Intensity-based hierarchical clustering in CT-scans : application to interactive segmentation in cardiology

Jonathan Hadida, Christian Desrosiers and Luc Duong

Software and IT department at École de Technologie Supérieure, Montreal Canada

ABSTRACT

The segmentation of anatomical structures in Computed Tomography Angiography (CTA) is a pre-operative task useful in image guided surgery. Even though very robust and precise methods have been developed to help achieving a reliable segmentation (level sets, active contours, etc), it remains very time consuming both in terms of manual interactions and in terms of computation time. The goal of this study is to present a fast method to find coarse anatomical structures in CTA with few parameters, based on hierarchical clustering. The algorithm is organized as follows: first, a fast non-parametric histogram clustering method is proposed to compute a piecewise constant mask. A second step then indexes all the space-connected regions in the piecewise constant mask. Finally, a hierarchical clustering is achieved to build a graph representing the connections between the various regions in the piecewise constant mask. This step builds up a structural knowledge about the image. Several interactive features for segmentation are presented, for instance association or disassociation of anatomical structures. A comparison with the Mean-Shift algorithm is presented.

Keywords: CTA, segmentation, hierarchical clustering

INTRODUCTION

Computed tomography angiography (CTA) is routinely used for high resolution visualization of cardiovascular diseases, such as coronary artery disease and valvular heart disease. Recent advances in CTA offer a resolution sufficient for the visualization of ventricules, atria, pericardium, aorta and coronary arteries. However, these anatomical structures are combined with other organs and are difficult to isolate based solely on the image intensity due to radiographic noise and CT reconstruction artifacts. The main purpose of this study is to find an appropriate data representation that allows selecting quickly and interactively anatomical structures in CT-scans. Specifically, we aimed at obtaining a coarse segmentation of the main anatomical structures in the heart with *as less parameters as possible* to allow a better interactivity with the clinician. The coarse segmentation by intensity is consistent with the facts that in the one hand, enhancing contrast agents are often used to highlight specific structures (like coronary arteries for instance), and in the other hand, each structure is characterized by a specific attenuation coefficient of X-rays (related to its density and thickness). As a result, each structure is represented by a specific range of gray levels in the histogram.

1. LITTERATURE REVIEW

The main purpose of segmentation is to achieve region clustering in a given input dataset. Since the intensity in CT acquisition has a biophysical meaning related to the density and thickness of the imaged tissues, it is pertinent to consider the intensity as the main criterion for anatomical clustering in CT-scans.

In the litterature, one of the early attempts similar to our approach to achieve clustering in the histogram for image processing purpose is stated in 1979 by Otsu[1] . In this paper, he aims at separating a given number of classes in the histogram by computing the best possible thresholds between classes. Later on, as the research in pattern classification evolved, the problem of clustering for segmentation in image processing received a remarkable attention, and a myriad of solutions were proposed to solve it.

L. Duong, C. Desrosiers and J. Hadida are with the Software and IT department at ETS, 1100 Notre-Dame Ouest, Montreal QC, H3C 1K3, Canada. E-mails: jonathan.hadida.1@ens.etsmtl.ca, luc.duong@etsmtl.ca, christian.desrosiers@etsmtl.ca

Medical Imaging 2011: Visualization, Image-Guided Procedures, and Modeling,
edited by Kenneth H. Wong, David R. Holmes III, Proc. of SPIE Vol. 7964, 79641R
© 2011 SPIE · CCC code: 1605-7422/11/$18 · doi: 10.1117/12.878285

Downloaded From: http://spiedigitallibrary.org/ on 09/12/2012 Terms of Use: http://spiedl.org/terms

What these solutions generally have in common is that they consider the intensity as a *feature* for each pixel, and their goal is then to suggest a *generative model* to group the pixels in the resulting *feature space*. One of the most robust and efficient method following this approach is the Mean-Shift algorithm presented by Comaniciu[2] . This method exploits the mean-shift clustering algorithm to achieve image processing tasks such as filtering or segmentation, by locating the modes of a density function estimated from the input data in the $L^*u^*v^*$ feature-space, and assigning a label to each pixel corresponding to one of these modes.

Other clustering methods were used for segmentation purposes, such as the Fuzzy C-means method[3] for example. Advances in machine learning also allowed to automate algorithms by "teaching" them how to proceed on precomputed learning databases. The work of Zheng[4] is a recent example of the possibilities offered by machine learning. Energetical approaches with the use of snakes led to the creation of *active contours* and their use in medical image segmentation. A good recent example is the work of Rivest-Henault[5] for the segmentation of major aorto-pulmonary collateral arteries (MAPCAs). However, there are inherent drawbacks to use these methods for segmentation : methods using machine learning require a learning database, generally involving a significant number of manually segmented CT-scans. The methods involving active contours require an intervention by the clinician to specify seed points inside the structures of interest, and the clustering methods usually require the number of clusters to find as a prior.

2. METHOD

Our study does not extend these latest methods, but rather, we looked for a fast and efficient method to obtain quickly, and with as less parameters as possible, a coarse segmentation of anatomical structures in CT-scans. The resulting segmentation can then be refined with the use of more efficient methods if needed. For example with active contours, this can be considered as an automatic intialization, which dismisses the intervention of the clinician in the segmentation process. We also provide an efficient and practical representation of morphological knowledge, in the form of a hierarchical graph, that may serve either of these purposes :

- Allow to select automatically computed Regions Of Interest (ROIs) to apply a local treatment without affecting the rest of the image,
- Consider the use of the structural knowledge extracted from the input data to achieve affine-invariant recognition or automatic labeling,
- Provide the clinician with a simple, efficient and powerful interactive tool for segmentation.

To achieve this, we needed a valid model to describe the imaging modality we are studying. The fact that intensities in CT-scans represent normalized Hounsfield Units[6] , and that these precisely represent the densities of the scanned tissues, implies that the ideal CT-slice should look like a piecewise constant image, where each biological structure (muscle, vessel, atheroma, etc.) would be represented by a mean gray level, with small variations due to thickness variations and texture irregularities.

However, the amount of radiation that would be needed to tend to this ideal CT would put at risk the health of the patient. In fact, the amount of radiation is rather usually set to reach the minimum definition needed for the diagnosis. The consequence is that the observed acquisitions are subject to a *noise*, which is usually modeled as an additive Gaussian noise with a deviation coefficient varying for each gray level[6] . If we denote u the gray level, and U the grayscale domain, the noise is therefore stated by the following transformation :

$$\forall u \in U, \quad \mathcal{F} : u \mapsto u + \Sigma(u)\eta \tag{1}$$

where $\Sigma(u)$ is a static deviation that varies with the gray level u, and η is a centered random variable of time.

