Semi-Automatic Anatomical Tree Matching for Landmark-Based Elastic Registration of Liver Volumes

Klaus Drechsler, Cristina Oyarzun Laura, Yufei Chen, Marius Erdt
Fraunhofer Institute for Computer Graphics Research, Fraunhoferstr. 5, 64283 Darmstadt, Germany
E-Mail: {Klaus.drechsler, cristina.oyarzun, yufei.chen, marius.erdt}@igd.fraunhofer.de

ABSTRACT
One promising approach to register liver volume acquisitions is based on the branching points of the vessel trees as anatomical landmarks inherently available in the liver. Automated tree matching algorithms were proposed to automatically find pair-wise correspondences between two vessel trees. However, to the best of our knowledge, none of the existing automatic methods are completely error free. After a review of current literature and methodologies on the topic, we propose an efficient interaction method that can be employed to support tree matching algorithms with important pre-selected correspondences or after an automatic matching to manually correct wrongly matched nodes. We used this method in combination with a promising automatic tree matching algorithm also presented in this work. The proposed method was evaluated by 4 participants and a CT dataset that we used to derive multiple artificial datasets.

Keywords: Liver, Anatomical Trees, Tree Matching, Interactive Refinement, Visualization

1. INTRODUCTION

The registration of liver volumes gathered from Computed Tomography (CT) or Magnetic Resonant Tomography (MRT) is a necessary step for the qualitative and quantitative comparison of pre- with postoperatively gathered data to validate the outcome and accuracy of the surgery with a resection plan. Recently, Lange et al. [1] reported a method based on interactively chosen corresponding landmarks and non-rigid registration to validate the resection plan using pre- and postoperative CT data. The outcome of the surgery is judged based on the remaining vessel parts which were resected or not resected as planned, instead of a comparison between planned and resected liver volumes.

The registration of liver volumes gathered from preoperative CT or MRT and intraoperative three-dimensional ultrasound is another important application to assist the surgeon during the resection of the liver (hepatectomy). The basic idea is to visualize the registration results during intervention to improve the orientation of the surgeon.

Even though in the case of outcome validation, the liver deformations are not as much as in the situation during the intervention. Significant deformations are still present, either because of breathing motion or simply because parts of the liver have been resected. Due to the soft-tissue nature of the liver, non-rigid registration methods should be used to take deformations into account.

Different methods have been proposed to register liver volumes. The most promising ones are, in our opinion, landmark-based methods (e.g. [1, 2]). They are based on branching points of the vessel trees as anatomical landmarks inherently available in the liver. The rough idea is described as follows. After the segmentation of the liver vessels, a skeleton is created. The skeleton is then transformed into a formal graph representation where the root, the leaves and branching points are represented by attributed nodes. The formal graph representations serve as input to a graph matching algorithm whose task is to automatically find pair-wise correspondences between two vessel trees. The matching is then used to deform one dataset to match the other. The quality of the matching directly influences the results of the registration algorithm. However, to the best of our knowledge, no existing automated tree-matching algorithm is able to correctly match all available nodes and has been sufficiently tested. Charnoz et al. [3], for example, reported an algorithm able to match 90-95% of the nodes, while the algorithm by Lohe et al. [4] could match 80-90% of the nodes. Metzen et al. [5] reported that their algorithm produced no false matching. However, they evaluated their algorithm only with one liver dataset and compared only a few matchings that were manually selected as ground truth. Their algorithm found only 50% of those matches. The use of approximating graph-matching methods can be problematic, because wrongly matched correspondences may lead to contortions of the interpolation [2]. The reasons for false matchings are mainly due to partial volume effects, inaccurate segmentation algorithms and low spatial resolutions (especially when ultrasound is being used) the extracted formal trees to be matched are defective and incomplete. This leads to the need of efficient and intuitive interaction mechanisms. Lange et al. noted that the interactive determination of landmarks is tedious and time-consuming [1]. They motivated the need for an efficient interaction mechanism based on the fact that usually only about 5-6 and rarely up to 10 branching points can be selected during the available time in the operating room [6]. They proposed a method which visualizes the vessels as surfaces; when the surface near a branch is clicked by the user, the nearest branching point is chosen automatically.

The published works in this area concentrated either on completely manual or fully automatic landmark placement. In this article, we try to bridge the gap between completely manual landmark placement and fully automatic tree matching by proposing an efficient interactive tree matching method with several visualization features that can be used to support automatic tree matching algorithms with important pre-selected correspondences or after an automatic matching to manually correct wrongly matched or unmatched nodes. This method will be combined with a recently published promising tree matching algorithm to match liver vessel trees. The proposed method was evaluated by 4 participants. A CT dataset was employed to derive multiple artificial datasets. The application of this semi-automatic tree matching method is not limited to CT, but also to match vessel trees extracted from different modalities such as 3D ultrasound. Thus, our method could add value to recently proposed registrations methods like those in [1, 6], where landmarks must be interactively chosen.

The paper is organized as follows. In section 2, an extensive review of related work is presented. In section 3, our methodology with emphasis on interactive and automated tree matching is described. In section 4, the results of four experiments to evaluate the interactive and automated tree matching components are presented and discussed. Section 5 concludes this work.

2. RELATED WORKS

2.1 Organ Segmentation

Numerous 2D and 3D automatic and semi-automatic methods for liver segmentation have been proposed, such as statistical shape models [7], atlas registration [8], level-sets [9], graph-cuts [10], and rule-based systems [11]. Amongst them, model-based approaches have been established as one of the most successful methods since they incorporate *a-priori* information about the expected shape. Therefore, they generally perform well in regions with poor organ boundaries and produce few segmentation artifacts. An overview of state of the art model-based approaches that incorporate statistical shape information can be found in [7].

Another class of segmentation algorithms is based on registration of the input image with a probabilistic label image, *atlas*. Each voxel in the *atlas* has an assigned probability, which classifies it into one tissue class. By registering the atlas with the input image, the tissue of interest can be identified. The quality of the resulting segmentation depends primarily on the performance of the registration method. Due to the large inter-patient variability of organs' shape and appearance, a robust registration is problematic. Therefore, *atlas*-based registration is often used as a pre-segmentation step to compute starting positions for other segmentation algorithms [8].

A popular method for image segmentation takes the level-set approach based on the definition of a cost function to control the evolution of a front (contour) towards the boundaries of the organ [8, 9]. The cost function is commonly based on a term that represents image features (edges) and a term to ensure smoothness of the contour. The level-set method is efficient for image segmentation and can also be used in combination with model-based approaches.

Another well known technique for image segmentation is the graph-cut algorithm where all voxels are connected with each other forming a graph [10]. Starting with manually or automatically placed seed regions for fore- and background, the remaining voxels of the image are classified based on finding the best *cut* through the graph. This cut is based on a cost that is assigned to every link of the graph (such as a boundary cost). Graph-cut methods are mainly used for manual or interactive segmentation, since the result is strongly influenced by the placement of the initial seeds.

In a rule-based segmentation of the liver, a set of rules using *a priori* knowledge about human anatomy in CT-data sets is defined for segmentation [11]. Those rules incorporate intensity distributions, neighborhood relations or geometric features of organs and structures. The liver is segmented by first excluding structures that are easy to detect, like background air and lungs. Based on that knowledge, more complicated structures are successively segmented until finally enough information is available to extract the liver. While such a technique can produce good results, it is mainly based on empiric knowledge. Finding appropriate rules therefore requires a complex analysis of a large and representative test data base.

A comprehensive overview of the results from the MICCAI ’07 Liver challenge is given in [12] where 16 teams evaluated their algorithms on a common database. Model-based techniques placed top regarding the automatic approaches and could also compete well with interactive methods on the majority of test cases.

2.2 Vessel segmentation

This section gives only a rough overview of vessel segmentation methods. A comprehensive survey can be found in [13].

Thresholding-based methods classify voxels according to their intensity values. However, they completely neglect available *a priori* knowledge about the objects to be segmented.

Boundary-based methods use gradient filters to detect object boundaries. The result is then further processed to produce closed curves that represent an object. Region-based methods avoid the major drawback of thresholding-based techniques by incorporating knowledge about the object to be segmented. They make use of the fact that voxels within the same region belong together and have similar intensity values.

