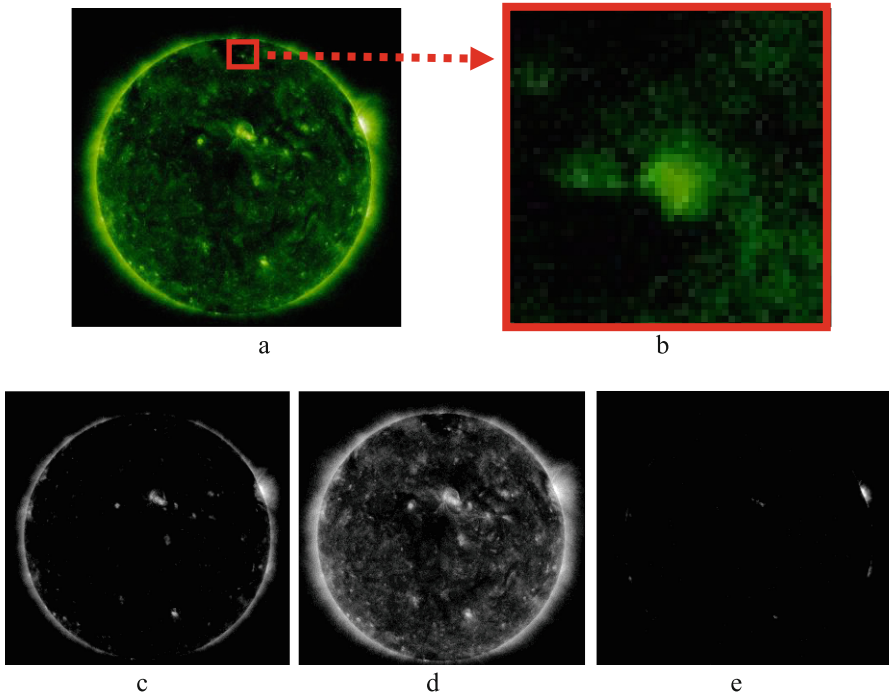


Fig. 3. Solar image of AIA 94Å. (a) Original image with a CBP marked, (b) a zoomed view of the CBP, (c) Red channel of the original RGB-color image, (d) Green channel, (e) Blue channel. (Image courtesy of NASAS/DO) (Color figure online)

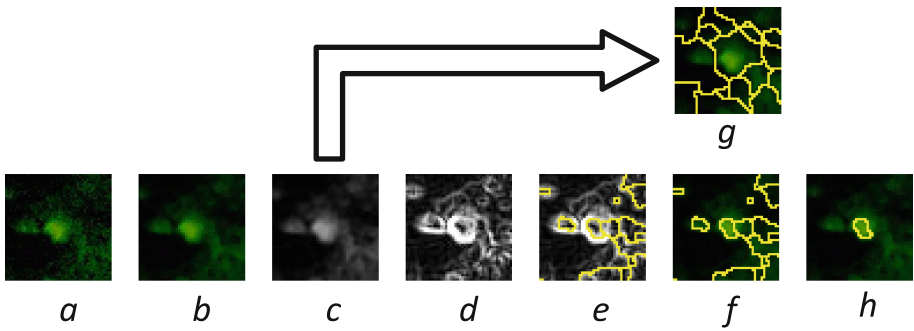


Fig. 4. Example of a CBP detection. (a) original image; (b) median filtered image; (c) green channel; (d) gradient image using Sobel operator; (e) GPL segmentation contours over the gradient image; (f) and (g) GPL segmentation contours over the green image respectively with and without gradient preprocessing; (h) final GPL result. (Color figure online)

Fig. 4) one can see that in the upper image (g) CBP is correctly segmented, although the region boundaries do not overlap the CBP boundaries. However, by applying the GPL to the gradient image (f) this problem is resolved and the CBP boundaries match the segmentation boundaries.

3.2 GPL Image Segmentation

By visualizing an image gradient in a quiver plot it can be noticed that in the proximity of higher intensity regions gradient vectors are directed towards their intensity maximum (Fig. 5a). The confluence of several ascending paths reveals the presence of a regional intensity maximum and the pixels belonging to these paths define its area of influence. This is the principle of the Gradient Path Labeling segmentation algorithm [20]. The algorithm is divided into two main stages: label propagation and labels merging.

