Spatially Aware Patch-Based Segmentation (SAPS): An Alternative Patch-Based Segmentation Framework

Zehan Wang, Robin Wolz, Tong Tong, and Daniel Rueckert

Department of Computing, Imperial College London, UK
zehan.wang06@imperial.ac.uk

Abstract. Patch-based segmentation has been shown to be successful in a range of label propagation applications. Performing patch-based segmentation can be seen as a k-nearest neighbour problem as the labelling of each voxel is determined according to the distances to its most similar patches. However, the reliance on a good affine registration given the use of limited search windows is a potential weakness. This paper presents a novel alternative framework which combines the use of kNN search structures such as ball trees and a spatially weighted label fusion scheme to search patches in large regional areas to overcome the problem of limited search windows. Our proposed framework (SAPS) provides an improvement in the Dice metric of the results compared to that of existing patch-based segmentation frameworks.

Keywords: patch-based segmentation, label propagation, multi-atlas, nearest neighbour search, spatial.

1 Introduction

Accurate segmentations in medical imaging form a crucial role in many applications from patient diagnosis to clinical research. The amount of data generated from medical images can take a substantial amount of time for clinicians to manually segment, often becoming prohibitive as a regular task. Consequently, automatic methods for performing these tasks are becoming more important for image analysis. However, obtaining accurate results is highly important and still poses a challenge in many medical imaging applications.

Patch-based approaches for label propagation [1], [2] have been shown to be a robust and effective solution for applications in medical images. These methods label each voxel of a target image by comparing the image patch centred on the voxel with patches from an atlas library and choosing the most probable label according to the closest matches.

In this paper, we propose an alternative framework for patch-based segmentation which uses efficient k nearest neighbour structures, such as ball trees and a spatially weighted label fusion method which is loosely based on a non-local means approach [3] to allow segmentation of data with greater variability in alignment after affine registration. We validate this approach on 202 images from the ADNI database and compare the results with an existing method.

B.H. Menze et al. (Eds.): MCV 2012, LNCS 7766, pp. 93–103, 2013.

2 Methods

2.1 Pre-processing

Atlases are all registered to a common template space using affine registration and intensities are normalised using the method proposed by Nyúl and Udupa [4]. A general mask is then created for each label of interest in the atlas by taking the union of the labels from all the training data and dilating the result. This mask is used to narrow the search space and restrict search to valid areas where a label might appear. The mask needs to be large enough to allow for possible variations in anatomical variability, but not too large as this would make the search process less efficient.

The training data is also denoised to improve robustness. We used Total Variation denoising as a quick and easy to apply method which is effective in regularizing images without smoothing boundaries and edges [5].

2.2 *k*NN Data Structure Construction

Performing patch-based segmentation can be seen as a k-nearest neighbour problem as the labelling of each voxel is determined according to the distances to its most similar patches. An exhaustive search would have a computational complexity that is linearly proportional to the size of the dataset and can be quite prohibitive in large datasets, especially given the number of voxels that require this process in an image. This is one reason why existing methods use a small search volume size, such as in the region of $11 \times 11 \times 11 = 1331$ voxels, and why a good alignment of images is required.

To increase the search volume size without a detrimental impact to the search speed, an efficient kNN search data structure is required. Any exact kNN data structure could be used in this framework, but in our implementation, a ball tree [6] was used. Ball trees provide much better search performance than kd trees or brute force searches for high dimensional data [7]. Ball trees are metric trees which use a given distance metric to partition the data so that only a small part of the data need to be queried. The distance metric used must obey the triangle inequality for metric trees to work correctly. Since Euclidean distances are used in both patch based comparisons and atlas selection, and this obeys the triangle inequality, ball trees can then be used to provide the results to kNN queries.