The method presented by Selle et al. [14] is a combination of boundary- and region-based methods to segment liver vessels through a two step approach. In the first step, the vessels are enhanced by applying a median-filter to reduce salt and pepper noise and blurring the image with a gauss-filter. In parallel, a Laplace-filter is employed to detect edges which are then enhanced in the blurred image. In the second step, a region-based method is used to segment the liver vessels. This method has the following drawbacks. Median-filtering results in a loss of fine details, and gauss-filtering usually does not preserve object-boundaries and also results in a loss of fine details by blurring away small structures. Furthermore, a set of seed points is necessary from where the region growing process starts and thresholds that define a class of similar intensity values must be specified. Blurring of object boundaries is avoided by enhancing the edges in the blurred image. However, the loss of fine details remains.

Manual selection of seed points is error-prone but segmentation results depend on the choice of good seed points, making it sometimes difficult to reproduce the results. Region-based methods that do not depend on seed points have been proposed for example by Lin et al. [15].

Wrongly selected thresholds lead to an under- or over-segmentation. The former leads to an incomplete segmentation while the latter leads to a leaking into surrounding structures, thus segmenting too much of the image. Selle et al. proposed a method to automatically suggest a reasonable threshold for liver vessel segmentation [16], based on the observation that the number of segmented voxels is approximately linear for a threshold range where only vessels are segmented. The slope changes dramatically when the region grower starts to leak. In this case, a disproportionately high amount of liver parenchym is segmented. This can be used to fit two regression lines to the two characteristic parts of the curve. The crossing of both lines is suggested as a threshold. However, while providing a good guess of the threshold, it is not guaranteed that the value is the best. Thus, an application must still allow for user interaction.

An ideal filter should enhance vessels by sharpening their edges and removing noise in homogenous regions while preserving small structures. Image filtering based on anisotropic diffusion [17] is able to overcome the aforementioned shortcomings of median- and gauss-filtering [15].

Multiscale-based methods observe an image at different scales by using methods from differential geometry to analyze the present structures. Usually a linear/Gaussian scale space is used and the second order structure at different scales is analyzed [18, 19]. The main disadvantage is that due to the Gaussian scale space, small structures can be removed and nearby vessels can be detected as one. Further developments try to overcome these problems by using, for example, non-linear scale space [20] or taking first and second order structures into account [21].

2.3 Skeletonization

A variety of skeletonization methods have been developed. In general, we are interested in methods that retain the topology of the vessel tree, force the skeleton to be geometrically in the middle of the vessel, and preserve the connectivity. Methods based on distance transforming convert the original image into feature and non-feature elements, generate the distance map where each element gives the distance to the nearest feature element, and then detect ridges in the distance map as skeletal points (2D) or voxels (3D). These methods fulfill the geometrical requirement, but are sensible to noisy disturbances, and generally do not guarantee the skeleton connectivity [22, 23].

Methods based on voronoi diagrams calculate the voronoi diagram generated by the boundary points or voxels; when it goes to infinity, the corresponding diagram converges to the skeleton [24]. These methods satisfy both topological and geometrical requirements. However, it is a time consuming process especially for large objects. Therefore, these methods are not suitable for medical images.

Methods based on thinning remove from the surface the points or voxels that do not change the topology of the object until a single point/voxel wide skeleton is left. These methods preserve the topology and connectivity of the skeleton and guarantee the medial position of the skeleton, although the results are somewhat not in the completely accurate position [25, 26].

2.4 Tree matching

In principle, automatic tree matching algorithms can be divided into exact and inexact algorithms. In the case of medical imaging, due to the use of different segmentation methods as well as to the difference in resolution levels between different modalities, the extracted trees do not have the same number of nodes. Some branches will be missing, no existing matches will appear, and two or more nodes will appear very close to each other when there should be only one. Therefore, inexact matching algorithms seem to be more appropriate in this area. Nevertheless, exact matching algorithms have been adapted for medical imaging. Metzen et al. [5] used the exact matching algorithm of Pelillo et al. [27] that finds the maximal clique of an association graph using replicator equations by previously generating a particular association graph assigning the nodes and branches taking into account their attributes. Good results were achieved by testing the algorithm with only two datasets. For both datasets, the trees contained approximately 200 nodes. They considered 34 assignments as ground truth and obtained as the result approximately 80 assignments. For the first dataset (portal vein), there were no wrong matches taking into account the ground truth matches; however for the second dataset (bronchi tree), the results showed 4 errors. It is worth mentioning that only 17 ground truth matches in case of the liver and 21 in case of the bronchi were found.

Some algorithms, such as the graduated assignment algorithm of Gold and Rangarajan [28] and the fuzzy graph matching algorithm of Medasani et al. [29], are not considered for the application in medical imaging. For our application, it is of interest to consider some other inexact algorithms that have been considered for medical imaging. Among these inexact algorithms, the one by Charnoz et al. [30] generates a series of matching hypothesis based on the attributes of the branches that are continuously updated to achieve the best matches (search from root to leaves). The main drawback of this algorithm is that it depends very much on the attributes. Branches that have no similar attributes are not further considered, and when this conclusion is wrong and the branch is close to the root, the derived results are not as good as expected [5]. Another work to be mentioned is that by Tschirren et al. [31], which carries out a pruning of the spurious branches followed by a rigid registration and a search of the maximum clique. The basis of this algorithm (specially thought for the human airway) is to detect as a previous step the major branch points of the trees. This is not easy for liver trees, as pointed out by Metzen et al. [5]. For trees of 200-300 branch points, 92.9% of the matches verifiable by the ground truth matches were correct.

Kaftan et al. [32] introduced a new concept for tree matching algorithms. Instead of being based on matching nodes or branches, the algorithm matches whole paths. Unfortunately, the results (87% of the paths correctly matched) are not as good as some of the other algorithms mentioned above, since every wrongly matched path corresponds to many wrongly matched nodes.

In their model-based approach, Graham and Higgins [33] looked for some local valid matches according to a series of previously defined primitives such as leaves, bifurcations nodes, etc. The local valid matches found are iteratively compared using a similarity measure obtained from the attributes of nodes and branches to achieve the final best possible global match. The algorithm has good results but being new it has not been tested with many datasets yet. For two trees of 341 and 131 nodes, respectively, 115 nodes were matched correctly.

2.5 Non-rigid landmark-based registration

After the tree matching, n landmarks s_i of a moving image and corresponding landmarks d_i of a fixed image are found ($i=0,\ldots,n-1$). The resulting point pairs (d_i,s_i) are used to carry out an elastic registration. Therefore, a smooth function f: $\mathrm{R}^3 \rightarrow \mathrm{R}^3$ is searched that maps s_i to d_i. Thus, $d_i = f(s_i)$.

Splines have been commonly used in the literature to handle this task. Thin plate splines were used to evaluate the target registration error (TRE) in an experiment with 30 datasets of patients [2]. The 75% percentile of the TRE was 6.1 mm. It is worth noting that it was concluded that thin plate splines are not very efficient to extrapolate the deformations in areas where no landmarks are given. Thus, the TRE is small in the depth of the liver, because many branch points are available. In contrast, the TRE is big near the surface of the liver, because landmarks are very sparsely distributed. However, this is not necessarily a problem as computer based navigation support is mostly needed in scenarios where the tumor lies in the depth of the liver and not at the surface where the doctor can easily feel and possibly see the tumor.

The method in [1] used not only branching landmarks of the vessel tree, but also additional landmarks on the vessel segment between two branching landmarks. Interpolating thin plate splines were compared with approximating and interpolating Gaussian elastic body splines. An average weighted Euclidean distance of 1.0 mm was achieved after a registration with approximating Gaussian elastic body splines. The same authors in [1] presented a hybrid multi-modal (CT, 3D-US) non-rigid registration algorithm based on anatomical landmarks and intensity information in [6]. They concluded that a registration accuracy of 3 mm was feasible.

3. METHODOLOGY

The processing steps necessary to carry out a semi-automatic tree matching of the liver vasculature are described in this section. The main difference between our work and recently published works is that we try to bridge the gap between completely manual landmark placement and fully automatic tree matching. We combine an interactive tree matching component with an automatic tree matching algorithm. Figure 1 presents an overview of these steps which are described in the following subsections.