The labelling process starts by a sequential pixel analysis following a top-left to bottom-right direction. In this sequence each unlabeled pixel will receive a new label (sequential number) that will be propagated to other pixels along the gradient path. This path is composed by azimuths, obtained using the 3×3 Sobel operator (Fig. 5b), that follow an ascending intensity path. This propagation continues until an already labelled or outside image boundaries pixel is found (Fig. 5c and d). Whenever we have the confluence of two paths, i.e., the propagation ends on a labelled pixel, the two path labels are defined as *equivalents* (for example labels 2, 4, 6, 10 in Fig. 5d and e).

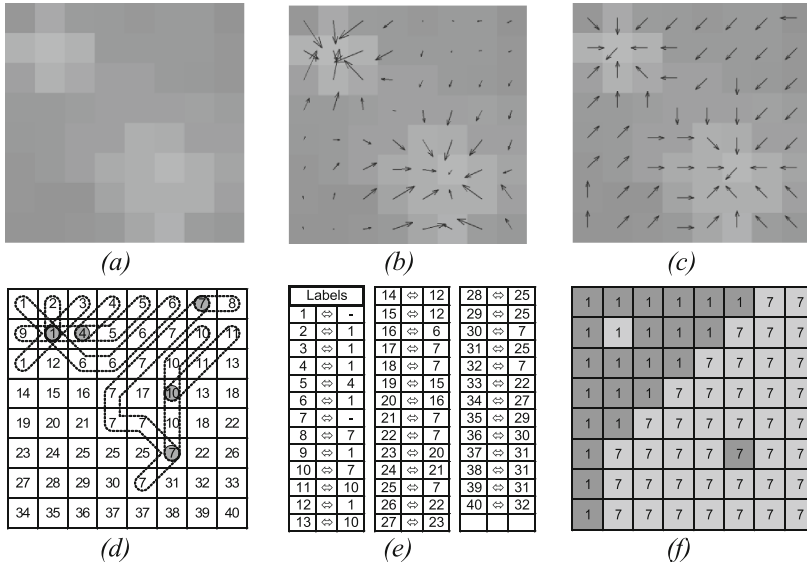


Fig. 5. Example of GPL segmentation algorithm. (a) Original image; (b) Image gradient; (c) Label propagation directions; (d) Initial label propagation – (label propagation on the two upper rows); (e) Label equivalences table; (f) Segmentation results and their maximum point highlighted.

After analyzing all the image, the equivalent labels will be merged together and replaced by the lowest one (Fig. 5f). The GPL result is a segmented image with as many labels as higher intensity regions.

The GPL algorithm has a tendency to produce over-segmented images, in particular when they have flat valleys or flat hills. A merging procedure was introduced to overcome this problem. The merging algorithm is based on a connectivity graph analysis where adjacent regions are merged if they can be connected by a path that does not go lower than a predefined amplitude. At this stage regional maximums are detected and their area of influence is defined.

3.3 CBP Matching

The GPL segmentation produces several regions from which the CBP must be selected. From the segmented regions the most discriminant CBP features are based on color and intensity rather than shape, since CBPs do not have a predefined and static shape. The gradient of the image and the maximum, minimum and mean intensities for each color channel are extracted, as well as the centroid (x, y) location and the difference to its previous location. From a statistical analysis of the extracted features, the most discriminant ones are red and green maximum and mean intensities and centroid location difference. Using these parameters, a multi-criteria score Eq. (6) was computed considering 50 % for location and 50 % for color features. As a result, the highest scored region is selected as the tracked CBP. As it can be seen in the Fig. 4h, GPL is able to detect the CBP boundaries, although at the cost of an over-segmentation effect. By computing the ranking Eq. (6) we are able to identify and select the correct CBP region and track it along consecutive images.

$$\begin{aligned}
 score = & 50\% * rank(centroid\ distance) \\
 & + 12.5\% * rank(red\ maximum) \\
 & + 12.5\% * rank(red\ mean) \\
 & + 12.5\% * rank(green\ maximum) \\
 & + 12.5\% * rank(green\ mean)
 \end{aligned} \tag{6}$$

Due to the CBP diversity in intensity and shape this ranking based criteria obtained a better selection accuracy when compared to a relative or absolute one, obtained using computational intelligence techniques such as decision trees. However, whenever the CBP is very fade or simply disappears this selection process will fail to reject all regions. The use of computational intelligence could be applied after the selection process to reject or accept the selected region as a CBP. Since in this work this step was not implemented, regions are selected only if a manual evaluation is available.