According to this model, the histogram of the CT-scan should be composed with a sum of Gaussians, each of which should be centered on the mean gray level corresponding to a visible structure in the scan, but which deviation parameter is unknown.

Downloaded From: http://spiedigitallibrary.org/ on 09/12/2012 Terms of Use: http://spiedl.org/terms

The method we present in this paper splits into four steps :

1. A non-parametric kernel-based clustering method to find the set of Gaussians that compose the histogram. Once this set is computed, a first Piecewise Constant Mask (PCM-1) is created.
2. A classes association method that aims at reducing the number of subdivisions of the histogram found at the previous step. The resulting subdivision leads to the computation of a second PCM (PCM-2).
3. A connected-space clustering algorithm that leads to the indexation of each cluster in the considered PCM (1 or 2). The result is called the Indexed Clusters Mask (ICM).
4. Finally, a hierarchical graph is computed to represent the relationships between the different clusters. These relationships define the anatomical topology of the image.

2.1 Non-parametric kernel-based clustering

The goal here is to define a non-parametric algorithm to find a sum of Gaussians that compose the histogram. In the following, $\mathcal{H}$ will be the histogram defined continuously on the grayscale domain U, typically equal to $[0, 255]$. We also define a general Gaussian kernel $\mathcal{N}_{\mathbf{p}}(u) = \mathcal{H}(\mu) exp\left(-\frac{(u-\mu)^2}{\sigma^2}\right)$ with parameters $\mathbf{p} = (\mu, \sigma)$.

Because its values represent a number of pixels, the meaningful values of the histogram are the values greater than 1. This means that the sum of Gaussians we are trying to find only needs to approximate the histogram with a unitary precision. Our target is thus to find a number n of Gaussians such that :

$$\int_U \left| \mathcal{H}(u) - \sum_{i=1}^{n} \mathcal{N}_{\mathbf{p_i}}(u) \right| du \leq 1 \tag{2}$$

The same statement is also true for the kernels we manipulate, which means that we have to restrict their definition on a domain on which their value is greater than 1. That is $\mathcal{D}(\mu, \sigma) = \left[\mu \mp \sigma \sqrt{\ln\left(\mathcal{H}(\mu)\right)}\right]$ if $\mathcal{H}(\mu) > 1$.

We now have all the tools we need to achieve the clustering. What remains to be defined is how we are going to find the number n of Gaussians, and what their parameters $\{\mathbf{p_i}\}_{i=1}^{n}$ might be. This is actually no different than a classical optimization problem where we try to reach the target-inequality (2). The question "*How*" is equivalent to the definition of an objective function to maximize or to minimize, which we will call the *clustering criterion*. For example, we can say that we want to minimize the number of clusters n to find, and this would mean that we want to find a small number of anatomical structures in the CT-scan.

If globally solving this optimization problem is actually possible in a nonparametric way using Dirichlet Processes[7] , it is theoretically quite complex and in first approximation, we can rather consider that solving this problem locally gives a satisfactory result for an application in coarse segmentation.

Solving the optimization locally means that instead of trying to minimize directly the number n of Gaussians, we will recursively select the fitting Gaussian that maximizes a given *clustering criterion*, substract it to the histogram, and stop the recursion if the target-inequation (2) is satisfied.

To clearly summarize the clustering process :

- The goal is to find a sum of Gaussians that approximate the histogram with a unitary precision,
- For that purpose, we recursively find the "*best*" possible kernel (which implies the computation of its parameters μ and σ) and substract it to the histogram until the target-inequation (2) is satisfied,
- Defining what is the "*best*" kernel at each step is equivalent to define a *clustering criterion* ; the "*best*" kernel is then simply the kernel which parameters maximize this criterion.

In the following, we tried two different clustering criteria : maximizing the *height* of the kernel at each step, and maximizing its *area*. A comparison of the results is shown on Fig. 1.

Downloaded From: http://spiedigitallibrary.org/ on 09/12/2012 Terms of Use: http://spiedl.org/terms

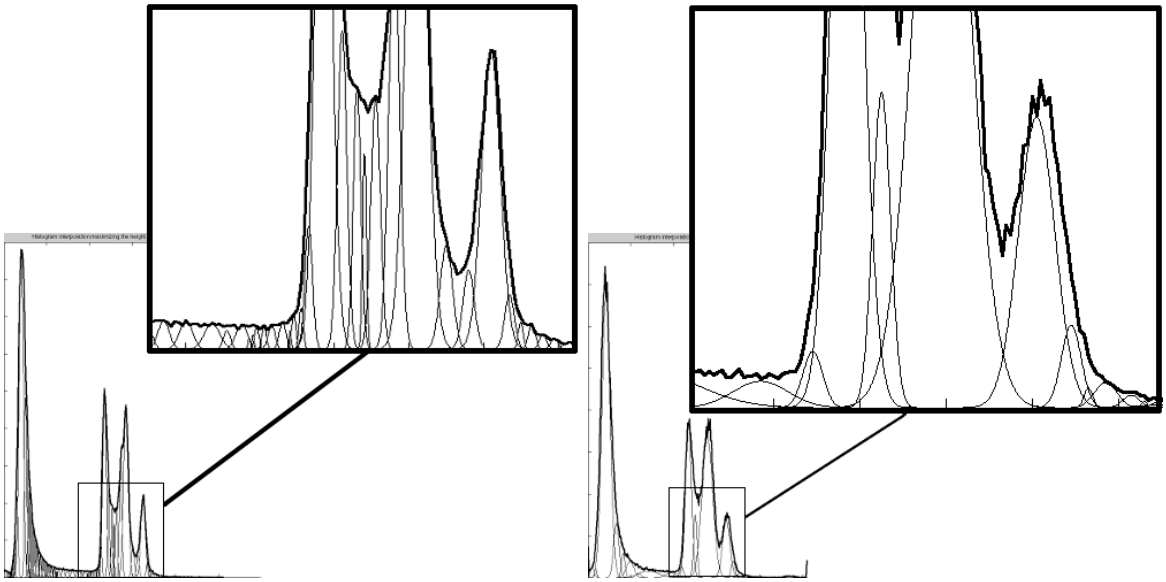

Figure 1. Comparison of the clustering results for the two presented clustering criteria. The clustering on the left was achieved by maximizing the *height* of the kernel at each step, whereas the clustering on the right maximized its *area*.

