Fast Clustering for Interactive Tractography Segmentation

***Abstract*—We developed a novel interactive system for human brain tractography segmentation to assist neuroanatomists in identifying white matter anatomical structures of interest from diffusion magnetic resonance imaging (dMRI) data. The difficulty in segmenting and navigating tractographies lies in the very large number of reconstructed neuronal pathways, i.e. the streamlines, which are in the order of hundreds of thousands with modern dMRI techniques. The novelty of our system resides in presenting the user a clustered version of the tractography in which she selects some of the clusters to identify a superset of the streamlines of interest. This superset is then re-clustered at a finer scale and again the user is requested to select the relevant clusters. The process of re-clustering and manual selection is iterated until the remaining streamlines faithfully represent the desired anatomical structure of interest. In this work we present a solution to solve the computational issue of clustering a large number of streamlines under the strict time constraints requested by the interactive use. The solution consists in embedding the streamlines into a Euclidean space and then in adopting a state-of-the art scalable implementation of the k-means algorithm. We tested the proposed system on tractographies from amyotrophic lateral sclerosis (ALS) patients and healthy subjects that we collected for a forthcoming study about the systematic differences between their corticospinal tracts.**

***Index Terms*—diffusion MRI ; tractography ; dissimilarity representation; clustering ; interactive segmentation**

I. Introduction

Diffusion magnetic resonance imaging (dMRI) [1] techniques provide non-invasive images of the brain white matter. dMRI quantifies locally, i.e. in each voxel, the diffusion process of the water molecules which are mechanically constrained in their motion by the axons of the neurons. Reconstruction and tracking algorithms [2] transform dMRI data into a set of streamlines, i.e. 3D polylines that approximate the neuronal pathways. The whole set of streamlines is called tractography and it represents the anatomical connectivity of the brain. See for example Figure 1.

This work focuses on computer-assisted tractography segmentation and describes the solution we developed to build a software system to support neuroanatomists and medical doctors in studying the white matter. Our work is motivated by a clinical research hypothesis about the characterisation of the amiotrophic lateral sclerosis (ALS) disease. We collected dMRI data from ALS patients and healthy controls with the aim of studying the effects of the ALS disease on the corticospinal tract (CST) (see Figure 1I), an anatomical structure that connects cortical motor areas to the spine and the body. The CST is known to be affected by ALS [3] and for this reason, our long term goal is to characterise these effects through tractography data. The first task in this endeavour is to segment the CST from the full brain tractography of each subject.

Tractography segmentation can be performed manually or automatically. Despite an increasing literature in automatic segmentation (see a brief review in [4]), the application in the clinical domain usually rely on manual segmentation. The manual segmentation process usually consists in selecting the subset of the streamlines connecting a few manually located regions of interest[1]. This task is a lengthy and complex one, for two reasons: first the tractography is a very large set of streamlines, in the order of 3×10^5, which makes it intrinsically difficult both to inspect and to unfold anatomical structures (see Figure 1A). Second, the reconstruction of the streamlines is frequently suboptimal due to the noise in the measurement process and to the limitations of reconstruction algorithms. For this reason, a single neuronal pathway may be just partially reconstructed, resulting in multiple disconnected polylines. These *broken* streamlines would be discarded by the manual segmentation procedure mentioned above, because not connecting the regions of interest defined by the expert.

Differently from the previous approaches of tractography segmentation, we conceived a novel computer-assisted interactive process based on clustering algorithms, which aims at greatly reducing the time required to manually segment a given anatomical white matter structure of interest. Our approach is based on a fast-clustering technique by means of which the expert is presented with a summary of the streamlines, i.e. the clusters represented by their medoids [2]. The expert manually selects the medoids/clusters of interest in order to remove most of the streamlines not related to the anatomical structure of interest. Interacting with the summary, instead of the actual streamlines, is much simpler for the user. In the example of Figure 1, the user selects 15000 streamlines (see 1C) of the 3×10^5 streamlines (see 1A) just by clicking on 20 of the 150 medoids (see 1B, the selected medoids are shown in white). The process of reclustering the selected streamlines and of manual selection by the expert is iterated until the expert is confident of having segmented the structure of interest (see Figure 1I).