4 Results and Discussions

4.1 Dataset and Experiment Setup

As benchmark data we use SDO-AIA corona images, downloaded from NASA-SDO repository¹. We have used selected images taken at 9.4 nm wavelength in the timespan between 14 September 2010 and 20 October 2010. For comparison purposes we use the manual CBP marking done by an expert in [30] and the database used in [19]. The tracking process starts by choosing a CBP to track. In this comparative study we the CBP marking data done by an expert as our benchmark data. After choosing a CBP, we run PSO/Snake and GPL algorithms independently on that CBP. Each CBP is chosen in the first image and then tracked automatically until it disappears below a predefined size. GPL uses similar criteria, but the decision about stopping the algorithm is embedded in the CBP matching unit. That is if a CBP cannot be matched with a predefined confidence level, the tracking stops. Determining the exact moment for start and ending the lifetime of a CBP is subjective. In this study, since we are comparing the precision of tracking the CBP movement, we consider the life time of CBP according to the manual data availability. For PSO/Snake algorithm the input images are converted to 8-bit grayscale and image force is calculated by a gradient of Gaussian functional with $\sigma = 3$. Images are resized to 512×512 pixels. Altogether, in GPL algorithm we have tested 41 CBP structures, being tracked in 6 days (3098 measurements). PSO/Snake parameters and running conditions are presented in Table 4.

A screenshot of PSO/Snake CBP tracking is shown in Fig. 6. The red circle shows the initial snake around a CBP chosen by an operator. After detection, each CBP is characterized by the heliographic coordinates their center of mass. On the next frame the previous CBP contour is used as a baseline to automatically track its new position. In Fig. 7 is presented the snake contour evolution for a tracked CBP. It shows that, due to the dynamic nature of PSO/Snake hybrid algorithm, detected contours are flexible and can adjust to the dynamic shape and size of deformable objects like CBPs. Figure 8 shows an example of GPL segmentation and tracking of a CBP region in five consecutive images.

4.2 Comparison of Results

First we compare the precision of the GPL and PSO/Snake algorithm in tracking CBP movements against the ground truth obtained by manual tracking data. The calculated angular rotational velocity of solar corona is compared against reported values in the literature to test the feasibility of using these methods in solving real world problems. In the manual procedure [30], an expert operator determined the CBPs positions as they moved. In our study we use the manual CBP markings as our benchmark data.

Several parameters that were reported in the original reference papers [30] and [19] were used for comparison purposes in this paper. Reference [30] manually marks CBPs for the entire dataset, while in [19] PSO/Snake algorithm is used for automatic CBP

¹ <http://sdo.gsfc.nasa.gov/data/aiahmi/browse.php>.

Table 4. PSO/Snake parameters and running conditions

Parameter	Value
No. of iterations	750
No. of particles	15
Image force	Gradient of Gaussian
Image force normalization?	Yes
Gradient sigma	3
C_1, C_2, C_3	Dynamic
C_4	1
r_1, r_2, r_3	Random numbers in range (0,1)
w	0.01
Max velocity	1

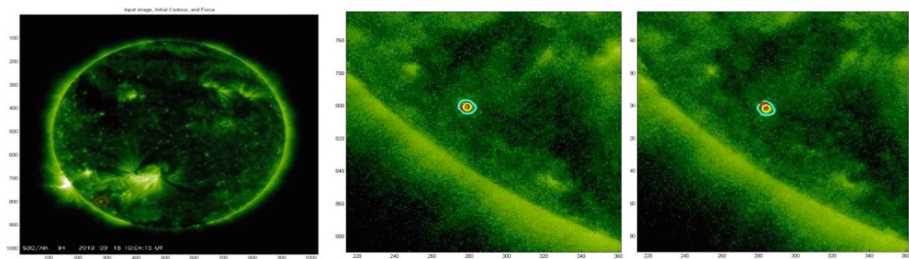


Fig. 6. Screenshot of the PSO/Snake CBP tracking. (Left) initial snake on the first image at time t , (Center) the detected CBP at time t , (Right) the tracked CBP at time $t + 10$. The boundary fitting snake is shown with cyan contour, expert’s marking is shown with red square and the yellow circle is the center of mass for the tracked CBP by PSO/Snake. (Color figure online)