2.1.1 Clustering by maximizing the height

The height of the kernel $\mathcal{N}_{\mathbf{p}}$ is defined as $\mathcal{H}(\mu)$. Maximizing the height at each step therefore simply consists in finding the global peak of the histogram at each step, and more especially the corresponding μ. The following computation of the parameter σ is described in Appendix A.

2.1.2 Clustering by maximizing the area

The area of $\mathcal{N}_{\mathbf{p}}$ on its restricted definition domain $\mathcal{D}(\mu, \sigma)$ is:

$$\mathcal{A}(\mu, \sigma) = \mathcal{H}(\mu) \int_{\mathcal{D}(\mu,\sigma)} exp\left(-\frac{(u-\mu)^2}{\sigma^2}\right) du = \sigma\sqrt{\pi}\mathcal{H}(\mu)\mathrm{Erf}\Big(\sqrt{\ln\left(\mathcal{H}(\mu)\right)}\Big)$$

where $\mathcal{H}(\mu) > 1$ and $\mathrm{Erf}(\cdot)$ is the Gauss error function.

This formula raises a difficulty, because the computation of the kernel's parameter μ therefore requires the parameter σ, but the computation of the parameter σ also requires the parameter μ. To escape this dead end, we suggest to compute at each step the parameter σ for all the gray levels u for which $\mathcal{I}(u) \neq \varnothing$ (see Appendix A), and the corresponding area. What remains to be done in order to maximize the area of the kernel after this computation is simply to select the gray level u_0 for which the area is maximal, and set $\mu = u_0$.

Additionally, there is an exception to handle for the computation of the area at the boundaries of the grayscale domain U, because the interval $\mathcal{D}(\mu, \sigma)$ might actually be truncated there. The general formula to compute the area of Gaussian kernels is the following :

$$\forall x \in \mathbb{R}, \quad \int_{\mu}^{\mu+x} \mathcal{N}_{(\mu,\sigma)}(u)du = \frac{\sigma\mathcal{H}(\mu)\sqrt{\pi}}{2}\mathrm{Erf}(\tfrac{x}{\sigma})$$

This formula allows to compute precisely the area of any portion of kernel at the boundary of the grayscale domain U, which further enables to select the actual kernel of greatest area.

Downloaded From: http://spiedigitallibrary.org/ on 09/12/2012 Terms of Use: http://spiedl.org/terms

2.1.3 Building the first piecewise constant mask

To go back to our application, this clustering method enabled us to assess a certain number of classes in the histogram, and therefore a certain subdivision of the grayscale domain U. Indeed, to compute the first Piecewise Constant Mask (PCM-1), we just need to apply this subdivision in each CT-slice.

We achieve this by simply assigning to each gray level u, the mean value of the class that has the greatest value at this point. That is :

$$u \mapsto \mu_j \quad \text{with} \quad j = arg \max_{i=1..n} \mathcal{N}_{\mathbf{p}_i}(u)$$

An example result can be seen on Fig. 6.

It should be noted that the clustering process presented above does not depend on the dimensionality of the processed data (only on the size of the grayscale domain U), which means that the computation time is identical for 2D or 3D data, once the histogram is computed.

Nevertheless, this algorithm typically ends up with an important number of classes in the histogram, which means that too much details are captured. The next section aims at reducing this number if needed by associating classes between them.

2.2 Classes association

The number of classes found at the previous step was typically too important and as a result, the first PCM was too detailed ; the classes did not represent a sufficiently broad range of gray levels. This step aims at reducing the number of classes by introducing what we call a *ratio of integrity*.

2.2.1 Ratio of intergrity

Let us consider two overlapping direct neighbour classes A and B, and note $\mathcal{A}(X)$ the area of a class X. We define for each couple of class the following *ratios of integrity*, which lie between 0 and 1 :

$$\rho(A/B) = \frac{\mathcal{A}(A \backslash B)}{\mathcal{A}(A)} \qquad \text{and} \qquad \rho(B/A) = \frac{\mathcal{A}(B \backslash A)}{\mathcal{A}(B)}$$

To decide wether or not A and B should be merged, we introduce the first threshold parameter of our method ρ_{thresh}. We then take the following decision :

- If $\rho(A/B) < \rho_{thresh}$, then A becomes a part of B, and *vice versa*,
- If both ratios are lesser than ρ_{thresh}, then the class of smaller area is associated to the class of bigger area.

Finally, if a class B has to be merged with its two neighbour classes A and C, B becomes a part of the class that has both the biggest area and the biggest ratio of integrity :

$$arg \max_{X \in \{A,C\}} \Big(\mathcal{A}(X) \rho(X/B) \Big) = arg \max_{X \in \{A,C\}} \Big(\mathcal{A}(X \backslash B) \Big)$$

The details of classes association for a Gaussian kernel can be found in Appendix B. A practical demonstration case of association is explained in Fig. 2.

Downloaded From: http://spiedigitallibrary.org/ on 09/12/2012 Terms of Use: http://spiedl.org/terms

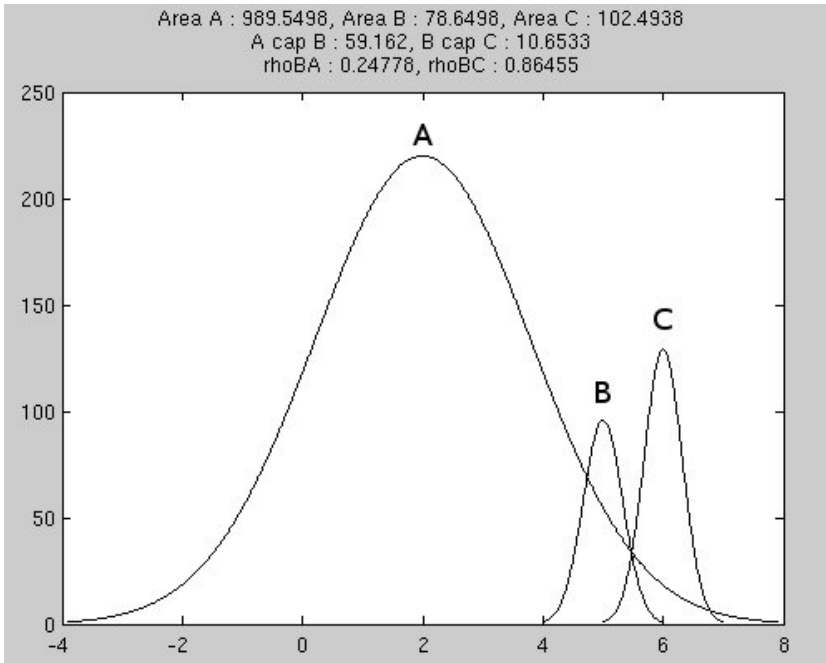

Figure 2. Illustration of the association decision taken depending on the type of intersection. Our decision concerns the association of class B. B has two intersections with A, and one intersection with C. A and C intersect as well in two points, but this is not important in the decision process. The ratios of integrity were computed by using the developments in Appendix B. In this case, if we chose $\rho_{thresh} = 0.9$ (90% integrity required to survive), B has to be merged with both A and C. Following the presented process, we would then decide to merge B with the class that has the biggest *pure* area. We would therefore compute $\mathcal{A}(A \backslash B) = \mathcal{A}(A) - \mathcal{A}(A \cap B) = 930$ and similarly $\mathcal{A}(A \backslash C) = 92$, and we would chose to merge B with A.