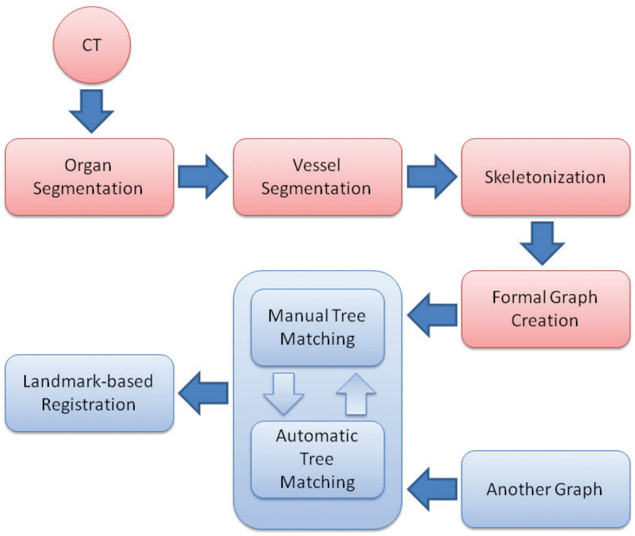


Figure 1. Overview of the processing chain for a landmark-based registration of organs based on their vessel trees. The steps colored in red are preoperative. The blue colored steps are during intervention or postoperative in case of a pre- and postoperative outcome validation.

3.1 Organ segmentation

Before further processing, the whole liver is segmented through a model-based approach for the following reasons. First, it is a very efficient way to reduce the amount of data that will be processed in subsequent steps. Second, it allows for an appealing visualization of the whole liver before and after registration (Figure 2). Third, because we are interested in the vessels of the liver that are connected to a huge vessel system spreading through the whole body, it makes the vessel segmentation step much easier.

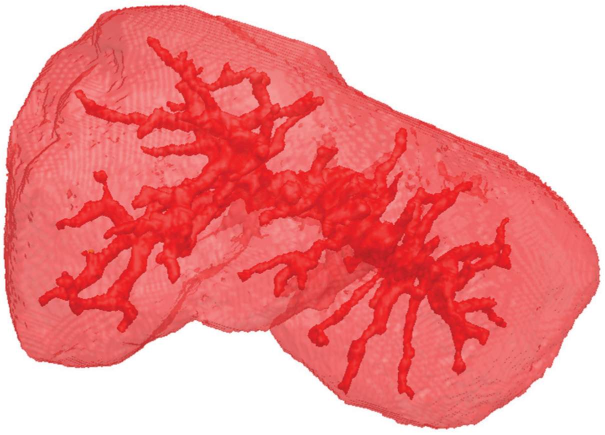


Figure 2. Visualization of a segmented liver with its portal vein.

Figure 3 depicts the workflow of our model-based organ segmentation system. First, an organ-specific model is loaded and positioned in the selected dataset by the user. Our model consists of two parts: a geometric shape that is derived from a labeled organ atlas [34] and a per-vertex definition of local shape constraints that control the local stiffness and adaptation force of the model, respectively. These constraints have been empirically identified based on the analysis of edge quality and similarity of gray values of neighboring structures using an image data base of 16 pathology-free cases. After preprocessing (see Figure 3), a multistage adaptation is applied as follows. In order to coarsely adapt the model to the underlying data while preserving the global shape, only affine deformations are allowed initially. In this stage, the model coarsely adapts to the real boundaries by searching for edges and typical grey value profiles along the model's normals. The resulting boundary mesh is affinely registered with the original mesh using the ICP algorithm. This procedure is repeated until there is no significant change in the transformation between two consecutive iterations.

The final segmentation is computed using the optimization in [35] enhanced by a per-vertex weighting using the local shape constraints described above. After some iterations of this free form deformation, the result only varies minimally and the adaptation process is stopped.

Figure 4 exhibits the results of the affine and free form deformation. At any step of this procedure, the user has the opportunity to refine the result if necessary using 3D

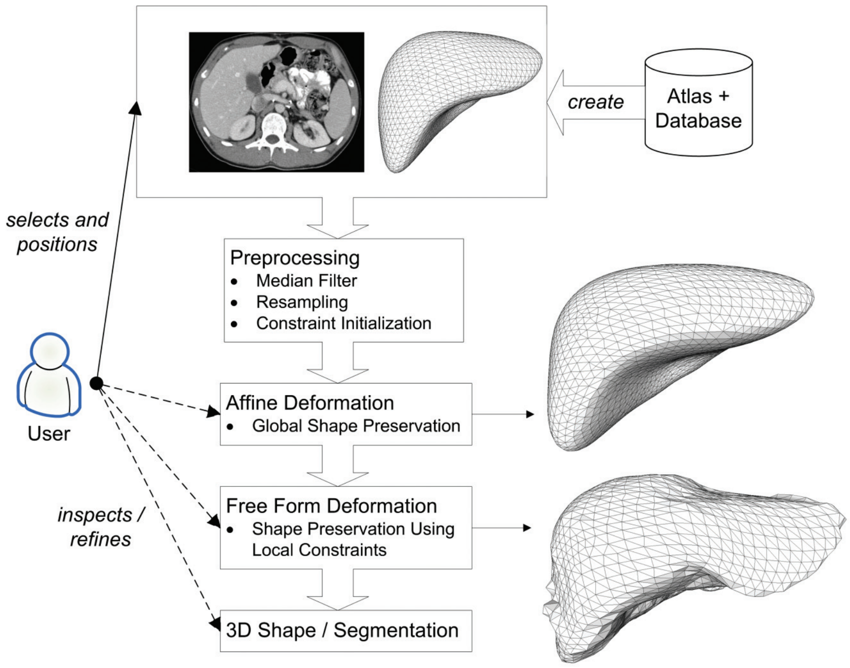


Figure 3. Overview of the model-based segmentation workflow. Mandatory and optional user interactions are displayed as solid and dotted lines, respectively.

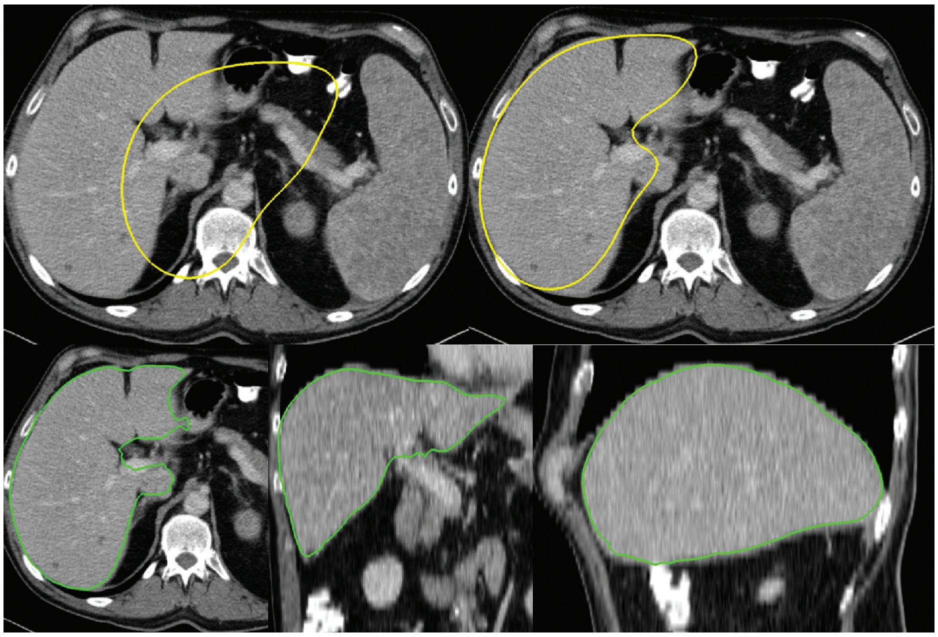


Figure 4. Top row: Affine Registration of the organ model with the dataset. Left: initial position, right: end result. Bottom row: End result of the model-based segmentation after free form deformations.

mesh refinement tools. This generally makes the segmentation more robust, for example, in cases which contain pathologies.