In principle all patches could be stored in a single tree, however, the memory requirements would grow prohibitively large as the number of atlases increases as well as giving decremental search performances. So instead, a ball tree is constructed offline for each label in each atlas region of every atlas in the training set. Each patch stored in the ball tree also has its spatial coordinates within the template space stored with it. This information is used in a soft-weighting scheme when performing patch selection as spatial correspondence can help distinguish between patches with homogeneous intensities which provide very little structural information. This is particularly the case in brain images where patches from different structures of the brain can be very similar when only voxel intensities are compared.

Fig. 1. Example: ball tree construction from patches. Split the brain into 2 regions centred around the left and right hippocampus and create a tree in each region for each label, including the background label.

2.3 Search Strategy

Target images undergo the same pre-processing steps as the training images prior to segmentation as some degree of spatial correspondence is required for an effective segmentation.

Atlas Selection by Region. For each of the regions of interest that requires labelling, the nearest N atlases are found for each region by comparing their Euclidean distance. Using a limited selection of the best subjects from the atlas library has been shown to provide more effective segmentation results [8]. Another kNN data structure such as the ball tree can be built offline to allow fast atlas selection in the case of a large atlas library. The corresponding kNN data structure for those atlas regions are then used for segmentation. By performing atlas selection on the regional level, more appropriate atlases can be chosen for each region rather than selecting a single set of atlases to use for the whole image. This can be improve the accuracy of segmentations in cases where images differ in their similarity from region to region. For example, for performing a hippocampus segmentations, the set of atlases selected for the left hippocampus can differ to those selected for the right hippocampus.

Patch Search and Label Fusion. The corresponding kNN data structures for the nearest N regions are then used for finding the nearest k patches for each voxel location i in target image x. The Euclidean distance between the patch, $P(x_i)$, in the target image and the nearest k patches, $\{P(y_j)\}$, from the atlas library are weighted with the Euclidean distance on their spatial location to provide an overall weighting for each label. An additional weighting, α, can be

applied to control the influence of spatial correspondence. The resulting weighting for label l at voxel i is then determined by the sum of weights between patch $P(x_i)$ and the k nearest patches, $\{P(y_j)\}$ as follows:

$$w_{l_i}(x) = \sum_{j=1}^{k} w_l(x_i, y_j) \tag{1}$$

coordinate where

$$w_l(x_i, y_j) = e\frac{-\{||P(x_i) - P(y_j)||_2^2 + \alpha||x_i - y_j||_2^2\}}{h^2} \tag{2}$$

h is a decay parameter which controls the level of influence of patches as the distance increases, an automatic estimation of this parameter is used for each voxel based on the minimum distance between patch $P(x_i)$ and the nearest k patches, weighted by their spatial coordinates:

$$h^2(x_i) = \min\{||P(x_i) - P(y_j)||_2^2 + \alpha||x_i - y_j||_2^2\} \tag{3}$$

An overall weight for each label at each voxel i is then calculated from the sum of the distances of these patches and the resulting label is decided based on majority voting of the labels according to these weights:

$$L(x_i) = \arg\max_l w_{l_i}(x) \tag{4}$$

3 Experiments and Results

3.1 Dataset

Images from the Alzheimers Disease Neuroimaging Initiative (ADNI) database (www.loni.ucla.edu/ADNI) were used for validation. These images consists of 202 subjects (68 normal, 93 with mild cognitive impairment, 41 with Alzheimer's disease) imaged using different scanners. Reference segmentations were obtained semi-automatically using a commercially available high dimensional brain mapping tool (Medtronic Surgical Navitgation Technologies, Louisville, CO) by propagating 60 manually labelled images. Images were pre-processed by the ADNI pipeline [9] and were linearly registered to the MNI152 template space using affine registration.

To test the proposed framework, a leave-one-out validation strategy was applied where each image was segmented in turn, using the remaining dataset as the atlas database. A patch size of $7 \times 7 \times 7$ was used whilst we experimented with the number of atlases used, N, the spatial weights, α, and the number of nearest neighbours for each patch, k. All image intensities were normalised and scaled to the same range and TV denoising [5] was applied to the training data.