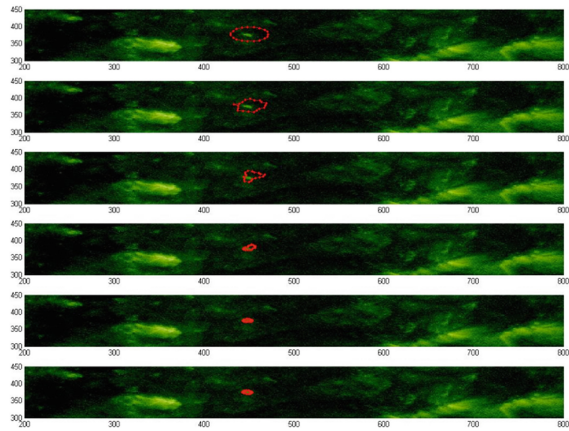


Fig. 7. The evolution of the snake to detect CBP boundaries can be seen.

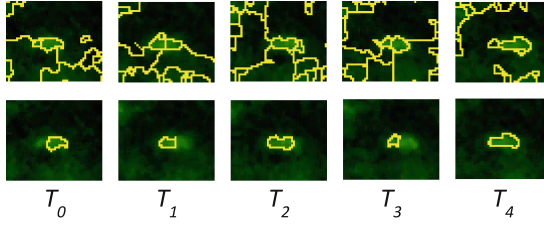


Fig. 8. Examples of GPL segmentation and tracking of the CBP region in five consecutive frames.

tracking. Based on the CBP positions, angular rotational velocity of the Sun is calculated. Both references report ω (angular rotation velocity) and $\Delta\omega$ (measurement error at 95 % confidence level). In this study we report measurements for these parameters using the GPL algorithm. Figure 9 shows the calculated rotational speed in different solar latitudes in this study in comparison with the values obtained by other authors. Further details can be found in [30]. The + markers in this image show the tracked CBP structures that, as it can be seen, are well scattered between the $\pm 60^\circ$ latitude but not in proximity of the solar limb. It is a technically challenging problem to study the solar disk near the limb, since the projection of the sphere shape of the Sun into a 2D image causes projection errors that are even more important in the limbs. For this reason, we focus on comparing the results in the $\pm 5^\circ$ solar latitude (Fig. 9 Bottom). This figure illustrates the main results comparing the PSO/Snake algorithm vs. GPL algorithm for calculating solar differential rotation. As it can be seen in the figure, both the PSO/Snake and GPL algorithms perform well in approximating the solar rotational velocity in comparison with the ground truth and other reported results. Both methods generate acceptable results matching the scientific community state of the art data. However, PSO/Snake has better conformity with previous results, and a more smooth curve-fitting results, which improves its extendibility in dealing with bigger databases.

To take a closer look at the performance of two methods, results obtained for some sample CBP structures is reported in Table 5. This table shows the comparison between the results obtained with the manual CBP tracking, the results obtained by the PSO/Snake hybrid algorithm and the GPL algorithm for some sample structures. In this table, the structure is the identifier of CBP, b is the heliographic latitude of CBP and ω E is the orbital angular rotation velocity of the Earth that can be looked up from solar almanacs. As with Fig. 9, Table 5, shows that the obtained results in both methods are very close to the results of manual CBP tracking, but with some deviations.

By extending the CBP tracking results for whole life-time of a CBP, we can calculate the accumulated error for the calculated ω by each CBP. Figure 10 show the deviation of the results from the benchmark data for b , ω , $\Delta\omega$, and ω E, for the PSO/Snake results and the GPL results of all CBP structure during their lifetime. The PSO/Snake algorithm measures the rotational speed of the Sun within ± 0.2 of the benchmark data most of the time. For the GPL algorithm, the slight error in calculation of b leads to a greater offset in ω results. The error in this case is -18 degree/day in the