3.2 Vessel segmentation

In order to segment the liver vessels, we rely on datasets of contrast enhanced CT that are routinely performed in local clinics. The segmentation involves three steps depicted in Figure 5. First, a median filter is applied to reduce salt and pepper noise from the segmented organ. A variation of the classic Perona-Malik anisotropic diffusion equation [17] known as modified curvature diffusion equation (MCDE) [36] is then applied to the volume, which is less sensitive to contrast and preserves finer detailed structures. The effect of anisotropic diffusion is that it blurs the image while preserving edges. Figure 6 shows one slice of a contrast enhanced CT dataset (a), after median filtering and anisotropic diffusion (b), and without median filtering (c). The latter resulted in scattered object boundaries, which could be significantly improved by applying a median filter beforehand. Finally a region growing with a seed point at the root of the portal vein is used to segment the vessel tree. The threshold for the region growing is set manually with a slider. If a potential leak occurred, the region growing step is repeated with a modified threshold. The aforementioned filters were provided by the ITK toolkit[1].

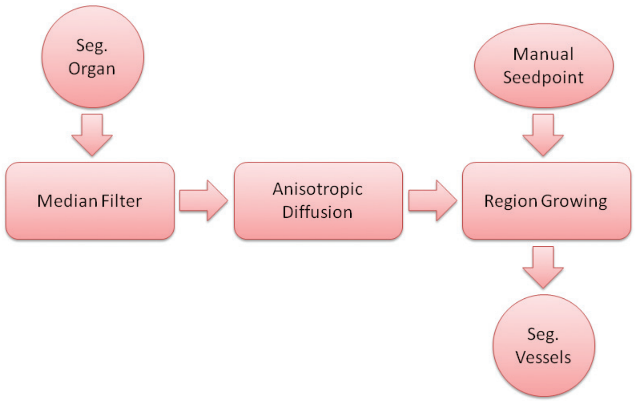


Figure 5. Processing steps for a semi-automatic vessel segmentation.

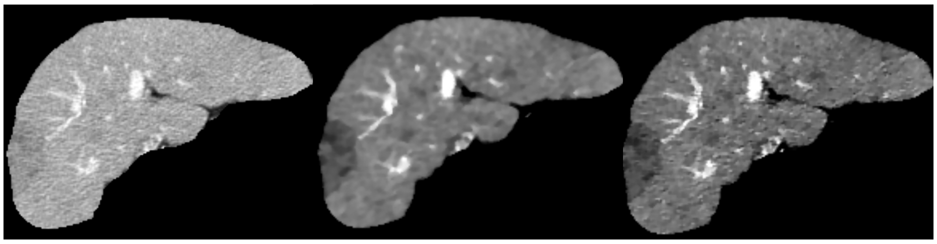


Figure 6. (a): A slice of a contrast-enhanced liver CT dataset. (b): The same slice after median filtering and anisotropic diffusion. (c): The result of anisotropic diffusion without preceding median filtering.

[1]http://www.itk.org

Because of imaging artifacts, limited spatial resolution and the segmentation algorithm itself, the segmented vessels can contain internal cavities, holes and bays which we remove with some filling-holes techniques together with morphological operations.

3.3 3-D skeletonization

The aim of the skeletonization is to extract a single-voxel wide central axis from the 3D binary liver vessel segmentation, where 0 indicates the background and 1 indicates the vessel. The basic idea of using 3D method is to repeat eroding all deletable voxels until no more change occurs, as Lee et al. did in [37]. A 3×3×3 lattice is built to examine the local connectivity of a voxel, as shown in the left image of Figure 7. The 26-neighborhood connectivity is illustrated on the right, with 6-neighborhood connectivity voxels marked in green. The deletable voxels include:

1. Voxels whose 6-neighbors do not have at least one value of "1" (border voxels).
2. Voxels that are not at the end of a line; i.e., among their 26-neighbors, more than one voxel have a value of "1".
3. Voxels that are invariant in terms of the Euler characteristic [37]; i.e., within their 26-neighbors, the Euler characteristic is preserved.
4. Simple voxels; i.e. their deletion does not change the number of connected components in the 26-neighborhood [37].

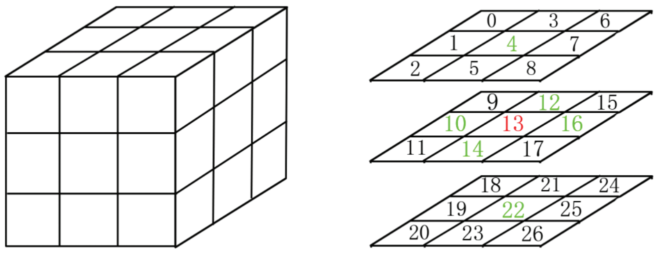


Figure 7. Neighborhood of a voxel [38].

The output is a binary image that contains a single-voxel wide skeleton marked as "1".

3.4 Formal graph creation

The formal graph representation of a vessel tree skeleton, which only preserves the topological structure of the tree, is an essential step for the following tree matching and registration. The graph can be determined from the skeleton by finding topological voxels. Topological voxels are those with only one neighbor (end-voxel), with more than two neighbors (branch-voxel), and the remaining voxels (regular-voxel). End-voxels can easily be found by counting the skeleton voxels in a 26-neighbourhood. However, when it comes to branch-voxels, this approach is not feasible. As shown in Figure 8, there are four voxels that have more than two neighbors (in red and green).

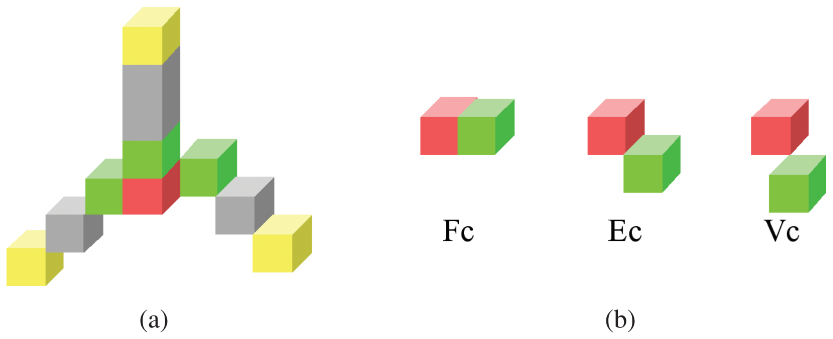


Figure 8. (a): Blocks in red and green are branch-voxel candidates, blocks in grey are regular-voxels, and those in yellow are end-voxels. (b): Different voxel connections in 3D space: Face-connection (Fc), Edge-connection (Ec), and Vertice-connection (Vc) [38].

The best candidate among all possible branch-voxels should have the highest connectivity. Connectivity is quantified by evaluating a cost function for branch-voxel candidates defined as the following:

$$Cost = 3 \cdot \sum Fc + 2 \cdot \sum Ec + \sum Vc \quad (1)$$

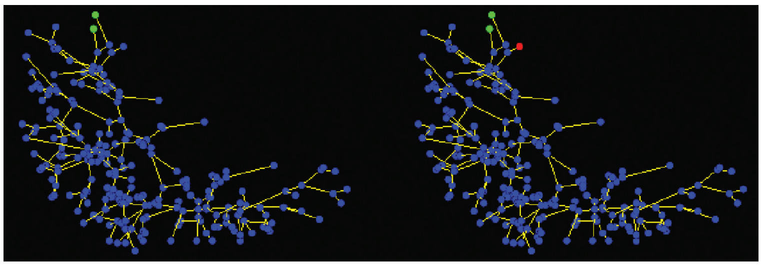
where Fc, Ec, and Vc are defined in Figure 8. The voxel with the highest cost is selected as branch-voxel. In Figure 8, this is the voxel in red. Figure 9 displays the resulting graph. Further details of this approach can be found in [38]. After the formal graph is created, attributes like exit angle, mean diameter and length of the edges are calculated to construct a similarity measure for the graph matching algorithm to be explained in section 3.6.