2.2.2 Association computation

What we get from the clustering process is a class-mapping that associates a *class number* to a *gray level*, from which we compute the PCM-1. What we get from the previous computations is an associations mapping that associates a *class number* to another *class number*, or the same one if there is no change.

Computing the new class-map from the associations map is not straightforward, because in the general case, the associations to compute take the form of an oriented non-connex graph, in which we have to select only the *root nodes*, and to assign every gray level to the correct class among the remaining classes.

If this graph is acyclic, an easy solution is to recursively apply the associations mapping to the initial class-map until there is no change between two recursions. That way we ensure that the new class-map contains only the root nodes of the association graph (at each recursion, we "climb" one level closer to the root), and we by-pass the problem of associating each gray level to one of these classes because we process the class-map directly.

However, if the required integrity threshold is high, cycles may appear in the associations graph, which would bring the previous process to an endless loop. We solve this problem in Appendix C. An example association process is shown in Fig. 3 with the following associations map :

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 1 & 4 & 5 & 5 & 5 \end{bmatrix}$$

Furthermore, it might be necessary to apply this association algorithm several times to ensure that all the remaining classes have an intergrity ratio greater than ρ_{thresh}. The resulting class-map leads to the computation of a second Piecewise Constant Mask (PCM-2).

Downloaded From: http://spiedigitallibrary.org/ on 09/12/2012 Terms of Use: http://spiedl.org/terms

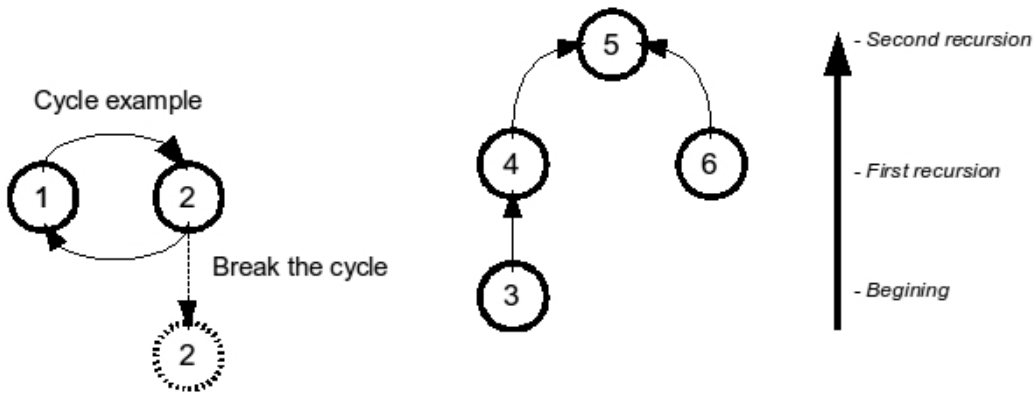

Figure 3. Example association process. At the beginning the six classes are represented in the classes map. Before beginning the recursion, we detect and break the cycle between 1 and 2 using the method described in Appendix C. At the first recursion, the remaining classes are $(2, 4, 5)$. At the second recursion, only 2 and 5 remain. The recursive association litterally makes us "climb" to the roots of the tree. The arrow on the right illustrates that for an association tree with a maximum depth n, the depth at recursion k is $(n - k)$.

The visual effect of this algorithm is to associate neighbour regions in the image that have a close gray level intensity. This association allows to litterally discard details in the image and extends the size of the visible clusters.

2.3 Topological knowledge representation

The previous parts of the method described how to compute a Piecewise Constant Mask (PCM-1 or 2) containing a relatively small number of regions. The goal of this next step is first to index all the connex regions in the PCM, which results in another mask that we call the Indexed Clusters Mask (ICM), and secondly to build the hierarchical graph from this mask.

The computation of the ICM is easily done by assigning a unique label to every connex region in a given PCM. In the following we define the hierarchical graph and present how to compute it efficiently.

2.3.1 Hierarchical graph

The hierarchical graph is a representation of the structural knowledge computed from the input image/volume :

- Each node of the graph corresponds to a cluster in the ICM,
- Each edge of the graph means that the two connected nodes are neighbours in the image/volume,
- The graph is supposedly oriented, which means that the relative information are different from the "point of view" of both nodes.

Even though this representation is simple and brings no direct added value for segmentation, the possibilities it offers for image processing are countless. Indeed, it defines a proper context to extract and use topological information from the input dataset :

- Each node can be caracterized itself with an unlimited number of features (mean intensity, size, shape, intensity covariance matrix, principal axes, etc.)
- Each edge can store an unlimited number of features caracterizing the relationship between the two connected nodes (relative position, intensity difference, shape similarity, etc.)

Downloaded From: http://spiedigitallibrary.org/ on 09/12/2012 Terms of Use: http://spiedl.org/terms