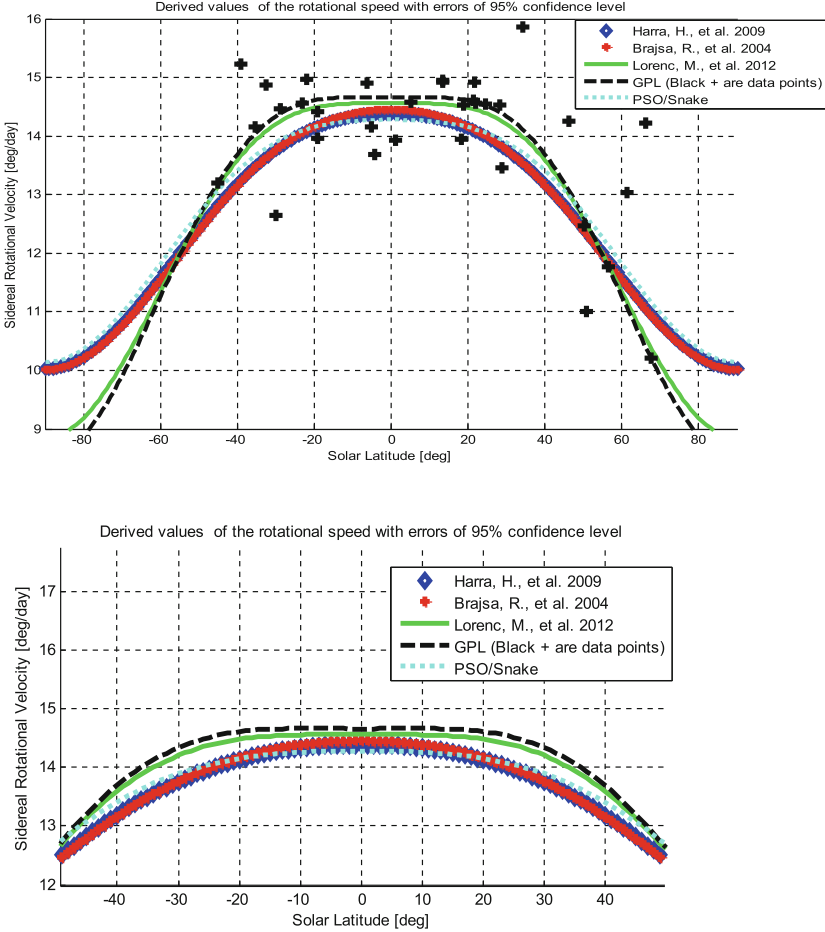


Fig. 9. Rotational speed in different solar latitudes calculated by tracking CBPs in comparison with the values obtained by other works (*Top*). The black dashed curve and the cyan dotted curve show the fit to the mean $\omega(b)$ values as a function of latitude b , for the GPL (present work) and the PSO/Snake algorithms (based on [19]) respectively. Results from [30–32] are superimposed for comparison. The black + markers are the calculated GPL data points (*Bottom*). A cropped and zoomed view of the plot above, confined in the ± 50 degree solar latitude. In both images the confidence level is set to 95 %. (Color figure online)

highest. This shows that accumulated deviation error for PSO/Snake algorithm is less than the GPL method. It should be noted that part of this deviation is due to code implementation differences, which, in precise calculations, impose a minute variation. It is also worth mentioning that, in several cases, results displayed bigger differences. Closer investigation by a solar physicist expert (co-author), found out that the PSO/Snake hybrid algorithm behaves consistently and that the user-error is the main cause.

Table 5. Comparison of the results reported in [30] with results from PSO/Snake [19] and GPL algorithms for some sample structures

Structure	Parameter	PSO/Snake	Benchmark	GPL
xy0510.01	b	67.103	66.7	67.57
	ω	11.213	10.295	10.21
	$\Delta\omega$	± 0.642	± 0.327	± 0.411
xy0510.03	b	21.057	20.5	21.367
	ω	14.387	14.586	14.611
	$\Delta\omega$	± 0.303	± 0.099	± 0.182
xy0510.04	b	-32.27	-33.8	-32.409
	ω	13.803	13.648	14.878
	$\Delta\omega$	± 0.342	± 0.209	± 0.432
xy0510.07	b	28.170	27.8	28.290
	ω	15.112	14.478	14.530
	$\Delta\omega$	± 0.439	± 0.116	± 0.226