3.5 Interactive tree matching

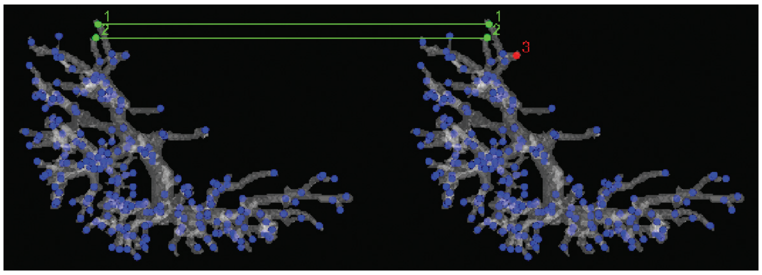
The formal graph representations serve as input to a graph matching algorithm whose task is to automatically find pairwise correspondences between two vessel trees. The matching is then used to deform one dataset to match the other. The quality of the matching directly influences the results of the registration algorithm. However, to the best of our knowledge, no existing automatic tree-matching algorithm produces perfect results. Therefore, efficient mechanisms that allow for a manual correction must be integrated into the complete registration workflow, an innovative approach of this work described in the next section.

3.5.1 Interaction and visualization mechanisms

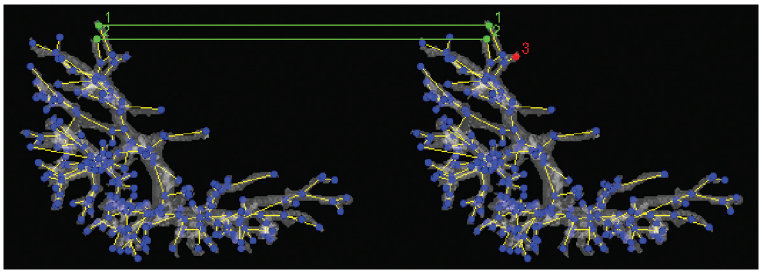
Matching all nodes of two trees manually can be an error prone and time consuming task. On the other hand, automatic methods are known to produce mismatches. The present approach represents a compromise between the two. Naturally, the tree matching step is divided into two sub-steps: interactive/manual matching and automatic



(a)



(b)



(c)

Figure 9. Visualization features. (a): a plain graph and colored nodes to visualize matches. (b): a 3D surface of the segmented vessel tree and lines and numbers to visualize the matches. (c): the surface and graph together with lines and numbers.

matching (Figure 1). The order of execution can be variable. The user can first interactively match a few (important) nodes to support the automatic matching algorithm, or first let the automatic algorithm generate a matching that can be interactively corrected. In either case, it is important that the interactive part is as intuitive as possible to reduce human errors. The following features are implemented to reach this goal:

- Both trees are visualized side by side in 3D and allow rotation for viewing from different perspectives.
- Vertices from the graph are visualized as spheres.
- Edges are visualized as lines (feature A).
- Different colors for the spheres are used in order to visualize the current status (selected, matched, unmatched).

- A matching between two nodes can be visualized by drawing identical numbers next to both nodes (feature B).
- A matching between two nodes can be visualized by drawing a line between both nodes (feature C).
- A 3D model of the segmented vessel tree is visualized transparently in the background of both graphs in order to increase the orientation by providing additional visual impressions like the diameter of a vessel (feature D).

Figure 9 depicts an impression of features A-D. The first row shows two graph representations visualized as yellow edges (feature A), as well as blue, green and red nodes where blue nodes are not matched, green nodes are already matched, and the red node is currently selected. The second row shows two graph representations without yellow edges, but with a semitransparent 3D surface model of the segmented vessels as background (feature D). Furthermore, already matched nodes are visualized by a line between the matching nodes (feature C) and numbers (feature B). In the third row, all available features A-D are turned on. In summary, feature A visualizes the connection between two nodes in an abstract way by drawing a line between them, while feature D makes the appearance of the tree more natural by drawing the segmented vessels semi-transparent in the background. Feature B visualizes a matching by adding information (numbers) locally to a node, while feature C adds information globally by drawing a line between two corresponding nodes of two trees. The interaction is through a pointing device such as a mouse, with the left click of the mouse to select nodes and to match with other nodes, and the right click of the mouse to ’unmatch’ nodes (remove from a matching).

3.6 Automatic tree matching

The algorithm chosen for the automatic tree matching is the one designed by Graham and Higgins [33]. It is a model-based approach to find the globally optimal match between two trees by considering that both trees come from an initial common structure that has undergone certain topological deformations. Relating it to medical imaging, the common structure could be the liver of a patient and both trees could correspond to the liver in different respiratory cycles or from different modalities. Therefore, the resulting trees will be different containing false branches, missing branches, etc. To deal with all these deformations, Graham and Higgins built a mathematical framework, that consisted of a series of primitives to limit the number of valid matches, and a similarity measure to determine the best between all the valid matches.

Considering trees that have no more than trifurcations, Graham and Higgins defined six primitives. Every pair of the nodes to be matched must belong to at least one of these primitives; otherwise they are not valid matches and are not considered to calculate the global optimal match. On one hand, the trees under study are considered to have planted roots, in other words, trees whose root has a degree of one (only one child). This is the first primitive. On the other hand, a node in the common tree can be a leaf or a bifurcation. With these three primitives, a Primary Deformation Model is built. This basic model deals with missing and false branches. However, these are not the only possible types of deformations that can be found between two trees extracted from medical images. Actually, it is not unusual to find situations where a trifurcation on one of the trees has a correspondence of two close bifurcations on the other tree. Similarly, nodes with reversed topological order are often found. This leads to three more primitives: trifurcations, two close bifurcations on both trees, two close bifurcations on one tree and a trifurcation on the other tree. The trees in Figure 10 demonstrate examples of the discussed primitives. Each node could belong to more than one primitive; for example, a node with a degree of 2 (bifurcation), could be a bifurcation or two close bifurcations. To determine which of the primitives to choose for a determined match, a similarity measure is employed. It uses the attributes of the nodes and branches of the trees to determine which of the primitives will lead to an optimal match. The matches with the greater similarity measure will be chosen and stored for further usage. The search for valid matches starts with the leaves and continues until it reaches the root. For all possible node pairs, it calculates their similarity measure taking into account the optimal similarity measure of the previously calculated node pairs.

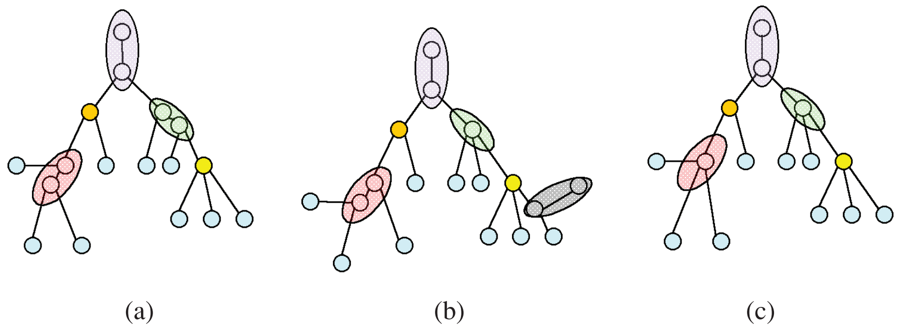


Figure 10. Primitives of the algorithm. (a), (b): two different trees. (c): the common structure. The primitives are represented by colours as follows: blue – leaves, orange – bifurcations, yellow – trifurcations, violet – planted root, red – close bifurcation/close bifurcation, green – close bifurcation/trifurcation, grey – false branch.