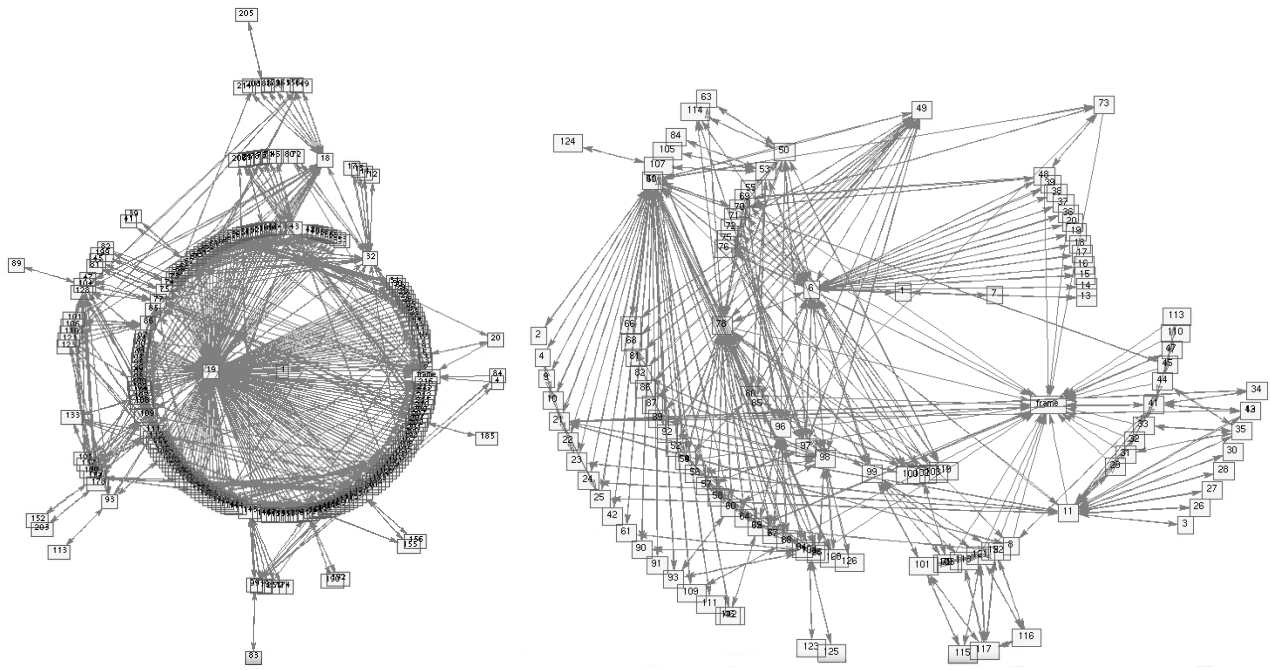

Figure 4. Example hierarchical graphs computed from CT-slices. The graph on the left corresponds to the use of the clustering maximising the height of the kernel, with an 85% association. It contains 226 nodes. The graph on the right corresponds to the use of the clustering maximizing the area, without association. It contains 126 nodes.

These information are stored in a structural dataset $\mathfrak{S}$ composed with :

- A matrix $\mathfrak{S}_N$ in which the ith row is the vector of caracteristics of node i,
- A third-order tensor $\mathfrak{S}_E$ in which the vector $\left(\mathfrak{S}_E(i,j,k)\right)_k$ contains the information relative to the edge from node i to node j.

In the following, we describe how to compute efficiently the hierarchical graph from the ICM. We then discuss the possibilities offered by the use of the structural dataset as previously defined.

2.3.2 Efficient computation of the hierarchical graph

The ICMs we were able to compute on several CT-slices typically contained an important number of small clusters (several thousands of clusters smaller than 15 pixels on 512×512 slices). Even if these small clusters did not affect significantly the quality of the coarse segmentation (the anatomical structures were still clearly visible), the computation of the hierachical graph was affected.

Simply discarding the small clusters in the ICM would result in a segmentation mask with holes, which could be acceptable. However, to quickly reduce the number of small clusters in the ICM we should : first, include a holes-filling process in the indexation step to remove most of the small clusters, and secondly, associate the remaining small clusters with their most representative neighbour (the neighbour that shares the largest border with the current cluster). An example case of such processing is shown on figure Fig. 5.

The holes-filling process and the computation of the neighbourhood for each cluster were surprisingly by far the most costly tasks in the whole process of segmentation. But it was necessary both to associate the small clusters, and to compute the structure of the hierarchical graph. The computation of the neighbourhood is however a very simple algorithm, essentially parallelizable, and its implementation on GPU should present no challenge if it needs to be computed fast.

Ultimately, the computation of the hierarchical graph is achieved by computing its binary adjacency matrix with as many rows/columns as there are of clusters in the ICM, and for each row (*i.e.* each node), putting 1 in the column corresponding to a neighbour cluster in the ICM. Examples are shown on figure Fig. 4.

Downloaded From: http://spiedigitallibrary.org/ on 09/12/2012 Terms of Use: http://spiedl.org/terms

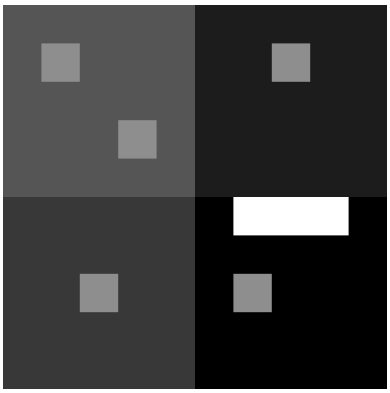

Figure 5. Example of small clusters association on a 10×10 pixels image. We chose to discard the regions smaller than 4 pixels : the lighter gray holes will be filled during the indexing of the four main squares. The white stripe between the two right main squares will be discarded and has to be associated afterwards by analysing its neighbourhood. In this case, either with a 4 or 8-connected neighbourhood, it will be associated to the darker main square.

3. RESULTS

The results were computed on a Mac Mini Intel Core 2 Duo 2.53GHz with 4Go RAM. The implementation used Matlab v7.8. Results on an example CT-slice are shown on figure Fig. 6 ; the clustering in the histogram affected the CT-slice as a denoising filter. The derived ICMs show the possible segmentation capabilities offered.

The method involving the clustering maximizing the area can be used with only **one** parameter, which specifies the minimum size that the final visible clusters should have. In our examples, we set this parameter to 90 pixels for all the computation on 512×512 slices. Note that with an increasing association level (integrity ratio threshold ρ_{thresh}), the ICM computed from the clustering method maximizing the height becomes similar to the one maximizing the area. This means that if the level of details captured is not satisfactory with only one parameter (area method), another parameter can be introduced to *increase* the level of details ($1 - \rho_{thresh}$ for instance).

Even though we tried to optimize as much as possible the implementation in Matlab, the computation times (Tab. 1) remain relatively high. However, the computation of the PCMs is very fast for both methods, which allows the use of our work for interactive segmentation. The clustering part of the algorithm (indexing, smoothing, graph computation) is costly mainly because of the holes filling and cluster's neighbours computations. Hopefully, these simple algorithms could run faster with an implementation in Java, C or on GPU.

For volumetric results (see Fig. 7), the implementation in Matlab could not handle the indexation of all the clusters in the PCM, nor the 3D rendering. We therefore extracted and processed dynamically the selected region directly from the PCM, and exported the result to achieve the rendering with XIP (eXtensible Imaging Platform). It should be noted that the result presented in Fig. 7 was obtained without any parameter.