In this work we describe the algorithmic solution we adopted in order to build the interactive tractography segmentation tool. The core of the problem is to cluster a large number of streamlines in no more than a few seconds, to allow a comfortable interactive user experience to the expert. The proposed solution combines two state-of-the-art elements: first a recently proposed Euclidean embedding algorithm for streamlines, i.e. the dissimilarity representation with the scalable *subset farthest first* (SFF) prototype selection policy [5]. This embedding provides fast and accurate vectorial

[1] See for example http://www.trackvis.org.

[2] A medoid is the element of a cluster closest to its centre.

representation of streamlines. Second, a recently proposed improvement of the k-means clustering algorithm called *mini-batch* k-means [6] (MBKM). This algorithm, which require the data to lie in a vector space, drastically reduces the convergence time to the actual clusters in case of large and very-large sets of objects. We claim that the dissimilarity embedding together with the MBKM algorithm provides a viable solution to problem of fast clustering of streamlines.

The paper is structured as follows. In Section II the algorithmic elements of the proposed method are formally described. In Section III we describe the segmentation process and report the details of the actual use of the proposed solution in the context of the CST segmentation. We quantitatively describe the segmentation process and provide timings to evaluate the viability of the proposed solution. In Section IV we discuss the results and we show that the proposed solution confirms our claims.

II. Methods

In the following we describe the elements that we evaluated in order to build the proposed method. After introducing the notation we formally describe the dissimilarity representation, i.e. a Euclidean embedding for streamlines, and then we present the mini-batch k-means algorithm.

A. Notation

Let $X \in \mathcal{X}$ be a streamline, i.e. a sequence of points in 3D space $X = ((\mathbf{x}_1), \ldots, (\mathbf{x}_n))$, $\mathbf{x} = (x, y, z) \in \mathbb{R}^3$. Notice that in general n is different from one streamline to another, which means that streamlines are heterogeneous objects that cannot be directly represented as vectors of the same vector space. Let $T = \{X_1, \ldots, X_M\}$ be a brain tractography, for which $M \approx 3 \times 10^5$ usually. Let $d : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}^+$ be a distance function between streamlines. A common distance between streamlines is the symmetric minimum average distance (see [7]) defined as

$$d(X_a, X_b) = \frac{1}{2}(\delta(X_a, X_b) + \delta(X_b, X_a)) \tag{1}$$

where $\delta(X_a, X_b) = \frac{1}{|X_a|} \sum_{\mathbf{x}_i \in X_a} \min_{\mathbf{y} \in X_b} ||\mathbf{x}_i - \mathbf{y}||_2$.

B. The Dissimilarity Representation

The *dissimilarity representation* [8] is a lossy Euclidean embedding algorithm that maps general objects into $\mathbb{R}^p$. In our case the objects are the streamlines, as it was previously proposed in [5]. The dissimilarity representation is a function $\phi^d_\Pi(X) : \mathcal{X} \mapsto \mathbb{R}^p$ s.t.

$$\phi^d_\Pi(X) = [d(X, \tilde{X}_1), \ldots, d(X, \tilde{X}_p)] \tag{2}$$

where d is a given distance function between streamlines, and $\Pi = \{\tilde{X}_1, \ldots, \tilde{X}_p\} \subset \mathcal{X}$ is a given set of p streamlines called *prototypes* or *landmarks*. The quality of the Euclidean embedding is strongly dependent on the choice of d and on the selection of the prototypes (see [5], [9]). In this work we adopt the distance of Equation 1, as suggested in [5].