Many different attributes could be used for the calculation of the similarity measure. We use a combination between the length and the angle of the branches. The expression to determine the similarity measure is the following:

$$S\left(T_1, T_2 \mid \phi\right) = \mu S_{length} + (1-\mu) S_{angle}, \quad \mu \in [0,1] \tag{2}$$

where μ is a parameter that gives different weight to the similarity measure obtained from different attributes depending on the desired importance they have in the calculation. T_1 and T_2 represent both trees and ϕ represents a match, with $\phi(u)$ being the node that is matched to the node u. S_{length} and S_{angle} correspond to the similarity measure of a match taking into account their length and angle attributes, respectively. S_{length} is calculated as:

$$S_{length}\left(T_1,T_2 \mid \phi\right)=\sum_{(u_1,v_1)\in E(T_1[\phi])}\sigma_{length}\left(u_1,v_1,\phi\left(u_1\right),\phi\left(v_1\right)\right) \tag{3}$$

where $E(T[\phi])$ is the set of previously matched directed edges of a tree T. σ_{length} is the similarity between two nodes and is calculated as:

$$\sigma_{length}\left(x_1,y_1,x_2,y_2\right)=\begin{cases} -1, & if\, R_{length}\left(x_1,y_1,x_2,y_2\right)>t_r \\ \dfrac{-2R_{length}\left(x_1,y_1,x_2,y_2\right)}{t_r-1}+\dfrac{t_r+1}{t_r-1}, & otherwise \end{cases} \tag{4}$$

x_i and y_i are nodes of the trees. t_r is the saturation threshold. R_{length} is the ratio between the length of the branches (x_1,y_1) and (x_2,y_2). For ratios greater than t_r, the match is not considered. R_{length} is calculated as:

$$R_{length}\left(x_1,y_1,x_2,y_2\right)=\max\left(\frac{\lambda(x_1,y_1)}{\lambda(x_2,y_2)},\frac{\lambda(x_2,y_2)}{\lambda(x_1,y_1)}\right)\in[1,\infty] \tag{5}$$

$\lambda(x_i, y_i)$ calculates the length of the path from node x_i to node y_i. S_{angle}, σ_{angle} and R_{angle} are calculated in a similar way.

4. RESULTS

For evaluation, we used one dataset of liver acquired with a GE Medical Systems LightSpeed 16 CT Scanner. The dimension of the dataset was 512x512x291 with a spacing of 0.64x0.64x1.0 mm and a slice thickness of 1.25 mm. The methods presented above were implemented to create the formal graph representations. Two experiments were conducted using equal (subsection 4.1) and unequal (subsection 4.2) trees to measure the matches per minute that a user can produce using our interactive tree matching component with various visualization features turned on or off. Another experiment was performed to find out how easy it is was to detect and correct wrong matches due to automatic tree matching (subsection 4.3). The last experiment was conducted to evaluate the performance of the implemented tree matching algorithm (section 4.4). Among the participants in the first three experiments were computer scientists and engineers familiar with medical related topics in general and graph theory in particular.

4.1 Interactive tree matching with equal trees

In order to evaluate the amount of manual matches a user can produce using our interactive tree matching component, we prepared an experiment with a small group of four participants and 8 different setups for each participant. In each setup, one or more of the features A-C were turned on or off. Feature D was always absent for two participants, while it was always present for the others. Two identical trees were displayed according to the setup (examples can be seen in Figure 9), and the task for each participant with each of the eight setups was to correctly match as many nodes as possible in one minute. The participants were given two minutes before the test to become familiar with the software. For each setup and participant, the number of matchings was noted. After finishing the experiment, the participant was asked which visualization was preferred. In the case where no feature was turned on, the participants could only see the nodes of the tree. We expected that in this case the participants should perform worst. Similarly, they should also perform poorly in setups where feature D was absent and feature B and/or C were present. Table 1 exhibits the results of this experiment in matches per minute.

Because of the small number of participants, the results are not statistically significant; however, it suggests a trend of what is possible. During the first experiment, we observed that P1 solved the task slower but more thorough than the other three participants. We also noticed that all participants concentrated first on the nodes near the leaves, because they were easier to match. The two participants to whom feature D was available mentioned that it was very helpful for orientation. They also preferred to have feature A turned on, and so did the participants to whom feature D was not available. None of the participants favored feature C during their task, mainly because it made the interaction more confusing. Three participants indicated that this feature was very helpful for visualizing and validating the results of an automatic matching. Only one favored feature B in this case.

Table 1. Results of the first experiment with two identical trees. The numbers represent the correct matches per minute of a participant Pi for a given setup. Wrong matches, if any, are denoted in parentheses.

Setup	D present			D absent			
	P1	**P2**	**Avg.**	**P3**	**P4**	**Avg.**	**Avg.**
-	13	20	16.5	26	26	26	21.25
A	15	27	21	23	25	24	22.5
B	10	14 (1)	12	22	22	22	17
AB	14	23	18.5	24	27	25.5	22
C	14	25	19.5	23	25	24	21.75
AC	16	24	20	24 (1)	28	26	23
BC	17	27	22	23	26	24.5	23.25
ABC	17	23	20	22 (2)	21	21.5	20.75

4.2 Interactive tree matching with unequal trees

In the second experiment, we displayed two different trees (see Figure 11) to the participants to simulate a case where the second tree was derived from an ultrasound volume. Such a tree has fewer braches and possibly a slightly changed topology. Similar to the first experiment, the task was to match as many nodes as possible in one minute, but the participants were explicitly requested to start at the root and match the inner nodes first. Branches and nodes near the main veins are usually more meaningful and thus this task simulated a more realistic clinical application. Table 2 shows the results of this experiment.

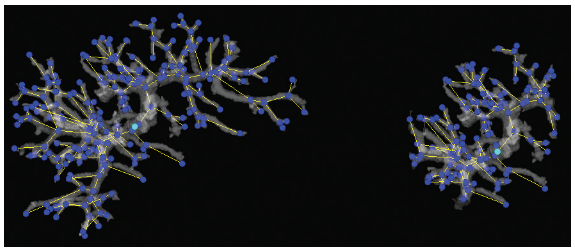


Figure 11. Two different trees for the second experiment.

Table 2. Results of the second experiment with two different trees. The numbers represent the correct matches per minute of a participant P_i for a given setup. Wrong matches, if any, are denoted in parentheses.

Setup	D present			D absent			
	P1	**P2**	**Avg.**	**P3**	**P4**	**Avg.**	**Avg.**
-	10	17	13.5	19	(1)	17	1815.75
A	11	9	10	16	19	17.5	13.75
B	9	10	9.5	20	15	17.5	13.5
AB	8 (1)	13	10.5	17	20	18.5	14.5
C	7 (2)	18	12.5	20 (1)	9 (1)	14.5	13.5
AC	10 (1)	18	14	14	19	16.5	15.25
BC	8	13	10.5	16	15	15.5	13
ABC	11	11 (1)	11	14 (1)	17	15.5	13.25

Compared to the first experiment, the matches per minute decreased significantly in the second experiment which was most likely due to the added complexity. It was pretty difficult to execute the given task without looking closely at the trees, thus entailing additional time. In both experiments, the matches per minute for the setup without any feature were surprisingly high, despite the expected worst performance. The reason may lie in the similarity of the trees. Nodes in both trees formed clusters that were easily detectable in both trees, even without visualizing structural/topological information. This could also explain the relatively high matches per minute for setups where only matchings and no topological information were visualized (A and D absent, B and/or C present). Thus, it is questionable if these setups are relevant to real clinical applications where probably no such similar clusters are present. However, if they are present, they should provide additional support during the matching task.

There is one outliner where only feature C was available. Participant 4 matched with this setup only 9 nodes although he was one of the participants with the most matches per minute. The only possible explanation was that probably he did not concentrate during this experiment. Taking into account that his performance with the same setup was much better in the first experiment and that the results of the other three participants were not conspicuous either, we can conclude that features A and C helped to increase the matches per minute. Feature C visually connects two nodes of different trees, and based on this connection, feature A helps to orient to match the neighboring nodes.

This experiment showed that it is practicable to reach 10-15 matches per minute (depending on the experience and skill of the user). If a liver vessel tree is assumed to have about 200 (more likely) to 1000 (less likely) nodes, and that the current matching algorithms correctly match 90-95% of the nodes (e.g., [3]), with our method, it will take 40-120 seconds (with a tree size of 200 nodes, the used automated tree matching produces 90-95% correct matchings and the user is able to match 10-15 nodes per minute) to 2:20-10 minutes (with a tree size of 1000 nodes, the used automated tree matching produces 90-95% correct matchings and the user is able to match 10-15 nodes per minute) to correct automatically generated matchings. The time needed to carry out an interactive matching also depends on if a registration of the whole liver or only important parts are of interest. In the latter case, the time would be further reduced.

