Use of Particle Swarm Optimization for ODF maxima extraction

Mouloud KACHOUANE-Non Member, Thininane MEGHERBI-Non Member, Fatima OULEBSIR-BOUMGHRAR-EMBS Member, and Rachid DERICHE-EMBS Member

Abstract—Fiber tracking is gaining more and more interest in the neuroscience research field and clinical practice, for its ability in revealing the structural connectivity; the quality of the fiber tracking depends in great extent, on fiber directions extraction. The PSO algorithm could give good approximation of these directions.

I. MATERIALS AND METHODS

The diffusion tensor MRI (DTI) was introduced by Basser et al. 1994 [1], it captures and quantifies the free or constrained water molecules movement; symmetric positive definite second order diffusion tensors were used to model the profile of the diffusion for each voxel, with the assumption of a single bundle fiber per voxel. Thus, tractography algorithms based on DTI may produce unreliable results. To overcome this limitations, new methods and techniques of High Angular Resolution Diffusion Imaging (HARDI) [2] have been proposed: the Q-ball imaging (QBI), [3] with a spherical sampling of the diffusion space, Diffusion Spectrum Imaging (DSI) with a sampling of the entire Cartesian grid 3D space diffusion, spherical deconvolution techniques.

These techniques allow the reconstruction of the multi-fiber by calculating probability density function (PDF) which is estimated by the Orientation Density Function (ODF) whose maxima are aligned with the actual directions of the fibers. The importance of tractography in clinical studies makes the ODF maxima extraction a crucial post-processing step, since the ODF provides the angular information by having its maxima aligned on the underlying fiber directions.

In the present work, we propose a new ODF maxima search approach using the algorithm Particle Swarm Optimization (PSO), to efficiently and accurately extract all the fibers directions. PSO introduced by Kennedy and Eberhart [4] in 1995, is an optimization technique of adaptive research based on population. Through a trial and error process [5], PSO assigns a randomized velocity to each potential solution, called particle, and then fly through the problem space.

A. The PSO Algorithm

Each particle represents a potential solution in the search space. The new position of a particle is determined by its own value and that of its neighbors. In the algorithm \( \bar{x}_i(t) \) is the position of the particle \( P_i \), at instant \( t \), which changes by adding a velocity \( \bar{v}_i(t) \) to its current position. \( \bar{v}_i(t) \) is the particle \( i \) velocity at the instant \( t \) and \( x_i(t) \) is the position of the particle \( i \) at instant \( t \), parameters: inertia \( \omega \), learning coefficients \( c_1 \) and \( c_2 \) are constants set by the user, \( 0 \leq \omega \leq 1.2 \), \( 0 \leq c_1 \leq 2 \) and \( 0 \leq c_2 \leq 2 \) [6] coefficients \( r_1 \) and \( r_2 \) are random numbers drawn at each iteration, \( g(t) \) is the best solution found so far and \( t \) \( x_p(t) \) is the best solution found by the particle \( P_i \); in the following algorithm \( N \) is the number of particles; \( p_{best_i} \) Best fitness obtained for the particle \( P_i \); \( \bar{x}_{pbest_i} \) Position of the particle \( P_i \) for her best fitness; \( \bar{x}_{gbest} \) Position of the particle with the best fitness of all.

PSO Algorithm

\begin{align}
\text{Begin} \\
\text{Repeat} \\
\text{For } i \text{ from } 1 \text{ to } N \text{ do} \\
\text{If } (F(\bar{x}_i) > p_{best_i}) \text{ Then} \\
\quad p_{best_i} \leftarrow F(\bar{x}_i) \\
\quad \bar{x}_{pbest_i} \leftarrow \bar{x}_i \\
\text{End If} \\
\text{If } (F(\bar{x}_i) > g_{best}) \text{ Then} \\
\quad g_{best} \leftarrow F(\bar{x}_i) \\
\quad \bar{x}_{gbest} \leftarrow \bar{x}_i \\
\text{End If} \\
\text{End For} \\
\text{For } i \text{ de } 1 \text{ à } N \text{ faire} \\
\quad \bar{v}_i \leftarrow \bar{v}_i + r_1 c_1 (\bar{x}_{pbest_i} - \bar{x}_i) + r_2 c_2 (\bar{x}_{gbest} - \bar{x}_i) \\
\quad \bar{x}_i \leftarrow \bar{x}_i + \bar{v}_i \\
\text{End For} \\
\text{Until (the process converges)}.
\end{align}

\text{End}

To optimize the algorithm and find the maxima efficiently, we evaluated the behavior of PSO according to different values of the constant parameters: inertia and learning coefficients \( c_1 \) and \( c_2 \).