An efficient procedure to select effective prototypes in the case of tractography data was presented in [5]: the *subset farthest first* (SFF) algorithm. This procedure is a scalable approximation of the well known farthest first traversal (FFT) algorithm. The FFT algorithm selects one streamline at random from the tractography as the first prototype $\tilde{X}_1$ and then iteratively adds a new prototype as the streamline maximising the distance to the already selected prototypes. The SFF algorithms is a stochastic scalable version of FFT, which subsamples $m = \lceil cp \log p \rceil$ streamlines from the whole tractography, and then applies FFT to the subsample. For the case of tractography data, when $c >= 3$ the SFF algorithm is comparable to the FFT algorithm with high probability, following the proof in [10] and the empirical results in [5].

C. Mini-Batch k-means

The k-means clustering problem is a cornerstone of the clustering literature. Given k, the number of clusters, the problem is to find k cluster centres $C = \{\mathbf{c}_1, \ldots, \mathbf{c}_k\}$, $\mathbf{c} \in \mathbb{R}^p$, and to assign each element of the vectorial dataset $\Phi(T) = \{\phi(X_1), \ldots, \phi(X_M)\} \subset \mathbb{R}^p$ to the closest cluster[3]. The k-means problem is then to compute centres C such as to minimise the loss function $f(C) = \sum_{\phi(X) \in \Phi(T)} D(\phi(X), C)^2$, where $D(\phi(X), C) = \min_{\mathbf{c} \in C} ||\phi(X) - \mathbf{c}||_2$ is the distance between $\phi(X)$ and the closest centre. The exact solution of the k-means problem is NP-hard and the computational complexity of the standard algorithm, the Lloyd's algorithm, has been proved to be $O(M^{34})$ in the general case [11], even though much less in practical applications. Nevertheless the standard algorithm is impractical when clustering tractography data in an interactive setting, as we show in Section III.

The *mini-batch k-means* (MBKM) algorithm [6] is a recently proposed modification of the standard algorithm that is able to reduce the computational costs by orders of magnitude. The intuitive idea is to use a stochastic gradient descent approach to find the centres C starting from a random initialisation. This idea was introduced in [12] where the points of the dataset were given one at a time in an online fashion.

Instead of updating the centers with one streamline at a time, the MBKM algorithm proposes to use multiple random subsets of the dataset, i.e. the *mini batches*, to update the cluster centres and to estimate the per-centre learning rates. As soon as the objective function $f(C)$ converges the process stops. The pseudocode algorithm of MBKM is shown in [6] and we do not report it here for lack of space.

The computational complexity of the MBKM algorithm is not known in the general case but empirical results in [6] show a reduction of two orders of magnitude in computation time with respect to the standard k-means. We present analogous results on tractography data in Section III.

III. Experiments

In the following we briefly describe the dataset we collected for the validation of the proposed approach. Then we describe the details of the interactive segmentation process on those data which is depicted in Figure1 and we conclude with the actual timings of the competing approaches (see Table I).

[3]From now on, we denote $\phi^d_\Pi(X)$ as $\phi(X)$ to simplify the notation without introducing ambiguity.

A. ALS Dataset

The data was recorded with a $3T$ scanner at *** ANONYMISED ***. It consisted of 12 ALS patients and 12 healthy controls (64 gradients; b-value$= 1000$.; anatomical scan ($1 \times 1 \times 1mm^3$)). We reconstructed the streamlines using EuDX, a deterministic tracking algorithm [13] from the DiPy library [4]. The tractography was then embedded in $\mathbb{R}^p$ using the dissimilarity representation presented in Section II-B with $p = 40$ and the SFF prototype selection procedure ($c = 3$) as suggested in [5]. The prototype selection and the actual embedding of $\approx 3 \times 10^5$ streamlines required ≈ 180s. The resulting matrix $\phi(T) \in \mathbb{R}^{300K \times 40}$ was computed once and stored, so that the time to compute the projection did not affect the interactive segmentation.