Due to a lower resolution, a graph created from ultrasound data usually contains fewer branches than that from CT data. Therefore, in matching these two graphs, it is no necessary to display the complete graph extracted from CT. To this end, we suggest implementing an additional feature that allows reducing the height of the bigger tree to that of the smaller one.

4.3 Correcting wrong matches after automatic tree matching

In the third experiment, the participants were asked to correct wrong matches due to the automatic tree matching. The participants were given three tree pairs where the automatic tree matching algorithm produced known mismatches. We did not disclose the number of wrong matches to the participants. The result was that they were only able to correct obvious mismatches, but failed to find not so obvious ones.

The results of this experiment demonstrated the current limitations of our system. While lines between two wrongly matched nodes were very helpful when one of the involved nodes was far away from the correct node, the opposite was true when it was very close. In the former case, the line between the matched nodes showed an eye-catching atypical course compared to the other lines. In the latter case, the line did not attract any attention and was mostly overseen by the participants. To complicate matters further, lines of correct matches obstructed lines of wrong matches, making it even harder to identify.

4.4 Automatic tree matching

To evaluate the automatic tree matching algorithm, we used one tree with 192 nodes and applied some topological deformations to randomly remove 25% and 50% of the nodes, respectively. As shown in Figure 12, when one node is removed, its children are not removed but become children of the parent of the removed node, thus altering the topology of the tree. This experiment was done 30 times and 30 different trees were obtained. The number of matches from all the available nodes and the number of wrong matches were analyzed, as exhibited in Tables 3 and 4. The average of wrong matches was 1.84 when 25% of the nodes were removed, and 3.2 when 50% of the nodes were removed.

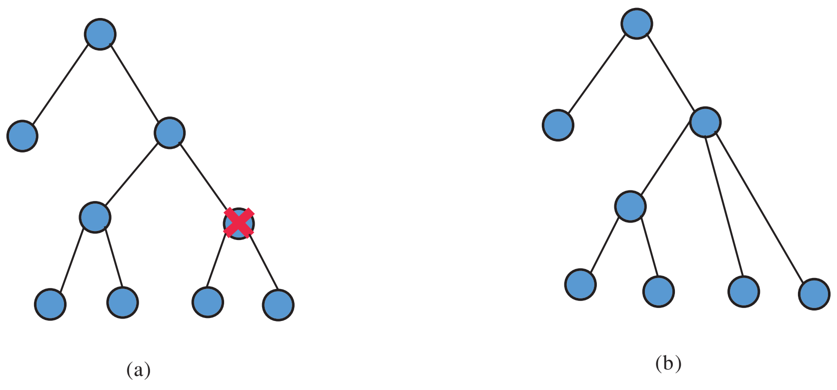


Figure 12. Additional topological deformations. (a): Random removal of a node. (b): The resulting tree.

Table 3. Results of 30 experiments (E_1 to E_{30}) with 25% of the nodes of a tree removed.

	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10
Matches	130	120	117	102	100	113	132	115	88	103
Wrong matches	0	0	1	1	1	1	0	0	4	0
	E11	**E12**	**E13**	**E14**	**E15**	**E16**	**E17**	**E18**	**E19**	**E20**
Matches	118	87	121	115	70	93	124	118	118	105
Wrong matches	0	2	1	0	3	3	3	4	0	1
	E21	**E22**	**E23**	**E24**	**E25**	**E26**	**E27**	**E28**	**E29**	**E30**
Matches	85	107	84	97	118	104	62	108	114	120
Wrong matches	3	1	0	1	0	1	0	1	2	0

Table 4. Results of 30 experiments (E_1 to E_{30}) with 50% of the nodes of a tree removed.

	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10
Matches	56	53	50	66	25	49	56	39	54	48
Wrong matches	1	0	7	2	5	2	2	4	5	4
	E11	**E12**	**E13**	**E14**	**E15**	**E16**	**E17**	**E18**	**E19**	**E20**
Matches	27	42	38	71	60	45	52	46	35	34
Wrong matches	0	3	1	6	2	1	3	2	5	1
	E21	**E22**	**E23**	**E24**	**E25**	**E26**	**E27**	**E28**	**E29**	**E30**
Matches	46	50	58	58	59	46	63	43	24	69
Wrong matches	1	1	5	0	5	3	0	7	2	4

Figure 13 shows a typical result of a larger number of matches usually achieved in our experiments. In some situations, not many matches were achieved (nodes in blue in Figure 13) due to big topological deformations. Random removal of the nodes resulted in high ramifications in some of the experiments; for example, trifurcations on one tree could become 6-furcations after node removal in the other tree. This could be solved by adding more primitives to the algorithm, but considering its final application in matching liver trees, it is unlikely to have such topologies.

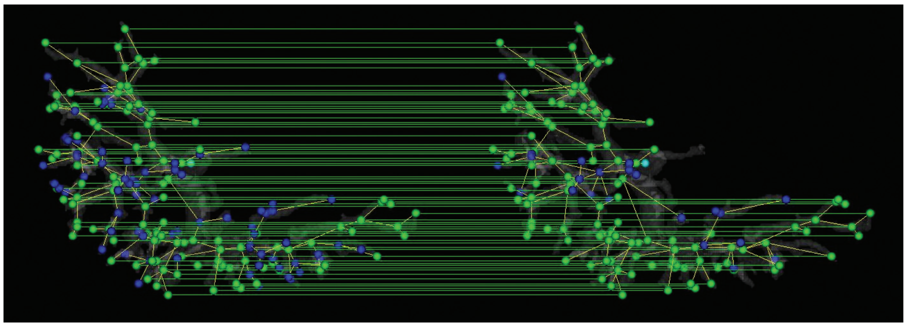


Figure 13. Result of one of the experiments described in section 4.4 with 120 nodes correctly matched and no wrong matches.

5. CONCLUSION

In this paper, we presented a method to semi-automatically match liver vessel trees, including several visualization features of the interactive tree matching component to improve the performance. According to our experiments, 10-15 matches per minute are feasible to preselect important nodes before an automatic matching algorithm is applied. Lines to visualize matches and lines to visualize the topology of the graph were found to be very helpful. Visualizing only nodes without any topological information lead to surprisingly high matches per minute due to similar clusters of nodes in both trees. Detecting wrongly matched nodes by an automatic tree matching is still a difficult task to the present system. Therefore, we came to the conclusion that it is important to use graph matching algorithms that do not match nodes at all rather than wrongly matching them, in order to decrease the complexity for the user to detect the wrong matches. The present graph matching algorithm performed very well in this sense. Even with topologically very different trees, the algorithm never had more than seven wrong matches. The output of our semi-automatic tree matching method can be used to drive a non-rigid landmark-based registration.

Future work will focus on the visualization of wrongly matched nodes through, for example, coloring based on the probability of being a wrong or correct match. Furthermore, a feature will be implemented to visualize matches iteratively from the root to the leaves in order to simplify the detection of wrong matches. The present algorithms for segmentation and graph creation will be further evaluated with more clinical datasets. Another focus will be on the application of the proposed semi-automatic tree matching approach to datasets containing tumors, particularly multiple tumors, and the relevant non-rigid registration.

ACKNOWLEDGEMENT

This work was partially supported by the Fraunhofer Internal Programs under Grant No. MAVO 817 775.