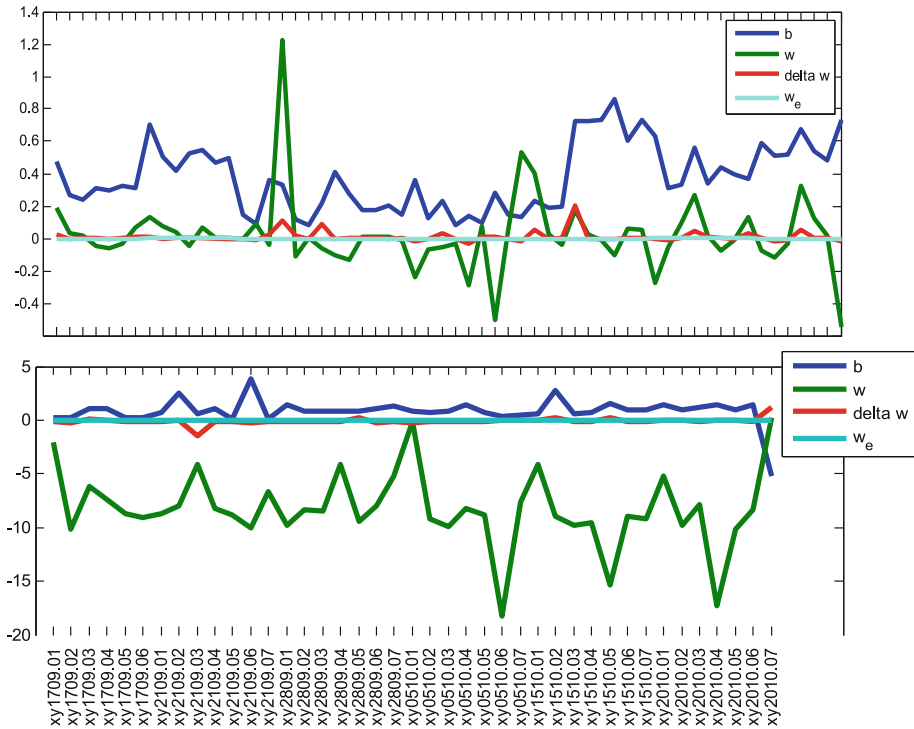


Fig. 10. Deviation of the results from the benchmark data. The differences in b , ω , $\Delta\omega$, and ω_E are represented with blue, green, red and cyan lines. (Top) The PSO/Snake results error (based on [19]) compared to the benchmark data are within 1.5 degree/day. (Bottom) The GPL results error compared to the benchmark data are within 18 degree/day. (Color figure online)

5 Conclusions

In our previous paper [19] we presented the PSO/Snake hybrid algorithm for solving a real solar physic and space weather problem, i.e. tracking CBPs and calculating solar differential rotation. This study compares the PSO/Snake hybrid algorithm and Gradient Path Labeling algorithm. Both methods are applied for determining the speed of sidereal angular rotation of the Sun by analyzing solar images and tracking specific features on a series of successive images. Based on the analysis of the results and comparison with a manual method the resulting values of rotational speed are reliable for both methods. The deviation between these two techniques from the benchmark was similar and both methods calculated solar rotational velocity comparable to the established values reported in the literature. However, by calculating the accumulated deviation during the extended lifetime of all CBP structures, PSO/Snake algorithm demonstrated better performance and lower accumulated deviation. This is because of the dynamic nature of the PSO/Snake algorithm.

For tracking CBP in every new image, information about the CBP profile and its contour is passed from the previous image to the current state, and these data are used only as starting point to recalculate the CBP contour in the current image. The GPL algorithm benefits from the lower computational complexity outperforming the PSO/Snake in calculation speed. PSO/Snake algorithm is an iterative process, which increases its execution time. Particularly calculating image force matrix is the bottleneck of this algorithm. Although GPL achieved reasonable results, PSO/Snake showed a superior performance especially handling the incomplete information and extendibility of the method.

The hybrid combination of PSO with snake model, preserved the PSO simplicity and overcome some of the snake drawbacks. Adding two new terms to the PSO velocity calculation increased the robustness of the algorithm allowing it to evolve even if some component is missing or misleading. The particle/snaxels velocity information embedded in this algorithm, makes it more suitable for object tracking in image processing applications, since it adapts itself to the movement of the object in the images. PSO/Snake algorithm is particularly good in handling noisy data, such as tested solar images. CBPs are small in size and the solar images that are used to study them have a fair amount of pixel-size structures that make detecting CBPs more difficult.