B. The Interactive Segmentation Process

We describe the segmentation process following the example of the CST segmentation of one subject illustrated in Figure 1. The full tractography (1A) of ≈ 250000 streamlines was initially clustered in $k = 150$ clusters and the medoids were presented to the user (1B). We observed that $k = 150$ was approximately the highest number of medoids the users could comfortably interact with in the 3D scene when the whole tractography was presented. Then user selected 20 clusters by clicking on the corresponding medoids (1B, in white). These clusters corresponded to a set of ≈ 15000 streamlines (1C). These streamlines were re-clustered into $k = 50$ clusters (1D) and the user selected 25 of them (1D, in white). We observed that 50 medoids are approximately the highest number a user can comfortably interact with after the initial selection from the full tractography. The 25 selected clusters corresponds to ≈ 3000 streamlines (1E) that are then re-clustered into $k = 50$ clusters (1F). In two further steps the user reduced the selected streamlines to ≈ 1500 (1G) and then to ≈ 500 (1I) thus reaching the desired segmentation of the CST. We observed that a trained neuroanatomist could segment the CST in approximately in 5 minutes.

The average timings of the clustering algorithms of Section II are reported in Table I. In the first column (size) are reported the size of the subset of streamlines that were clustered. The second column (k) reports the number of clusters, according to the notes expressed above. The third (k-means) and the fourth (MBKM) report the time for clustering[5]. The Fifth column reports the size (b) of the mini-batches for the MBKM, which was always 100 except for the full tractography for which we observed a significant gain in time when increasing it to 1000. The sixth column reports the time to compute the medoids from the centroids provided by k-means and MBKM. Each medoid was computed with simple exhaustive search within each cluster. The time to compute all medoids was always negligible with respect to the clustering time. All computations were performed on a standard desktop computer.

[4]http://www.dipy.org

[5]The clustering of the whole tractography can be computed once and stored, so its time does not affect the interactive use.

size	k	k-means	MBKM	b	medoids
500	50	$0.3s$	$\mathbf{0.2}s$	100	$0.003s$
1000	50	$0.6s$	$\mathbf{0.2}s$	100	$0.004s$
5000	50	$6.1s$	$\mathbf{0.4}s$	100	$0.009s$
10000	50	$14.4s$	$\mathbf{0.6}s$	100	$0.018s$
15000	50	$29.9s$	$\mathbf{0.7}s$	100	$0.026s$
250000	150	$> 1000s$	$\mathbf{13.3}s$	1000	$0.72s$

TABLE I

FOR A GIVEN NUMBER OF STREAMLINES (1ST COLUMN, SIZE) AND A GIVEN NUMBER OF CLUSTERS (2ND COLUMN, k) THE TIME TO COMPUTE THE CLUSTERING WITH k-MEANS AND MBKM IS REPORTED IN THE 3RD AND 4TH COLUMNS, RESPECTIVELY. THE SIZE (B) OF THE MINI-BATCHES FOR MBKM IS IN THE 5TH COLUMN. THE TIME TO COMPUTE THE MEDOIDS FROM THE CENTROIDS IS IN THE 6TH COLUMN.

The tractography segmentation tool was implemented in Python code on top of the DiPy, Fos and OpenGL [6]. The free software project of the segmentation tool, currently in alpha stage, is hosted *** ANONYMISED ***. The code of the dissimilarity representation is from [5] [7] and that of the k-means, the MBKM and the k-means++ is from scikit-learn [14] [8].

IV. CONCLUSION

We created a software tool to support the interactive segmentation of tractography data with pattern recognition algorithms. In order to handle the computational burden of clustering a large number of streamline under strong time constraints, we proposed a solution based on the dissimilarity representation and the MBKM algorithm. As shown in Table I (4th column) the time required to cluster the streamlines with the proposed solution was always the lowest and always < 1s during interactive use, thus meeting the requirements for a comfortable user experience. Conversely, the time required by the standard k-means algorithm was inadequate (see the 3rd column in Table I). As future work we plan to investigate further pattern recognition algorithms to better support the expert during tractography segmentation. To conclude we plan to improve the software segmentation tool in order to make it production stable in near future.