We have chosen to minimize four functions of dimension 2:

Two functions: Rosenbrock and Zakharov functions with a single minimum and Shafer and Himmelblau functions with multi-minimum.

B. PSO parameterization

a. Inertia

To evaluate the effect of inertia we set learning factors at \( c_1 = c_2 = 2 \), and do five tests for each value of inertia \( \omega \).

The results of these tests are represented on fig. 1, for different values of \( \omega \) between 0 and 1. For functions with only one minimum, the search for the optimum is achieved efficiently by using fixed values of inertia below 0.8. To deal with multi-minima problems, the best result is obtained when
the inertia is variable and decreases between 0.9 and 0.4 in each iteration.

![Reaching the optimum according to the inertia](image)

Fig. 1. PSO performance in function of inertia values

b. Learning coefficients

We conducted other tests to evaluate the effect of learning factor, \( w = 0.8 \) for first two functions and decreasing from 0.9 to 0.4 for the others. Table I illustrates the success rates for four benchmark functions when the learning factors, \( c_1 \) and \( c_2 \), are both set from 1 to 3.

<table>
<thead>
<tr>
<th>TABLE I. LEARNING COEFFICIENTS TESTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>( c_1/c_2 )</td>
</tr>
<tr>
<td>----------------</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
</tbody>
</table>

For single minimum functions (table 1.a and 1.b), the algorithm is efficient when the sum of the two coefficients is in the vicinity of four. In case of multi-minimum, PSO loss effectiveness for high values of learning factors.

C. Orientation Distribution Function (ODF)

The ODF function is estimated and expressed in the spherical harmonics basis:

\[
S(\theta, \phi) = \sum_{j=1}^{R} c_j Y_j(\theta, \phi)
\]

\( Y_j \) are the spherical harmonics components, \( \theta \) and \( \phi \) are the spherical coordinates.

Since the spherical function ODF is assumed to be real and antipodally symmetric, a modified version of this basis, real symmetric spherical harmonics, has been introduced in [4] and given by the following expressions:

\[
Y_j = \begin{cases} 
\sqrt{2} \text{Re}(Y_{jm}^m), & \text{if } m < 0 \\
Y_{jm}^m, & \text{if } m = 0 \\
\sqrt{2} (-1)^{m+1} \text{Im}(Y_{jm}^m), & \text{if } m > 0 
\end{cases}
\]

(4)

The index \( j \) is the order and \( m \) the degree.

\[
j(l, m) = (l^2 + l + 2)/2 + m
\]

(5)

D. Method

The core idea of our proposition is the use of the Particle Swarm Optimization.

The first step is to construct a 2D image from the ODF values (eq. 3) considering \( \theta \) and \( \Phi \) as the resulted image coordinates and the ODF value as the gray level [7]. In this case the image would have (180, 360) as dimensions. The next step is to search maxima using the PSO algorithm.

The PSO algorithm is efficient to extract the global maxima, indeed using the coordinates of each point and its gray level value as fitness, PSO allows finding the global maxima accurately. However, this algorithm is not suitable to detect local maxima, therefore we propose to divide the image into blocks. And launch a swarm of PSO in each block.

We tested our parameterized PSO on some synthetic ODF images of 2 and 3 lobes, and the error was of magnitude 1/100 of a degree which is better of our previous results [7].

![Two lobes ODF maxima extracted by PSO](image)

Fig. 2. Two lobes ODF maxima extracted by PSO

RESULTS

The first tests of PSO parameters that we conducted on two dimensional functions allowed us to use the algorithm efficiently to extract global maxima. That’s promising results suggested us to use Particle Swarm Optimization for the ODF maxima search.

By adapting the PSO to the local maxima search, we was able to extract the ODF maxima with precision on synthetic data.

Finally, we think really that PSO can be suitable to ODF maxima extraction, and as future work we intend to conduct more validation tests on synthetic noisy data, phantom and real diffusion data. Furthermore, we can compare it to other techniques.

REFERENCES