PSO/Snake algorithm is not a general problem solver like PSO. It takes advantage of PSO dynamics, for pushing particle to find the object boundaries in an image. The PSO/Snake algorithm has already been successfully tested for detection and tracking of small deformable structures such as endothelium cells from cornea microscopic images [17] and tracking sunspots [18] and its successful application on CBP test problem can be extended to other problem domains with similar nature. It can be applied in real world problems with dynamic and uncertain nature, where tracking a deformable-shape object is desired.

This work does not consider the splitting and merging of CBPs. As future work we plan to extend the capability of the algorithms for automatic detection of CBPs and analyzing their shape relative to other objects adjacent to them. This will enable the algorithm to determine if a CBP is splitting or merging. A combination of the current

PSO/Snake algorithm for tracking and the GPL capability in CBP matching can be used for implementation of this feature.

Acknowledgments. This work was partially supported by Fundação para a Ciência e a Tecnologia (FCT), MCTES, Portugal through grants SFRH/BPD/44018/2008 (I.D.) and SFRH/BD/62249/2009 (E.S.) and by FCT Strategic Program UID/EEA/00066/203 of UNINOVA, CTS. We would like to also thank the SDO (NASA) and AIA science team for the provided observational material.

References

1. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: IEEE International Conference on Neural Networks, Proceedings, pp. 1942–1948. IEEE, Perth (1995)
2. Clerc, M.: Particle swarm optimization. ISTE Ltd (2006)
3. Morel, J.-M., Solimini, S.: Variational methods in image segmentation: with seven image processing experiments, vol. 14. Springer, Boston (2012)
4. McInerney, T., Terzopoulos, D.: Deformable models in medical image analysis: a survey. *Med. Image Anal.* **1**, 91–108 (1996)
5. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: active contour models. *Int. J. Comput. Vis.* **1**, 321–331 (1988)
6. Ballerini, L., Bocchi, L.: Multiple Genetic Snakes for Bone Segmentation. In: Raidl, G.R., Cagnoni, S., Cardalda, J.J., Corne, D.W., Gottlieb, J., Guillot, A., Hart, E., Johnson, C.G., Marchiori, E., Meyer, J.-A., Middendorf, M. (eds.) *EvoIASP 2003, EvoWorkshops 2003, EvoSTIM 2003, EvoROB/EvoRobot 2003, EvoCOP 2003, EvoBIO 2003, and EvoMUSART 2003*. LNCS, vol. 2611, pp. 346–356. Springer, Heidelberg (2003)
7. Tsechpenakis, G., Rapantzikos, K., Tsapatsoulis, N., Kollias, S.: A snake model for object tracking in natural sequences. **19**, 219–238 (2004)
8. Niu, X.: A Geometric active contour model for highway extraction. In: *Proceedings of ASPRS 2006 Annual Conference*, Reno, Nevada (2006)
9. Wildenauer, H., Blauensteiner, P., Hanbury, A., Kampel, M.: Motion detection using an improved colour model. In: Bebis, G., et al. (eds.) *ISVC 2006*. LNCS, vol. 4292, pp. 607–616. Springer, Heidelberg (2006)
10. Karlsson, A., Stråhlén, K., Heyden, A.: A fast snake segmentation method applied to histopathological sections. In: *Energy Minimization Methods in Computer Vision and Pattern Recognition*. pp. 261–274. Springer Berlin Heidelberg (2003)
11. Tseng, C., Hsieh, J., Jeng, J.: Active contour model via multi-population particle swarm optimization, (2009)
12. Li, R., Guo, Y., Xing, Y., Li, M.: A Novel Multi-Swarm Particle Swarm Optimization algorithm Applied in Active Contour Model. In: *Intelligent Systems, 2009. GCIS '09. WRI Global Congress on*. pp. 139–143. IEEE (2009)
13. Ballerini, L.: Genetic snakes for medical images segmentation. In: Poli, R., Voigt, H.-M., Cagnoni, S., Corne, D.W., Smith, G.D., Fogarty, T.C. (eds.) *EvoIASP 1999 and EuroEcTel 1999*. LNCS, vol. 1596, pp. 59–73. Springer, Heidelberg (1999)
14. Nebti, S., Meshoul, S.: Predator prey optimization for snake-based contour detection. *Int. J. Intell. Comput. Cybern.* **2**, 228–242 (2009)