V. ACKNOWLEDGMENT

*** ANONYMISED ***

REFERENCES

[1] P. J. Basser, J. Mattiello, and D. LeBihan, "MR diffusion tensor spectroscopy and imaging." *Biophysical journal*, vol. 66, no. 1, pp. 259–267, Jan. 1994.

[2] S. Mori and P. C. M. van Zijl, "Fiber tracking: principles and strategies a technical review," *NMR Biomed.*, vol. 15, no. 7-8, pp. 468–480, 2002.

[3] M. Cosottini, M. Giannelli, F. Vannozzi, I. Pesaresi, S. Piazza, G. Belmonte, and G. Siciliano, "Evaluation of corticospinal tract impairment in the brain of patients with amyotrophic lateral sclerosis by using diffusion tensor imaging acquisition schemes with different numbers of diffusion-weighting directions." *Journal of computer assisted tomography*, vol. 34, no. 5, pp. 746–750, 2010.

[4] X. Wang, W. E. Grimson, and C.-F. F. Westin, "Tractography segmentation using a hierarchical Dirichlet processes mixture model." *NeuroImage*, vol. 54, no. 1, pp. 290–302, Jan. 2011.

[6]http://fos.me , http://opengl.org

[7]https://github.com/emanuele/prni2012_dissimilarity

[8]http://scikit-learn.org

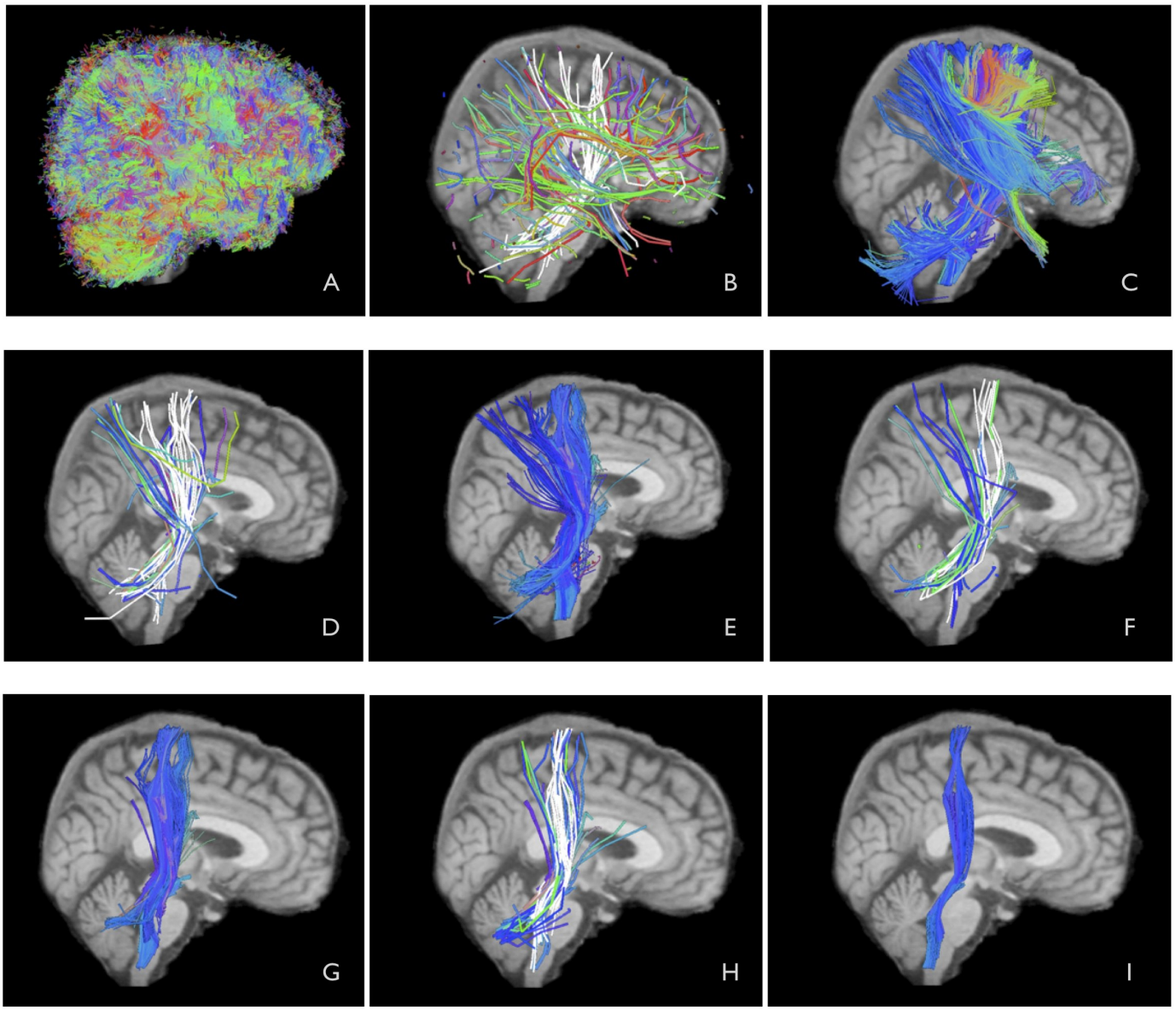

Fig. 1. The segmentation process. (A) Full tractography $\approx 3 \times 10^5$ streamlines; (B) Computation of 150 clusters and selection of 20 clusters (in white); (C) $\approx$ 15000 streamlines corresponding to previous selection; (D) Computation of 50 clusters and selection of 25 clusters; (E) $\approx$ 3000 streamlines corresponding to the previous selection; (F) Computation of 50 clusters and selection of 15 clusters; (G) $\approx$ 1500 streamlines corresponding to the previous selection; (H) Computation of 50 clusters and selection of 25 clusters; (I) $\approx$ 500 streamlines corresponding to previous selection and representing the segmented CST.