	Maximizing height	Maximizing area
PCM	0.09s *(with 85% association)*	1.4s *(without association)*
Indexing	0.6s	0.6s
Holes filling	16s	7s
Small clusters association	5s	11s
Final neighbours computation	4s	2s
Graph computation	0.9s	0.5s

Table 1. Rounded average computation times in seconds on 20 CT-slices.

Downloaded From: http://spiedigitallibrary.org/ on 09/12/2012 Terms of Use: http://spiedl.org/terms

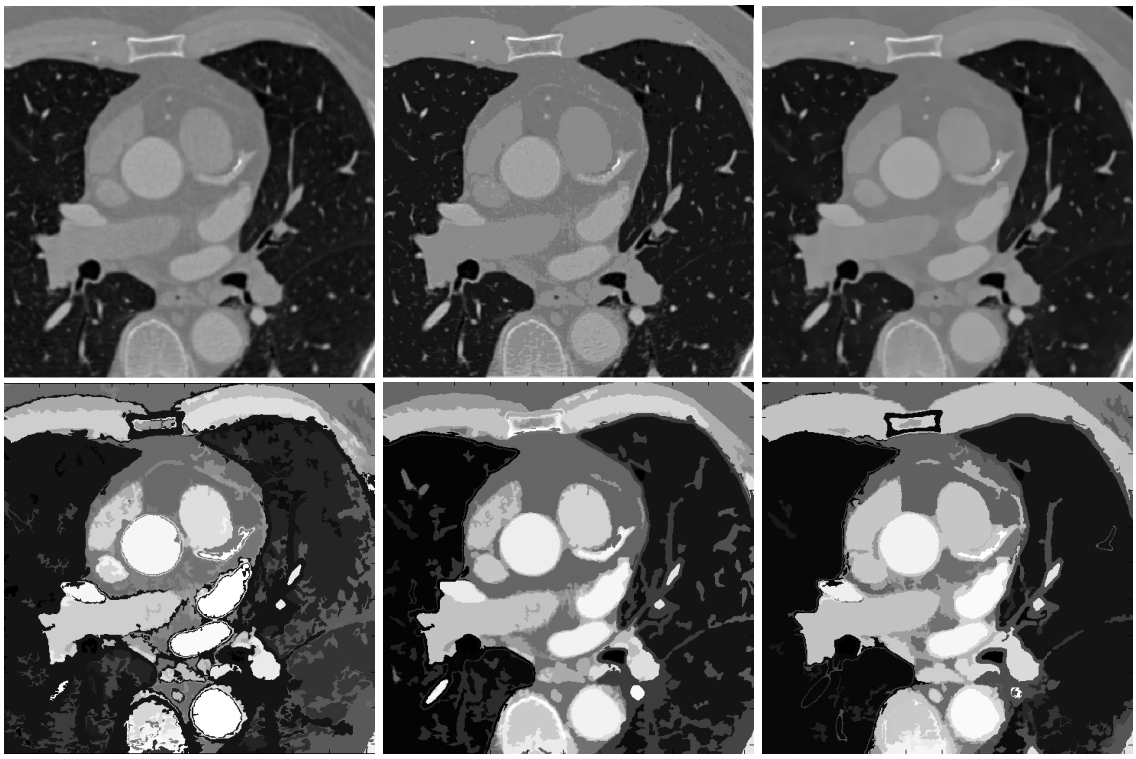

Figure 6. Example results obtained on a CT-slice. On the top-row, from left to right : the original slice ; an example PCM-1 using the clustering maximizing the area without association ; the mean-shift filtered slice with *SpatialBandwidth*=8 and *ColorDistance*=5. Bottom-row, from left to right : ICM computed with clustering maximizing the height and an 85% association (226 clusters) ; same ICM but using a 93% association (148 clusters) ; ICM corresponding to the previous PCM-1 (126 clusters).

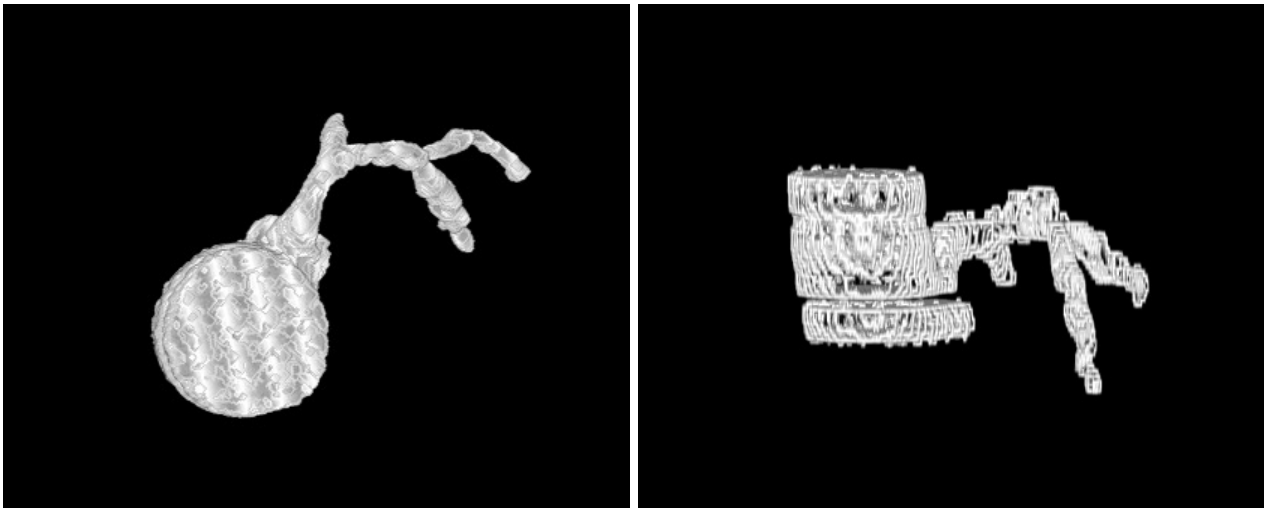

Figure 7. Coarse segmentation of the aorta computed from a 175-slices cardiac CT-scan. The volume was extracted directly from the PCM-1 using the clustering method maximizing the area ; this volume was therefore extracted from the CT-scan without any parameter.