REFERENCES

1. Lange, T., Wörz, S., Rohr, K., and Schlag, P.M., Landmark-based 3d Elastic Registration of Pre- and Postoperative Liver CT Data", *Bildverarbeitung für die Medizin (BVM)*, 2009.
2. Hassenpflug, P., *Gefäß basierte Registrierung zur Computer gestützten Navigation in der Leberchirurgie*, Ph.D. Dissertation, Ruprecht-Karls-Universität, 2004.
3. Charnoz, A., Agnus, V., Malandain, G., Soler, L., and Tajine, M., Tree Matching Applied to Vascular System, *Graph-based Representation in Pattern Recognition*, Lecture Notes in Computer Science, Springer, 2005, 3434, 183-192.
4. Lohe, T., Kröger, T., Zidowitz, S., Peitgen, H.-O. and Jiang, X., Hierarchical matching of anatomical trees for medical image registration, *Proceedings of the First International Conference on Medical Biometrics (ICMB)*, Lecture Notes in Computer Science, Springer, 2008, 4901, 224-231.
5. Metzen, J.H., Kröger, T., Schenk, A., Zidowitz, S., Peitgen, H.-O. and Jiang, X., Matching of tree Structures for Registration of Medical Images, *Graph-Based Representations in Pattern Recognition*, Springer, 2007.
6. Lange, T., Papenberg, N., Heldmann, S., Modersitzki, J., Fischer, B., Lamecker, H. and Schlag, P.M., 3D Ultrasound-CT Registration of the Liver Using Combined Landmark-Intensity Information", *International Journal of Computer Assisted Radiology and Surgery (CARS)*, 2009, 4, 79-88.
7. Heimann, T. and Meinzer, H.-P., Statistical Shape Models for 3D Medical Image Segmentation: A Review, Medical Image Analysis, 2009, 13(4), 543-563.
8. Furukawa, D., Shimizu, A. and Kobatake, H., Automatic Liver Segmentation based on Maximum a Posterior Probability Estimation and Level set Method, *Proc. MICCAI Workshop 3-D Segmentat. Clinic: A Grand Challenge*, 2007, 117–124.
9. Pan, S. and Dawant, B.M., Automatic 3-D Segmentation of the Liver from Abdominal CT Images: A Level-set Approach, in: Sonka, M. and Hanson, K.M., eds*., Proc. SPIE Med. Imag.: Image Process*, 2001, 4322, 128–138.
10. Beichel, R., Bauer, C., Bornik, A., Sorantin, E., and Bischof, H., Liver Segmentation in CT Data: A Segmentation Refinement Approach, *Proc. MICCAI Workshop 3-D Segmentat. Clinic: A Grand Challenge*, 2007, pp. 235–245.
11. Schmidt, G., Athelogou, M.A., Schönmeyer, R., Korn, R. and Binnig, G., Cognition Network Technology for a Fully Automated 3-D Segmentation of Liver, *Proc. MICCAI Workshop 3-D Segmentat. Clinic: A Grand Challenge*, 2007, 125-133.
12. Heimann T. et al., Comparison and Evaluation of Methods for Liver Segmentation from CT Datasets, *IEEE Trans Med Imaging*, 2009, 28(8), 1251-1265.
13. Kirbas, C. and Quek, F.K.H., Vessel Extraction Techniques and Algorithms: A Survey, *Proceedings of the third IEEE Symposium on Bioinformatics and Bioengineering*, 2003, 238-245.
14. Selle, D., Preim, B., Schenk, A. and Peitgen, H.-O., Analysis of Vasculature for Liver Surgical Planning, *IEEE Transactions on Medical Imaging*, 2002, 21(11), 1344-1357.
15. Lin, Z., Jin, J. and Talbot, H., Unseeded Region Growing for 3D Image Segmentation, *Selected Papers from the Pan-Sydney Workshop on Visualisation*, Australian Computer Society, Inc., 2001, 31-37.
16. Selle, D., Spindler, W., Preim, B. and Peitgen, H.-O., Mathematical Methods in Medical Imaging: Analysis of Vascular Structures for Liver Surgery Planning, In: Enquist, B. and Schmid, W., eds., *Mathematics Unlimited - 2001 and Beyond,* Springer*,* 2000, 103-109.
17. Perona, P. and Malik, J., Scale-Space and Edge Detection Using Anisotropic Diffusion, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1990, 12(7), 629-639.
18. Sato, Y., Nakajima, S., Shiraga, N., Atsumi, H., Yoshida, S., Koller, T., Gerig, G. and Kikinis, R.,

Three-dimensional multi-scale line filter for segmentation and visualization of curvilinear structures in medical images, *Medical Image Analysis,* 1998*, 2*, 143-168.

19. Frangi, A. F., Niessen, W. J., Vincken, K. L. and Viergever, M. A., Multiscale Vessel Enhancement Filtering, *Medical Image Computing and Computer-Assisted Interventation (MICCAI)*, Springer, 1998*,* 1496.
20. Bauer, C. and Bischof, H., A Novel Approach for Detection of Tubular Objects and its Application to Medical Image Analysis, *Proceedings of the 30th DAGM Symposium on Pattern Recognition*, Springer, 2008.
21. Bennink, H.E., van Assen, H.C., Streekstra, G.J., ter Wee, R., Spaan, J.A.E. and ter Haar Romeny, B.M., A Novel 3D Multi-scale Lineness Filter for Vessel Detection, in: Ayache, N., Ourselin, S. and Maeder, A.J., eds, *Medical Image Computing and Computer-Assisted Interventation (MICCAI)*, Springer, 2007*, 4792*, 436-443.
22. Choi, W.-P., Lam, K.-M. and Siu, W.-C., Extraction of the Euclidean Skeleton Based on a Connectivity Criterion, *Pattern Recognition*, 2003, 36(3), 721-729.
23. Latecki, L., Li, Q., Bai, X. and Liu, W., Skeletonization Using SSM of the Distance Transform, *International Conference on Image Processing (ICIP)*, 2007, 5, 349-352.
24. Ilg, M. and Ogniewicz, R., The Application of Voronoi Skeletons to Perceptual Grouping in Line Images, *Proceedings of the 11th International Conference on Pattern Recognition*, 1992, 3, 382-385.
25. Palágyi, K., Tschirren, J. and Sonka, M., Quantitative Analysis of Intrathoracic Airway Trees: Methods and Validation, *Information Processing in Medical Imaging*, Springer, 2003, 2732, 222-233.
26. Lee, T., Kashyap, R.L. and Chu, C., Building Skeleton Models via 3-D Medical Surface/Axis Thinning Algorithms, *Graphical Models and Image Processing*, Academic Press, 1994, 56, 462-478.
27. Pelillo, M., Siddiqi, K. and Zucker, S., Matching Hierarchical Structures Using Association Graphs, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1999, 21(11), 1105-1120.
28. Gold, S. and Rangarajan, A., A Graduated Assignment Algorithm for Graph Matching, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1996, 18(4), 377-388.
29. Medasani, S., Krishnapuram, R. and Choi, Y., Graph Matching by Relaxation of Fuzzy Assignments, *IEEE Transactions on Fuzzy Systems*, 2001, 9(1), 173-182.
30. Charnoz, A., Agnus, V., Malandain, G., Soler, L. and Tajine, M., Tree Matching Applied to Vascular System, *Graph-based Representation in Pattern Recognition*, Springer, 2005, 183-192.
31. Tschirren, J., McLennan, G., Palagyi, K., Hoffman, E. and Sonka, M., Matching and Anatomical Labeling of Human Airway Tree, *IEEE Transactions on Medical Imaging*, 2005, 24(12), 1540-1547.
32. Kaftan, J.N., Kiraly, A.P., Naidich, D.P. and Novak, C.L., A Novel Multipurpose Tree and Path Matching Algorithm with Application to Airway Trees, *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, 2006, 215-224.
33. Graham, M.W. and Higgins, W.E., Globally Optimal Model-based Matching of Anatomical Trees, *Proceedings of SPIE - Medical Imaging 2006: Image Processing*, 2006, 614415.
34. VOXEL-MAN Group, Voxel-Man Organ Atlas, University Medical Center Hamburg-Eppendorf, 2008.
35. Lorenz, C. and Berg, J.v., A Comprehensive Shape Model of the Heart, *Medical Image Analysis*, 2006, 10(4), 657-670.
36. Whitaker, R. and Xue, X., Variable-Conductance, Level-Set Curvature for Image Denoising, *Proceedings of the International Conference on Image Processing*, 2001, 3, 142-145.
37. Lee, T.-C., Kashyap, R.L. and Chu, C.-N., Building Skeleton Models via 3-D Medial Surface/Axis Thinning Algorithms, *Graphical Models and Image Process (CVGIP)*, 1994, 56(6), 462-478.
38. Chen, Y., Laura, C.O., and Drechsler, K., Generation of a Graph Representation from Threedimensional Skeletons of the Liver Vasculature, *Proceedings of the 2nd International Conference on BioMedical Engineering and Informatics (BMEI)*, 2009.