[5] E. Olivetti, T. B. Nguyen, and E. Garyfallidis, "The Approximation of the Dissimilarity Projection," in *IEEE International Workshop on Pattern Recognition in NeuroImaging*. IEEE, 2012.

[6] D. Sculley, "Web-scale k-means clustering," in *Proceedings of the 19th international conference on World wide web*, ser. WWW '10. New York, NY, USA: ACM, 2010, pp. 1177–1178.

[7] S. Zhang, S. Correia, and D. H. Laidlaw, "Identifying White-Matter Fiber Bundles in DTI Data Using an Automated Proximity-Based Fiber-Clustering Method," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 5, pp. 1044–1053, Sep. 2008.

[8] E. Pekalska, P. Paclik, and R. P. W. Duin, "A generalized kernel approach to dissimilarity-based classification," *J. Mach. Learn. Res.*, vol. 2, pp. 175–211, 2002.

[9] E. Pekalska, R. Duin, and P. Paclik, "Prototype selection for dissimilarity-based classifiers," *Pattern Recognition*, vol. 39, no. 2, pp. 189–208, Feb. 2006.

[10] D. Turnbull and C. Elkan, "Fast recognition of musical genres using RBF networks," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 17, no. 4, pp. 580–584, Apr. 2005.

[11] D. Arthur, B. Manthey, and H. Röglin, "k-Means Has Polynomial Smoothed Complexity," in *Proceedings of the 2009 50th Annual IEEE Symposium on Foundations of Computer Science*, ser. FOCS '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 405–414.

[12] L. A. Bottou and Y. Bengio, "Convergence Properties of the K-Means Algorithms," in *Advances in Neural Information Processing Systems 7*, 1995, pp. 585–592.

[13] E. Garyfallidis, "Towards an accurate brain tractography," Ph.D. dissertation, University of Cambridge, 2012.

[14] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, 2011.