Downloaded From: http://spiedigitallibrary.org/ on 09/12/2012 Terms of Use: http://spiedl.org/terms

4. DISCUSSION

The main contributions of this work are to offer a very simple method based on histogram analysis to obtain a coarse segmentation of CT-scans datasets, and to present an efficient, and to our knowledge unprecedented, representation of its structural topology. Our histogram processing method, although significantly simpler, allowed to compute similar results compared to the mean-shift algorithm[2] . However, for general segmentation purposes other than segmentation in CT-scans, the mean-shift filtering could be used similarly to process the hierarchical graph. This representation offers countless possibilities for further data processing, such as :

- Interactive association : the association of two clusters in the ICM is as simple as merging two nodes in the hierarchical graph.
- Interactive dissociation : a node can split in two or more nodes using a simple intensity analysis based on common clustering methods, such as Fuzzy C-means for example[3] .
- Local processing : if an algorithm is designed for a precise task on a precise anatomical structure, the coarse regions where it applies can be selected instantly in the graph, therefore reducing the computation time by focusing the processed area, and without affecting the rest of the dataset.
- Dynamic adaptive processing : similarly, algorithms applied to the entire dataset can use the local structural information extracted from the graph to adapt dynamically their parameters in the current processed area.
- Pattern recognition : defining an appropriate distance involving the desired caracteristics to compare two structural datasets will allow to compare any CT-scans relatively to these caracteristics. Topologically similar areas can then be detected in different images by pairing nodes in their hierarchical graphs. This could allow more complex tasks such as automatic labelling for instance.

CONCLUSION

We presented a fast method with few (or no) parameters for the coarse segmentation of CT-scans based on histogram processing, and showed application results on cardiac CT-scans. In this method, we designed a non-parametric clustering process for the histogram, according to a model of the imaging modality. Our results showed the validity of this model.

The main contribution of this work is the definition of a light and efficient representation of the topological knowledge as a hierarchical graph. This type of representation is, to our knowledge, unprecedented. It offers promising potential applications for further processing on CT acquisitions, such as immediate features for interactive segmentation, or potential applications in pattern recognition. Further study on the structural dataset as we defined it would be the most pertinent first step towards the continuation of our work.

Downloaded From: http://spiedigitallibrary.org/ on 09/12/2012 Terms of Use: http://spiedl.org/terms

APPENDIX A. COMPUTE THE DEVIATION PARAMETER OF A KERNEL

The computation of the deviation parameter σ requires the first parameter μ.

Given the parameter μ, computing σ simply consists in finding the *widest* possible kernel centered at μ that fits the histogram $\mathcal{H}$. We remark that the values of the histogram have an actual meaning as *integers* only, because they represent a number of pixels. This gives us a value-tolerance of 1 for the kernel that best fits the histogram at point μ. Provided that $\mathcal{H}(\mu) > 1$, we therefore state that : $\forall u \in U,\ \mathcal{N}_{\mathbf{p}}(u) < \mathcal{H}(u) + 1$. Which is equivalent to :

$$\forall u \in U,\ \mathcal{H}(\mu) exp\left(-\frac{(u-\mu)^2}{\sigma^2}\right) < \mathcal{H}(u) + 1 \Longrightarrow \forall u \in U \mid 0 \leq \mathcal{H}(u) < \mathcal{H}(\mu) - 1, \quad \sigma < \frac{|u-\mu|}{\sqrt{\ln\left(\frac{\mathcal{H}(\mu)}{1+\mathcal{H}(u)}\right)}}$$

We developped the calculus for the last inequality to highlight that it is valid only on the interval $\mathcal{I}(\mu) = \left\{u \in U,\ 0 \leq \mathcal{H}(u) < \mathcal{H}(\mu) - 1\right\}$, which gives us the suitable values of σ given the parameter μ and if $\mathcal{H}(\mu) > 1$:

$$\sigma \in \left\{ k \min_{\mathcal{I}(\mu)} \left(\frac{|u-\mu|}{\sqrt{\ln\left(\frac{\mathcal{H}(\mu)}{1+\mathcal{H}(u)}\right)}} \right) \right\}_{k \in]0,1[}$$

These values are all suitable for both clustering criteria, and nothing specifies that we should chose k as close to 1 as possible. However in our application, we wanted to split the grayscale domain in as less classes as possible (*i.e.* as less kernels as possible), which means that σ should take the largest possible value at each step. We set k to 0.99 in our implementation.

Note that during the computation, it might happen that $\mathcal{I}(\mu) = \varnothing$. In this case, according to the previous equations, σ can supposedly take any positive value. This case needs to be handled because we need to give a value to σ at each step. Possible solutions include assigning a default value σ_0 in these cases, or chosing another point μ where $\mathcal{I}(\mu)$ is not empty for instance. In our implementation, we chose the second solution.

APPENDIX B. CLASSES ASSOCIATION FOR GAUSSIAN KERNELS

In the following, $\mathcal{G}_1$ and $\mathcal{G}_2$ are two Gaussian-kernel classes.

B.1 Intersection between classes

The intersection of $\mathcal{G}_1$ and $\mathcal{G}_2$ corresponds to the equation :

$$u^2\left(\sigma_2^2 - \sigma_1^2\right) + 2u\left(\sigma_1^2\mu_2 - \sigma_2^2\mu_1\right) + \sigma_2^2\mu_1^2 - \sigma_1^2\mu_2^2 + \sigma_1^2\sigma_2^2 \ln\left(\frac{\mathcal{H}(\mu_2)}{\mathcal{H}(\mu_1)}\right) = 0 \tag{3}$$

We denote Δ' the reduced discriminant of Eq. (3). If $\Delta' \geq 0$, the general solutions of Eq. (3) are :

$$u_{1,2} = \frac{\sigma_2^2\mu_1 - \sigma_1^2\mu_2 \pm \sqrt{\Delta'}}{\sigma_2^2 - \sigma_1^2}$$

We go to the next step if one of the solutions $u_1, u_2 \in \mathcal{D}_{1,2} = \mathcal{D}(\mu_1, \sigma_1) \cap \mathcal{D}(\mu_2, \sigma_2)$.