3.2 Implementation

The main framework was implemented in Python using open source modules such as NumPy, SciPy and SciKit-learn. The computation time is around 10 minutes for each image using 8 cores clocked at 2.67GHz each when using 20 atlases and using the 100 nearst neighbours. Given that Python is an interpreted language, further speed ups can be achieved if the framework was implemented in C/C++.

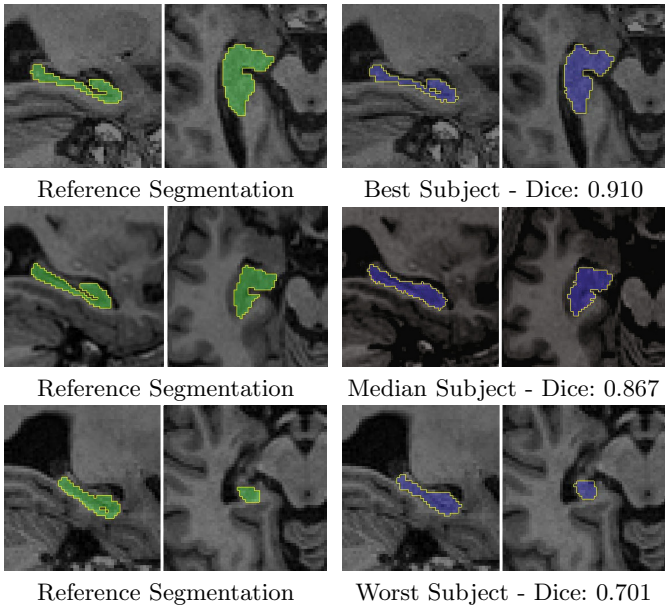

Fig. 2. Segmentations of the right hippocampus with parameters $N = 40$, $k = 79$, $\alpha = 13$

3.3 Effect of the Number of Nearest Patches and Atlases Used

With the spatial weight fixed at $\alpha = 13$, we experimented using a range of values for the number of patches, k, as well as the number of atlases, N. k is dependent on N as using more atlases would present a bigger selection of patches to choose from and we see in figure 3 that the optimal k value differs for the different N values.

Generally, we find that accuracy increases as k increases, but reaches a limit after $k > 60$. There is an increase in computational cost as k increases as more comparisons must be made in the kNN data structures, so it is most computationally optimal to select the lowest k value that provides the desired segmentation accuracy.

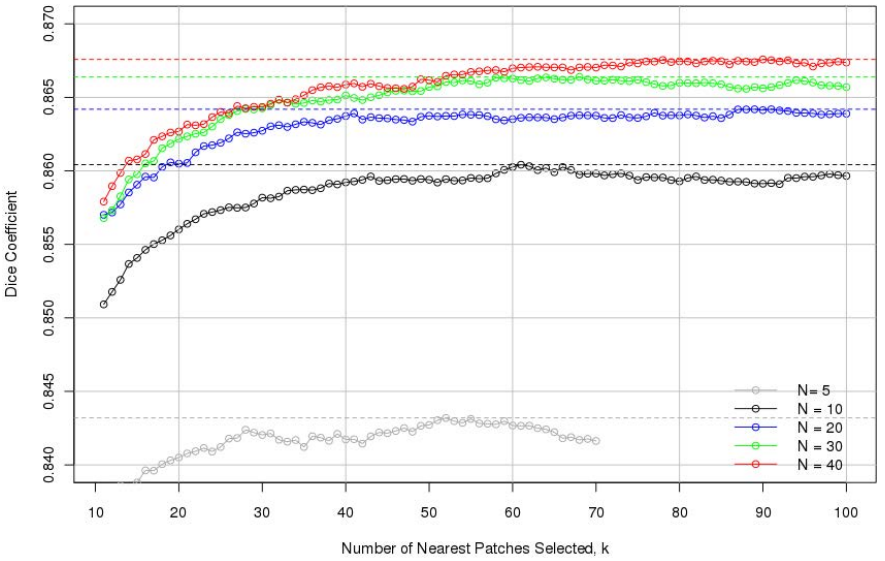

Fig. 3. Median Dice coefficients for the whole hippocampus whilst using a range of k values with different N values