15. Zeng, D., Zhou, Z.: Invariant topology snakes driven by particle swarm optimizer. In: 2008 3rd International Conference on Innovative Computing Information and Control. p. 38. IEEE (2008)
16. Shahamatnia, E., Ebadzadeh, M.M.: Application of particle swarm optimization and snake model hybrid on medical imaging. In: 2011 IEEE Third International Workshop on Computational Intelligence in Medical Imaging. pp. 1–8. IEEE, Paris, France (2011)
17. Sharif, S.M., Qahwaji, R., Shahamatnia, E., Alzubaidi, R., Ipson, S., Brahma, A.: An efficient intelligent analysis system for confocal corneal endothelium images. *Comput. Methods Programs Biomed.* **122**, 421–436 (2015)
18. Shahamatnia, E., Dorotovič, I., Ribeiro, R.A., Fonseca, J.M.: Towards an automatic sunspot tracking: Swarm intelligence and snake model hybrid. *Acta Futur.* **5**, 153–161 (2012)
19. Shahamatnia, E., Dorotovič, I., Fonseca, J.M., Ribeiro, R.A.: An evolutionary computation based algorithm for calculating solar differential rotation by automatic tracking of coronal bright points. *J. Sp. Weather Sp. Clim.* **6**, A16 (2016)
20. Mora, A.D., Vieira, P.M., Manivannan, A., Fonseca, J.M.: Automated drusen detection in retinal images using analytical modelling algorithms. *Biomed. Eng. Online.* **10**, 59 (2011)
21. Brajša, R., Wöhl, H., Ruždjak, V., Clette, F., Hochedez, J.-F.: Solar differential rotation determined by tracing coronal bright points in SOHO-EIT images I. Interactive and automatic methods of data reduction. *Astron. Astrophys.* **374**, 309–315 (2001)
22. Gálvez, A., Iglesias, A.: A new iterative mutually coupled hybrid GA–PSO approach for curve fitting in manufacturing. *Appl. Soft Comput.* **13**, 1491–1504 (2013)
23. Shahamatnia, E., Dorotovi, I., Mora, A., Fonseca, J., Ribeiro, R.: Data inconsistency in sunspot detection. In: Filev, D., et al. (eds.) *Intelligent Systems 2014*, pp. 567–577. Springer, Cham (2015)
24. Chen, B., LAI, J.H.: Active contour models on image segmentation: a survey. *J. Image Graph.* **1**, (2007)
25. Horng, M.-H., Liou, R.-J., Wu, J.: Parametric active contour model by using the honey bee mating optimization. *Expert Syst. Appl.* **37**, 7015–7025 (2010)
26. Van den Bergh, F.: An analysis of particle swarm optimizers, (2002)
27. Shahamatnia, E., Dorotovic, I., Fonseca, J., Ribeiro, R.: On the importance of an automated and modular solar image processing tool. In: *Proceedings of the European Planetary Science Congress (EPSC)*, Portugal (2014)
28. Hakkinen, A., Muthukrishnan, A.-B., Mora, A., Fonseca, J.M., Ribeiro, A.S.: Cell Aging: a tool to study segregation and partitioning in division in cell lineages of *Escherichia coli*. *Bioinformatics* **29**, 1708–1709 (2013)
29. Häkkinen, A., Muthukrishnan, A.B., Mora, A., Fonseca, J.M., Ribeiro, A.S.: Cell Aging: a tool to study segregation and partitioning in division in cell lineages of *Escherichia coli*. *Bioinformatics* **29**, 1708–1709 (2013)
30. Lorenc, M., Rybanský, M., Dorotovič, I.: On rotation of the solar corona. *Sol. Phys.* **281**, 611–619 (2012)
31. Hara, H.: Differential rotation rate of X-ray bright points and source region of their magnetic fields. *Astrophys. J.* **697**, 980 (2009)
32. Brajša, R., Wöhl, H., Vršnak, B., Ruždjak, V., Clette, F., Hochedez, J.-F., Roša, D.: Height correction in the measurement of solar differential rotation determined by coronal bright points. *Astron. Astrophys.* **414**, 707–715 (2004)

Transactions on Computational Collective Intelligence
XXIV

Nguyen, N.-T.; Kowalczyk, R.; Filipe, J. (Eds.)

2016, IX, 169 p. 78 illus., Softcover

ISBN: 978-3-662-53524-0