Downloaded From: http://spiedigitallibrary.org/ on 09/12/2012 Terms of Use: http://spiedl.org/terms

B.2 Integrity ratios computation

B.2.1 Case of one intersection

If $u_1 = u_2$ and $u_1 \in \mathcal{D}_{1,2}$, or one of the solution only is in $\mathcal{D}_{1,2}$, there is only one intersection between the two classes. We denote u_0 this intersection. Assuming that $\forall u < u_0,\ \mathcal{G}_1(u) > \mathcal{G}_2(u)$, the easiest quantity to compute is :

$$\mathcal{A}(\mathcal{G}_1 \cap \mathcal{G}_2) = \frac{\sigma_2 \mathcal{H}(\mu_2)\sqrt{\pi}}{2}\left(\mathrm{Erf}\left(\sqrt{\ln\left(\mathcal{H}(\mu_2)\right)}\right) - \frac{\mu_2 - u_0}{|\mu_2 - u_0|}\mathrm{Erf}\left(\frac{|\mu_2 - u_0|}{\sigma_2}\right)\right) + \frac{\sigma_1 \mathcal{H}(\mu_1)\sqrt{\pi}}{2}\left(\mathrm{Erf}\left(\sqrt{\ln\left(\mathcal{H}(\mu_1)\right)}\right) + \frac{\mu_1 - u_0}{|\mu_1 - u_0|}\mathrm{Erf}\left(\frac{|\mu_1 - u_0|}{\sigma_1}\right)\right)$$

The integrity ratio for class $\mathcal{G}_1$ relatively to class $\mathcal{G}_2$ (symetrical for class $\mathcal{G}_2$) is then given by :

$$\rho\big(\mathcal{G}_1/\mathcal{G}_2\big) = 1 - \frac{\mathcal{A}(\mathcal{G}_1 \cap \mathcal{G}_2)}{\mathcal{A}(\mathcal{G}_1)}$$

B.2.2 Case of two intersections

If $u_1, u_2 \in \mathcal{D}_{1,2}$ and $u_1 < u_2$, there are two different intersections between the two classes. Assuming that $\mu_1 < \mu_2$ and that $\forall u \in [u_1, u_2],\ \mathcal{G}_1(u) \geq \mathcal{G}_2(u)$, we compute as in the previous paragraph :

$$\mathcal{A}(\mathcal{G}_1 \cap \mathcal{G}_2) = \frac{\sigma_2 \mathcal{H}(\mu_2)\sqrt{\pi}}{2}\left(\mathrm{Erf}\left(\frac{|u_1 - \mu_2|}{\sigma_2}\right) - \frac{\mu_2 - u_2}{|\mu_2 - u_2|}\mathrm{Erf}\left(\frac{|u_2 - \mu_2|}{\sigma_2}\right)\right) + \frac{\sigma_1 \mathcal{H}(\mu_1)\sqrt{\pi}}{2}\left(2\mathrm{Erf}\left(\sqrt{\ln\left(\mathcal{H}(\mu_1)\right)}\right) - \mathrm{Erf}\left(\frac{|u_1 - \mu_1|}{\sigma_1}\right) - \mathrm{Erf}\left(\frac{|u_2 - \mu_1|}{\sigma_1}\right)\right)$$

Finally, if $\mu_2 < \mu_1$, the formula is not exactly symetrical (a sign coefficient changes) :

$$\mathcal{A}(\mathcal{G}_1 \cap \mathcal{G}_2) = \frac{\sigma_1 \mathcal{H}(\mu_1)\sqrt{\pi}}{2}\left(\mathrm{Erf}\left(\frac{|u_2 - \mu_1|}{\sigma_2}\right) - \frac{u_1 - \mu_1}{|u_1 - \mu_1|}\mathrm{Erf}\left(\frac{|u_1 - \mu_1|}{\sigma_1}\right)\right) + \frac{\sigma_2 \mathcal{H}(\mu_2)\sqrt{\pi}}{2}\left(2\mathrm{Erf}\left(\sqrt{\ln\left(\mathcal{H}(\mu_2)\right)}\right) - \mathrm{Erf}\left(\frac{|u_1 - \mu_2|}{\sigma_2}\right) - \mathrm{Erf}\left(\frac{|u_2 - \mu_2|}{\sigma_2}\right)\right)$$

The resulting integrity ratios are computed the same way as in the previous paragraph.

APPENDIX C. DETECT AND BREAK CYCLES IN ASSOCIATION MAPPINGS

Here, we suggest a fast and easy way to detect and break cycles in the presented association process. Beware that this method works only in our case, where the association is done only with the direct neighbours. Given a classes map in the form of a vector $(c_1, ..., c_n)$, the association mapping is fully defined with a $2 \times n$ matrix which coefficients $a_{i,j}$ are such that $a_{1,i} = c_i$ and $a_{2,i} = c_{f(i)}$ where $f(\cdot)$ represents the association process and $f\big(\{1..n\}\big) \subset \{1..n\}$ (if the two sets are equal, there is no association).

We define the following sum and product coefficients :

$$\forall i \in \{1..n\}, \quad s_i = a_{1,i} + a_{2,i} \qquad \text{and} \qquad p_i = a_{1,i}a_{2,i}$$

and then compute the following vector :

$$L = \begin{bmatrix} \quad |s_{i+1} - s_i| + |p_{i+1} - p_i| \quad \end{bmatrix}_{i \in \{1..n-1\}}$$

The cycles involving two neighbour classes in the association process correspond to the zeros of L. For example, if $L(i) = 0$, then c_i and c_{i+1} form a cycle, which we simply break by assigning $a_{2,i} = c_i$ or $a_{2,i+1} = c_{i+1}$.

Downloaded From: http://spiedigitallibrary.org/ on 09/12/2012 Terms of Use: http://spiedl.org/terms

REFERENCES

[1] Otsu, N., "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man and Cybernetics* **9**, 62–66 (January 1979).

[2] Comaniciu, D. and Meer, P., "Mean shift: a robust approach toward feature space analysis," *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **24**, 603 –619 (May 2002).

[3] Bellemare, C. and Meunier, J., "Structural method for tracking coronary arteries in coronary cineangiograms," *Canadian Conference Computer and Robot Vision* , 378–384 (2004).

[4] Zheng, Y., Barbu, A., Georgescu, B., Scheuering, M., and Comaniciu, D., "Fast automatic heart chamber segmentation from 3d ct data using marginal space learning and steerable features," in [*ICCV 2007*], 1 –8 (2007).

[5] Rivest-Henault, D., Duong, L., Lapierre, C., Deschenes, S., and Cheriet, M., "Semi-automatic segmentation of major aorto-pulmonary collateral arteries (mapcas) for image guided procedures," *Medical Imaging 2010: Visualization, Image-Guided Procedures, and Modeling* **7625**(1), 76250B, SPIE (2010).

[6] Preim, B. and Bartz, D., eds., [*Visualisation in Medicine, Theory, Algorithm, and Applications*], Morgan Kaufmann (2007).

[7] Teh, Y. W., Jordan, M. I., Beal, M. J., and Blei, D. M., "Hierarchical Dirichlet processes," *Journal of the American Statistical Association* **101**(476), 1566–1581 (2006).

Downloaded From: http://spiedigitallibrary.org/ on 09/12/2012 Terms of Use: http://spiedl.org/terms