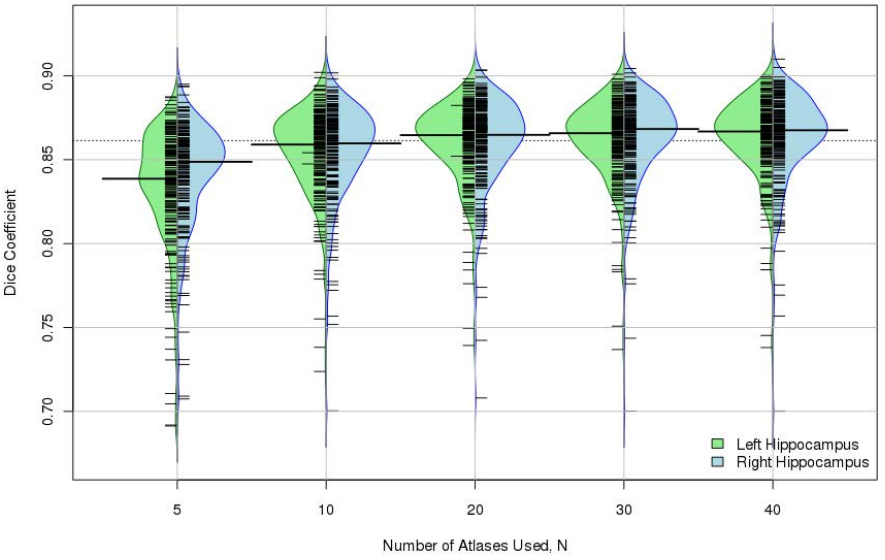

Fig. 4. Beanplot showing overall Dice coefficients distributions for a range of N values with $k = 64$. Large thick lines indicate medians, dotted line indicates median across all k values. The shape of the "bean" shows the distribution of the results and individual data points are shown as small lines on the bean.

Table 1. Dice Coefficients for the hippocampus (HC) when using different number of atlases, N, with $k = 64$. Highest values are show in bold.

N	Left HC			Right HC			Overall		
	Best	Worst	Median	Best	Worst	Median	Best	Worst	Median
5	0.887	0.691	0.839	0.895	0.707	0.849	0.886	0.719	0.842
10	**0.902**	0.724	0.860	0.902	0.700	0.859	0.898	0.719	0.860
20	0.898	**0.740**	0.864	0.904	**0.708**	0.865	0.899	**0.724**	0.864
30	0.901	0.737	0.866	0.904	0.700	**0.868**	0.899	0.719	0.866
40	0.900	0.738	**0.867**	**0.910**	0.700	0.868	**0.902**	0.719	**0.867**

An increase in the number of atlases used generally increases segmentation accuracy, but the gain accuracy after $N > 20$ is marginal. Given that the computational cost increases linearly with the number atlases used, this suggests that using more than 30 atlases would not provide a sufficient trade-off between the extra time spent and the accuracy gained. Our findings on here agree with those presented in [1] on the number of training subjects used, with proportional gains in accuracy as N increases.

3.4 Effect of the Spatial Weight, α

Experiments using several values for spatial weights, α, showed that using spatial information to provide a soft-weighting has significant impact on the

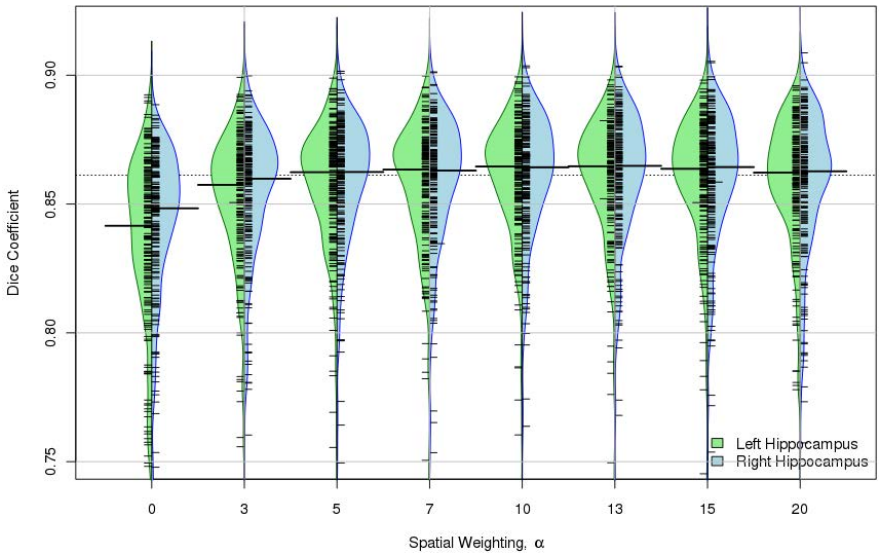

Fig. 5. Beanplot showing Dice coefficients distributions for a range of spatial weighting values, α with $N = 20$, $k = 64$. Large thick lines indicate medians, dotted line indicates median across all α values. The shape of the "bean" shows the distribution of the results and individual data points are shown as small lines on the bean.

Table 2. Dice Coefficients for the hippocampus (HC) when using different spatial weights, α, with $k = 64$ and $N = 20$. Highest values are show in bold.

α	Left HC			Right HC			Overall		
	Best	Worst	Median	Best	Worst	Median	Best	Worst	Median
0	0.892	0.669	0.842	0.889	0.674	0.848	0.884	0.702	0.844
5	0.899	0.736	0.862	0.902	0.700	0.862	0.897	0.718	0.862
10	**0.900**	**0.744**	**0.865**	0.906	0.692	0.864	0.899	0.723	0.863
13	0.898	0.740	0.864	0.904	0.708	**0.865**	0.899	**0.724**	**0.864**
15	0.898	0.736	0.864	0.905	0.704	0.864	**0.900**	0.720	0.863
20	0.860	0.729	0.862	**0.909**	**0.710**	0.863	**0.900**	**0.724**	0.862

segmentation accuracy (see figure 5 and table 2). The distribution of the results as seen in the beanplots shows that the consistency of the results increases significantly when we use spatial information. The values attempted suggests that segmentation accuracy peaks between $\alpha = 12$ and $\alpha = 13$. If the spatial weighting is too high, there is a detrimental effect on the segmentation accuracy as this soft-weighting becomes too restrictive when comparing patch intensities.

3.5 Effect of Denoising

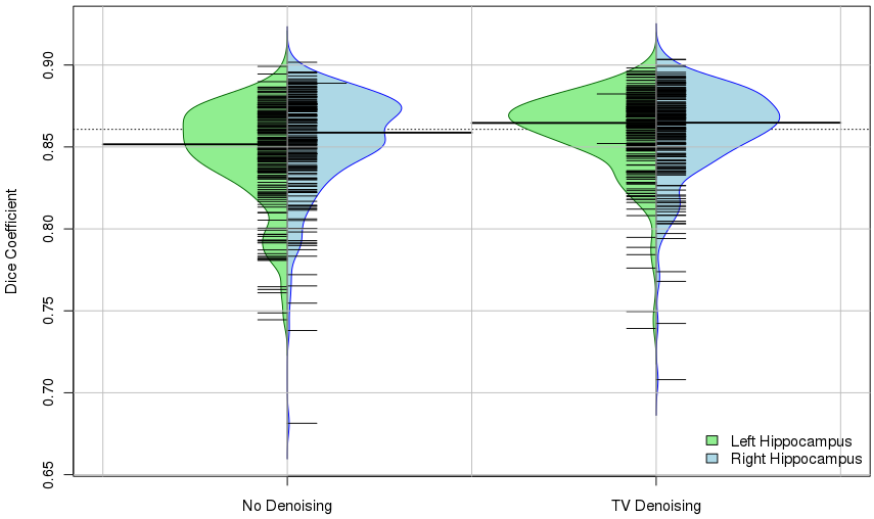

Fig. 6. Dice coefficients distributions for results using denoised and non-denoised training data with $N = 20$, $k = 64$, $\alpha = 13$. Large thick lines indicate medians, dotted line indicates median across both datasets. The shape of the "bean" shows the distribution of the results and individual data points are shown as small lines on the bean.

Comparing results from using non-denoised training data to those from using denoised training data, it can be seen that using denoised training data provides an improvement to the median segmentation accuracy (see figure 6). Further to this, the range of the results is significantly smaller with a more favourable distribution when using denoised training data, suggesting that this does indeed improve the generality and robustness of the framework.

3.6 Comparison of Results to an Existing Method

Finally, with the same dataset of ADNI images, we compared the results obtained by our proposed method to that using the method described in [1] (see figure 7 and table 3), with 10 training atlases in both cases. It can be seen that our

Table 3. Median Dice coefficients for the hippocampus (HC) comparing with the existing method in [1] with the number of atlases, $N = 10$ (*and* $N = 40$ *for reference*). Proposed method uses $k = 64$, $\alpha = 13$ as its other parameters.

Method	Left HC			Right HC			Overall		
	Best	Worst	Median	Best	Worst	Median	Best	Worst	Median
Existing[1]	0.894	0.696	0.842	**0.910**	0.644	0.848	**0.901**	0.709	0.844
Proposed, $N = 10$	**0.902**	**0.724**	**0.860**	0.902	**0.700**	**0.859**	0.898	**0.719**	**0.860**
Proposed, $N = 40$	*0.900*	*0.738*	*0.867*	*0.910*	*0.700*	*0.868*	*0.902*	*0.719*	*0.867*

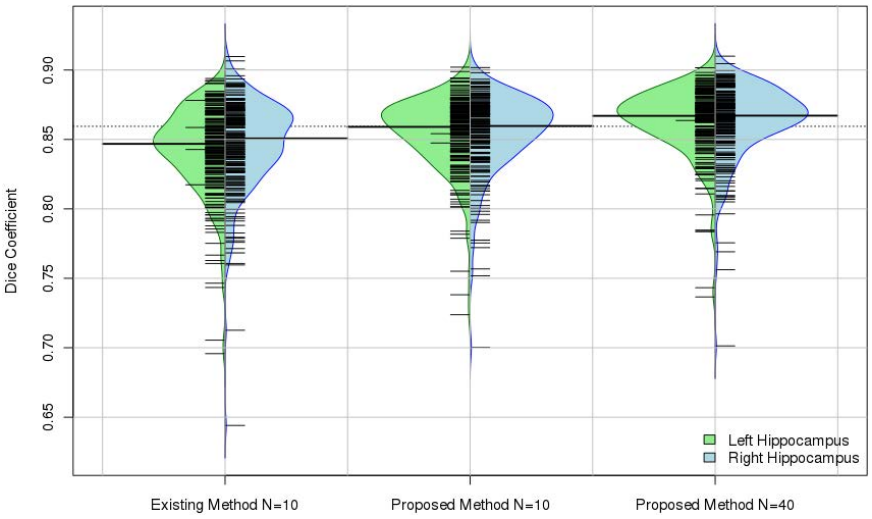

Fig. 7. Dice coefficients distributions for results comparing SAPS with an existing method [1]. Other parameters for SAPS are $k = 64$, $\alpha = 13$. Large thick lines indicate medians, dotted line indicates median across both datasets. The shape of the "bean" shows the distribution of the results and individual data points are shown as small lines on the bean.

method generally outperforms the existing method and is more robust. The two methods performs quite similarly when no spatial information is used (see table 2). This is because the label fusion would be equivalent to the non-local means method if the spatial weight, α, is 0.

Employing Welch's two sample t-test on these results gave p-values of 0.00003, 0.007 and 0.004 for the left, right and overall hippocampus respectively. Additionally, our proposed method has a 0.05 decrease in the standard deviation of the results.

4 Conclusion

We have presented a new generalized framework for applying patch based segmentation which is able to robustly segment data in conditions where images can have large variations in alignment by looking at a much larger search windows in addition to applying a spatial location weighting to each patch. We validated the proposed framework against 202 ADNI images of patients at various stages of Alzheimer's disease and achieved an overall median dice coefficient of 0.867 using patches from the 40 most similar atlases. The framework allows a trade-off between segmentation accuracy and speed. If we use patches from half as many atlases, we can complete the segmentation in half as much time and are still able to attain a median dice coefficient of 0.864. At the lowest limit tested, using 5 atlases is still able to yield a median Dice coefficient of 0.842 for the whole hippocampus whilst taking around 2-3 minutes on a machine with 8 cores.

In future work, we plan on further validating our proposed framework using a multi-scale extension against different anatomical structures and image types. We are currently working on a multi-scale extension to speed up segmentation of large structures such as bones in knee images or when performing brain extraction.

References

1. Coupé, P., Manjón, J.V., Fonov, V., Pruessner, J., Robles, M., Collins, D.L.: Patch-based segmentation using expert priors: application to hippocampus and ventricle segmentation. NeuroImage 54(2), 940–954 (2011)
2. Rousseau, F., Habas, P., Studholme, C.: A supervised patch-based approach for human brain labeling. IEEE Transactions on Medical Imaging 30(10), 1852–1862 (2011)
3. Coupe, P., Yger, P., Prima, S., Hellier, P., Kervrann, C., Barillot, C.: An optimized blockwise nonlocal means denoising filter for 3-D magnetic resonance images. IEEE Transactions on Medical Imaging 27(4), 425–41 (2008)
4. Nyúl, L.G., Udupa, J.K.: On standardizing the MR image intensity scale. Magnetic Resonance in Medicine 42(6), 1072–1081 (1999)
5. Chambolle, A.: An Algorithm for Total Variation Minimization and Applications. Journal of Mathematical Imaging and Vision 20(1), 89–97 (2004)
6. Omohundro, S.M.: Five Balltree Construction Algorithms. Technical Report 1, International Computer Science Institute (1989)

7. Kumar, N., Zhang, L., Nayar, S.K.: What Is a Good Nearest Neighbors Algorithm for Finding Similar Patches in Images? In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part II. LNCS, vol. 5303, pp. 364–378. Springer, Heidelberg (2008)
8. Aljabar, P., Heckemann, R.A., Hammers, A., Hajnal, J.V., Rueckert, D.: Multi-atlas based segmentation of brain images: atlas selection and its effect on accuracy. NeuroImage 46(3), 726–738 (2009)
9. Jack, C.R., Bernstein, M.A., Fox, N.C., Thompson, P., Alexander, G., Harvey, D., Borowski, B., Britson, P.J., Whitwell, J., Ward, C., Dale, A.M., Felmlee, J.P., Gunter, J.L., Hill, D.L.G., Killiany, R., Schuff, N., Fox-Bosetti, S., Lin, C., Studholme, C., DeCarli, C.S., Krueger, G., Ward, H.A., Metzger, G.J., Scott, K.T., Mallozzi, R., Blezek, D., Levy, J., Debbins, J.P., Fleisher, A.S., Albert, M., Green, R., Bartzokis, G., Glover, G., Mugler, J., Weiner, M.W.: The Alzheimer's disease neuroimaging initiative (ADNI): MRI methods. Journal of Magnetic Resonance Imaging 27(4), 685–691 (2008)